



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

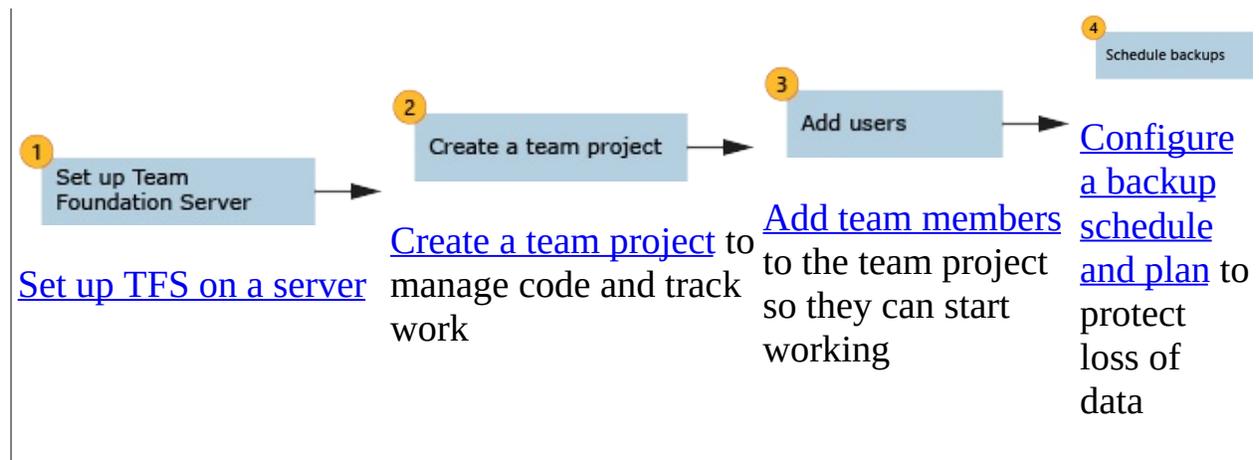
Administer Team Foundation Server

[See Also](#) [Send Feedback](#)

Administrators of an on-premises deployment of Visual Studio Team Foundation Server (TFS) install, configure, and maintain the deployment. They are tasked with supporting teams to have access to the resources to do their work. This includes managing permissions and access to servers, backing up and restoring data, and maintaining the security of the deployment.

Get started

If you're new to managing TFS, you can get a server up and running with these four steps.



Administer TFS users, groups, application-tiers, and databases

Manage users or groups in TFS



- [Add users to team projects](#)
- [Change access levels](#)
- [Restrict access to functions and tasks](#)
- [Set administrator permissions for team project collections](#)
- [Set administrator permissions for Team Foundation Server](#)

Configure and manage TFS resources

- [Configure an SMTP server to support alerts and feedback requests](#)
- [Change cache](#)



[settings for an application-tier server](#)

- [Manage team project collections](#)
- [Stop and start services, application pools, and websites](#)
- [Add SharePoint products to your deployment](#)
- [Support rollup of work and other fields](#)

Scale your deployment

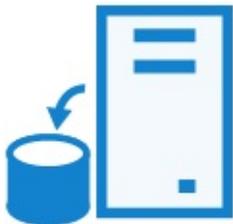
- [Move a team project collection](#)
- [Split a team project collection](#)
- [Move or clone Team Foundation Server \(hardware move\)](#)
- [Move Team Foundation Server from one environment to another](#)

Manage TFS reports, data warehouse, and analysis services cube



- Add reports to a team project
- Manually process the TFS data warehouse and analysis services cube
- Rebuild the TFS data warehouse and cube

Back up and restore TFS



- Configure a backup schedule and plan
- Understand backing up Team Foundation Server
- Manually back up Team Foundation Server
- Restore an application-tier server
- Restore data to the same location

- [Restore a deployment to new hardware](#)
- [Refresh the data caches on client computers](#)

[Technical Reference for Team Foundation](#)

- [Compatibility between Team Foundation clients and Team Foundation Server](#)
- [Architecture reference](#)
- [Permission reference for Team Foundation Server](#)
- [Command-line tools for TFS](#)
- [Work item field and database schema reference](#)
- [XML element reference](#)
- [Extending Visual Studio Application Lifecycle Management](#)



Administer build, test, and lab systems

[Configure and manage your build system](#)



- [Deploy and configure a build server](#)
- [Deploy and configure a build controller](#)
- [Deploy and configure build agents](#)
- [Scale out your build system](#)
- [Manage your build system](#)

[Setting Up Test Machines to Run Tests or Collect Data](#)

[Configure and administer Lab Management](#)

- [Configure Lab Management for](#)



SCVMM environments

- Planning Your Lab
- Security for test controllers and test agents

Additional Resources

- [TFS installation guide](#)
 - [Install TFS](#)
 - [Upgrade TFS](#)
 - [Configure features after an upgrade](#)
 - [Upgrade from Visual SourceSafe](#)
- [TFS Administration forum](#)
- [Visual Studio ALM Rangers Solutions and Projects](#)
- [Team Foundation Server Blog](#)
- [Technical Articles for Visual Studio Application Lifecycle Management](#)
- [Agile Development Showcase](#)
- [TFS on CodePlex](#)

See Also

Concepts

Track work with Visual Studio ALM and TFS

[What's new for Application Lifecycle Management in Visual Studio 2013](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Set up TFS on a server

[Send Feedback](#)

You can use a basic installation of TFS to share your code and improve collaboration with customers and team members. If you don't already have TFS, you can set it up on your own server by following the steps in this topic: [Install TFS](#), [Install the build service](#), and [Install Team Explorer](#).

If you don't want to manage your own server, you can use [Visual Studio Online](#) instead. If you'd rather install TFS in a different configuration, you can learn how in [Team Foundation Server install guide](#).

Install TFS

1.

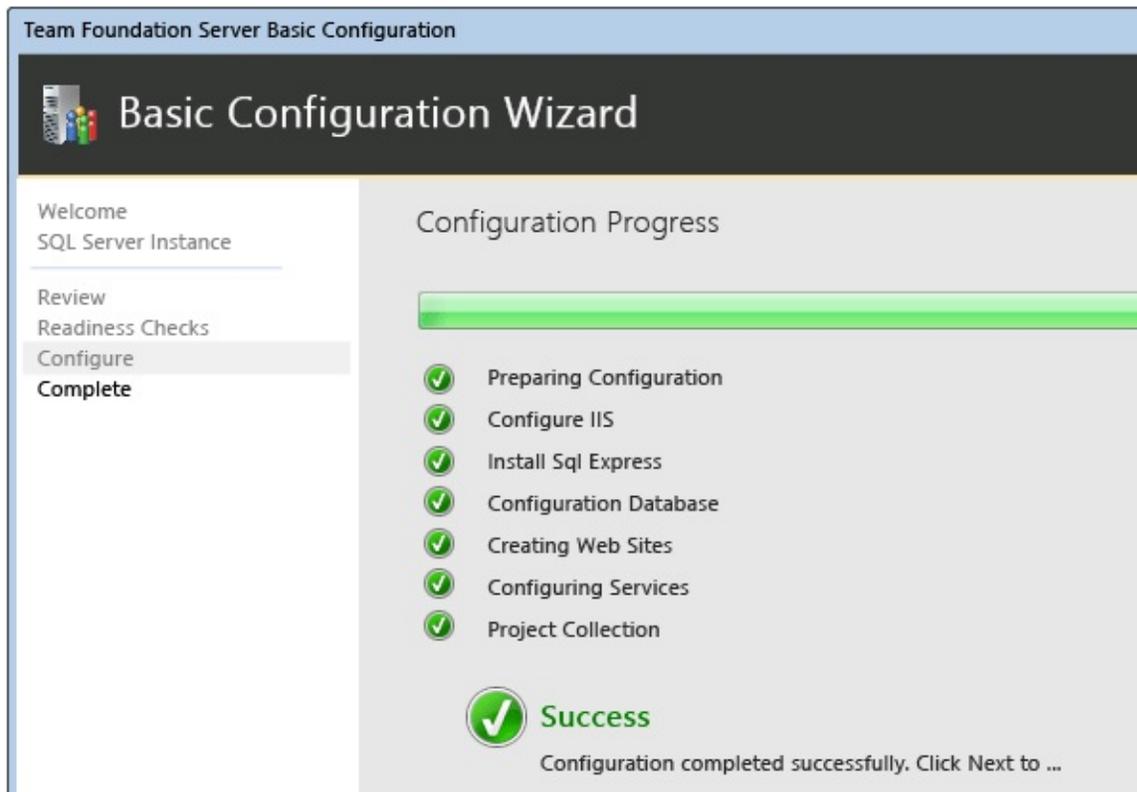
[Download TFS](#). You can install TFS on a client or on a server operating system. For clients, use Windows 7 or Windows 8, 32-bit or 64-bit. For servers, you have to use 64-bit. Go [here](#) for the complete list of system requirements for TFS.

If you install on a server operating system, you can add a SharePoint portal and SQL Server reporting later, even if you skip those features during initial installation.

- Use the Basic configuration wizard for a quick and simple installation.



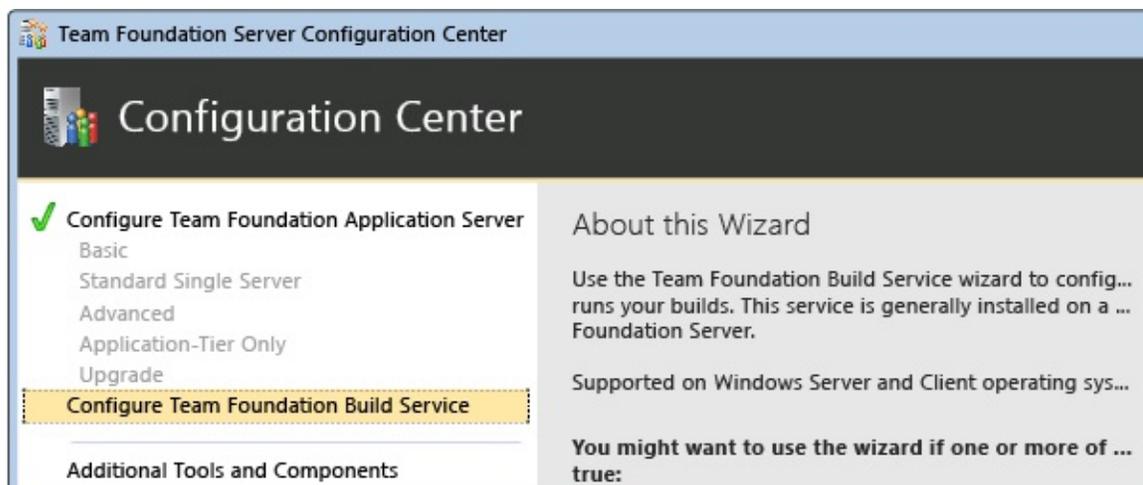
- After finishing, close the basic configuration wizard.



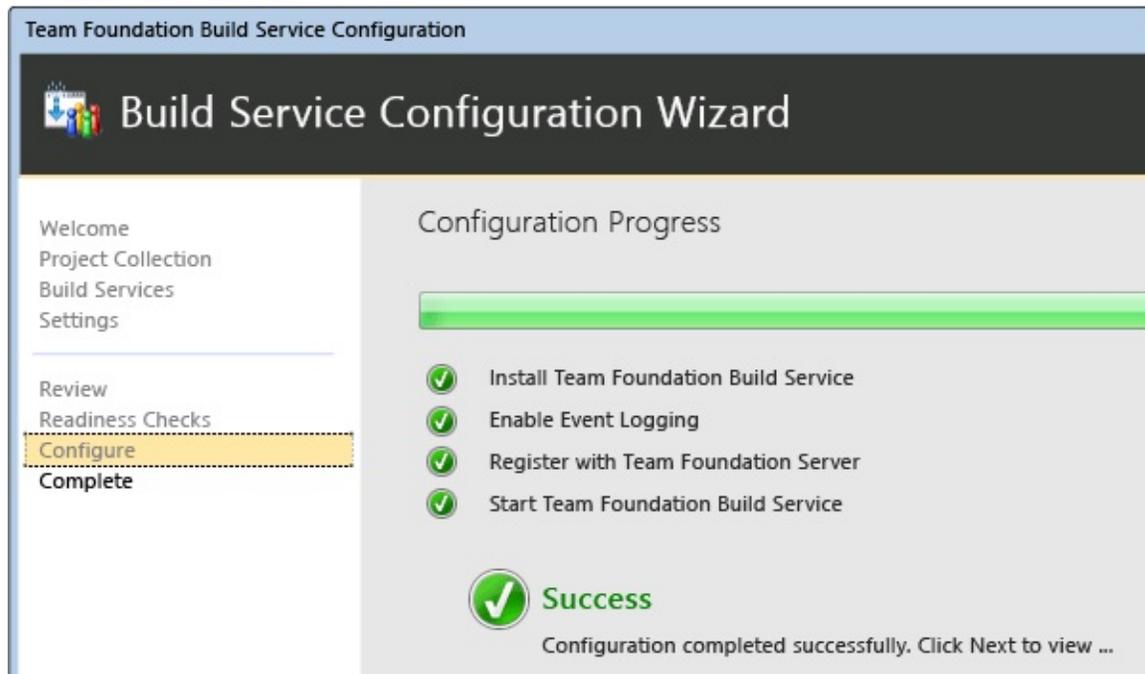
Once you've installed TFS, you might need to add antivirus exceptions. [Read this KB article for more information.](#)

Install the build service

1. After finishing the basic configuration wizard, you can install the build service on the same server. The build service automates builds of your software projects. Launch the build service configuration wizard.



2. Close the wizard after it finishes. Now your build server is ready to go.



■ Install Team Explorer

Team Explorer installs with each version of Visual Studio, or you can download it for free. Although Team Explorer isn't necessary for most administration tasks, it is required to create team projects, which your software teams will require, and to support work item tracking customization. Installing it on the server is a convenient way to ensure that you can create team projects whenever necessary without having to go to another computer.

- If you downloaded the DVD5 ISO image, browse the directory where you downloaded the image files, open the Team Explorer folder, and run the vs_TeamExplorer application.
- If you ran the tfs_server application to install TFS, then you'll need to download Team Explorer from the

[Visual Studio downloads page](#).

You are ready to connect to TFS and start creating your first team projects!

■ Try this next

- [Create a team project](#)

Q & A

Q: Can I install TFS in a different configuration?

A: Yes. You can install TFS in a variety of configurations, including ones that include SharePoint and reporting resources, ones that use network load balancing to manage workload between multiple application tiers, configure a proxy for remote offices, and a number of other options. You can learn more about installation options in the [Team Foundation Server install guide](#).

Q: How do I manage users in TFS?

A: Managing users individually is not recommended for any but the smallest deployments. You can use the built-in groups in TFS to manage your users, and you can also create groups to help manage users in your deployment. Read [Manage users or groups in TFS](#) to learn more.

Q: How does TFS manage its services?

A: Service accounts play a crucial role in TFS operations as described in [Service accounts and dependencies in Team Foundation Server](#).

Q: Where can I learn more about the underlying architecture in TFS?

A: See [Team Foundation Server architecture](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Create a team project

[Send Feedback](#)

You create a team project in Team Foundation Server (TFS) to establish a repository for source code and a place for teams to plan, track progress, and collaborate.

If you have a team project already, and want to start coding an application project, then see one of the following topics: [Develop your app in Team Foundation version control](#), [Set up Git on your dev machine](#), or [Creating Solutions and Projects](#).

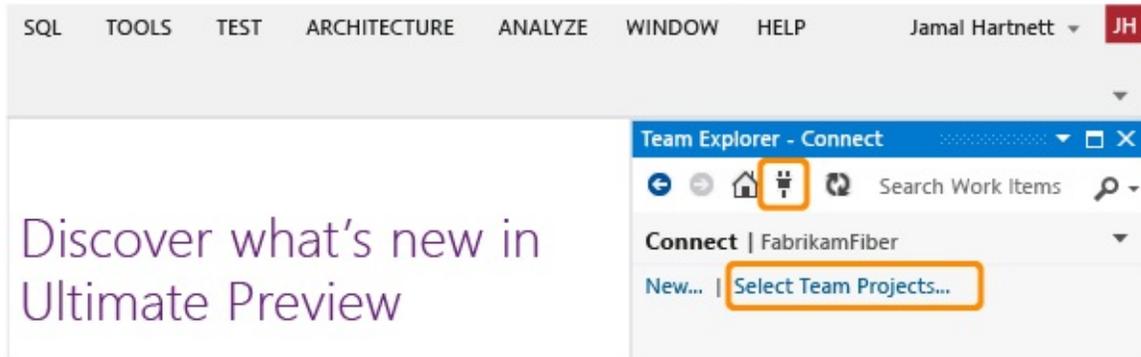
If you don't want to manage an on-premises server, you can [create a team project using Visual Studio Online](#). This topic applies only to creating a team project when you have deployed TFS on-premises.

1. If you're not a member of the Project Collection Administrators Group, [get added as one](#). To create team projects you must have the Create new projects permission set to Allow.
2. Ask your TFS administrator about the following resources and get additional permissions as needed:
 - Which team project collection you should connect to when you create your team project? If you installed TFS using the Basic Configuration Wizard, you have only one project collection named DefaultCollection. Unless you work in an enterprise organization and will be supporting hundreds of team projects, you should add all your team projects to a single project collection. If you need to create additional collections, see [Manage team project collections](#).
 - Has SQL Server Analysis Services and SQL Server Reporting Services

been configured for the deployment? If so, ask your administrator to [add you as a member of the Team Foundation Content Managers group](#) on the server that hosts SQL Server Reporting Services. Without these permissions, you'll be unable to create a team project.

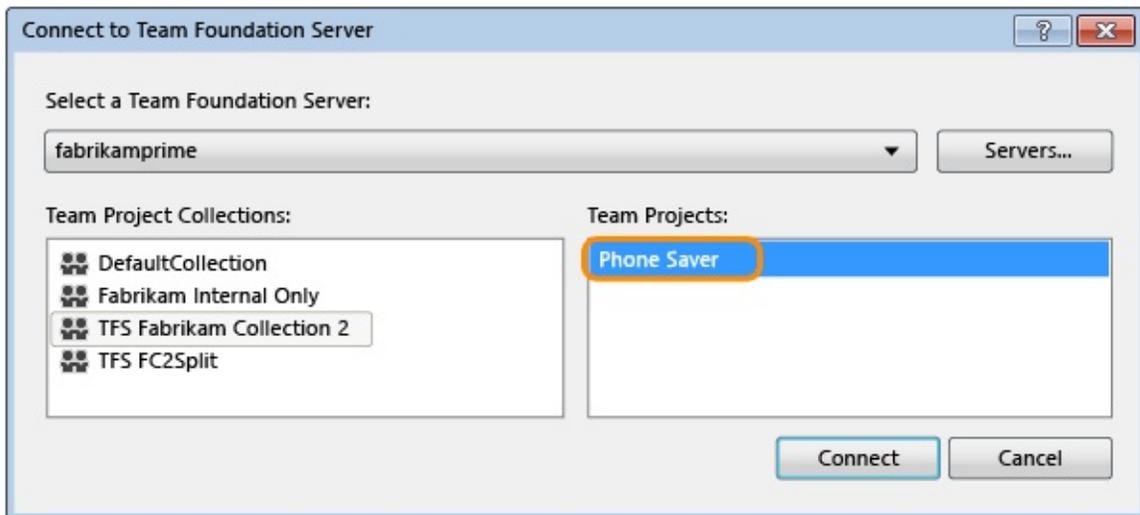
- Has a SharePoint Web application been configured for your deployment? If you want to configure a SharePoint portal when you create your team project, ask the SharePoint administrator to give you Full Control permissions on the server that hosts SharePoint Products. Otherwise, you can skip this step and configure a portal at a later time.

3. From Team Explorer 2013, connect to the server and team project collection where you want to create your team project.



Team Explorer is installed with Visual Studio. Or, you can install [Team Explorer for free](#).

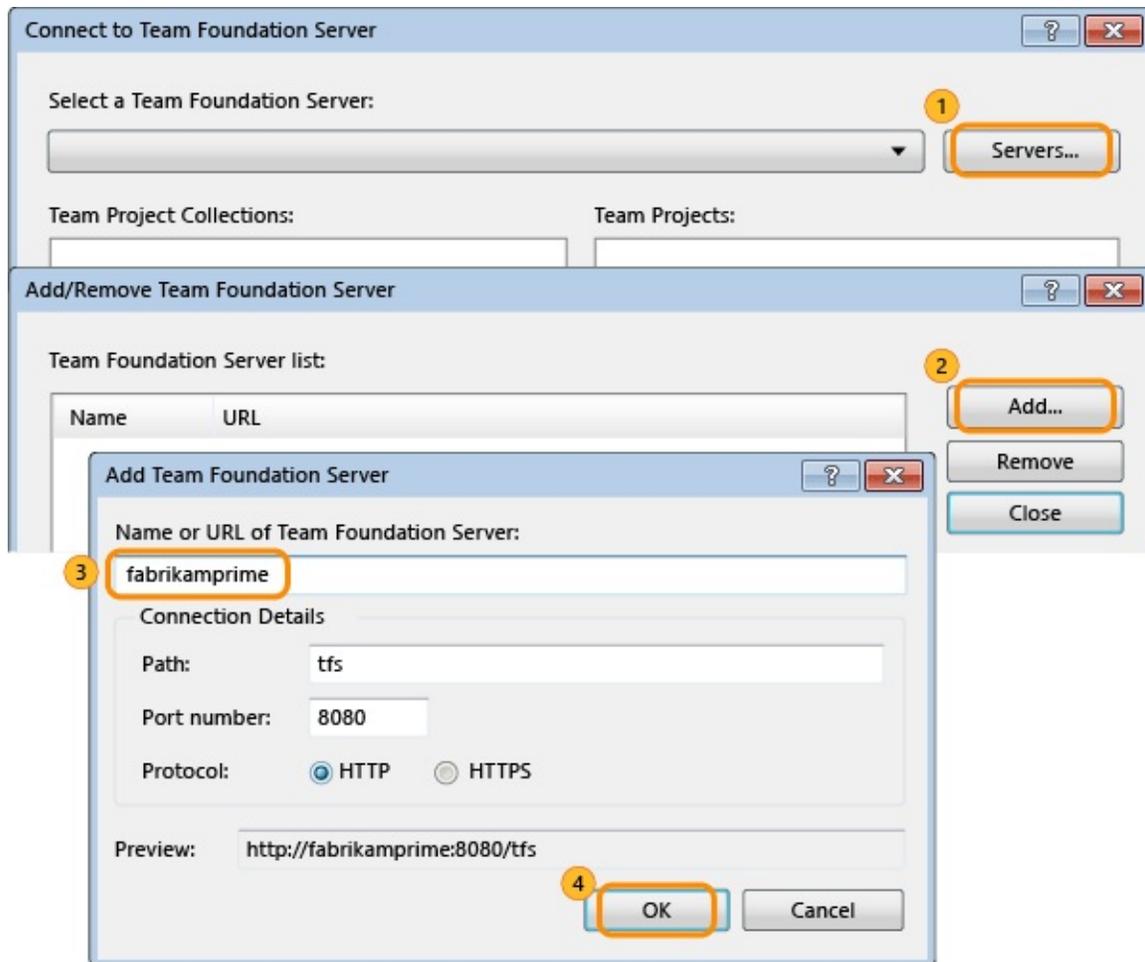
You must connect from a client that is at the same version level as TFS. That is, you must connect to TFS 2013 from Team Explorer 2013.



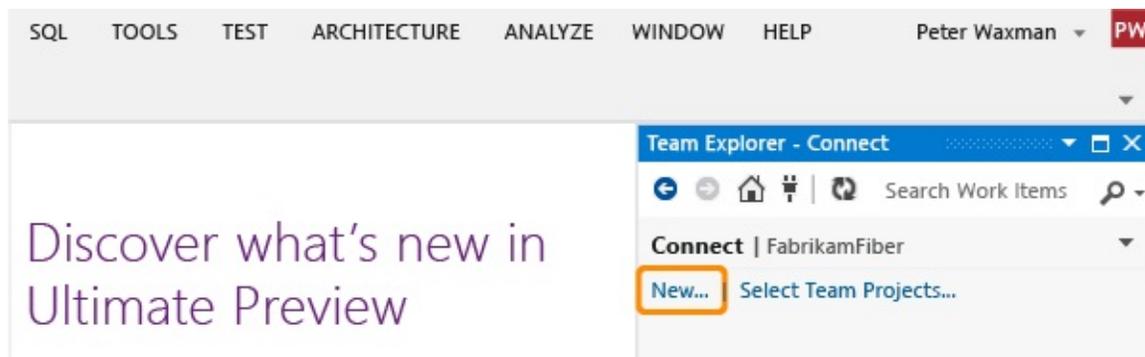
Tip

If you are running Team Explorer from a server that hosts SharePoint Products and SQL Server Reporting Services, you might need to run Visual Studio as an administrator.

4. If it's your first time connecting to TFS, you'll need to add TFS to the list of recognized servers.



5. Open the New Team Project Wizard.



6. Name the team project. Don't specify more than 64 characters.

New Team Project on FabrikamPrime/DefaultCollection



Specify the Team Project Settings

The New Team Project Wizard uses the team project name you type here when creating various components. After the team project is created, the name is used by team members to locate the team project.

Make sure that the name you pick for the team project is not already in use by Team Foundation Server or any other software used in the deployment (for example, SharePoint Products or SQL Server Reporting Services).

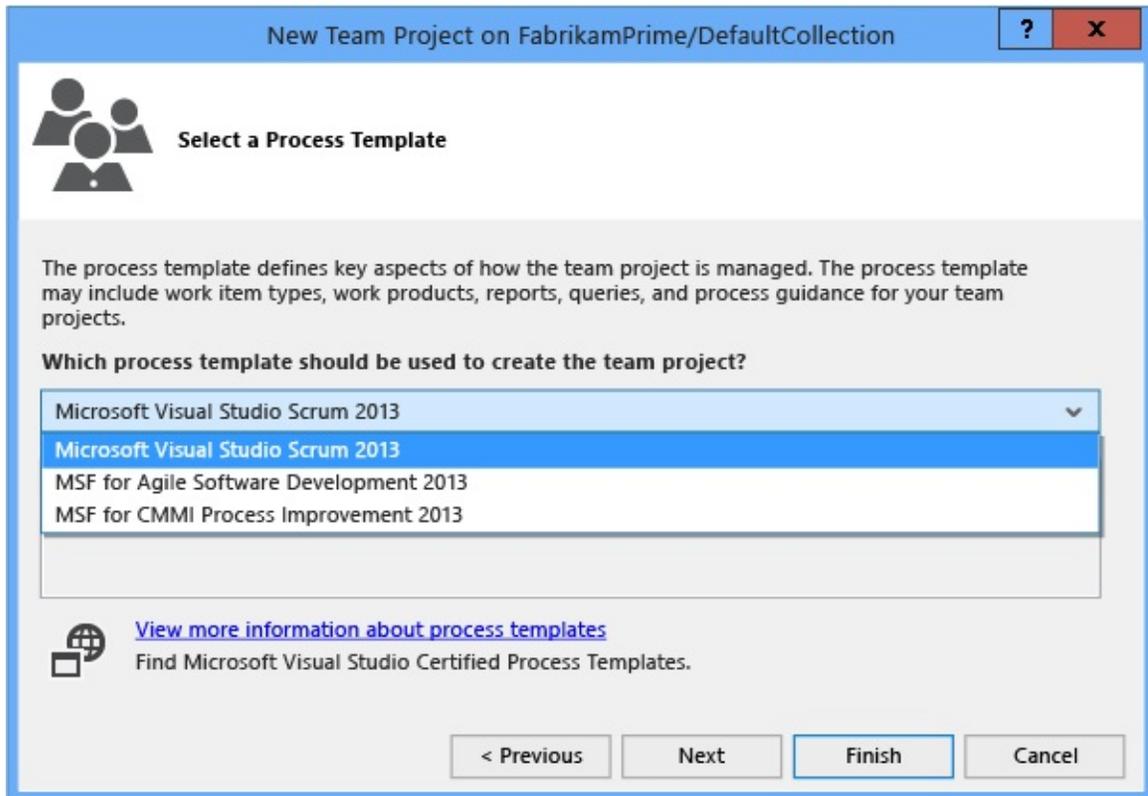
What is the name of the team project?

What is the description of the team project?

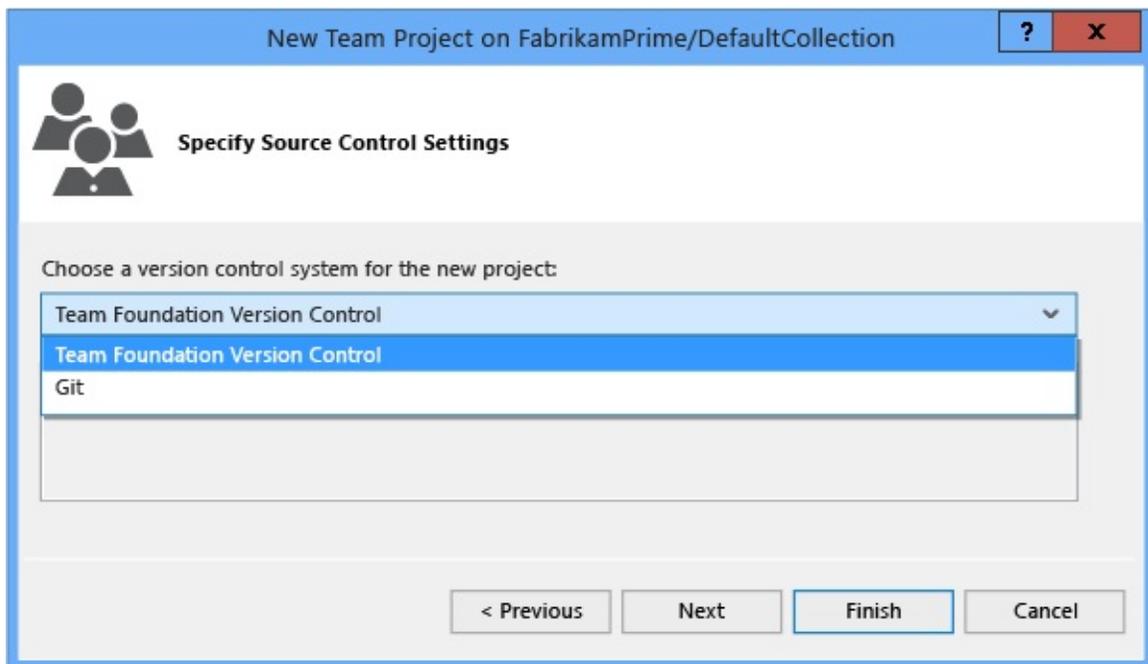
< Previous Next Finish Cancel

And, note that after you've created a team project, you can't change its name.

7. Choose a process template. For a comparison of the default process templates, see [Work with team project artifacts](#).

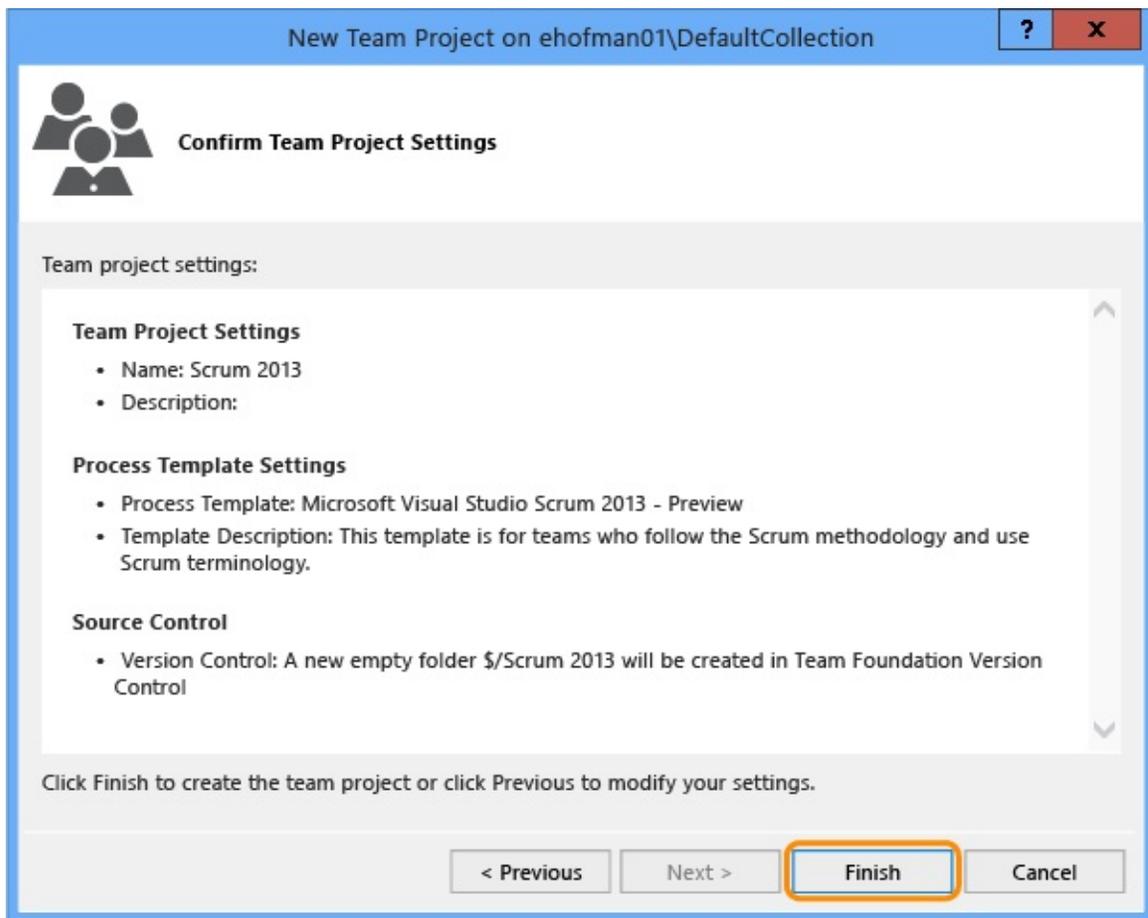


8. Choose your source control system.



Not sure which system to use? Learn more [here](#).

9. Unless your team project collection is configured to support a SharePoint project portal, you're done.



If the Next button is active, you can configure your project portal.

If the wizard encounters a problem, you'll receive an error message and a link to the project creation log. Review the log for specific errors and exceptions. See [Q & A](#) later in this topic for additional information.

10. When you're finished, you can see your team project in Team Explorer. You can also choose the Web Access link to connect to your team project using the Team Web Access.

Discover what's new in Ultimate 2013 Preview

You can find information about new features and enhancements in Ultimate 2013 Preview by reviewing the following sections.

[Learn about new features in Ultimate 2013 Preview](#)
[See what's new in .NET Framework 4.5.1](#)
[Explore what's new in Team Foundation Service](#)

[Minimize what's new information](#)

What's new on Microsoft Platforms

-  [Windows](#)
-  [Microsoft Azure](#)
-  [ASP.NET vNext and Web](#)
-  [Windows Phone](#)
-  [Microsoft Office](#)
-  [SharePoint Development](#)

Team Explorer - Home

Search Work Items

Home | FabrikamFiber

- Project
 - [Configure Workspace](#) | [Web Access](#) | [Task Board](#) | [Team Room](#)
-  **My Work**
-  **Pending Changes**
-  **Source Control Explorer**
-  **Work Items**
-  **Builds**
-  **Reports**
-  **Settings**
- Solutions
 - You must [configure](#) your workspace mappings to open solutions for this project.

▣ Try this next

[Add users to team projects](#) to enable other users to connect to your team project.

Also, with the team project created, you can start [developing your app in Team Foundation version control](#), [use Git](#), or [create an application](#).

To start planning work and collaborating as a team, see [Track work with Visual Studio ALM and TFS](#).

Q & A

Q: How do I add my custom process template to the list?

A: You'll need to first

[upload your template](#) using the Process Template Manager. To learn more about customizing a process template, go [here](#).

Q: Where is the log file located?

A: The log file is stored in %:\Users\user name\AppData\Local\Temp and labeled VSTS_TeamProjectCreation_yyyy_mm_dd_hh_mm_ss.log.

The log shows each action taken by the wizard at the time of the failure and may include additional details about the error. You can review the log entries to find network or file related issues by searching for Exception or Error.

Q: How do I resolve permission related errors?

A: If you receive an error message that states you don't have permission, go get those permissions: become a member of the [Project Collection Administrators group](#), [Team Foundation Content Managers group](#) become a member of the, and [Full Control permissions](#) on the server that hosts SharePoint Products.

Q: How do I resolve Error TF30169?

A: Error TF30169: The New Team Project Wizard was unable to download the process template {0}. indicates that SharePoint site process templates are not available on the server that hosts SharePoint Products.

Contact the system administrator for the server that hosts SharePoint Products and request the required process templates be added to the server. See SharePoint Products requirements for Team Foundation Server.

Q: How do I resolve Error TF30321?

A: Error TF30321: The name you typed is already used for another team project on the Team Foundation Server indicates that you should use a different name for your team project. The name you entered is either in active use or has undergone partial deletion, but not full deletion.

Q: How do I delete a team project?

A: You can delete a team project you no longer use, which helps simplify the navigation to team projects that are in use. See [Delete a team project](#).

Q: I deleted a team project, but now when I try to create a team project with the same name I get an error?

A: If a team project create or delete operation doesn't successfully finish, some components could be created or deleted even though others are not. In this event, you can't reuse the name associated with the team project.

To verify project deletion or remove remaining components associated with a partially deleted team project, use the [Delete a team project \[TFSDeleteProject\]](#). Then try again to create the team project with the same name.

Even with troubleshooting, you might not be able to use the same name. Some components of the deleted team project could be scheduled for deletion but not yet deleted.

Q: How do I resolve an error message related to a plug-in?

A: The process template used to create the team project contains several XML plug-in files. If one of these files contains a format or other error, an error message appears.

Review the project creation log to determine the plug-in that caused the error. After you discover the problem, you can either contact the developer or vendor that provided the plug-in, or attempt to fix the problem yourself. For more information, see [Customize a Process Template](#).

Q: How do I resolve a problem connecting to a server?

A: If you receive an error message about a problem connecting to a server, retrieving information from a server, or checking permissions to create projects, it could be caused by an incorrectly configured server in the deployment. This problem is especially common after a server move, failover, or other maintenance activity.

Contact the TFS system administrator and request that they verify the server configuration.

Q: Is there a diagnostic tool I can use that helps ensure my deployment is ready to create team projects?

A: Yes. You can use the [Best Practices Analyzer for Team Foundation Server](#). This tool performs specific checks to determine whether the deployment is configured to support the creation of a team project.

Q: My team project doesn't have reporting or a project portal. How do I add these resources?

A: See one of these topics:

- To add reporting: [Add reports to a team project](#).
- To add a SharePoint web application: [Add SharePoint products to your deployment](#).
- To configure a team project portal to use an existing website or SharePoint site: [Configure a project portal](#).

Q: Where can I go if I have more questions?

A: You can post a question or search for answers in the [Team Foundation Server - Project Management & Work Item](#) forum.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

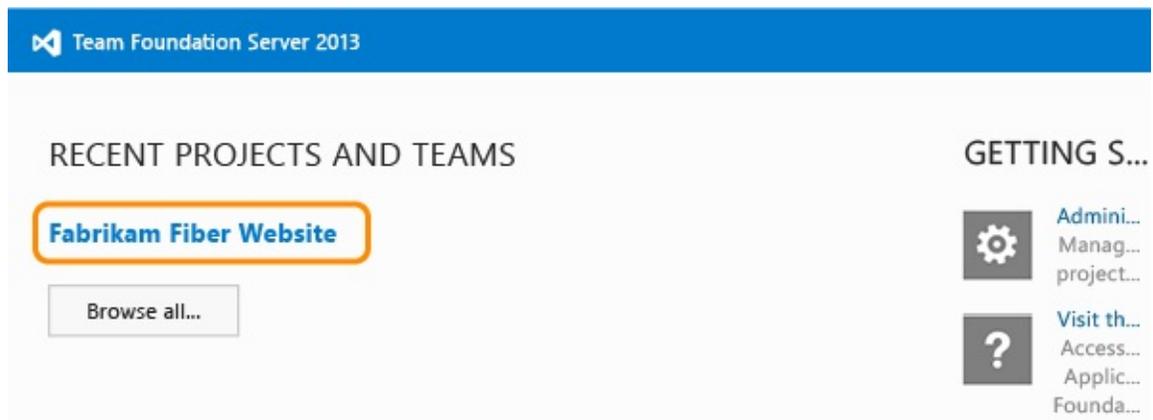
Add team members

[Send Feedback](#)

Members of a team project can contribute to source control, work item tracking, and other team activities. If you don't yet have a team project, create one. To add teams to a team project, go [here](#).

Add users or groups

1. If you're not yet an administrator, get added as a team administrator. Only administrators can add members to a team or team project.
2. On the TFS home page, choose your team or team project. If you don't see your team project listed, use Browse all to select it.



3. Manage members from the team home page.

The screenshot shows the Microsoft Teams interface for a team named 'Fabrikam'. At the top, there is a blue header with the text '...Website', the user name 'Raisa Pokrovskaya (Fabrikam)', and a 'Help' button. Below the header is a search bar labeled 'Search work items'. The main content area is divided into two columns. The left column features a '+ Bug' button and a 'more' dropdown menu. Below these is a 'Burndown' chart showing a red triangle. A yellow dashed box highlights a text area containing the text: '...them here on the team home page as tiles. ...itions and version control paths to your team.' The right column is titled 'ACTIVITIES' and lists several options: 'View backlog', 'View board', 'View work items', 'Request feedback', 'View reports', and 'Open new instance of Visual Studio'. Below this is a 'MEMBERS (1)' section showing the user 'Raisa Pokrovskaya (Fabrikam)' and a 'Manage all members...' button. At the bottom of the right column is an 'ADMINISTRATION' section with links for 'Configure schedule and iterations...' and 'Configure work areas...'.

4. Add an individual Windows user account or a group.

The screenshot shows a dialog box titled 'MANAGE MEMBERS OF FABRIKAM FIBER WEBSITE TEAM'. At the top left, there is an 'Add...' button with a dropdown arrow, a refresh icon, and a search field. Below the search field is a list of options: 'Add Windows user or group' (highlighted with an orange box), and 'Add TFS group'. To the right of the list is a table with one row containing the text 'NORTHAMERICA\ctsodev1' and a 'Remove' button. At the bottom right of the dialog box is a 'Close' button.

ADD A WINDOWS USER OR GROUP

To add a Windows user or group that is not currently known to Team Foundation Server, type the domain/user... the identity is known, just type the display name.

Identities [browse](#) | [check name](#)

The first time an account is added to TFS, you must enter the full domain name and the alias. Then you can then browse for that name by display name as well as account name. To learn more, see [Set up groups for use in TFS deployments](#).

Tip

You must enter user and group names one at a time. However, after entering a name, the account is added to the list, and you can type another name in the Identities text box before choosing to save your changes.

5. To verify that you've added all accounts, return to the home page and view the list of team members.

The screenshot shows the Microsoft Teams interface for a team named 'Fabrikam'. At the top, there is a blue header with the text '...Website', the user name 'Raisa Pokrovskaya (Fabrikam)', and a 'Help' button with a gear icon. Below the header is a search bar labeled 'Search work items'. The main content area is divided into two columns. The left column contains a '+ Bug' button, a 'more' dropdown, and a 'Burndown' chart showing a red shaded area that tapers to the right. Below the chart is a yellow highlighted box with the text: '...them here on the team home page as tiles. ...itions and version control paths to your team.' The right column contains an 'ACTIVITIES' section with links: 'View backlog', 'View board', 'View work items', 'Request feedback', 'View reports', and 'Open new instance of Visual Studio'. Below this is a 'MEMBERS (6)' section listing team members with their names and the organization 'Fabrikam': Johnnie McLeod, Christie Church, Nicole Zamora, Raisa Pokrovskaya, Francis Totten, and Jamal Hartnett. At the bottom of the members list is a link 'Manage all members...'.

6. Send the URL for your team project (for example, <http://fabrikamprime:8080/tfs/DefaultCollection/Fabrikam%20Fiber%20We>) to the new team members so that they can start contributing to the team project.

By default, team members inherit the permissions afforded to members of the team project Contributors group. Members of this group can add and modify source code, create and delete test runs, and create and modify work items. They can collaborate with other team members and check in work to the team's code base or collaborate on a Git team project.

Control Panel > DefaultCollection > Fabrikam Fiber Website

Overview Iterations Areas Team field **Security** Alerts Version Control

Groups Users

Create TFS group

Search...

Teams

- Fabrikam Data Project Team
- Fabrikam Fiber Website Team**

TFS Groups

- Build Administrator
- Contributors
- Project Administrators
- Project Valid Users
- Readers
- Reviewers

Fabrikam Fiber Website > Fabrikam Fiber Website T...

Permissions Members Member of

The default project team.

Permission	Member of
Create test runs	Inherited allow
Delete team project	Not set
Delete test runs	Inherited allow
Edit project-level information	Not set
Manage test configurations	Inherited allow
Manage test environments	Inherited allow
View project-level information	Allow
View test runs	Inherited allow

Clear explicit permissions

Save Changes Undo Changes

To learn more about permission settings, go [here](#).

Next steps

- If your deployment includes reporting or SharePoint Products, add users to those resources.
- Help protect against data loss by configuring backups for TFS.

Q & A

Q: How do I grant access to all features?

A: If you don't see test case management tools, team rooms, or Agile portfolio management features such as portfolio backlog levels, you might not have the correct level of access to the features in Team Web Access (TWA). Go here to learn how to grant or limit access to TWA features.

Q: How do I add a team administrator and what activities can they manage?

A: You can add a team administrator from the administration context for the team. Team administrators can add other team administrators, team members, team favorites. They can also customize the Kanban board and add members and events in a team room.

Q: How do I restrict access to select functions in a team project?

A: By default, users who have permissions to access one project within a collection can view other projects within that collection, even if they don't have permissions to modify work items or perform other actions in that project. You can restrict individuals or TFS groups from creating or modifying select artifacts.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Configure a backup schedule and plan

[Send Feedback](#)

If you administer systems, you're probably already familiar with all the reasons why a good backup set is crucial. No one wants to be the administrator of a server that goes down without a complete set of backups in place. Fortunately, you can back up data for TFS by using the Scheduled Backups tool in the administration console. If you regularly back up those databases, you reduce the risk of losing productivity or data because of equipment failure or other unexpected events. Unlike previous methods, this tool also backs up the SharePoint databases that TFS uses, if TFS is configured to use SharePoint.

⚠️ Important

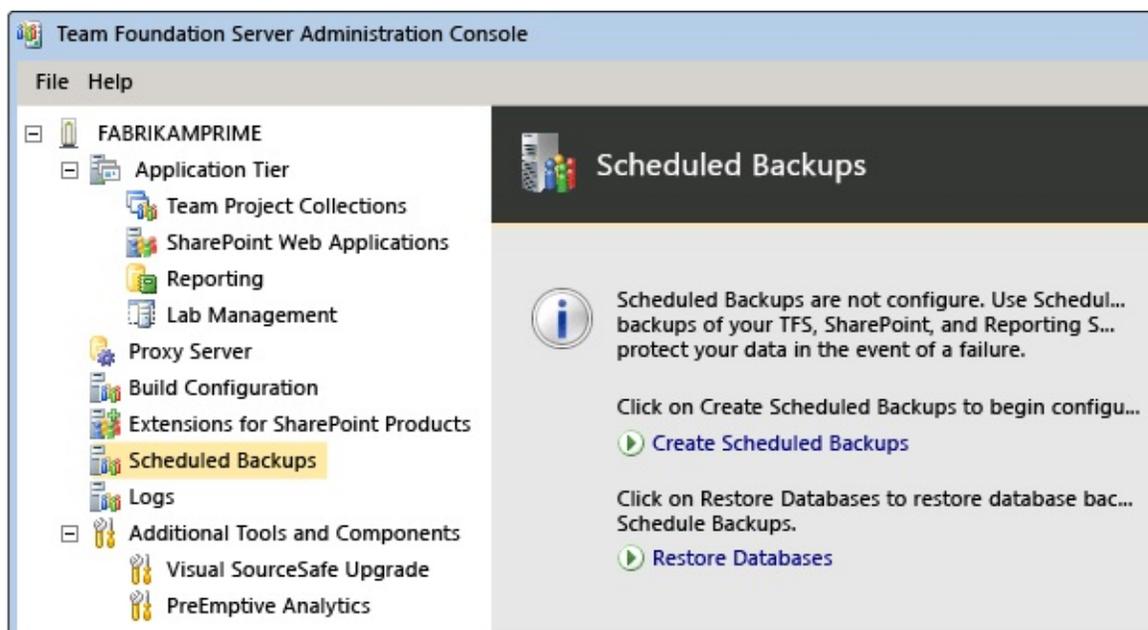
If you are using the Enterprise or Datacenter edition of SQL Server and you want to restore the collection and configuration databases to a Standard edition of SQL Server, then before you make the backup, you must turn off SQL Server compression by following the steps in the [Microsoft Knowledge Base article](#).

■ Create a backup plan

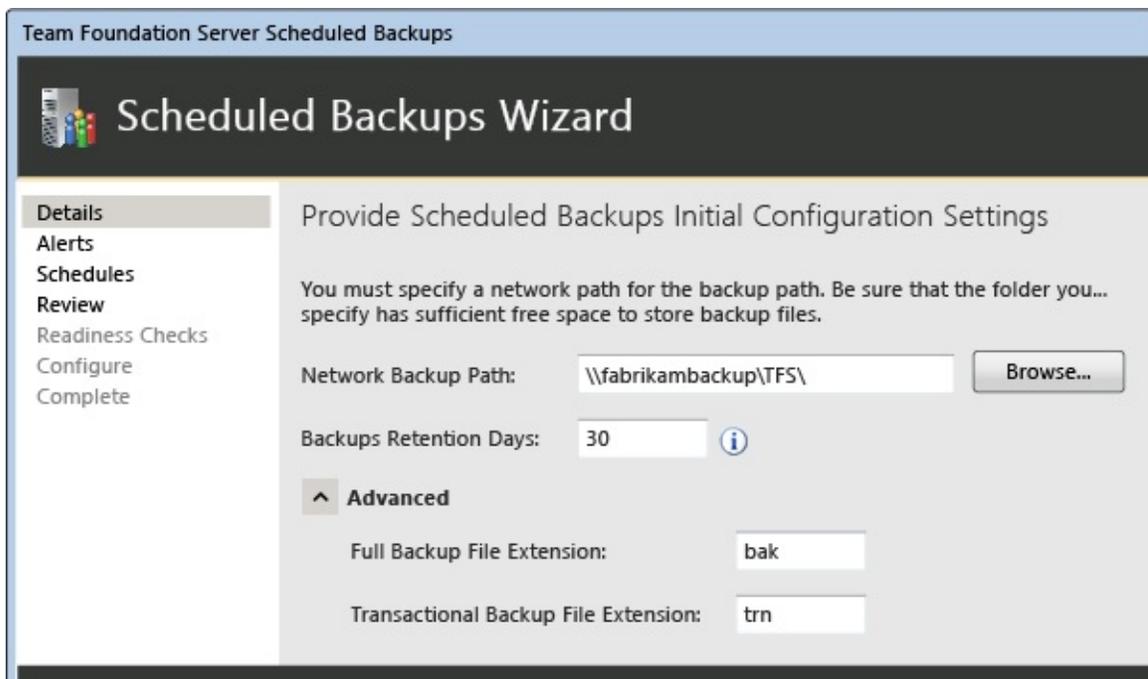
1. If you're not an administrator for TFS, a member of the SQL Server System Administrators group, and (if your deployment uses SharePoint Products) a member of the Farm Administrators group, get those permissions now.

In addition, the service account for TFS (TFSService) must have SQL Server Perform Back Up and Create Maintenance Plan permissions set to Allow on each instance of SQL Server that hosts the databases that you want to back up, and Full Control on the network share, folder, or storage device where the backups will be kept.

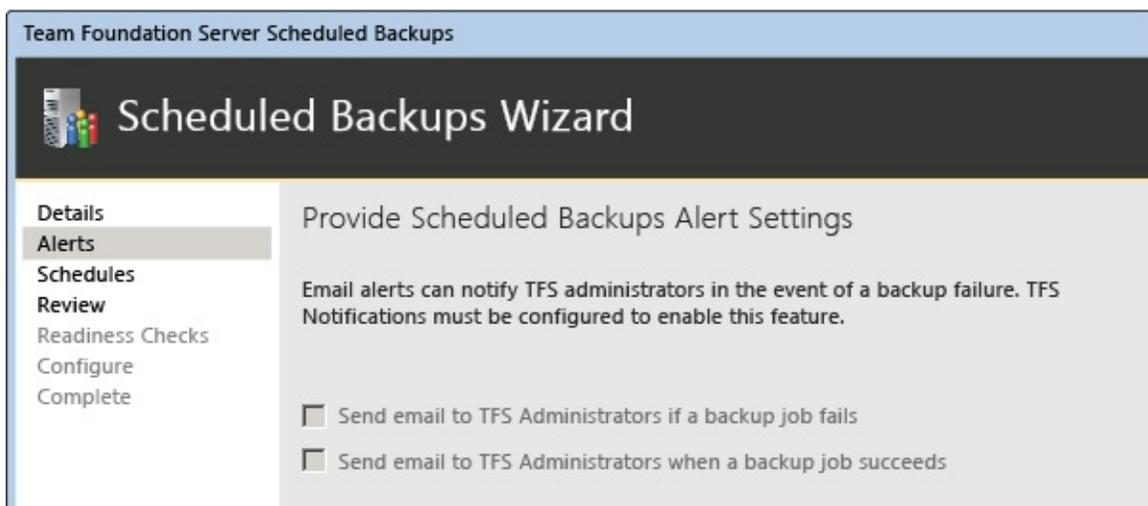
2. Open the administration console for TFS and on the Scheduled Backups page, launch the wizard for creating a backup schedule.



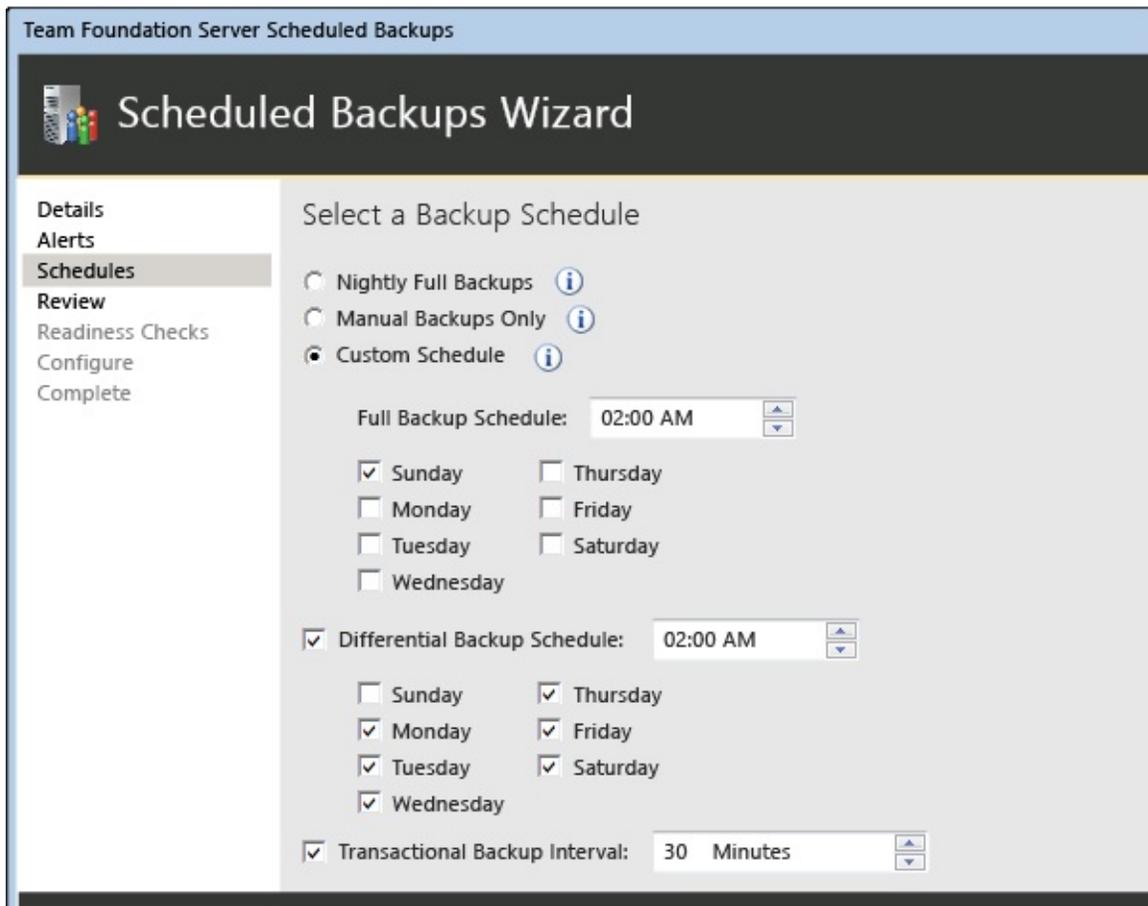
3. Backups must be stored on a network-accessible location, and both the account that configures the scheduled backup and the service account for TFS must have Full Control for that location. You can also choose how long a backup set will be kept and the file extensions used for backup types.



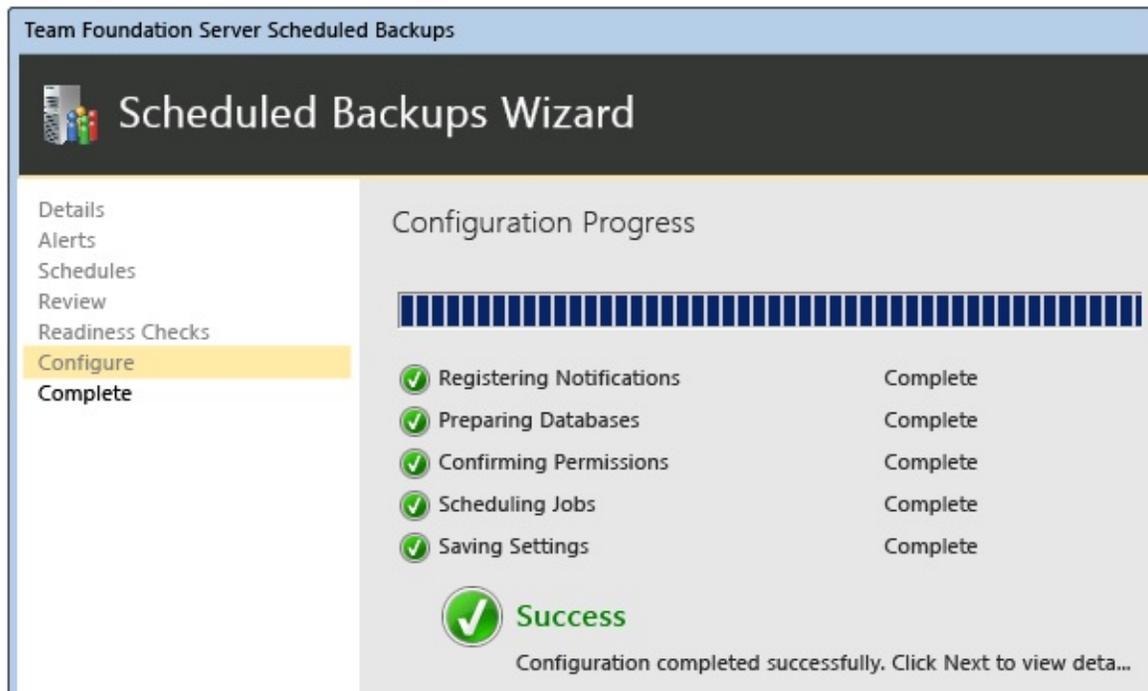
4. If your server is configured with SMTP support, you can select email alerts for specific events. If not, all selections will be dimmed.



5. Choose between two default schedules, or create your own custom schedule.



6. Complete the wizard. If your deployment uses reporting, you will be prompted for a password in order to back up the encryption key for reporting.



7. Once you've configured the scheduled backups, you can allow them to run as scheduled. You can also choose to take an immediate backup, which will back up your data right away while leaving your plan in place. This is particularly recommended if your scheduled backups will not occur for a significant amount of time, or if you do not already have a recent backup available.

Note

If another administrator chooses to take an immediate backup, that administrator must also have Full Control on the location where the backups are saved.

Q: Where can I learn more about backups in TFS?

A: You can learn more about the kinds of backups available in

[Understand backing up Team Foundation Server.](#)

Q: I don't want to use the Scheduled Backups tool. Are there other methods for backing up the system?

A: Yes. You can manually create backup plans. Learn how [here](#).

Q: How do I restore TFS from backups?

A: You use the Restore tool, but the steps vary depending on whether you need to restore data to a new server, or whether you want to restore data to the same server.

Q: Can I move TFS?

A: Yes, you can move TFS to new hardware. You can also change its environment, such as its domain.

- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Manage users or groups in TFS

[Send Feedback](#)

For users to have access to your team project in Team Foundation Server (TFS), you need to grant them access. If you're an administrator for a small team and restricting access isn't important, simply add your team members to TFS.

However, if you need to grant access to a large number of users who perform different roles within the team, then the recommended practice is to create Windows or Active Directory groups, add these groups to TFS groups, and add the same groups to grant access to additional resources.



The last three steps are optional. You only have to grant permissions for reports or the project portal if your team project has been provisioned with SQL Server Reporting Services or a SharePoint site. Also, you only have to change access levels if you need to enable an entire group to access premium features or if you want to enable stakeholders or non-licensed users limited access.

Grant users and groups access to team projects and resources:

- Set up groups for use in TFS deployments
- Add users or groups to team

projects

- Change access levels

Limit access to functions for select users or groups:



- [Restrict access to functions and tasks](#)
- Grant non-licensed users limited access
- Give reviewers permissions to provide feedback

Provide administrative access:

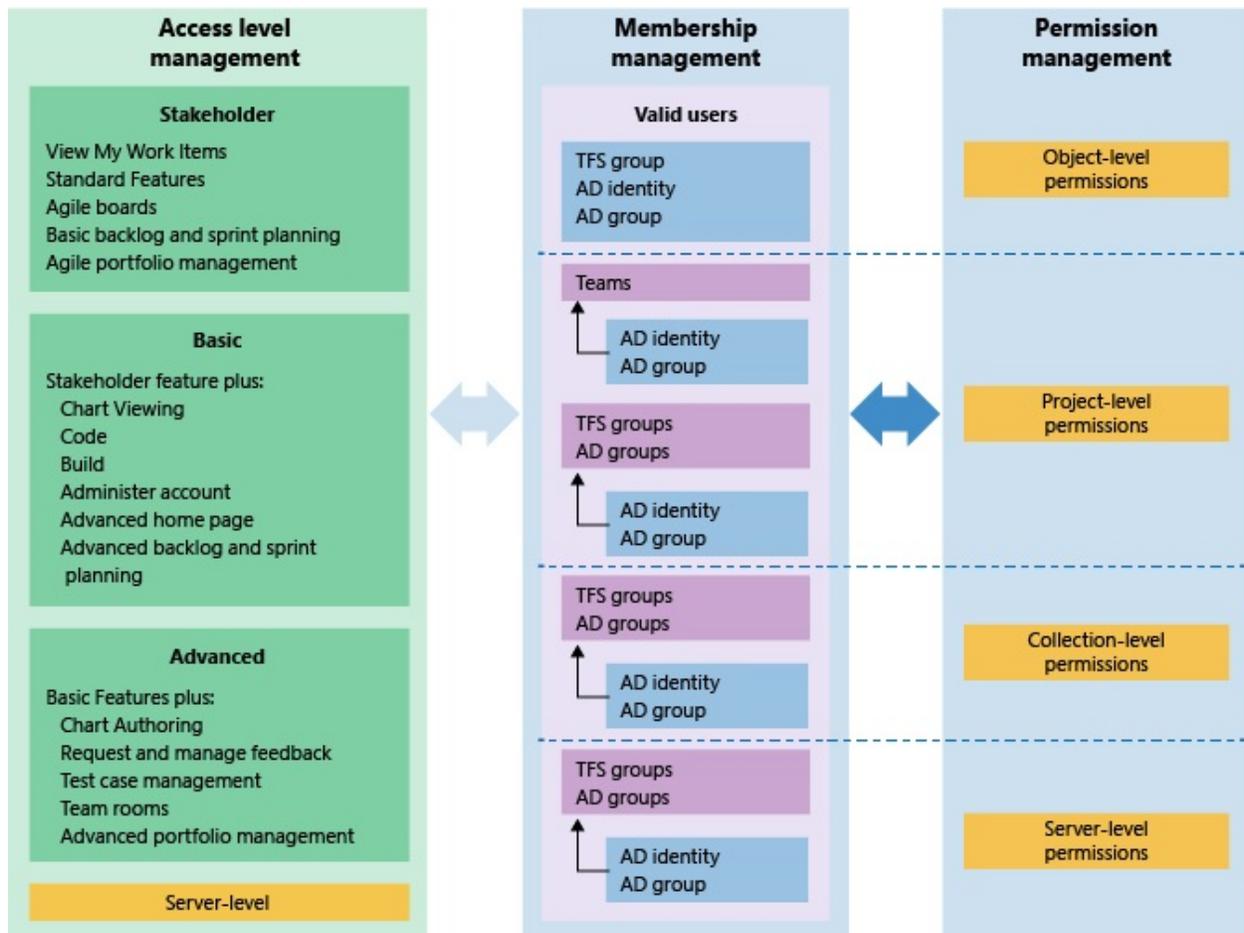
- Add a team administrator
- Add accounts to administer project collections
- Add accounts to administer TFS

How TFS manages access

If you need to make sure that the right users have the correct access or permissions to features and functions, it helps to understand that TFS controls access through three inter-connected functional areas:

- Access level management controls access only to features provided through TWA, the TFS web application. Based on their user license, administrators grant access to Basic, Advanced, or Stakeholder (previously labeled Standard, Full, and Limited) set of features. To learn more, see [Change access levels](#).
- Membership management supports adding individual Windows user accounts and groups to default TFS groups. Also, you can create TFS groups. Each default TFS group is associated with a set of default permissions. All users added to any TFS group are added to the Valid Users group. A valid user is someone who can connect to the team project.
- Permission management controls access to specific functional tasks at different levels of the system. Object-level permissions set permissions on a file, folder, build definition, or a shared query. Permission settings correspond to Allow, Deny, Inherited allow, Inherited deny, and Not set.

Each functional area uses groups to simplify management across the deployment. You add users and groups through the TFS web service administration pages. Permissions are automatically set based on the TFS group that you add users to, or based on the object, project, collection, or server level to which you add groups. On the other hand, access level management controls access for all users and groups at the server level.



Notes:

- AD: Active Directory. You can create local groups or Active Directory groups to manage your users. If you decide to use groups, make sure that membership in those groups is limited to TFS users. Because group membership can be altered by their owners at any time, if those owners did not consider TFS when they created those groups, their changes to membership can cause unwanted side effects within TFS.

Here's what you need to know about permission settings:

- Allow or Deny explicitly grants or restricts users from performing specific tasks, and are usually inherited from group membership.
- Not set implicitly denies users the ability to perform tasks that require that permission, but allows membership in a group that does have that permission set to take precedence, also known as Inherited allow and

Inherited deny.

- For most groups and almost all permissions, Deny trumps Allow. If a user belongs to two groups, and one of them has a specific permission set to Deny, that user will not be able to perform tasks that require that permission even if they belong to a group that has that permission set to Allow.

For members of the Project Collection Administrators or Team Foundation Administrators groups, Deny doesn't trump Allow. Permissions assigned to these groups take precedent over any Deny set within any other group to which that member might belong.

- Changing a permission for a group changes that permission for all users who are granted that permission through their membership in that group. In other words, depending on the size of the group, you might affect the ability of hundreds of users to do their jobs by changing just one permission. So make sure you understand the impact before you make a change.

Two useful tips for understanding the effects of change: The Member of tab shows the groups that an individual user or group belongs to. You can also hover over an inherited permission, and a Why? icon will appear. If you choose it, a dialog box will open with more information.

The screenshot shows the TFS Security interface for the 'Contributors' group. The breadcrumb path is 'Control Panel > CSI > Fabrikam Fiber'. The 'Security' tab is active, and the 'Member of' sub-tab is selected. The main content area shows the group's permissions:

Permissions	Members	Member of
Members of this group can add, modify, and delete items within the team project.		
Create tag definition	Inherited allow	Why?
Create test runs	Allow	
Delete team project	Deny	

Q & A

Q: I just want to access the code. How do I do that?

A: Once you've been added as a team member or contributor to TFS, see Team Foundation Version Control or Git.

Q: What permissions do I need before I can add users and manage permissions in TFS?

A: To add users or groups to your team project and manage permissions for you team project, you must belong to the Project Administrators group for that team project, or have Edit project-level permissions set to Allow. Project Administrators are granted the following default permissions.

Code	Work	Build	Test
Object and project-level:	Area node-level	Object and project-level:	Area node-level
TFVC: Administer labels Check in Check in other users' changes Check out Label Lock Manage branch Manage permissions Merge Read Revise other users' changes Undo other users' changes Unlock other users' changes	Create child nodes Delete this node Edit this node Edit work View permissions View work items Shared queries Contribute Delete Manage Permissions Read	Administer build permissions Delete build definition Delete builds Edit build definition Edit build quality Manage build qualities Manage build queue Queue builds Retain indefinitely Stop builds View build definition View builds	Manage test plans Manage test suites
Git repositories Administer Branch creation Contribute Note management Read Tag creation	Project-level Create tag definition Delete team project Edit project-level information View project-level information		Create test runs Delete test runs Manage test configurations Manage test environments View test runs

To manage users, groups, or permissions for all team projects within a collection, you must belong to the Project Collection Administrators group, or have Edit collection-level permissions set to Allow.

Q: What kind of permissions do my users need?

A: You grant permissions to users based on the tasks they perform in TFS. In general, you'll want to grant the minimum set of permissions users need to do their job. You can use the following default groups and their associated permissions to manage most users and meet their needs.

Users	Team Foundation Server	SharePoint Foundation or SharePoint Server	SQL Server Reporting Services
Add accounts for people who need view-only access to the project.	Readers	Visitors	Browser
Add accounts for people who contribute to or manage the development of the software project.	Contributors	Members	Browser
Add accounts for people who			

manage user Project	Team
access to the Administrators Owners	Foundation
project or	Content
need to	Manager
configure or	
customize	
the project.	

To simplify management across the three systems, consider using the [TFSAdmin tool from CodePlex](#).

Tip

Unlike Team Foundation Server and SharePoint Foundation, SQL Server Reporting Services does not distinguish between projects. Therefore, if you add a group to Reporting Services, that group will have the same permissions for reports across all the projects in the collection, regardless of their permissions in individual projects. Keep this in mind when choosing what groups to add.

Q: If I add team members to the Contributor role, will they have all the permissions they need?

A: The Contributor role grants the most common permissions developers need to contribute to a team project. However, it doesn't allow users to create shared queries or to add area or iteration paths. You have to grant these permissions separately. Go here for queries, and here for area and iteration paths.

Code	Work	Build	Test
Object and project-level:	Area-node level	Object-level:	Area-node level
TFVC: Check in Check out Label Lock Merge Read Git repositories Administer Branch creation Contribute Note management Read Rewrite and destroy history Tag creation	Edit work View permissions View work items Shared queries Read Project-level View project-level information	Edit build quality Queue builds View build definition View builds Edit build quality Queue builds View build definition View builds	Manage test plans Manage test suites Create test runs Delete test runs Manage test configurations Manage test environments View test runs

Q: How do I manage service account permissions?

A: See TFS service accounts and dependencies.

Q: What tasks are stakeholders with limited access able to perform?

A: See Work as a stakeholder.

Q: Is there a reference for all of the permissions in TFS?

A: Yes. See TFS permissions.

Q: Are their command line tools I can use to manage access and permissions?

A: Yes. You can use the TFSSecurity command-line utility to create, modify, and delete TFS users and groups, as well as modify permissions for users and groups.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Set up groups for use in TFS deployments

[Send Feedback](#)

Managing users in TFS is much easier if you create Windows or Active Directory groups for them, particularly if your deployment includes SharePoint Foundation and SQL Server Reporting Services.

Users, groups, and permissions in Team Foundation Server deployments

Team Foundation Server, SharePoint Products, and SQL Server Reporting Services all maintain their own information about groups, users, and permissions. To make managing users and permissions across these programs simpler, you can create groups of users with similar access requirements in the deployment, give those groups appropriate access in the different software programs, and then just add or remove users from a group as needed. This is much easier than maintaining individual users or groups of users separately in three separate programs.

If your server is in an Active Directory domain, one option is to create specific Active Directory groups to manage your users, like a group of developers and testers for all projects in the team project collection, or a group of users who can create and administer projects in the collection. Similarly, you can create an Active Directory account for services that can't be configured to use the Network Service system account as the service account. To do so, create an Active Directory account for SharePoint Foundation and as the read-access data source account for reports in SQL Server Reporting Services.

Important

If you decide to use Active Directory groups in TFS, consider creating specific ones whose purpose is dedicated to user management in TFS. Using previously existing groups that were created for another purpose, particularly if they are managed by others who are not familiar with TFS, can lead to unexpected user consequences when membership changes to support some other function.

The default choice during installation is to use the Network Service system account as the service account for Team Foundation Server and SQL Server. If you want to use a specific account as the service account for security purposes or other reasons, such as a scaled-out deployment, you can. You might also want to create a specific Active Directory account to use as the service account for

SharePoint Foundation and as the data source reader account for SQL Server Reporting Services.

If your server is in an Active Directory domain but you don't have permissions to create Active Directory groups or accounts, or if you're installing your server in a workgroup instead of a domain, you can create and use local groups to manage users across SQL Server, SharePoint Foundation, and Team Foundation Server. Similarly, you can create a local account to use as the service account. However, keep in mind that local groups and accounts are not as robust as domain groups and accounts. For example, in the event of a server failure, you would need to recreate the groups and accounts from scratch on the new server. If you use Active Directory groups and accounts, the groups and accounts will be preserved even if the server hosting Team Foundation Server fails.

For example, after reviewing business requirements for the new deployment and the security requirements with the project managers, you might decide to create three groups to manage the majority of users in the deployment:

- A general group for developers and testers who will participate fully in all projects in the default team project collection. This group will contain the majority of users. You might name this group TFS_ProjectContributors.
- A small group of project administrators who will have permissions to create and manage projects in the collection. You might name this group TFS_ProjectAdmins.
- A special, restricted group of contractors who will only have access to one of the projects. You might name this group TFS_RestrictedAccess.

Later on, as the deployment expands, you might decide to create other groups.

To create a group in Active Directory

- Create a security group that is a local domain, global, or universal group in Active Directory, as best meets your business needs. For example, if the group needs to contain users from more than one domain, the universal group type will best suit your needs. For more information, see

[Create a New Group \(Active Directory Domain Services\)](#).

To create a local group on the server

- Create a local group and give it a name that will quickly identify its purpose. By default, any group you create will have the equivalent permissions of the Users default group on that computer. For more information, see [Create a local group](#).

To create an account to use as a service account in Active Directory

- Create an account in Active Directory, set the password policy according to your business requirements, and make sure that Account is trusted for delegation is selected for the account. For more information, see [Create a New User Account \(Active Directory Domain Services\)](#) and [Understanding User Accounts \(Active Directory Domain Services\)](#).

To create a local account to use as the service account on the server

- Create a local account to use as the service account and then modify its group membership and other properties according to the security requirements for your business. For more information, see [Create a local user account](#).

Try this next

- [Add users to team projects](#)

Q & A

Q: Can I use groups to restrict access to projects or features in TFS?

A: Yes, you can. You can create specific groups for restricting access to projects, for managing access levels, and other purposes.

Q: Is there a simpler way to manage permissions across TFS, SharePoint, and reporting?

A: Not within TFS itself, but consider installing and using the [Team Foundation Server Administration tool from CodePlex](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Add users to team projects

[Send Feedback](#)

As roles and responsibilities change, you might need to change the membership or permission levels for individual members of a team project. This is particularly true if your project depends on resources that use SQL Server Reporting Services or SharePoint Products because permissions for those resources are managed separately.

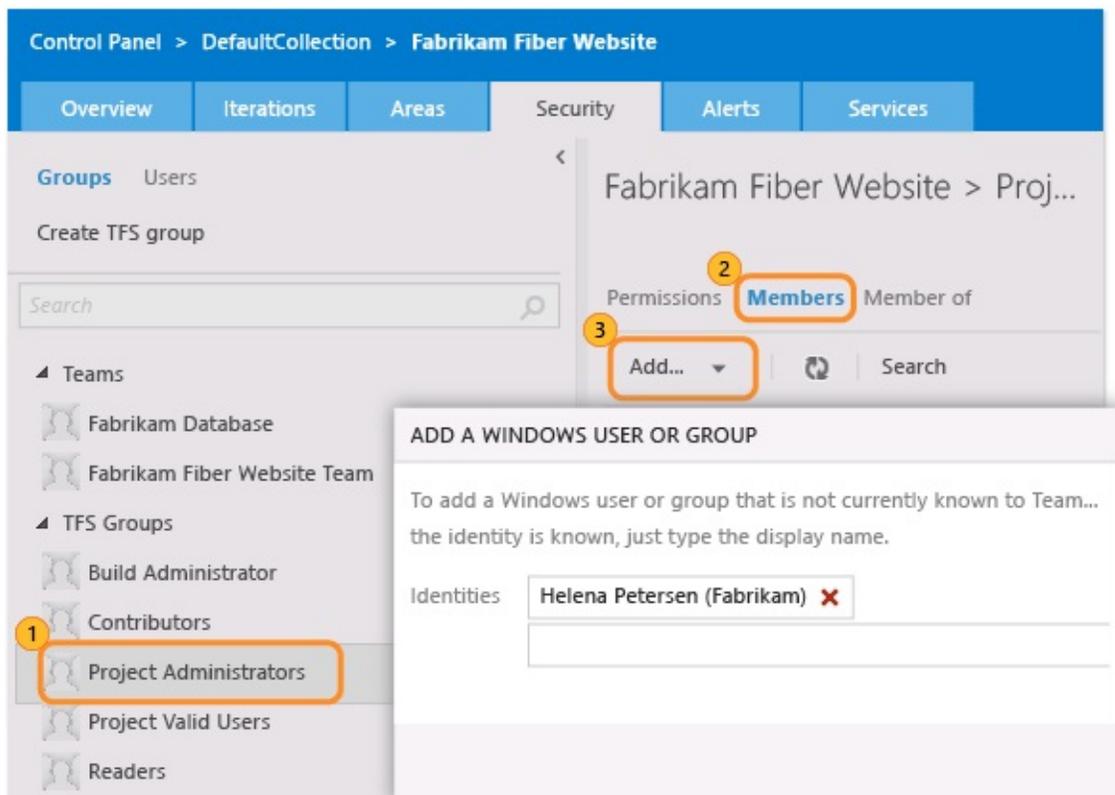
If all you want to do is add a user to an existing team in TFS, and you don't have to worry about specific permissions for other resources, skip this topic and simply add them to a team.

Permissions are different than access levels. Access levels control what features are visible to users in Team Web Access, and are dependent on user licenses; permissions control a user's ability to use features across TFS. If you're just trying to give someone access to a team room or to Agile portfolio management and test case management features, you'll want to change access levels, not permissions.

Verify your permissions in TFS

Before you change permission levels for others, make sure that you have the right level of permissions.

1. Open the administrative context for your team project. ⚙️
2. In the Security tab, under users, find your own name, and look at what groups you belong to and what permissions you have.
3. If you aren't a project administrator, you need to be. Find someone who is, and have them add you:



4. If you need to make changes at the team level, change your context to the team overview. If you aren't a team administrator, you can add yourself if you're already a project administrator. Otherwise, have someone add you.

Control Panel > DefaultCollection > Phone Saver > Phone Saver Team

Overview Security Alerts Version Control Services

Team Profile



Name
Phone Saver Team

Description
The default project team.

Administrators

Raisa Pokrovskaya (Fabrikam) ✘
Jamal Hartnett (Fabrikam) ✘

[+ Add](#)

Team Members

Add... ▼ | ↻

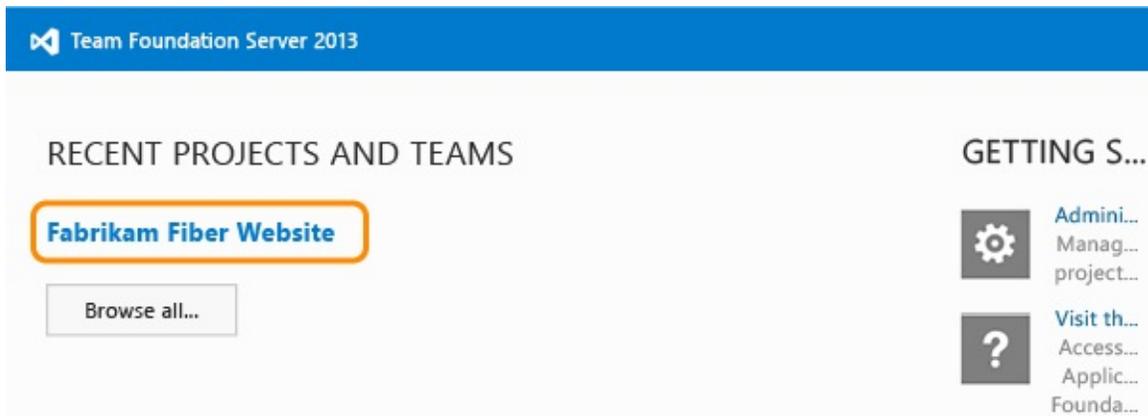
Display Name	User...
Christie Church (Fabrikam)	NOR...
Mateo Escobedo (Fabrika...	NOR...

5. Similarly, if you need to add users to SharePoint Products or SQL Server Reporting Services, make sure that you have the required permissions. For reporting, you must be either a member of the local Administrators group on the report server or

[be a member of a group specifically created to add users](#). The requirements for SharePoint Products are more complex. For more information about SharePoint 2013, [go here](#).

Add users to a project in Team Foundation Server

1. Open Team Web Access and navigate to the project where you want to add users or groups.



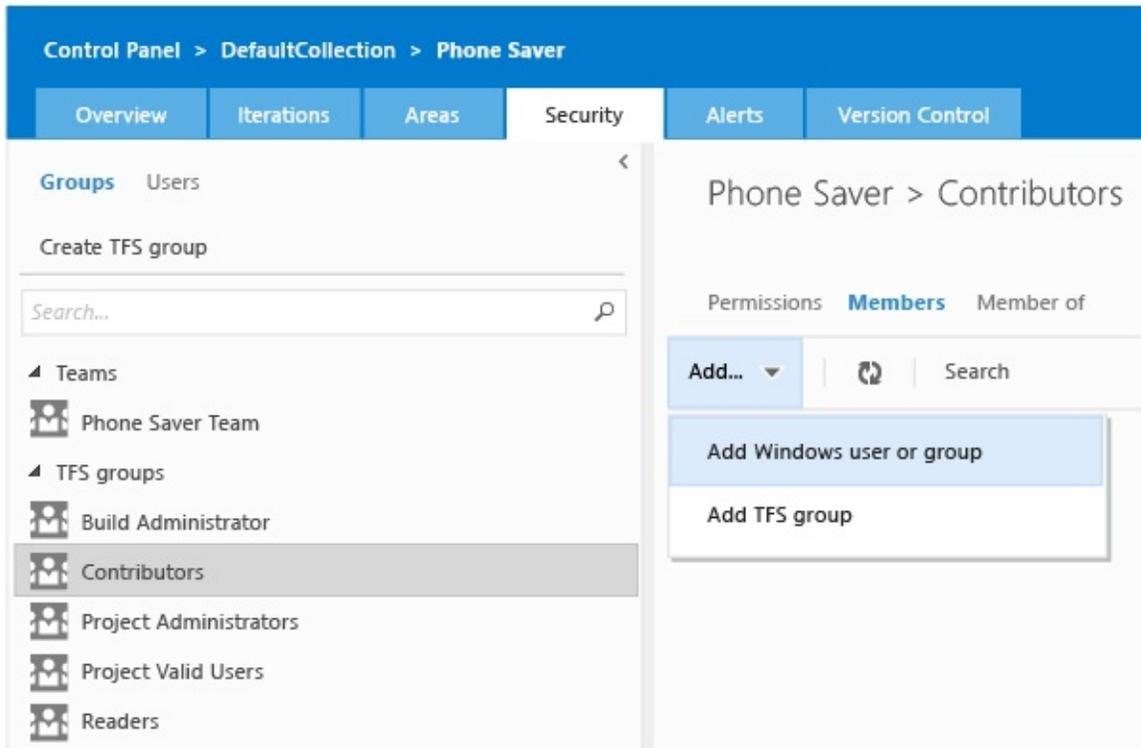
Tip: Managing user access to TFS is much easier if you add user groups, not individual users. Learn how to [Set up groups for use in TFS deployments](#).

Choose the gear icon  to open the administration context for the project, and then navigate to the Security tab.

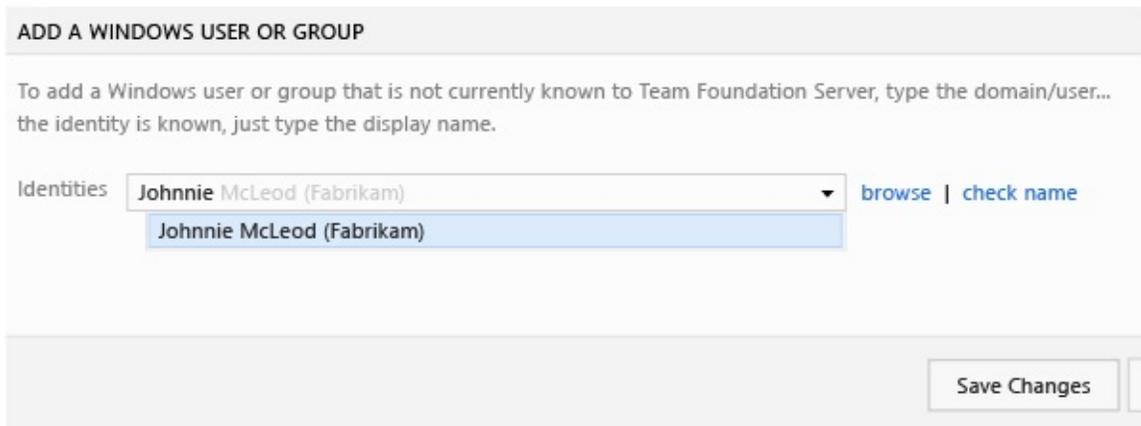
2. In Groups, choose one of the following:

- To add users who will require minimal access to the project, choose Readers.
- To add users who will contribute fully to this project, choose Contributors. By default, the team group created when you created the project is included as a member of the Contributors group, so you could choose to add the new user as a member of the team instead, and the user would automatically inherit Contributor permissions. For more information, see [Add team members](#).
- To add users who will act as project leads, choose Project Administrators.

After you have chosen one of those groups, add a user or a user group.



3. In Identities, specify the name of the user or group you want to add.



Tip

The first time you add a user or group to Team Foundation Server, you cannot browse to it or check the friendly name. After the identity has been added once in Team Foundation Server, you can just type the friendly name.

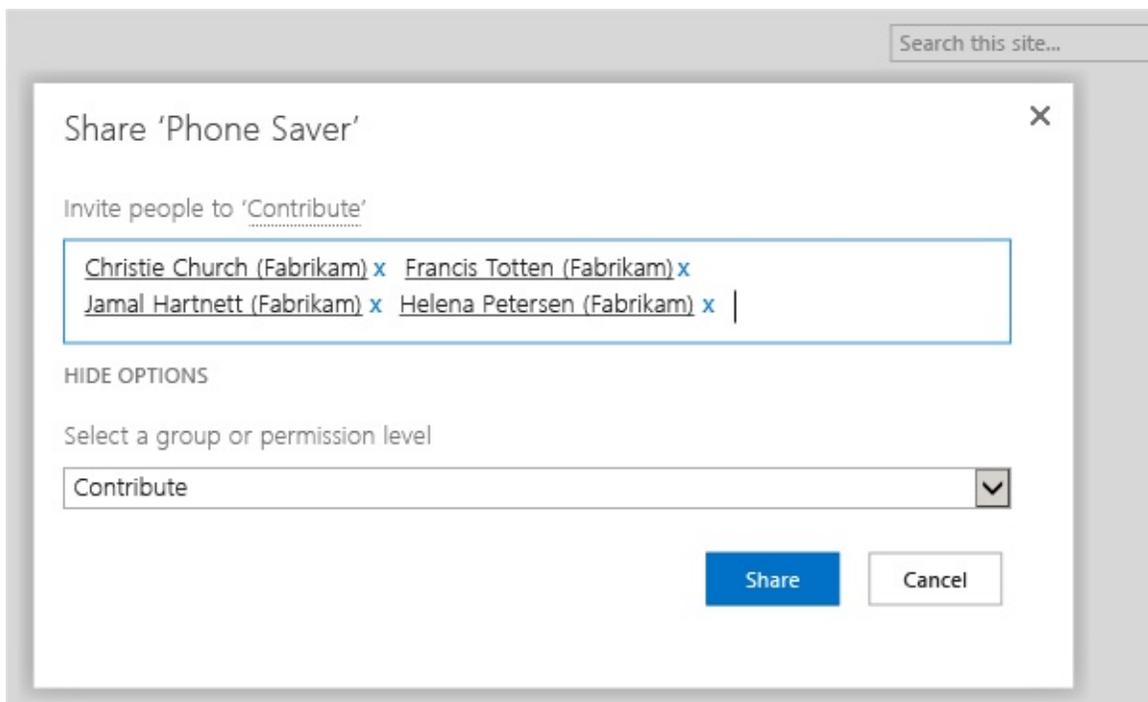
4. Depending on the user, you might want to customize their permissions for

other functionality in the project, such as [areas and iterations](#) or [shared queries](#). You can also control access to projects, version control, build, and work items; learn how in [Restrict access to functions and tasks](#).

If your TFS deployment is integrated with SharePoint Foundation, you'll need to manage membership in the SharePoint groups to grant permissions to the team project portal for your TFS users.

Add users to SharePoint Foundation

1. Open your project portal. (If you're not sure what the URL is, open Team Explorer, choose Settings, and then choose Portal Settings. The URL for the portal is listed.)
2. Choose Share, and add users or user groups to the appropriate SharePoint groups.



- To add users who will require minimal access to the project, choose Readers.
- To add users who will contribute fully to this project, choose Contributors.

- To add users who will act as project leads, choose Full Control.

For more information about users and groups in SharePoint Products,

[go here](#).

If your TFS deployment is integrated with SQL Server Reporting Services, you'll need to manage users in the appropriate SQL Server Reporting Services groups, or they won't be able to view or edit those reports.

Add users to SQL Server Reporting Services

1. Open Internet Explorer or another browser compatible with Reporting Services administration, and navigate to the following address, where ReportServer represents the name of the server that is running SQL Server Reporting Services:

`http://ReportServer/Reports/Pages/Folder.aspx`

2. On the Home page, choose Folder Settings, and in Security, choose New Role Assignment and add users.
 - To add users who can act as readers of or contributors to the project, select the Browsers check box.

Home

 SQL Server Reporting Services
New Role Assignment

Use this page to define role-based security for TfsReports.

Group or user name:

Select one or more roles to assign to the group or user.

<input type="checkbox"/> Role ↓	Description
<input checked="" type="checkbox"/> Browser	May view folders, reports and subscribe to reports.
<input type="checkbox"/> Content Manager	May manage content in the Report Server. This includes...
<input type="checkbox"/> My Reports	May publish reports and linked reports; manage folders, r...
<input type="checkbox"/> Publisher	May publish reports and linked reports to the Report Serv...
<input type="checkbox"/> Report Builder	May view report definitions.
<input type="checkbox"/> Team Foundation Content Manager	May manage Team Foundation Server related content in th...

- To add users who will act as project leads, select the Team Foundation Content Manager check box.

Home Home ...

 SQL Server Reporting Services
New Role Assignment

Use this page to define role-based security for Home.

Group or user name:

Select one or more roles to assign to the group or user.

<input type="checkbox"/> Role ↓	Description
<input type="checkbox"/> Browser	May view folders, reports and subscribe to reports.
<input type="checkbox"/> Content Manager	May manage content in the Report Server. This includes...
<input type="checkbox"/> My Reports	May publish reports and linked reports; manage folders, r... folder.
<input type="checkbox"/> Publisher	May publish reports and linked reports to the Report Serv...
<input type="checkbox"/> Report Builder	May view report definitions.
<input checked="" type="checkbox"/> Team Foundation Content Manager	May manage Team Foundation Server related content in th... and resources.

If you're a member of Team Foundation Administrators, you can verify what features are available for your users by default, and see whether any users are members of groups that have access outside of the default level.

Verify features available for a user or group of users

1. Open Team Web Access in administration mode, and choose Control Panel to navigate to the top-level administration context.
2. Choose the Access levels tab.
3. Choose the name of each license group in turn, and review the following information:
 - Which licensing group is set as the default group for the deployment. That group name will be followed by (Default). This is the group that all users of your deployment of Team Foundation Server will be assigned to by default.
 - Whether the user for whom you are determining licensing levels is a member of a different licensing group than the default group. If so, review the description of the features of that licensing group to better understand what features are and are not available to that user.
4. To review the licensing group membership of all users in the deployment at once, choose Export Audit Log. The membership information will be exported to a comma-delimited file. Save or open the file.

Q: Where can I learn more about permissions and built-in groups, like the Contributors group?

A: The following picture shows the default permission assignments made to the Contributors group.

Code	Work	Build	Test
Object and project-level:	Area-node level	Object-level:	Area-node level
TFVC: Check in Check out Label Lock Merge Read Git repositories Administer Branch creation Contribute Note management Read Rewrite and destroy history Tag creation	Edit work View permissions View work items Shared queries Read Project-level View project-level information	Edit build quality Queue builds View build definition View builds Edit build quality Queue builds View build definition View builds	Manage test plans Manage test suites Create test runs Delete test runs Manage test configurations Manage test environments View test runs

To learn more about other groups and their permission assignments, see

[Permission reference for Team Foundation Server.](#)

Q: How do I add someone as an administrator?

A: You'll need to add that user to groups at the project-collection level or at the server-level.

Q: Can I restrict access to a team project or to functional areas within a team project?

A: Yes. Learn how in [Restrict access to functions and tasks.](#)

Q: Can I set up default groups to use for licensing and managing users in team projects?

A: Yes, and it's a good idea to do so. See [Set up groups for use in TFS deployments.](#) If you want to define them prior to creating team projects, see [Configure initial groups, teams, members, and permissions.](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Change access levels

[Send Feedback](#)

Access levels allow groups of users to access features in Team Web Access (TWA) based on their license level when working in an on-premises deployment of Team Foundation Server (TFS). Certain features, such as authoring charts, test case management and team rooms, require Advanced access.

You only set access levels for on-premises TFS. For Visual Studio Online, [account licenses control access to premium features](#).

As an administrator, you assign users or groups of users to one of the three levels of access-Basic, Advanced, and Stakeholder-based on the license that each user has. With the release of [TFS 2013 Update 4](#), access level names have been renamed to align with Visual Studio Online license levels. Also, it's important to note that service accounts are added to the default access level. If you set Stakeholder as the default access level, you must add the TFS service accounts to the Basic or Advanced group.

Access level	License required
Basic (previously Standard)	TFS client-access license (CAL) or Visual Studio Professional with MSDN subscription
	One of these MSDN subscriptions: Visual

Advanced (previously Full) Studio Ultimate with MSDN, Visual Studio Premium with MSDN, MSDN Platforms, or Visual Studio Test Professional with MSDN.

Stakeholder (previously Limited) No license required. Assign Stakeholder access to customers or stakeholders that you want to collaborate with but who aren't on your team.

You can find out more about licensing from the [Visual Studio and MSDN Licensing White Paper](#).

Add a user or group to an access level

If you're managing access for a large group of users, a best practice is to first create either a Windows group or TFS group and add individual accounts to those groups.

1. From the TFS home page (for example, <http://myserver:8080/tfs>), go to the server administration page.



2. Select the level and then add the user or group.

Control Panel

Control panel | **Access levels** | Extensions

Export audit log
Stakeholder
Basic (default)
Advanced

Access levels

Name	
Advanced	
Features	View My Work Items Standard Features Agile boards Basic backlog and sprint planning tools Request and Manage Feedback Test case management Team rooms Agile Portfolio Management Chart Viewing Chart Authoring Code Build Administer account Advanced home page Advanced backlog and sprint planning tools Advanced portfolio management

Access levels only apply to features in Team Web Access. For more, informat...

Set as default access level | **Add...** | Refresh | Search

Display Name | Username or Scope

No identities found in current scope.

If you don't see the Access levels tab, you aren't a TFS administrator and don't have permission. Here's how to get permission.

Quick reference to access levels and features

The following table indicates which features users can access based on their access level. This information depends on your product version and is subject to change. These access levels reflect what is supported with TFS 2013.4 update. For a full comparison of products and features, go

[here](#).

Feature areas	Stakeholder	Basic	Advanced
View and edit all work items	✓(1)	✓	✓
Standard Features (2)	✓	✓	✓
Agile boards	✓(3)	✓	✓
Basic backlog and sprint planning tools	✓(4)	✓	✓
Agile Portfolio Management	✓(3)	✓(3)	✓

Chart Viewing	✓	✓	✓
Chart Authoring			✓
Code		✓	✓
Build		✓	✓
Request and manage feedback			✓
Test case management			✓
Team rooms			✓
Administer account		✓	✓
Advanced home page		✓	✓
Advanced backlog and sprint planning		✓	✓

tools

Advanced
portfolio
management



Notes:

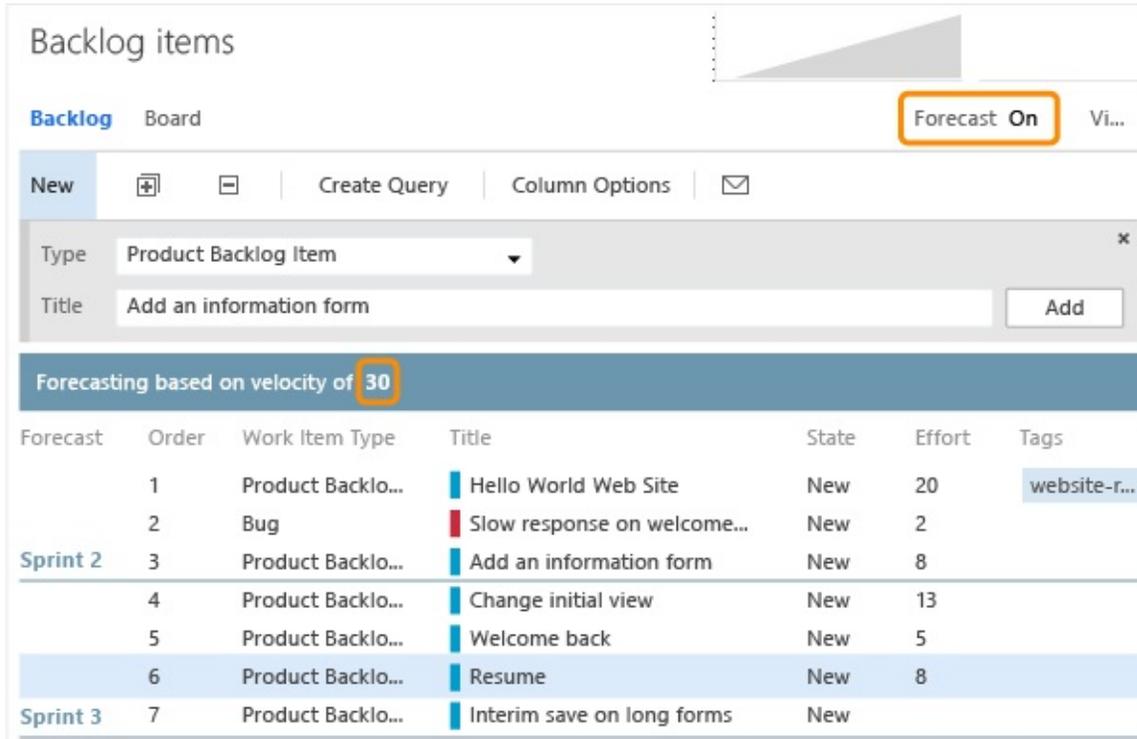
1. With Stakeholder access, users can create and modify all work items, and can create and save queries on all work items under their My Queries folder. (This is a change from Limited access in which users could create and modify only those work items that they created and query and view work items they created.)

Also, stakeholders can create and modify work items from Team Foundation clients such as Microsoft Excel, Microsoft Project, and Microsoft Feedback client.

2. Standard features include access to the Home and Work hubs.
3. Can view backlog pages and Kanban boards. Can add work items through the quick add panel, which are appear at the bottom of the list. Can't move items on the page or use other features.
4. Can view sprint pages and task boards. Can add work items, but can't use other sprint planning tool features.

Basic access

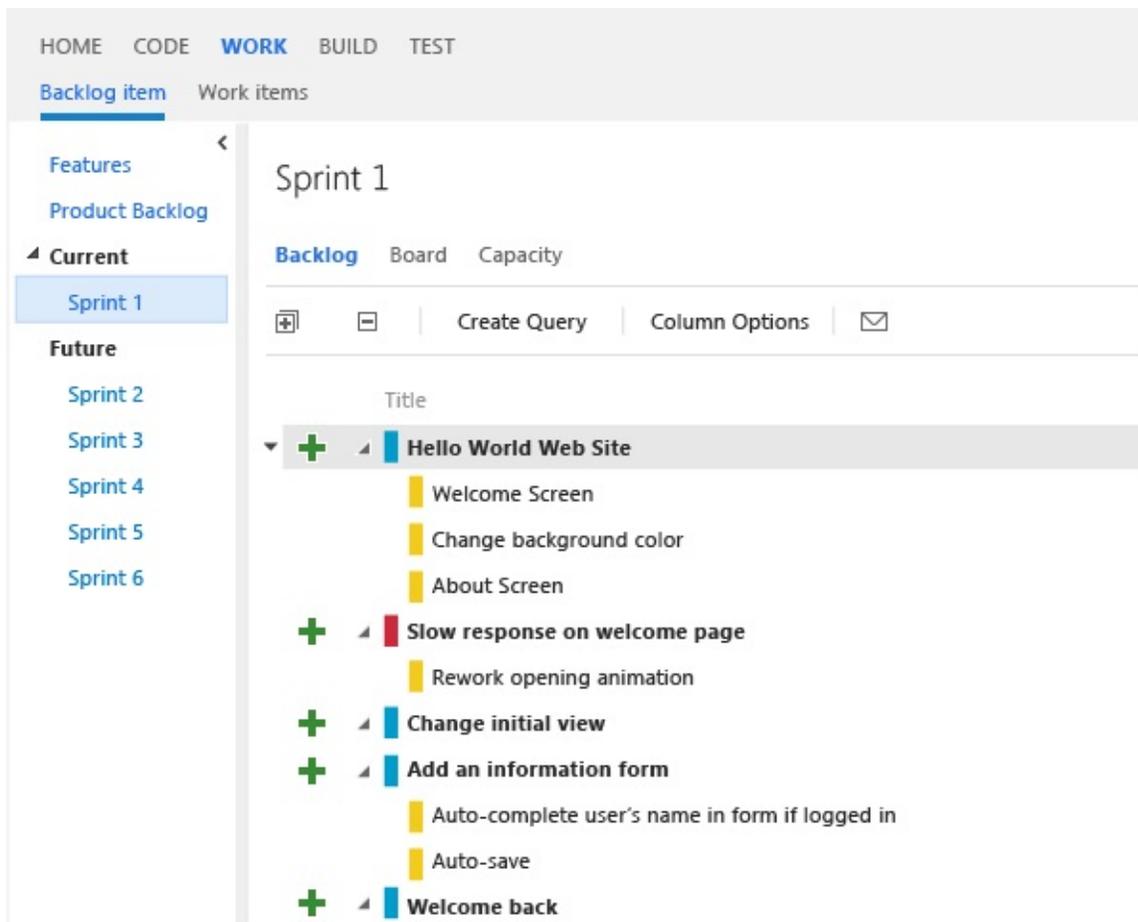
With Basic access, you can manage work in a product backlog...



The screenshot shows the 'Backlog items' view in Jira. At the top right, there is a 'Forecast On' button highlighted with an orange box. Below the header, there is a 'New' button and a search bar containing 'Add an information form'. A dropdown menu shows 'Product Backlog Item' as the selected type. Below this, a blue bar indicates 'Forecasting based on velocity of 30', with the number '30' highlighted in an orange box. The main area contains a table of backlog items:

Forecast	Order	Work Item Type	Title	State	Effort	Tags
	1	Product Backlo...	Hello World Web Site	New	20	website-r...
	2	Bug	Slow response on welcome...	New	2	
Sprint 2	3	Product Backlo...	Add an information form	New	8	
	4	Product Backlo...	Change initial view	New	13	
	5	Product Backlo...	Welcome back	New	5	
	6	Product Backlo...	Resume	New	8	
Sprint 3	7	Product Backlo...	Interim save on long forms	New		

...configure sprints with their own backlogs and task boards...



...and view work in progress on the Kanban board.

Backlog items

Backlog **Board**

Customize Columns

New	Approved	0/5	Committed	0/5	Done
<p>Hello World Web Site</p> <p>20</p>					
<p>Slow response on welcome page</p> <p>2</p>					
<p>Add an information form</p>					

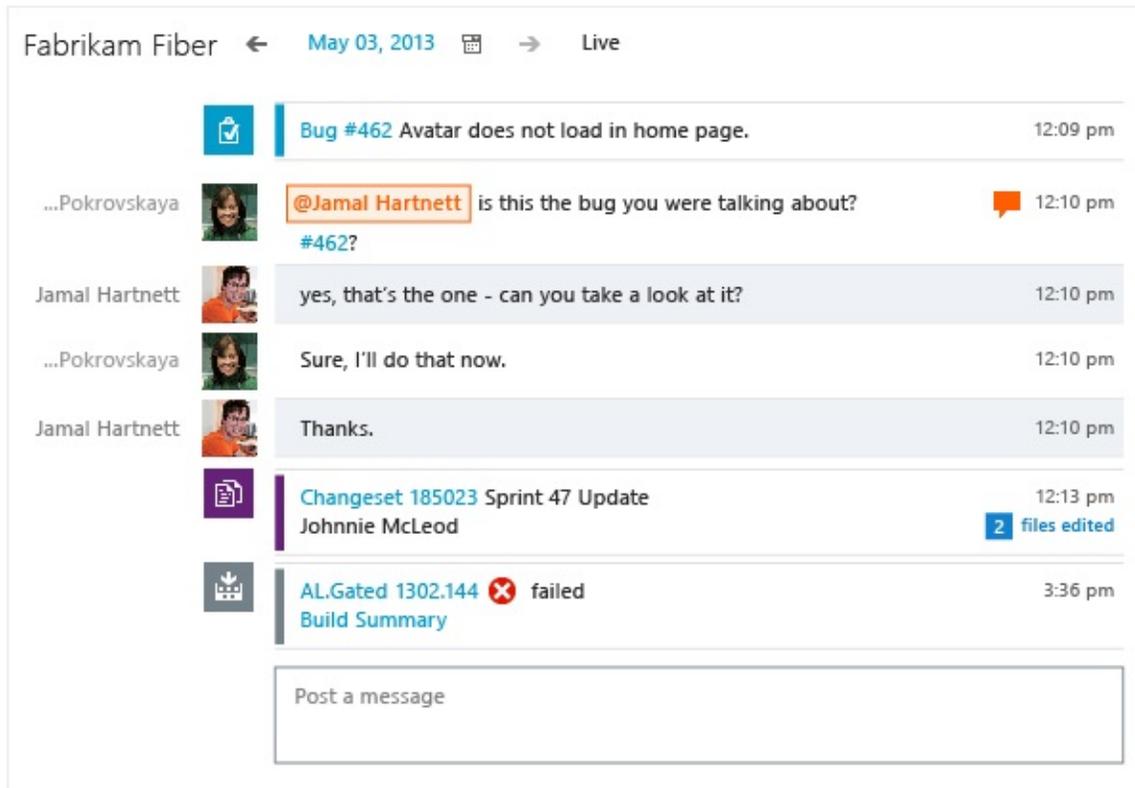
Advanced access

Advanced access includes access to everything included with Basic access, plus additional features.

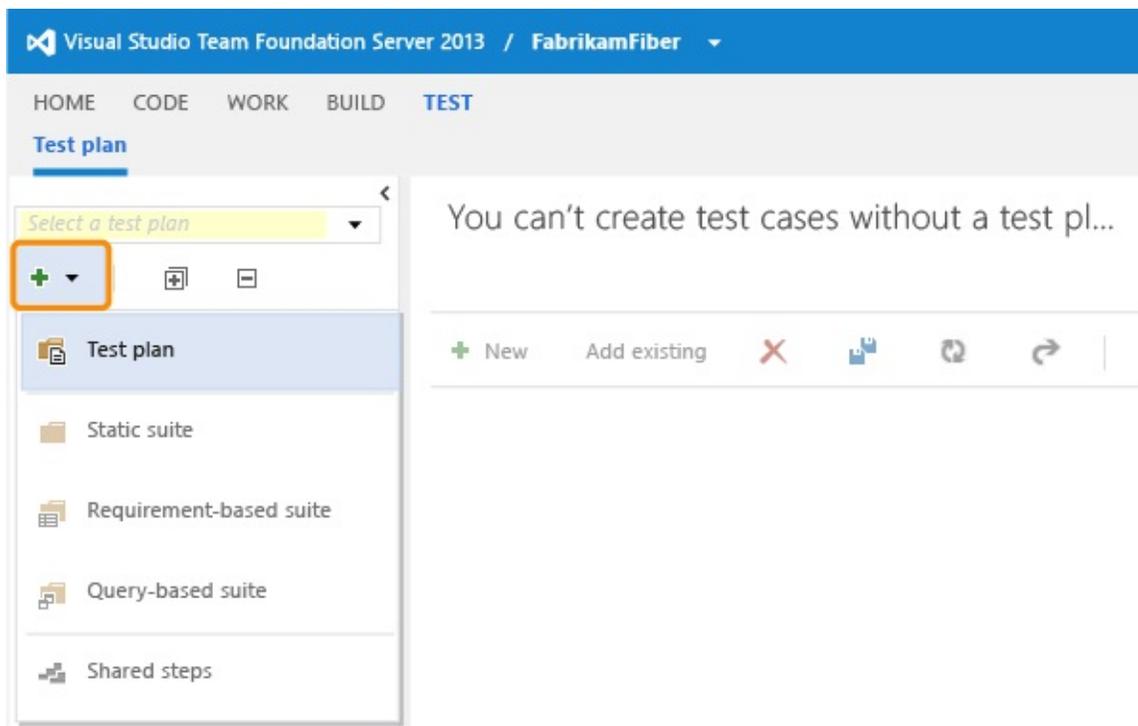
For example, with Advanced access, you can work with portfolio backlogs.

Work Item Type	Title	State
Feature	Mitigate impact fo low-coverage areas	In Progress
Product Backlo...	Coverage map on website	Committed
Task	Update map code to change colors based on coverage stren...	In Progress
Task	Auto-detect code to center map on current location	In Progress
Task	Update code to toggle between kilometers and miles	To Do
Bug	Autolocation data persisting in stale state between sessions	Committed
Product Backlo...	Performance boost in low-bandwidth mode	New
Product Backlo...	Data cache improvements	New
Feature	Utilize customer data in targeting expansion areas	New
Product Backlo...	Self-report coverage dead zones	Approved
Feature	Promote customer participation with Most Valued Customer	New
Feature	App of the Month Club	New
Feature	Increase developer excitement with Developer Incentive Program	New

You also have access to team rooms.



And you can use web-based test case management tools.



You can author charts to help your team visualize progress.

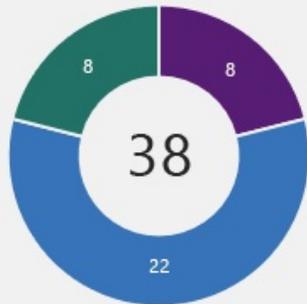
Active Bugs

Results Editor **Charts**

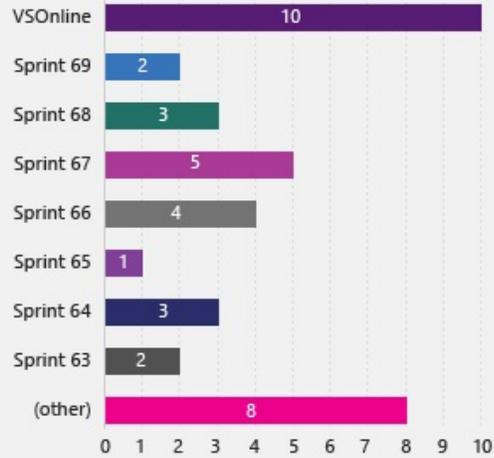
+ New chart

Active Bugs by Priority - Chart

1 2 3

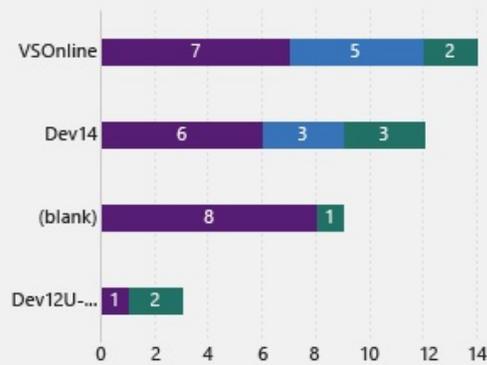


Active Bugs by Iteration - Chart



Active Bugs Stacked Bar by Release

2 3 1

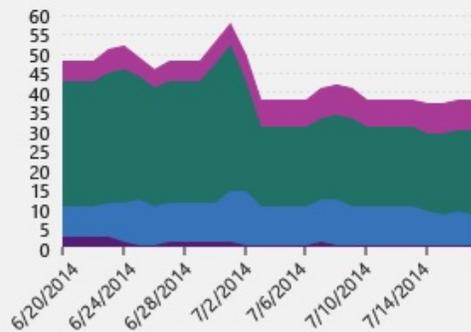


Active Bugs Pivot Table by Iteration

	VSONline	Sprint 67	Sprint 66	OneVS	(other)	Total
2	7	2	72	3	8	22
3	1	1	1	1	4	8
1	2	2	1	3	3	8
Total	10	5	4	4	15	38

Active Bugs Trend by Priority

0 1 2 3



Active Bugs Trend by Area

Modern WIT WIT Adapter



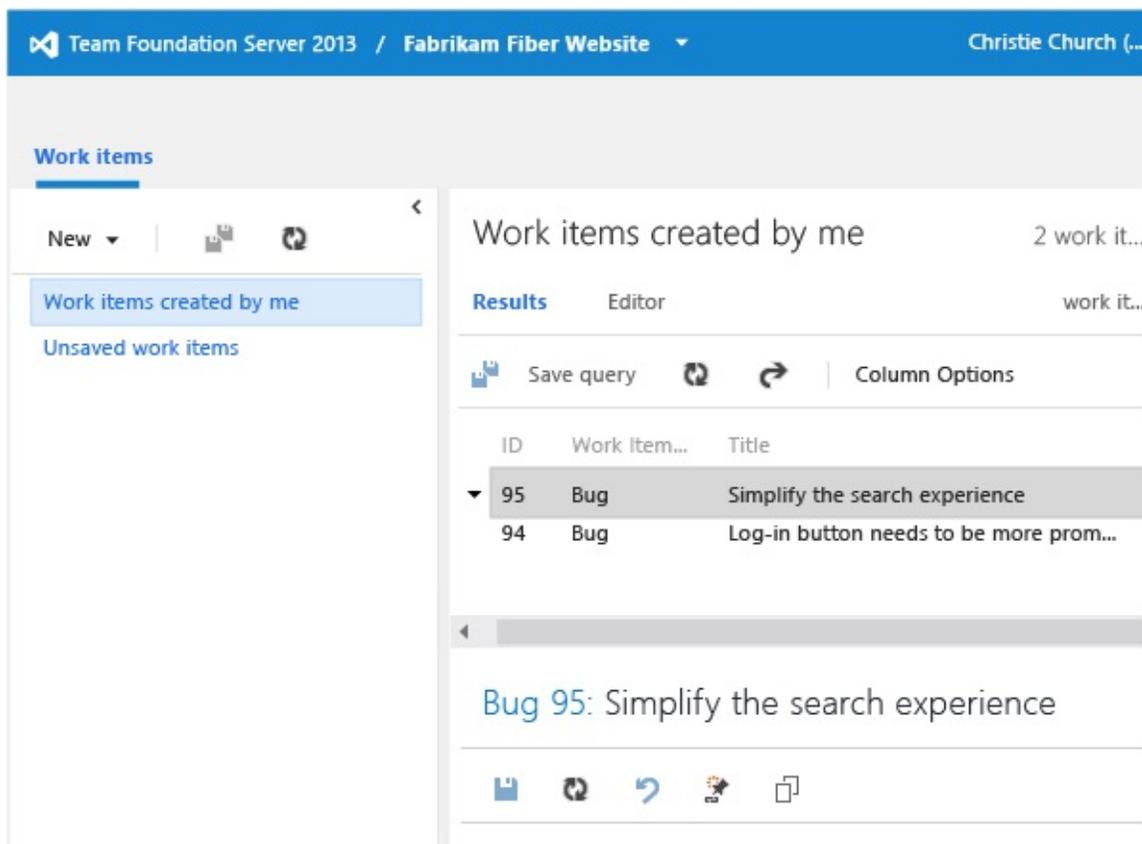
You can also request and manage feedback from customers.

Work	Create new ▾	Visual Studio
Backlog Taskboard Queries		 Open in Visual Studio Requires Visual Studio 2013  Get Visual Studio See Visual Studio downloads
Team rooms		Other links
 Agile Room 0 users in room		Request feedback View project portal View process guidance View reports Configure schedule and iterations... Configure work areas...

Stakeholder access

Users who have Stakeholder access can access team home pages, backlogs, and portfolio backlogs. They can view, create, and modify work items such as stories, features, bugs, and feedback responses. They can also view, create, and save queries and receive alerts when changes are made to work items.

Here's what Stakeholder access looks like.



To learn more, see [Work as a stakeholder](#).

To add a group of users to Stakeholder access

1. Create either a Windows group or TFS group.
2. Add the user accounts to the group you just created.

3. Add the group to Stakeholder access.

The screenshot shows the 'Control Panel' interface for TFS. The 'Access levels' tab is selected. The 'Stakeholder' access level is highlighted in the left sidebar. The main content area shows the 'Stakeholder' access level configuration, including its name and a list of features: 'View My Work Items', 'Standard Features', 'Agile boards', 'Basic backlog and sprint planning tools', and 'Agile Portfolio Management'. A blue banner below the features states: 'Access levels only apply to features in Team Web Access. For more information, see MSDN.' At the bottom, there is a section for adding identities. The 'Add' button is highlighted, and a dropdown menu is open, showing two options: 'Add Windows user or group' (highlighted) and 'Add TFS group'. The text 'No identities found in curr...' is visible below the dropdown.

Permissions and access levels

Of course, none of these levels of access expose information that you don't have permission to view. Make sure your users have both the permissions and the access levels they need. If they're members of the team, then they probably have the right permissions to use Basic and Advanced access.

- Do you want to add another team member? Add team members
- Do you need to grant permissions to people who aren't on the team? Give reviewers permissions to provide feedback
- Do you need to restrict access to certain features based on permissions?

[Restrict access to functions and tasks](#)

Q & A

Q: Can I customize the Agile planning tools pages?

A: Yes. See [Customize the Agile experience for a team project](#).

Q: What features are available to a user who belongs to two different groups?

A: If a user belongs to a group that has Basic access, and another group that has Advanced access, the user has access to all features available to Advanced access.

Q: Are service accounts assigned to an access level group?

A: By default service accounts are added to the default access level. If you make Stakeholder the default access level, you must add TFS service accounts to the Basic or Advanced group.

Q: Do service accounts require a license?

A: No. Service accounts don't require a separate license.

Q: Where can I learn more about licensing and MSDN subscriptions?

A: You can find out more about licensing [here](#).

Q: Where can I learn about the new features in TFS?

A: Learn all about what's new in TFS, Visual Studio, and ALM [here](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Restrict access to functions and tasks

[Send Feedback](#)

You can restrict access to many Team Foundation Server (TFS) tasks by setting the permission state to Deny through a security group. For a comprehensive list of default groups and permissions, see [Permission reference for Team Foundation Server](#).

Q: How do I restrict who can access or modify source code?

A: From the Version Control tab in the TWA administration context, you can set permissions for a group or individual.

The screenshot displays the security configuration for a repository. On the left, a list of user groups is shown, with 'Readers' highlighted. On the right, an 'ACCESS CONTROL SUMMARY' table lists permissions and their current status.

ACCESS CONTROL SUMMARY	
Shows information about the permissions being granted to this identity	
Administer labels	Not set
Check in	Not set
Check in other users' changes	Not set
Check out	Not set
Label	Not set
Lock	Not set
Manage branch	Not set
Manage permissions	Not set
Merge	Not set
Read	Allow
Revise other users' changes	Not set
Undo other users' changes	Not set
Unlock other users' changes	Not set

Buttons at the bottom: [Clear explicit permissions](#), , ,

For team projects that use Git for version control, you can set the following permissions.

Control Panel > DefaultCollection1 > GitTeam > GitTeam Team

Overview Security Alerts Version Control

Project-level security for GitTeam

Add... ▾

Search...

- Project Collection Administr...
- Project Collection Build Serv...
- Project Collection Service A...
- Build Administrators
- Contributors
- Project Administrators
- Readers**

ACCESS CONTROL SUMMARY

Shows information about the permissions being granted to this identity

Administrator	Not set
Branch creation	Not set
Contribute	Not set
Note management	Not set
Read	Allow
Rewrite and destroy hist...	Not set
Tag creation	Not set

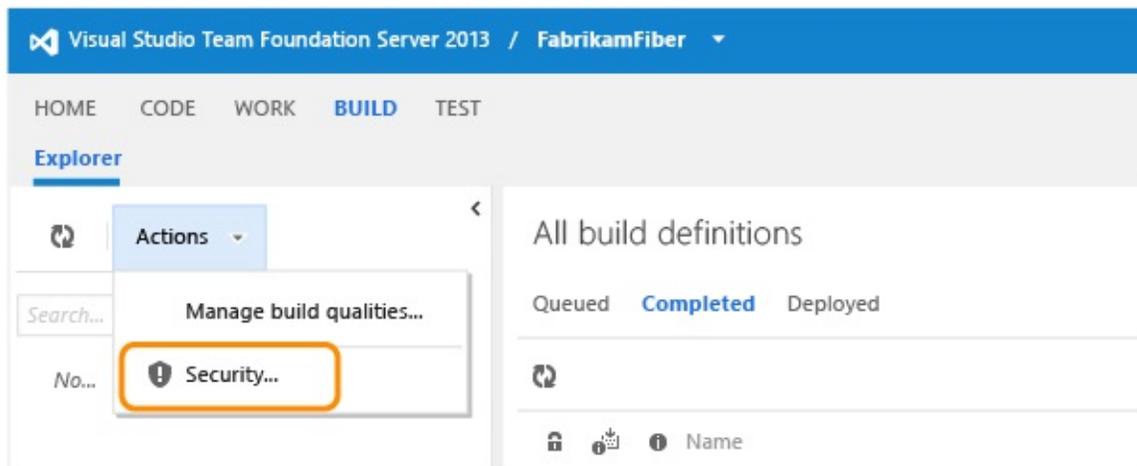
[Clear explicit permissions](#)

Remove Save Changes Undo C...

For additional information, see [Permission reference for Team Foundation Server](#).

Q: How do I restrict who can modify build definitions?

A: From the Build hub in TWA, you can set build permissions at the project level for a group or individual.



You can set permissions for the build operations shown in the following image.

\$/FabrikamFiber security

Add... ▾ Inheritance ▾

Search...

-  Project Collection Administrators
-  Project Collection Build Service Accounts
-  Project Collection Service Accounts
-  Build Administrators
-  Contributors
-  Project Administrators
-  Readers

ACCESS CONTROL SUMMARY
Shows information about the permissions being granted to this identity

Administer labels	Not set
Check in	Not set
Check in other users' changes	Not set
Check out	Not set
Label	Not set
Lock	Not set
Manage branch	Not set
Manage permissions	Not set
Merge	Not set
Read	Allow
Revise other users' changes	Not set
Undo other users' changes	Not set
Unlock other users' changes	Not set

[Clear explicit permissions](#)

Remove Save Changes Undo Changes

Also, you can set permissions by opening the ▾ context menu for a build definition.

For additional information, see [Permission reference for Team Foundation Server](#).

Q: How do I restrict access to team members changing a work item?

A: By setting permissions on an area path, you can deny a group or individual the ability to create or edit work items assigned under an area path.

Q: How do I restrict access to people creating specific work item types (WITs)?

A: You can restrict access in one of two ways:

- By adding WITs to the Hidden Categories group, you can prevent the majority of project contributors from creating them. You can create a hyperlink to a template that opens the work item form and share that link with those team members who you do want to create them.
- By adding a field rule to the workflow for the System.CreatedBy field, you can effectively restrict a group of users from creating a work item of a specific type. As the following example shows, the user who creates the work item must belong to the Allowed Group in order to save the work item.

Copy Code

```
<TRANSITION from=" " to="New">
  <FIELDS>
    <FIELD refname="System.CreatedBy">
      <VALIDUSER for="Allowed Group" not="Disallowed Group" .
    </FIELD>
  </FIELDS>
</TRANSITION>
```

For more information about how to customize WITs, see [Modify or add a custom work item type \(WIT\)](#).

Q: How do I restrict access to changing a work item based on certain conditions or field values?

A: Set a condition field rule, a condition-based field rule or a combination of the two that applies to a group. You can restrict changes from being made to a field by specifying a qualifying rule and making it apply for a specific group. Conditional rules can include CANNOTLOSEVALUE, EMPTY, FROZEN, NOTSAMEAS, READONLY, and REQUIRED elements.

Q: How do I restrict who can modify a shared work item query?

A: Set permissions on a shared query or query folder to restrict who can modify the query or queries within the folder.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

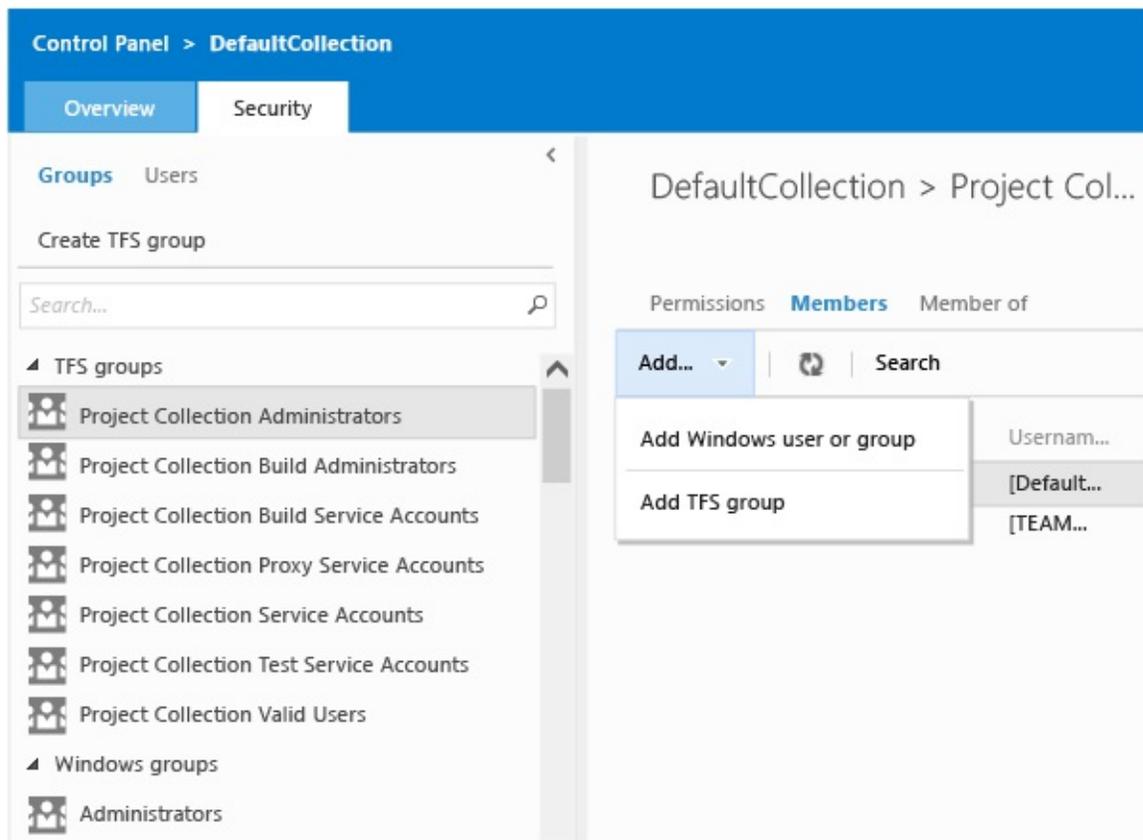
Set administrator permissions for team project collections

[Send Feedback](#)

In TFS, each team project collection is its own grouping of projects that can share reports, work items, and other items, all stored in a single database. Project collection administrators maintain the collection and administer permissions and security for other roles at the collection level.

Add a collection administrator in Team Foundation Server

1. Open Team Web Access and switch to administration mode by choosing the gear icon ⚙️.
2. Navigate to security at the collection level, and add a member to Project Collection Administrators.

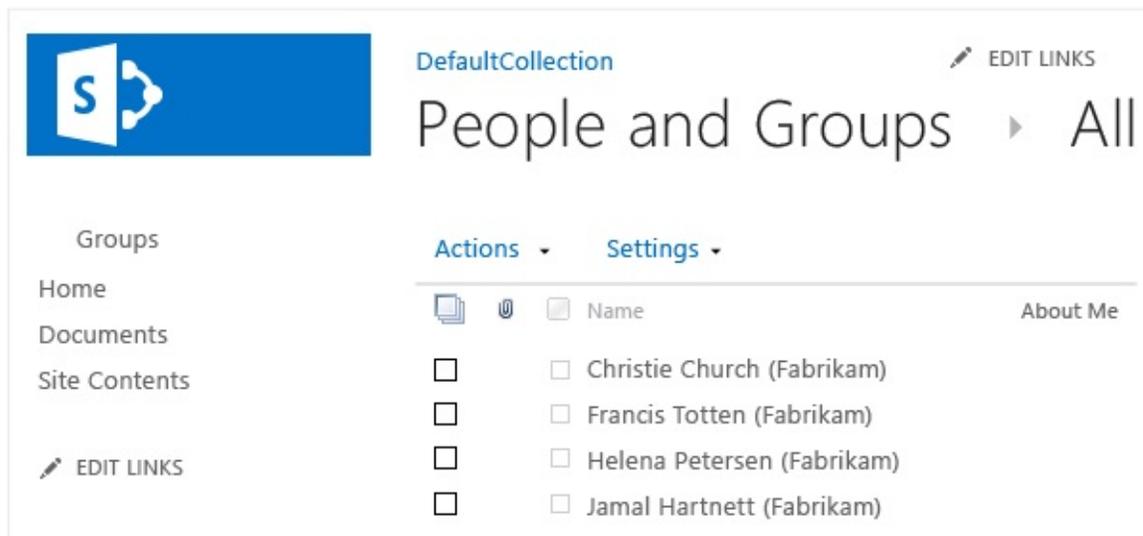


Add a user as a site collection admin in SharePoint Foundation

If your deployment is integrated with SharePoint Products, add team project collection administrators to the site collection administrators group in SharePoint Products. Skip this procedure if your deployment does not integrate with SharePoint.

1. Open SharePoint Central Administration.
2. Grant permissions that are appropriate for this user at the farm or the Web application level, depending on your security needs.

For optimum interoperability, consider adding users of the Project Collection Administrators group to the Site Collection Administrators group in SharePoint Products.



The screenshot shows the 'People and Groups' page in SharePoint. The page title is 'DefaultCollection' and the main heading is 'People and Groups' with a sub-heading 'All'. There are 'Actions' and 'Settings' dropdown menus. A table lists four users: Christie Church (Fabrikam), Francis Totten (Fabrikam), Helena Petersen (Fabrikam), and Jamal Hartnett (Fabrikam). Each user has a checkbox and an 'About Me' link.

<input type="checkbox"/>	<input type="checkbox"/>	Name	About Me
<input type="checkbox"/>	<input type="checkbox"/>	Christie Church (Fabrikam)	
<input type="checkbox"/>	<input type="checkbox"/>	Francis Totten (Fabrikam)	
<input type="checkbox"/>	<input type="checkbox"/>	Helena Petersen (Fabrikam)	
<input type="checkbox"/>	<input type="checkbox"/>	Jamal Hartnett (Fabrikam)	

■ Add users in Reporting Services

If your deployment is integrated with a report server, add team project collection administrators to the Team Foundation Content Manager group in SQL Server Reporting Services. Skip this procedure if your deployment does not integrate with a report server.

1. Open Internet Explorer running as an administrator.
2. In the Address bar, specify the following URL, where ReportServer is the name of the server that is running Reporting Services:
http://ReportServer/Reports/Pages/Folder.aspx

Important

If you are using a named instance, you must include its name in the path of the reports. You use the following syntax, where ReportServer is the name of the report server for Team Foundation and InstanceName is the name of the instance of SQL Server:

http://ReportServer/Reports_InstanceName/Pages/Folder.aspx

3. On the Home page, choose Folder Settings, and add the user by granting them the Team Foundation Content Manager role as a new role assignment.



SQL Server Reporting Services

New Role Assignment

Use this page to define role-based security for Home.

Group or user name:

Select one or more roles to assign to the group or user.

<input type="checkbox"/> Role ↓	Description
<input type="checkbox"/> Browser	May view folders, reports and subscribe to reports.
<input type="checkbox"/> Content Manager	May manage content in the Report Server. This includes...
<input type="checkbox"/> My Reports	May publish reports and linked reports; manage folders, r... folder.
<input type="checkbox"/> Publisher	May publish reports and linked reports to the Report Serv...
<input type="checkbox"/> Report Builder	May view report definitions.
<input checked="" type="checkbox"/> Team Foundation Content Manager	May manage Team Foundation Server related content in th... and resources.

Q & A

Q: When do I need to add someone to the team project collection administrator role in TFS?

A: It varies. For most organizations that use TFS, project collection administrators manage the collections that members of the Team Foundation Administrators group create, but members of the Project Collection Administrators group do not create the collections themselves. Project collection administrators also perform any operations that are required to maintain the collection, such as creating team projects, adding users to groups, or modifying the settings for the collection.

Q: What are the optimal permissions needed to administer a team project collection across all its components and dependencies?

A: A team project collection administrator for TFS must be a member of the following groups or have the following permissions:

- Team Foundation Server: Project Collection Administrators or the appropriate collection-level permissions set to Allow.
- SharePoint Products: If the collection is configured with a site collection resource, Site Collection Administrators.
- Reporting Services: If the collection is configured with reporting resources, Team Foundation Content Manager

Q: I'm an admin, but I don't seem to have all the permissions I need to add a team project collection administrator. What might I need?

A: These are the required permissions:

- In TFS, you must belong to the Project Collection Administrators group, or your View Server-Level Information and Edit Server-Level Information permissions must be set to Allow.
- To add permissions for SharePoint Products, you must be a member of the Site Collection Administrators group or the Farm Administrators group for SharePoint Products.
- To add permissions for Reporting Services, you must be a member of the Content Managers group or the Team Foundation Content Managers group for Reporting Services.

⚠Important

To perform administrative tasks such as creating team project collections, your user account requires administrative permissions, and the service account that the Team Foundation Background Job Agent uses also must have certain permissions granted to it. For more information, see

[Service accounts and dependencies in Team Foundation Server](#) and [Team Foundation Background Job Agent](#).

≡Q: Where can I find information about each individual permission?

A: You can find detailed information about individual permissions and their relationship to default groups in TFS in the [Permission reference for Team Foundation Server](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Set administrator permissions for Team Foundation Server

[Send Feedback](#)

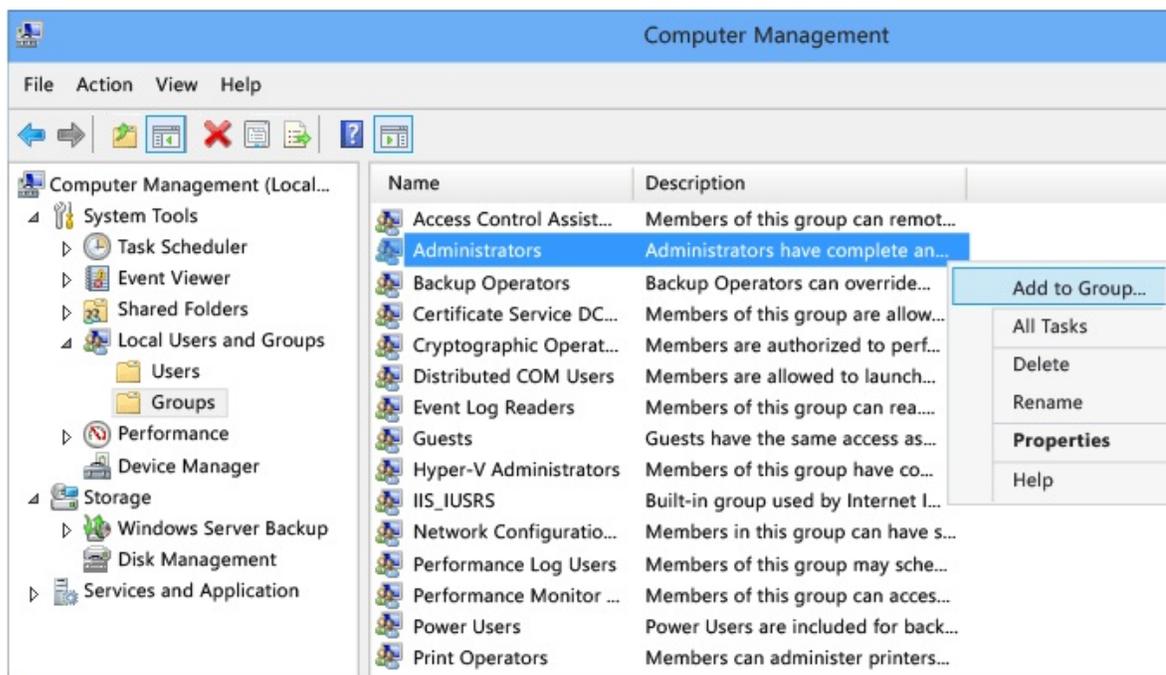
To perform system maintenance, schedule backups, add functionality, and do other tasks, administrators in Visual Studio Team Foundation Server (TFS) must be able to configure and control all aspects of TFS. That's why TFS administrators require administrative permissions in the software programs that TFS interoperates with.

You can quickly grant these permissions to administrators by adding them to the Team Foundation Administrators group in Team Foundation Server (TFS). However, you should only grant this level of permission to the minimum number of users needed to maintain TFS.

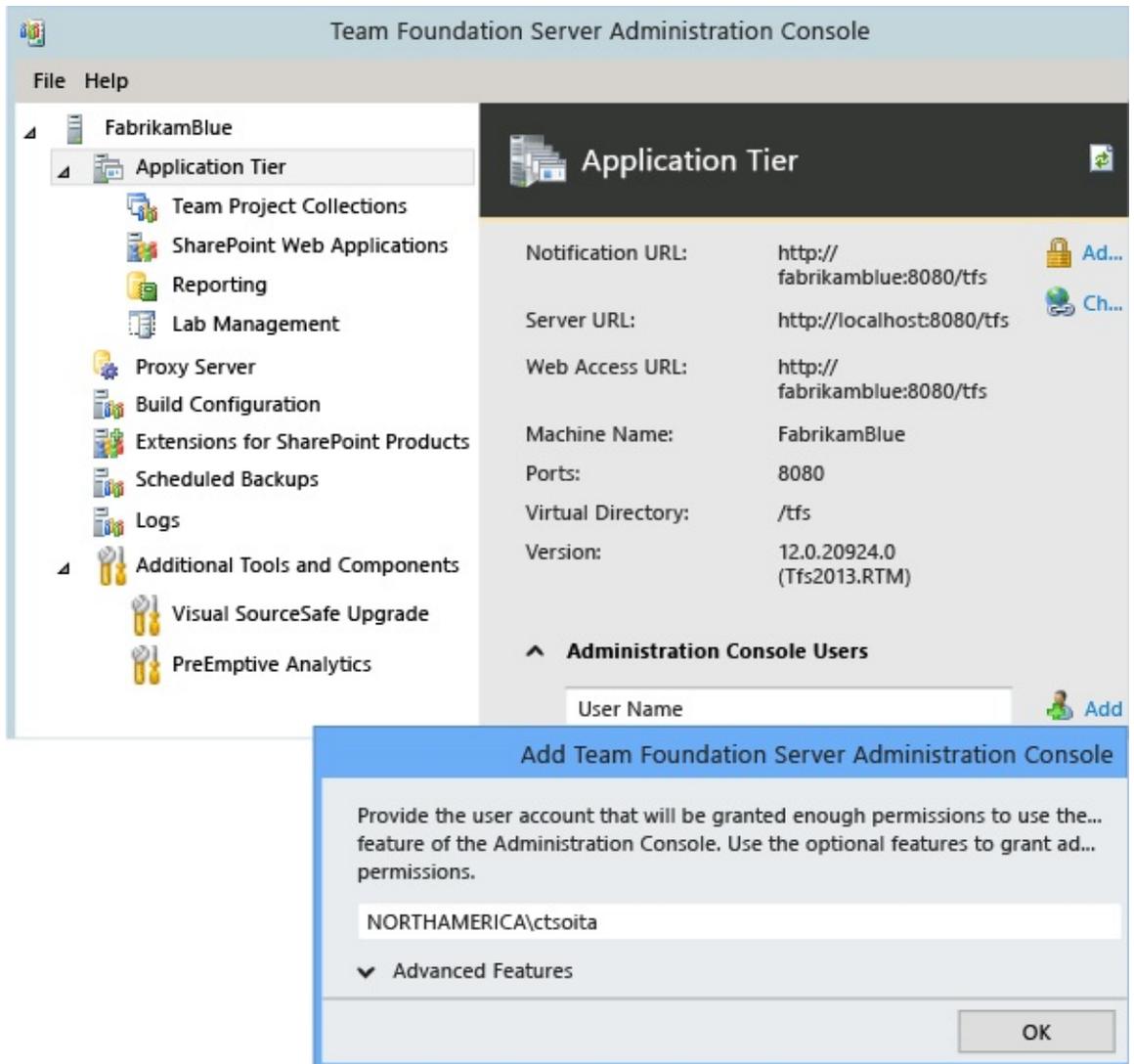
☐

Add users to the TFS local Administrators group and as an administrator for the Team Foundation Server Administration console

1. On the application-tier server, add the user to the local Administrators group.



2. Open the administration console and add a console user.



3. Review the progress to make sure that the user account is added to all aspects of the deployment, including SharePoint and reporting resources.

Add Administration Console User

 Successful completion (0 warnings).

To view the detailed log, [click here](#).

Status:

Adding account to collection databases ...
Changing collection: DefaultCollection
Added NORTHAMERICA\ctsoita to Tfs_DefaultCollection (FABRIKAM2008TFS)

Adding account to system databases ...
Adding account to master database ...
Added NORTHAMERICA\ctsoita to master

Adding to local groups ...
Adding NORTHAMERICA\ctsoita to IIS_IUSRS
Adding NORTHAMERICA\ctsoita to TFS_APPTIER_SERVICE_WPG

Adding account to Analysis Services database ...
Done.

Adding account to relational warehouse database ...
Added NORTHAMERICA\ctsoita to Tfs_Warehouse

Adding account to Reporting Services content manager role
http://fabrikam2008tfs/ReportServer
Done.

Adding account to SharePoint farm administrators
http://fabrikam2008tfs:17012/
The following service account was added as a member of the Farm Administrators group in SharePoint Prod...
Done.

Adding account to Team Foundation Administrators group...
Done.

Adding account to configuration database ...
Added NORTHAMERICA\ctsoita to Tfs_Configuration
VIEW ANY DEFINITION permission is granted to NORTHAMERICA\ctsoita login on FABRIKAM2008TFS SQL S...
ALTER ANY LOGIN permission is granted to NORTHAMERICA\ctsoita on FABRIKAM2008TFS SQL Server inst...

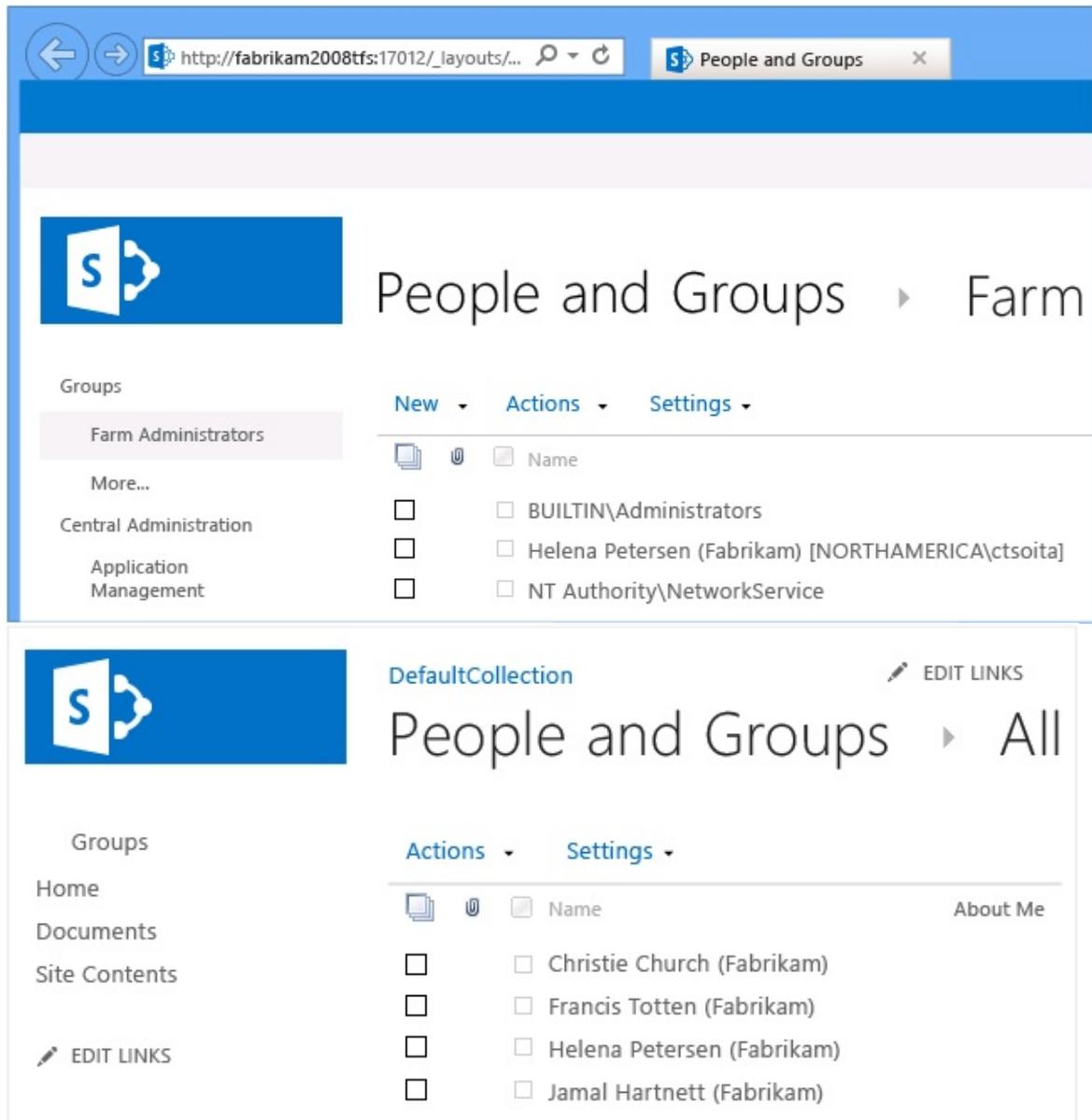
If you're running a standard single-server deployment, or a multi-server deployment without SharePoint or reporting, that's it! However, if you have multiple application tiers, you'll need to repeat these two steps on every application tier server. And if you have SharePoint or reporting on other servers, you might need to manually add administrative users to those products separately.

Grant administrative permissions in SharePoint Foundation

1. On the server that is running SharePoint Products, open SharePoint Central Administration.
2. Grant permissions that are appropriate for this user at the farm or the Web application level, depending on your security needs.

For optimum interoperability, consider adding users of the Team Foundation Administrators group to the following groups in SharePoint Products:

- Farm Administrators
- Site Collection Administrators group for all site collections that the deployment of Team Foundation Server uses



Grant administrative permissions in Reporting Services

1. Start Internet Explorer.
2. In the Address bar, specify the following URL, where ReportServer is the

name of the server that is running Reporting Services:
http://ReportServer/Reports/Pages/Folder.aspx

⚠ Important

If you are using a named instance, you must include its name in the path of the reports. You use the following syntax, where ReportServer is the name of the report server for Team Foundation and InstanceName is the name of the instance of SQL Server:

http://ReportServer/Reports_InstanceName/Pages/Folder.aspx

3. Choose Folder Settings, and then choose New Role Assignment.
4. Add the account name of the user or group to whom you want grant administrative permissions and grant them membership in the Team Foundation Content Manager role.

Home Home...

 SQL Server Reporting Services
New Role Assignment

Use this page to define role-based security for Home.

Group or user name:

Select one or more roles to assign to the group or user.

<input type="checkbox"/> Role ↓	Description
<input type="checkbox"/> Browser	May view folders, reports and subscribe to reports.
<input type="checkbox"/> Content Manager	May manage content in the Report Server. This includes...
<input type="checkbox"/> My Reports	May publish reports and linked reports; manage folders, r... folder.
<input type="checkbox"/> Publisher	May publish reports and linked reports to the Report Serv...
<input type="checkbox"/> Report Builder	May view report definitions.
<input checked="" type="checkbox"/> Team Foundation Content Manager	May manage Team Foundation Server related content in th... and resources.

Q & A

Q: Who should I add to the administrator role in TFS?

A: Administrators maintain at least one server that is running Team Foundation Server, and they administer permissions and security for other roles at the server level and at the level of team project collections. You'll need at least one administrator for your deployment. Depending on your availability needs, you might need to add more administrators to help ensure that there is someone available to perform administrator-level tasks on short notice.

For example, you need to add someone as an administrator if that person is expected to perform one or more of the following tasks:

- Create or delete team project collections
- Back up TFS
- Change access levels for Team Web Access
- Administer the reporting warehouse
- Change SharePoint Web applications used by TFS
- View and edit server-level permissions
- Trigger alert events

Q: What are the optimal permissions needed to administer TFS across all its components and dependencies?

A: Optimally, an administrator for TFS must be a member of the following groups or have the following permissions:

- Team Foundation Server: Team Foundation Administrators or have the appropriate server-level permissions set to Allow.

- Windows: the local Administrators group on the server that is running the administration console for Team Foundation. The administration console requires administrative permissions to operate correctly.
- SharePoint Products: the appropriate groups or permissions in SharePoint Central Administration. Depending on your deployment configuration and security requirements, you might not need to add the user to any groups in SharePoint Products. For optimum interoperability, consider adding them to the following SharePoint Products groups:
 - Farm Administrators
 - Site Collection Administrators group for all site collections that are used by the deployment of Team Foundation Server.
- Reporting Services: Team Foundation Content Manager and either sysadmin or the db_owner group membership for the configuration database, the reporting and analysis databases, and the databases for team project collections.
- SQL Server: sysadmin and serveradmin for all databases that TFS uses.

Q: Is there more than one way of granting administrator permissions in TFS?

A: Yes. You can grant administrative permissions for Team Foundation Server in two ways: from the administrative console or directly through each program for which you want to grant permissions. Granting permissions through the administrative console is simpler but has some requirements. Consider using the administrative console when all of the following conditions are true:

- Your deployment of Team Foundation Server is in a trusted environment where the service account for Team Foundation Server has permissions in SharePoint Products and SQL Server Reporting Services.
- All programs are running on the same computer (a single-server deployment).
- The security requirements for your deployment do not restrict granting one

or more of the permissions in the next bulleted list.

By default, adding users from the administration console grants them membership in the following groups in a single-server deployment of Team Foundation Server:

- Team Foundation Administrators group in Team Foundation Server
- The IIS_IUSRS and TFS_APPTIER_SERVICE_WPG groups in Internet Information Services (IIS)
- The Content Manager role in SQL Server Reporting Services, if reporting is configured
- The Farm Administrators group in SharePoint Products, if the deployment is configured to use SharePoint Products
- The DBO role and TFSExecRole for all databases that Team Foundation Server uses, including collection databases

Important

You cannot add a user to the local Administrators group by adding that user's account as a console user. You must manually add the user to that group before that user will have all the permissions that are required to open and use the console. In addition, if you want the user to have sufficient permissions to create a database as part of creating a team project collection, you must grant that user membership in the sysadmin role in SQL Server.

Granting permissions directly in each program in your deployment of Team Foundation Server is more time-consuming, but you can precisely configure the exact permissions that you want to grant to a user. Consider granting permissions directly in each program when any of the following conditions are true:

- Your deployment of Team Foundation Server is a multiple-server deployment.
- Your deployment is in an environment that has security restrictions between Team Foundation Server and the servers that are running SQL Server and

SharePoint Products.

- You want to configure different group memberships and permissions levels in SharePoint Products, SQL Server Reporting Services, and Team Foundation Server than those that are automatically granted from the administrative console.

Q: I'm an admin, but I don't seem to have all the permissions I need to add a TFS administrator. What else might I need?

A: These are the required permissions:

- Team Foundation Administrators group or have the View instance-level information and Edit instance-level information permissions set to Allow.
- If you want to add permissions for SQL Server Reporting Services, the Team Foundation Content Managers group or the System Administrators group.
- If you want to add permissions for SharePoint Products, the Farm Administrators group, the administrators group for the Web application that supports Team Foundation Server, or the SharePoint Administration group. Group membership will depend on the security architecture of your deployment and the group or groups to which you want to add the user.
- The sysadmin role in SQL Server on each server that hosts databases for Team Foundation Server.

⚠️ Important

To perform administrative tasks that involve database changes, such as creating team project collections, your user account requires administrative permissions, and the service account that the Team Foundation Background Job Agent uses also must have certain permissions granted to it. For more information, see [Service accounts and dependencies](#).

Q: What are the minimum permissions required for TFS to

connect to SQL Server?

A: To install, upgrade, and configure TFS, the user running the Team Foundation Administration console requires the following permissions and role memberships.

- Membership in the serveradmin server role
- ALTER ANY LOGIN, CREATE ANY DATABASE, and VIEW ANY DEFINITION server scoped permissions
- CONTROL permission on the master database.

If the user doesn't have these permissions and role memberships, TFS configuration operations will be blocked. When you add a user to the Administration Console Users group through the Team Foundation Server Administration Console, TFS attempts to grant these permissions and role memberships.

Q: Why are SQL Server permissions and memberships required?

A: Installing, upgrading, and configuring TFS involves a complex set of operations that require a high degree of privilege. These operations may include creating databases, provisioning logins for service accounts, and more. To ensure successful install, upgrade, and configuration, TFS checks that permissions are correctly assigned to ensure that the various operations can be done successfully. Even performing these checks requires a high degree of privilege. As such, these permissions and role memberships are required and cannot be bypassed.

Q: Can SQL Server permissions and role memberships be revoked after TFS is installed or upgraded?

A: Yes, as long as TFS service accounts are allocated the required permissions and role memberships as described in Service accounts and dependencies. Administrators only require permissions and role memberships described above when they need to install, upgrade, or configure TFS.

Q: Where can I find more information about each individual

TFS permission?

A: See [Permission reference for Team Foundation Server](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Configure and manage TFS resources

[Send Feedback](#)

You configure and manage Team Foundation Server (TFS) and additional resources to support your teams. These resources include the application-tier and data-tier server(s), team project collections which host team projects, and optional resources such as report servers, SharePoint products, build servers, lab management, and more. For information on TFS components and architecture, see [Team Foundation Server architecture](#).

Important

You should not manually modify any of the TFS databases unless you're either instructed to do so by Microsoft Support or when you're following the procedures described for manually backing up the databases. Any other modifications can invalidate your service agreement.

Through the Team Foundation Server Administration Console, you can manage and monitor your server configuration. Tasks you can accomplish through the administration console are indicated with this image: .

Application tier

- [Add administration console users](#)

- [Configure an SMTP server](#)

Data tier

- [Configure a backup schedule and plan](#)


[to support alerts and feedback requests](#)

Admin Console

- [Change cache settings for an application-tier server](#)

Admin Console

- [Manage access to features in Team Web Access](#)

- [Back up and restore TFS](#)
- [Manage TFS reports, data warehouse, and analysis services cube](#)
- [Understand backing up Team Foundation Server](#)

Team projects

Team project collections

- [Manage team project collections](#)
- Admin Console
- [Move a team project collection](#)
 - [Split a team project collection](#)

- [Create a team project](#)
 - [Delete a team project](#)
- Admin Console
- [Add reports to a team project](#)
 - [Configure or add a project portal](#)
 - [Support rollup of work and other fields](#)

Service accounts,
application pools,
and websites

Reports,
documents, and
dashboards

- [Add reports to a team project](#)

Admin Console

- [Manage TFS reports, data warehouse, and analysis services cube](#)

- [Add SharePoint products to your deployment](#)

- [Stop and start services, application pools, and websites](#)

- [Service accounts and dependencies in Team Foundation Server](#)

- [Change the service account or password for Team Foundation Server](#)

Admin Console

- [Change the service account or password for SQL Server Reporting Services](#)

Admin Console

Use the Team Foundation Administration Console to configure and manage your deployment

The administration console is installed by default on any server on which you install one or more of the following components: an application tier for TFS, TFS extensions for SharePoint Products, Team Foundation Build, and Visual Studio Lab Management

If all of your components are installed on a single server, the administration console provides management nodes for all components in your deployment. If, however, your deployment uses multiple servers, you must open the console on the server that is running the component that you want to manage.

1. If you're not a local administrator on the server where the console is running and an administrator in TFS, get those permissions.
2. To open the console:
 - From Start, open the application menu, choose Microsoft Team Foundation Server, and then choose Team Foundation Server Administration Console.

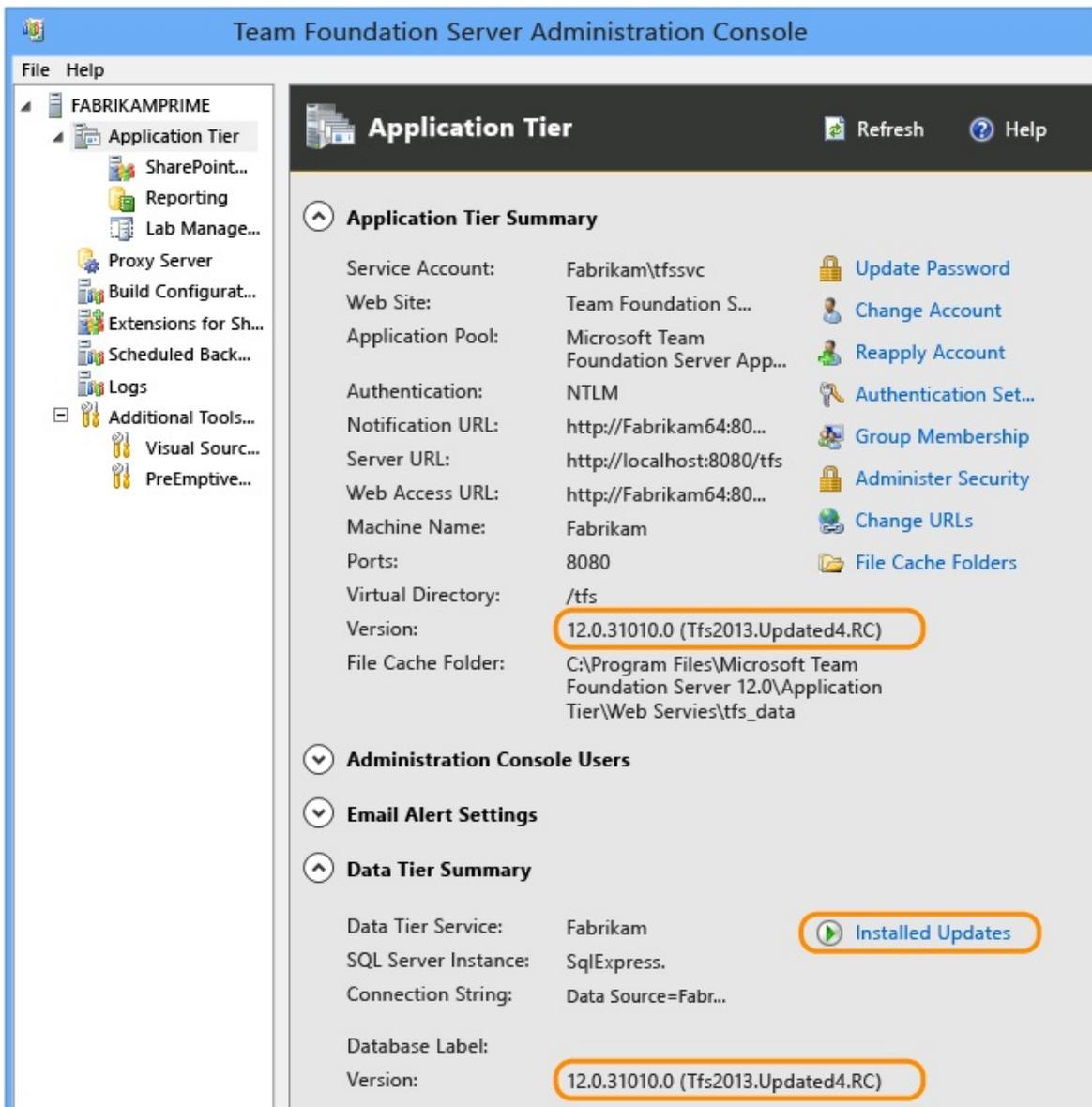
If Team Foundation Server Administration Console doesn't appear in the list of available applications, you might not have permission to open it. You can try to open it at a command prompt, but you might not be able to access some or all of the administration console's functionality.

- From the command line, open a Command Prompt window running as an administrator, and change directories to the TFS tools directory (by default, Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools), and enter TfsMgmt.exe.

View installed updates on your server

To determine what updates have been installed, you can open the administration console for Team Foundation or view the version number of a particular file.

1. Open the administration console and choose Application Tier and review the version numbers listed for the application-tier and data-tier.



The screenshot displays the Team Foundation Server Administration Console for the server FABRIKAMPRIME. The left-hand navigation pane shows the tree structure with 'Application Tier' selected. The main content area is titled 'Application Tier' and includes a 'Refresh' button and a 'Help' icon. The 'Application Tier Summary' section lists various configuration details:

Service Account:	Fabrikam\tfssvc	Update Password
Web Site:	Team Foundation S...	Change Account
Application Pool:	Microsoft Team Foundation Server App...	Reapply Account
Authentication:	NTLM	Authentication Set...
Notification URL:	http://Fabrikam64:80...	Group Membership
Server URL:	http://localhost:8080/tfs	Administer Security
Web Access URL:	http://Fabrikam64:80...	Change URLs
Machine Name:	Fabrikam	File Cache Folders
Ports:	8080	
Virtual Directory:	/tfs	
Version:	12.0.31010.0 (Tfs2013.Updated4.RC)	
File Cache Folder:	C:\Program Files\Microsoft Team Foundation Server 12.0\Application Tier\Web Services\tfs_data	

Below the summary are sections for 'Administration Console Users', 'Email Alert Settings', and 'Data Tier Summary'. The 'Data Tier Summary' section includes:

Data Tier Service:	Fabrikam	Installed Updates
SQL Server Instance:	SqlExpress.	
Connection String:	Data Source=Fabr...	
Database Label:		
Version:	12.0.31010.0 (Tfs2013.Updated4.RC)	

2. To learn when the software was installed, choose Installed Updates.

If the administration console provides isn't available, for example during a TFS upgrade, you can view the file version number of the Microsoft.TeamFoundation.Admin.dll file.

1. Open Windows Explorer (or File Explorer), and browse to \Program Files\Microsoft Team Foundation Server 12.0\Tools.
2. Expand the sub-menu for Microsoft.TeamFoundation.Admin.dll, and then choose Properties.
3. Choose the Details tab, and review the information under File Version.

Application areas and resource dependencies

You can add or remove resources to your deployment to better meet the changing needs of your business and the software projects that you support with TFS. You can add or remove reporting and Web resources at any time. You can also use more than one instance of SQL Server to host the databases for your deployment. For example, you can add a server that is running SQL Server Reporting Services to your deployment after you install and initially configure TFS. You can also upgrade the version of SharePoint Products that supports your deployment and add its capabilities of that product to the team projects that already exist in your deployment.

When you create a team project, you automatically gain access to the following functions:

- **Web access:** provides a web interface to TFS that grants access to team projects, Agile planning and tracking tools, version control, and builds. For an overview, see

[Work in Team Web Access \(TWA\)](#).

- **Source control repository** using Team Foundation version control or Git repositories.
- **Work item tracking:** teams can create work items and work item queries to track, monitor, and report on the development of a product and its features. A work item is a database record that stores the definition, assignment, priority, and state of work. Your team can create only those types of work items that are defined in the process template that is used to create the team project or types that are added to the team project after it is created.

Team members can work in TWA or Team Explorer. To learn more about these and other clients that connect to TFS, see [Choose the Team Foundation client to support your tasks](#).

The following table indicates the additional servers and functionality that you must configure for your team to have access to the corresponding feature. You can add resources before or after you have created your team project.

Feature area	Required resources	Related topics	Notes
Feedback requests and email alerts	SMTP Server	Configure an SMTP server to support alerts and feedback requests	<p>The SMTP server you configure supports all team projects created under the project collections defined on the application-tier. If the SMTP server isn't configured, then links may not appear to access the Request feedback and to set Alerts.</p>
			<p>The  Builds page lists the build definitions defined for your team project. This</p>

 Builds

Team
Foundation
Build

[Configure
and manage
your build
system](#)

page appears only when Team Foundation Build is installed and configured. Team Foundation Build enables your team to create and manage product builds. For example, a team can run daily builds and post them to a shared server. Team Foundation Build also supports build reports about the status and quality of each build.

Access to Team Foundation Build Service requires that the team project collection has been configured to

use a build controller. Each build controller is dedicated to a single team project collection. The controller accepts build requests from any team project in a specified collection. See Build the application.

The  Documents page appears only when your team project has been configured with SharePoint Products. After the team project is created, you can configure a SharePoint site or another web location as the project portal. You

[Add SharePoint products to your deployment](#)

How to: Configure a Default Location for

 Documents SharePoint (team project Products portal)

Team Project Portals may need to configure dashboard compatibility and configure an enterprise application definition. See also Share information using the project portal.

Microsoft Excel reports are uploaded to the  Documents folder when you configure your team project with a SharePoint site. With these reports you can track your team project's burn rate, bug backlog, software quality, test progress, and other metrics. Many of these reports display within your project's

Excel reports

SharePoint
Products

SQL Server
Analysis
Services

[Add
SharePoint
products to
your
deployment](#)

[Add a report
server](#)

dashboards. In addition to the SharePoint Products dependency, Excel reports depend on your team project collection that hosts your team project has been configured with both SQL Server Analysis Services.

For an overview of the default Excel reports, see Excel reports or Excel reports (CMMI).

If your team project doesn't have a SharePoint site, you can still use Excel to create status and trend reports. See Create Excel reports

from a work item query.

The  Reports page appears only when the team project collection that hosts your team project has been configured with both SQL Server Reporting Services and SQL Server Analysis Services. This page provides access to Report Manager and the default and custom reports that you upload to the server that hosts SQL Server Reporting Services.

For an overview of the default reports, see

 Reports

SQL Server
Analysis
Services

[Add reports
to a team
project](#)

SQL Server
Reporting
Services

Reports (SQL Server Reporting Services).

If some team members are located remotely from the main location for version control, you may want to install and configure Team Foundation Server Proxy to support them. TFS Proxy manages a cache of downloaded version control files in the location of the distributed team, which significantly reduces the bandwidth that is needed across wide area connections.

Remote-site support

Team Foundation Server Proxy

How to:
Install Team Foundation Proxy and set up a remote site
Configure Visual Studio to connect to TFS Proxy

Virtual environments

Visual Studio Lab Management

[Configure Lab Management for SCVMM environments](#)

If clients are configured to use Team Foundation Server Proxy, management of the files is transparent to the user. Any metadata exchange and file uploads continue to interface directly with TFS. See [Connect to team projects in Team Foundation Server](#).

By using Lab Management, your team can create, assign, and track virtual environments to support application development, deployment, and tests. Testing teams can test applications by using

Microsoft Test Manager when their team project is configured to use these virtual environments.

You can support a rapid release cadence and manage simultaneous releases using Release Management. Set up release paths that represent your stages from development to production. Run actions to deploy your app to an environment for that stage.

Automate deployments
Release Management
[Automate deployments with Release Management](#)

To support data to flow from work items TFS to tasks in enterprise project plans

Synchronize
with Project
Server

TFS-Project
Server
Integration

[Synchronize
Team
Foundation
Server with
Project
Server](#)

in Project Server, you install Team Foundation Server Extensions for Project Server on the server hosting Project Server. This integration enables project managers and software development teams to use the tools that they prefer, work at the level of precision that supports their needs, and share information transparently.

With PreEmptive Analytics for Team Foundation Server, development teams can capture

Capture
production
incidents

PreEmptive
Analytics [PreEmptive
Analytics](#)

production
incidents
(unhandled,
caught, and
thrown
exceptions)
and create and
update TFS
work items
based on
custom rules
and
thresholds.

You configure
PreEmptive
Analytics
through the
Team
Foundation
Administrator
Console.

Q & A

Q: What do I need to know about managing team projects?

A: Get started by reviewing the following topics:

- If you administer a small team, you can add team members directly to a team. However, if you need to manage a large number of users with different levels of access, see [Manage users or groups in TFS](#).
- You grant access to premium features—such as portfolio backlogs, team rooms, and test case management tools—provided through Team Web Access based on the license that each user has. See [Change access levels](#).
- If a team member has problems connecting to TFS, review the following topics: [Connect to team projects in Team Foundation Server](#) and [Configure features after a TFS upgrade](#).
- You can restrict access to a specific project, or limit access to particular resources.
- You, or members of the Project Collection Administrator or Project Administrator groups, manage many features of team projects and team project collections through the administrative context of Team Web Access.
- You can administer alerts that inform project members of changes in the team project.
- You can Customize work tracking objects to support your team's processes.
- Some upgrades require updating team projects as described in [Configure features after a TFS upgrade](#).

Q: Can I rename a team project or team project collection?

A: You can't rename a team project. You can rename a collection by first detaching the collection, editing its settings, and then attaching the collection.

See [Manage team project collections](#).

Q: How does TFS manage its services?

A: TFS and the products it integrates with include several services and service accounts which run on some or all of the logical tiers and physical computers that compose a deployment. You can use the same account for all of these service accounts, or you can use different accounts. Your choice of account or accounts to use depends on the configuration of your deployment, your security needs, and what components you have installed. For clarity, each service account is referred to separately by using placeholder names.

To learn more, see [Service accounts and dependencies in Team Foundation Server](#).

Q: How do I scale my deployment?

A: You can move TFS to a new server or add additional application-tier servers. You can also [Move Team Foundation Server from one environment to another](#).

Q: How do I monitor my deployment?

A: You can monitor TFS activity using the following tools:

- From the administration console Logs page, you can open a log file generated when configuring or servicing TFS.
- From the web administration page for the application-tier server, you can view activity logs and job monitoring charts.

`http://MyServer:8080/tfs/_oi/`

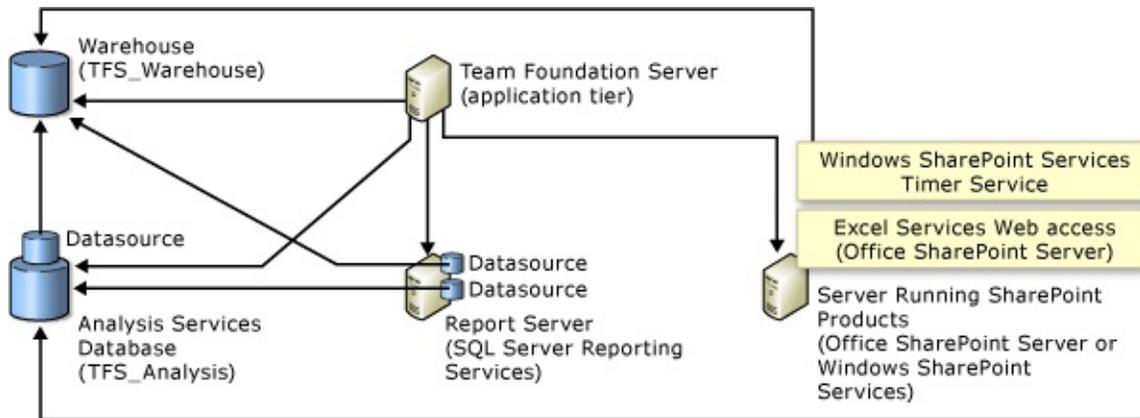
To learn more, see this blog post: [New tools for TFS Administrators](#).

- From the Control Panel context of TWA, you can export an audit log of all users and their access levels. See [Change access levels](#).

For additional monitoring tips, see this blog post: [What does a well maintained Team Foundation Server look like?](#)

Q: What do I need to know about maintaining multiple servers?

A: Depending on how you initially installed and configured Visual Studio Team Foundation Server (TFS), you might need to manage it across one or multiple servers to maintain and operate your deployment. For example, the following illustration shows the logical architecture of a deployment of TFS that is integrated with both SQL Server Reporting Services and SharePoint Products:



In this example, you have to manage the configuration of the components, the users, the groups, and permissions across several servers. You must configure firewalls or other devices to allow network traffic on the ports that are required for communication between the servers. You have to make sure that all services that the deployment requires are running and configured to start automatically. Finally, you must make sure that the service account that TFS uses is fully trusted by every computer in the deployment.

See also this blog post: [What does a well maintained Team Foundation Server look like?](#)

Q: Where can I learn more about TFS requirements?

A: See one or more of the following topics:

- The installation guide includes details about these elements:
 - system requirements
 - SQL Server requirements

- Active Directory requirements
 - SharePoint Products requirements
 - account requirements
 - language requirements
 - port requirements.
- You can also learn about requirements and infrastructure in these reference topics:
 - [Team Foundation Server architecture](#)
 - [Naming restrictions in Team Foundation](#)
 - [Service accounts and dependencies in Team Foundation Server.](#)
 - For licensing information, read the [licensing whitepaper](#).

Q: Are there any restrictions I should be aware of when maintaining TFS?

A: Yes. As mentioned above, you should never manually modify TFS databases, because that can invalidate your service agreement, block upgrades and patches, and result in data loss or corruption. There are only two circumstances where you should ever consider modification:

- You are instructed to do so by Microsoft Support.
- Your deployment needs require you to manually back up your TFS databases. In that case, you should only modify the databases as described here.

Q: What other resources are available?

A: The following resources and tools are available:

- You can post questions or search for answers in the MSDN forum: [Team](#)

[Foundation Server - Setup and Administration](#)

- [Team Foundation Server Blog](#)
 - [Microsoft Visual Studio Team Foundation Server 2013 Power Tools](#) includes the TFS Best Practices Analyzer, a Process Template editor, and additional tools.
 - [TFS CodePlex tools](#) provides access to open source software that provide extended functionality to interface with TFS.
-

- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Support rollup of work and other fields

[Send Feedback](#)

Rollup provides summed values of select fields for all child work items of a parent. Because Visual Studio Online and Team Foundation Server (TFS) support multiple levels of nesting, when you perform rollup, you want to make sure you don't double-count values. Most project managers are interested in getting rollup of estimated or completed work, effort, size, or story points.

Natively, Visual Studio Online and TFS provides rollup of Remaining Work for tasks on the Task board.

State	Original Estimate	Completed Work	Remaining Work	Title
New			78	Project 2013 test
New	6	0	6	Prepare to use the Windows Intune
New	12	0	12	Set up the Windows Intune Service
New	18	0	18	Maintain Windows Intune
New	12	0	12	Manage mobile devies using Wind...
New	30	0	30	Migrate computers from Windows

You can obtain rollup of additional data fields in Visual Studio Online or TFS data by using one of the following methods:

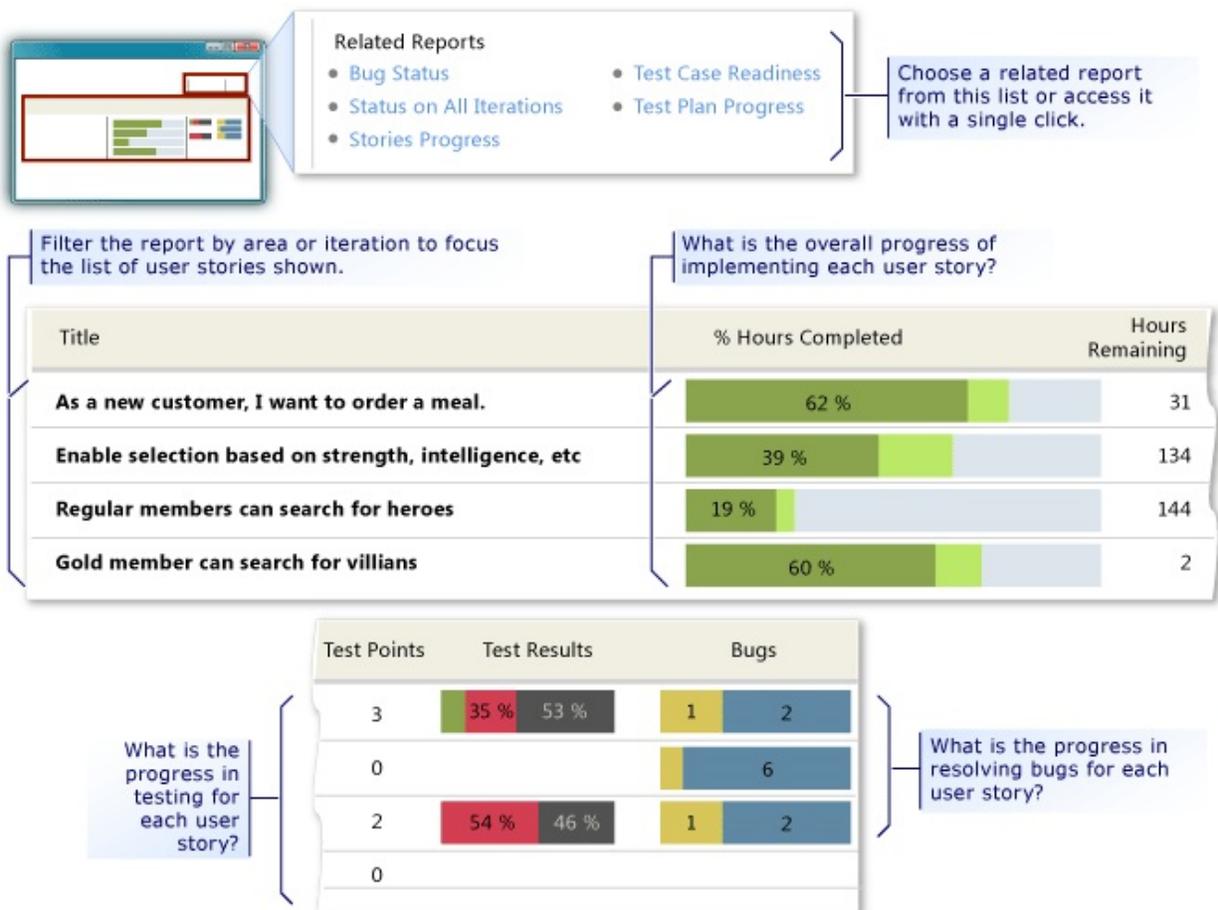
Method	Visual Studio	On-premises
--------	---------------	-------------

	Online	TFS
SQL Server Reporting Services report	✗	✓
Microsoft Project	✓	✓
Microsoft Excel	✓	✓
TFS-Project Server integration	✗	✓
Custom plug-in or extension	✗	✓

In Visual Studio Online, you're limited to the work item fields provided with the process template used to create your team project.

SQL Server Reporting Services report

Several OOB reports provide rollup. Here's an example of rollup of completed and remaining work that the Stories Overview report provides. This report is part of the default TFS Agile process template.



If you have SQL Server Analysis Services deployed, you can get rollup for backlog items from these reports. The refresh frequency for these reports is 2 hours.

- Backlog Overview (Scrum)
- Stories Overview Report (Agile)

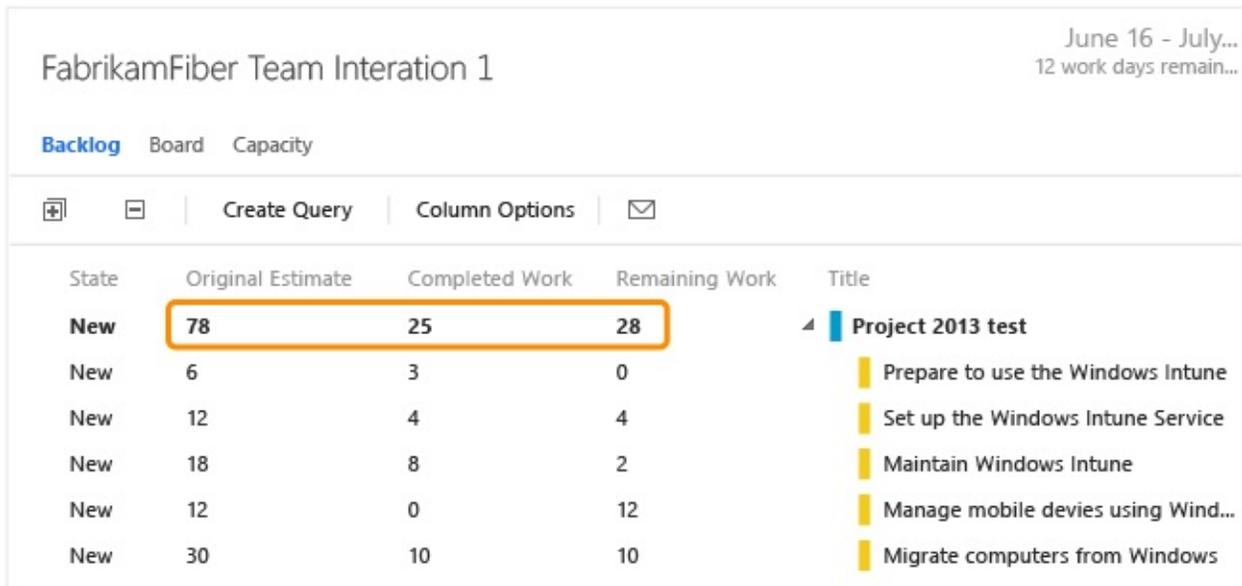
- Requirements Overview Report (CMMI)

If you need to add reports to your on-premises TFS deployment, see

[Add reports to a team project.](#)

Microsoft Project

Project natively supports rollup of summary tasks. With Project, you can round trip TFS data to obtain rollup values.



FabrikamFiber Team Iteration 1 June 16 - July...
12 work days remain...

Backlog Board Capacity

⊕ ⊖ Create Query Column Options ✉

State	Original Estimate	Completed Work	Remaining Work	Title
New	78	25	28	Project 2013 test
New	6	3	0	Prepare to use the Windows Intune
New	12	4	4	Set up the Windows Intune Service
New	18	8	2	Maintain Windows Intune
New	12	0	12	Manage mobile devies using Wind...
New	30	10	10	Migrate computers from Windows

To learn how, see Rollup estimated and actual work using Project.

Microsoft Excel

You can export a query to Excel that contains the work items you want to provide rollup. You can then write an Excel macro to get the sums and publish data back to TFS. To learn more about Excel and TFS integration, see [Bulk add or modify work items with Excel](#).

To learn more about Excel macros, see [Create or delete a macro](#).

■ **TFS-Project Server integration**

Like Project, Project Server natively supports rollup of summary tasks. If you have TFS-Project Server integration deployed, then you have rollup. To learn about how fields are synchronized, see [Understand how updates to specific fields are managed](#). If you need to add fields or change how fields are mapped, see [Customize the field mapping between TFS and Project Server](#).

■ Custom control or plug-in

You can write an extension using the client object model for Team Foundation to get rollup. Two code samples available on CodePlex that can get you started are [TFS Aggregator](#) and [TfsRollUp](#).

Rollup requirements

To support rollup, structure your work items according to the following recommendations:

- Use parent-child links to link work items that contain values that you want to rollup.
- Add required fields to the WITs that will capture the rollup values. Default fields used to schedule work are only present on the task work item. These fields are:
 - Original Estimate (Microsoft.VSTS.Scheduling.OriginalEstimate): The amount of work required to complete a task. (Agile and CMMI)
 - Completed Work (Microsoft.VSTS.Scheduling.CompletedWork): The amount of work that has been spent implementing a task. (Agile and CMMI)
 - Remaining Work (Microsoft.VSTS.Scheduling.RemainingWork): This field is used to support burndown charts.

If your team project was created using the Visual Studio Scrum process template, only Remaining Work is defined in the task.

To learn more about adding fields, see [Modify or add a field to support queries, reports, and workflow](#).

- Determine the unit of time used to track work and make sure it is used consistently across your team or organization. For example, you can track tasks using hours or days.
- Determine if you want to make rollup values read-only on the work item form. By making them read-only you prevent users from entering inaccurate data. You make fields read-only using the Control field Readonly attribute.

Q & A

Q: Can I get rollup of team capacity?

A: No. The data entered for team capacity isn't stored in the regular data stores.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

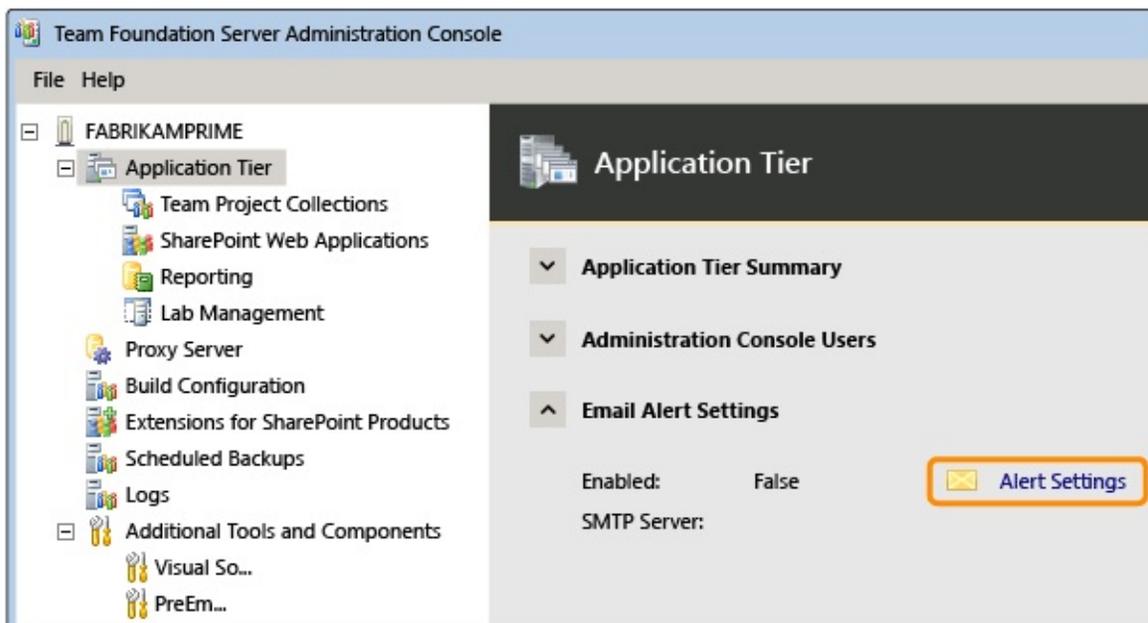
Configure an SMTP server to support alerts and feedback requests

[Send Feedback](#)

For feedback requests and alerts to work, you must configure an SMTP server for TFS.

Configure an SMTP server for TFS

1. If you aren't an administrator of TFS, get those permissions.
2. Open the Team Foundation Administration Console from the Start menu.
Or, at a command prompt, type **TFSMgmt.exe** (located in Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools).
3. Open email alert settings.



4. Enable email alerts and specify the SMTP Server and the email address to use for sending emails.

Email Alert Settings

Enabled Email Alerts

SMTP Server: Port:

This server requires a secured connection (SSL)

Send From Address:

Authentication

Use TFS Service Account

Use Specified

User Name:

Password:

Not Required

Advanced

Certificate Thumbprint:

OK Cancel

If your deployment is configured to use SSL, then select the second checkbox and enter a certificate. See also [Set up HTTPS with Secure Sockets Layer \(SSL\) for Team Foundation Server](#).

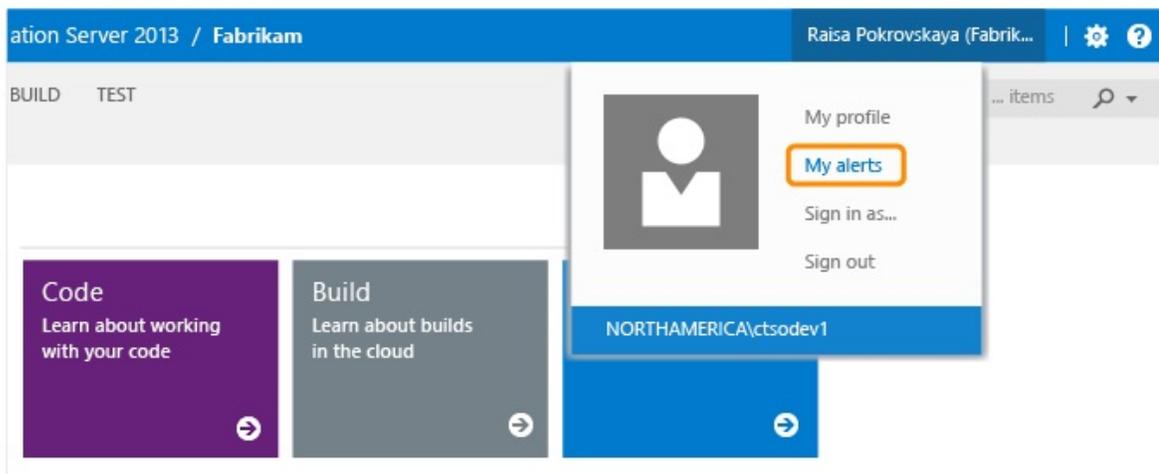
Troubleshooting tips:

- Make sure your firewall is configured to allow communications between TFS and your SMTP server.
- Your SMTP server must be configured to allow anonymous senders to send email, or you must have previously created an account specifically to use as the email account for alerts. If you specify the TFS service account (TFSService), this

account must be a domain account with permission to send email.

- If you used a system account (such as Network Service) as the TFS service account, leave the User and Password fields blank for the advanced configuration, and make sure that your SMTP server is configured to allow anonymous users to send mail. Alternately, specify an email-enabled domain account to use as the email account for alerts. If you do not, email notifications will not be sent.

5. To verify your configuration, open alerts management. You might need to refresh your browser to see this option if you just recently enabled an SMTP server.



Create an alert and then perform an action that will trigger the alert. You can set alerts for yourself or for a team.

This task supports Request and review feedback and Set alerts, get notified when changes occur.

Related content

- You can also use the TFSSConfig ConfigureMail command-line tool to configure the SMTP server.
 - To customize the format of TFS alert emails, you can modify the associated XSL transform files.
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Customize the format for TFS email alerts

[See Also](#) [Send Feedback](#)

You can customize the format of email notifications, or alerts, that team members subscribe to. These notifications are sent when changes occur to work items, code reviews, source control files, and builds. To customize their format, you can modify their associated XSL transform files. In addition, your team members can configure project alerts in Team Web Access (TWA) for various notifications and specify whether they want HTML or plain text as the format.

Alert format transform files are stored in the following folder on the application-tier server:

Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Application Tier\TFSJobAgent\Transforms\1033

You can use an XML editor, such as Visual Studio, to change any of the transform files. The following table lists the two transform files used when email notifications are sent for work item change events:

WorkItemChangedEvent.xsl	Formatting for HTML e-mail
WorkItemChangedEvent.plaintextXsl	Formatting for plaintext e-mail

The event service uses the .xsl and the plaintextXsl files to transform the XML data for an event into an e-mail message. Edit the .xsl file to get a different format for the email notification. For HTML-formatted email messages, the core layout for all events is stored in TeamFoundation.xsl. You should make a backup copy of any files that you want to change, and then test your changes. After you have made your changes, TFS will use the modified transform the next time it generates a notification for an event.

Note

The content of the emails that are issued is automatically generated from the TeamFoundation.xsl file and the above WorkItemChangedEvent xsl files. Modifying the TeamFoundation.xsl file is not recommended. If you do modify the contents of this file, you must thoroughly test your modifications. Incorrect modifications of this file can result in the failure of TFS email alerts and cause you to be unable to view work items, changesets, or files in a Web browser.

Requirements

To perform these procedures, you must be a member of the Administrators security group on the Team Foundation application-tier server.

To modify the alert format for work item changes

1. On your application-tier server, open this folder:

Drive:\%programfiles%\Microsoft Team Foundation Server
12.0\Application Tier\TFSJobAgent\Transforms\1033

2. Open WorkItemChangedEvent.xsl in Notepad or some other text editor.
3. Edit the file to show the message that you want in the notification email.

Note

You should make similar changes to the WorkItemChangedEvent.plaintextXsl for any users who have requested plaintext messages.

4. Save the WorkItemChangedEvent.xsl file.

Related content

- [Team Foundation Server Event Service](#)

See Also

Concepts

Set alerts, get notified when changes occur

Other Resources

[Configure an SMTP server to support alerts and feedback requests](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Change cache settings for an application-tier server

[See Also](#) [Send Feedback](#)

You can help increase or balance performance in your deployment of Team Foundation Server (TFS) by changing the settings of the cache for files that are under version control on the application-tier server. By default, this cache is enabled so that users can download files more quickly from the cache instead of directly from the database. As an administrator, you can change the settings of this cache any time after installation.

You can change the following settings:

- [Specify a different cache root folder.](#)
- [Change the limit at which old files are removed from the cache.](#)

You can perform these tasks by editing the web.config file for version control, which is located in the installation directory on the application-tier server.

Note

By default, the installation directory for the application tier is %programfiles%\Microsoft Team Foundation Server 12.0\Application Tier\Web Services.

Required Permissions

To perform these procedures, you must be a member of the Administrators security group on the application-tier server for Team Foundation.

For more information, see the [Microsoft website](#).

To specify a different cache root folder

1. On the application-tier server, create a cache folder.

You can create the folder on a local disk, in the UNC path, or on a mounted drive. For example, you might create the following folder:

```
d:\temp\cacheroot
```

Security Note

The cache folder stores sensitive information that is not encrypted. Therefore, you should make sure that only the service account of the application tier (TFSService) has Modify permissions to this folder.

2. Open the shortcut menu for the folder, and then choose Properties.

The Properties dialog box for the folder opens.

3. On the Security tab, choose Edit.

The Permissions dialog box opens.

4. Choose Add.

The Select Users, Computers, or Groups dialog box opens.

5. Add the local group **TFS_APPTIER_SERVICE_WPG**, and then choose **OK**.

6. Select the **Modify** check box, clear all other check boxes, and then choose **OK**.

7. In Windows Explorer (or File Explorer), browse to %programfiles%\Microsoft Team Foundation Server 2013\Application Tier\Web Services.

8. Open the web.config file in a text or XML editor, and then locate the

<appSettings> section.

9. Add a line to the appSettings section to point to the folder that you just created:

Copy Code

```
<add key="dataDirectory" value="NewCacheRootFolderLocation" />
```

For example, you would add the following line if you created a cache root folder that is named cacheroot in the temp directory of a hard drive, as in the earlier example:

Copy Code

```
<add key="dataDirectory" value="d:\temp\cacheroot" />
```

10. Save and close the web.config file.

Note

To maximize performance, copy the files from the old cache folder to the new cache folder.

11. Open a Command Prompt window, type **iisreset**, and then press ENTER.
12. Delete the old cache root folder.

Note

By default, the cache root folder is located at %programfiles%\Microsoft Team Foundation Server 12.0\Version Control Proxy\Web Services\VersionControlProxy\Data.

Changing limits for removing files from the cache

You can change the maximum limit on the amount of storage space that the application-tier server can use for caching files. When this limit is reached, a cleanup routine makes room for newly requested files by deleting those files that have not been accessed in the longest time.

To change the limit at which old files are removed from the cache

1. On the application-tier server, open Windows Explorer (or File Explorer), and browse to `\%programfiles%\Microsoft Team Foundation Server 12.0\Application Tier\Web Services`.
2. Open the `web.config` file in a text or XML editor, and then locate the `<appSettings>` element.
3. Add one of the following elements:
 - To specify a percentage of available disk space to fill before old files are removed, add the `PercentageBasedPolicy` element. You must specify a whole number as the value of this element.

For example, the following line specifies that the cache should fill up to 60% capacity of available disk space before old files are removed:

Copy Code

```
<add key="PercentageBasedPolicy" value="60" />
```

- To specify a fixed size in MB for the cache to reach before old files are removed, add the `FixedSizeBasedPolicy` element. You must specify a whole number as the value of this element.

For example, the following line specifies that the cache should reach

500 MB before old files are removed:

Copy Code

```
<add key="FixedSizeBasedPolicy" value="500" />
```

Note

If both the FixedSizeBasedPolicy and PercentageBasedPolicy elements are specified, the value of the FixedSizeBasedPolicy element is used, and the value of the PercentageBasedPolicy element is ignored.

4. Save and close the web.config file.
5. Open a Command Prompt window, type **iisreset**, and then press ENTER.

To change the amount of cache to free when removing old files

1. On the application-tier server, open Windows Explorer (or File Explorer), and browse to %programfiles%\Microsoft Team Foundation Server 12.0\Application Tier\Web Services\.
2. Open the web.config file in a text or XML editor, locate the <appSettings> element, and then add the CacheDeletionPercent element.

For example, the following line specifies to free 50% of the cache when removing old files:

Copy Code

```
<add key="CacheDeletionPercent" value="50" />
```

3. Save and close the web.config file.
4. Open a Command Prompt window, type **iisreset**, and then press ENTER.

See Also

Concepts

[Service accounts and dependencies in Team Foundation Server](#)

- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

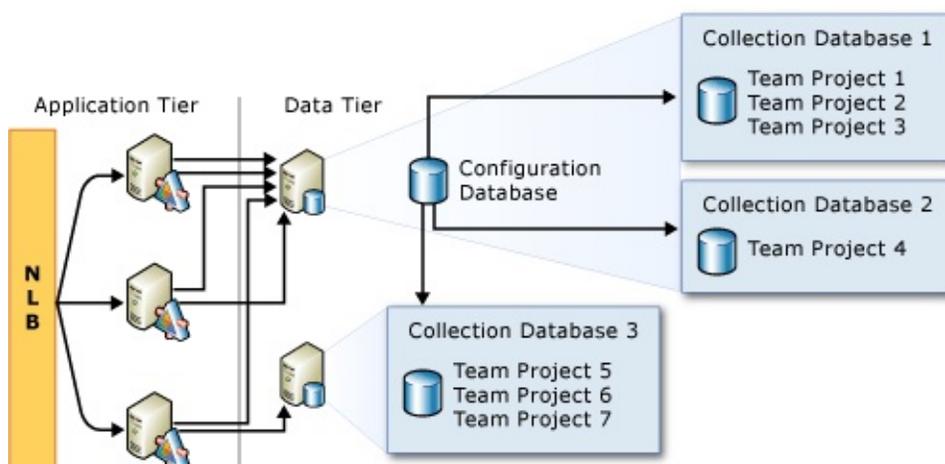
Manage team project collections

[Send Feedback](#)

When your Team Foundation Server (TFS) hosts multiple team projects, you can manage them more efficiently by grouping them together and assigning the same resources to them. For example, you can group projects that have similar requirements or objectives, such as all team projects that access a particular code base. You can then manage the group of team projects as an autonomous resource with its own user groups, server resources, and maintenance schedule.

A group of team projects is called a team project collection. When you install TFS, a default collection is created to contain all team projects. When you create a collection, you specify the logical and physical resources that team projects within that collection can use. All the artifacts and data that those team projects use are stored in the single database of the collection.

The following illustration shows how databases for team project collections are integrated with the logical architecture of TFS. When you create a team project, you specify the collection where its data will be stored.

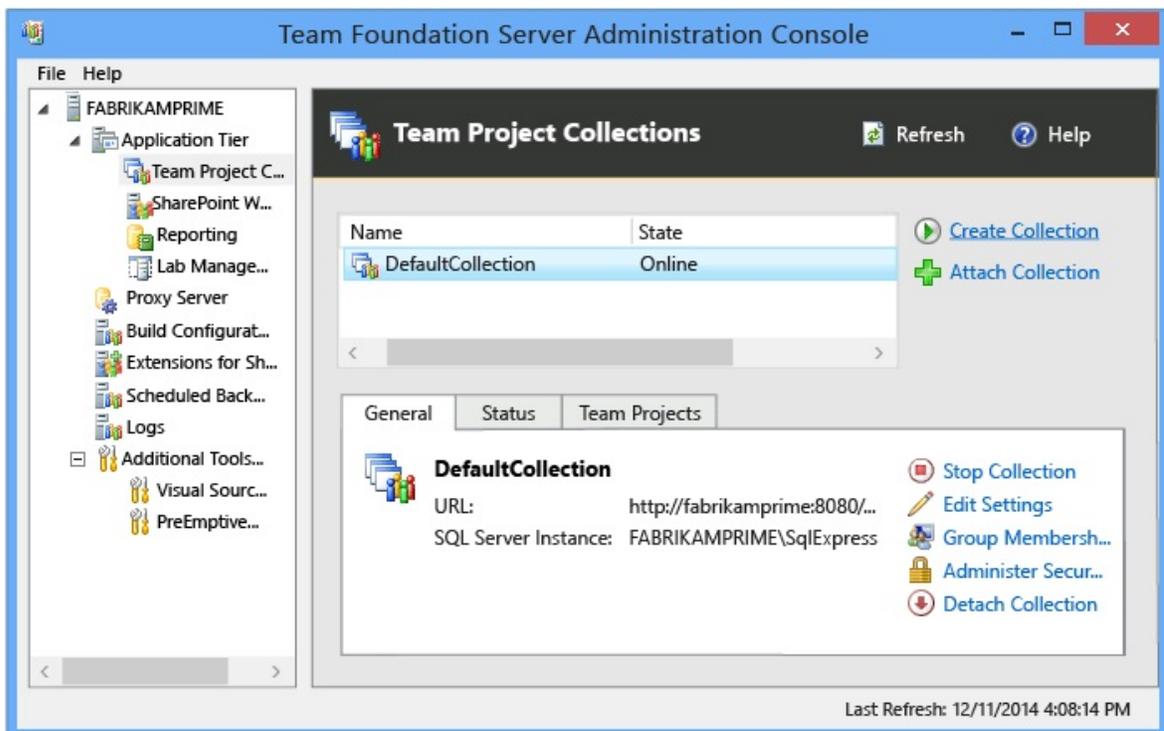


View information or configure existing team project collections

1. If you haven't been added as a TFS administrator, get added now.

You must be a member of the local Administrators group on the server where you want to open the console, and either a member of the Team Foundation Administrators group or your Edit Server-Level Information permission must be set to Allow.

2. Log on to the application-tier server for TFS, open the administration console for TFS, and open the Team Project Collections node.



To learn how to open the admin console, see Use the Team Foundation Administration Console to configure and manage your deployment.

3. Highlight the name of a collection and review the information provided from the various tabs. Some tabs only appear if the corresponding

application has been configured to support the application-tier in the TFS deployment.

You can perform the following tasks from the corresponding tab.

Tab	Tasks
General	<ul style="list-style-type: none"><li data-bbox="487 567 1395 1176">◦ Start Collection or Stop Collection: Start or stop a collection. Team projects become unavailable when you stop a collection. You typically stop a collection to support maintenance operations, such as moving or splitting a collection. <p data-bbox="487 1218 1395 1827">If the collection is started, only Stop Collection appears. If the collection is stopped, only Start Collection appears. Starting or stopping a collection can take several minutes to complete. You might need to choose Refresh to display the change in state.</p>

- Edit settings: Edit the collection's description or configuration.
- Group Membership: Add or remove users or groups as members of a collection. To learn more, see [Set administrator permissions for team project collections](#).
- Administer Security: Manage the permissions of a collection group. To learn more, see [Permission reference for Team Foundation Server](#).

Status View an activity log or rerun a job.

Team Projects View the team projects defined for a collection and [Delete a team project](#).

To learn more about team projects, see [Create a team project](#).

Reports
Folder

Configure the report server for use by the collection.

When you edit the default folder location, the operation will fail if you type the path of a folder that does not exist on the server and you do not have permission to create a folder on that server. You must specify an existing folder if you do not have permissions to create folders on that server.

To remove the default location for report subfolders, choose Clear Configuration.

Removing the configuration removes the reporting functionality for all existing and future team projects in the collection.

This tab only appears when you've added a report server to TFS.

View, configure, or remove the default root location for where team

project portals are created. The Create New Team Project Wizard creates team project portals at this location.

SharePoint Site If the SharePoint Web Application list is empty, the application-tier hasn't been configured with any applications.

This tab only appears when you've configured the application-tier with SharePoint Products. See [Add SharePoint products to your deployment](#).

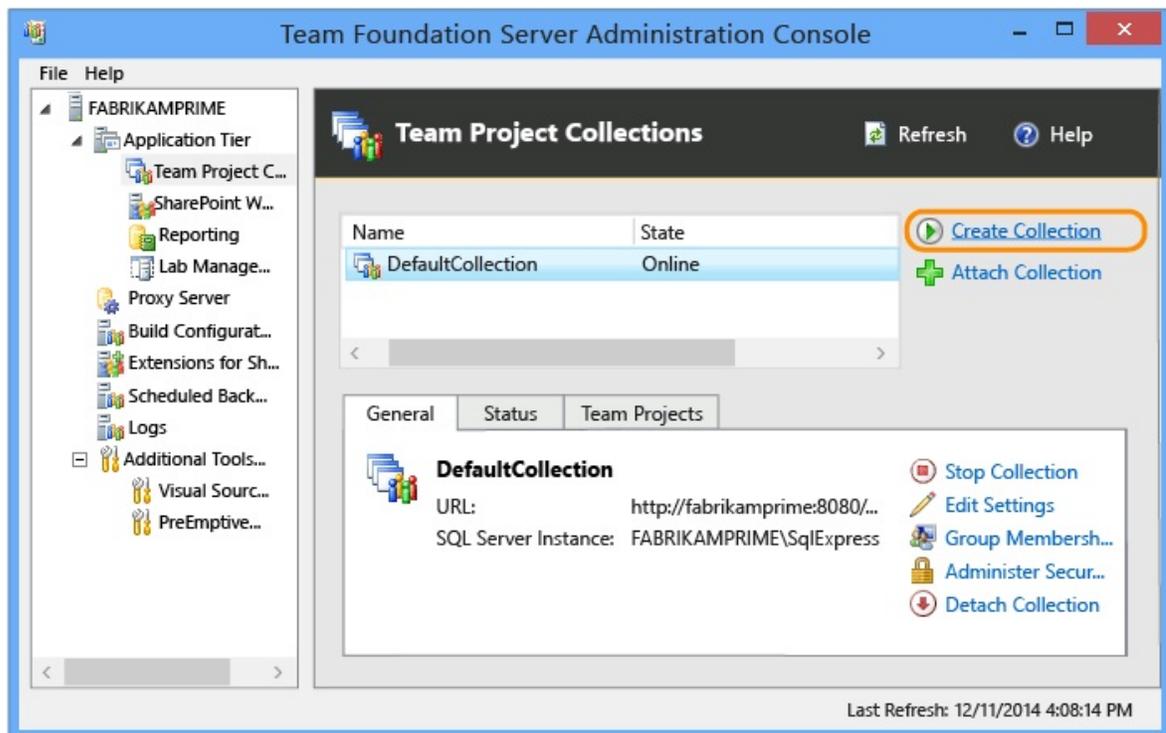
■ Create a team project collection

Before creating a team project collection, jump to this [section to learn more about the pros and cons of creating multiple team project collections](#).

1. If you haven't been added as an administrator to TFS, get added now.

You must be a member of the local Administrators group on the server where you want to open the console, and either a member of the Team Foundation Administrators group or your Edit Server-Level Information permission must be set to Allow.

2. From the administration console, open the Team Project Collections page and choose Create Collection.



3. Follow the guidance provided by the Create Team Project Collection wizard.

For the Name, specify a unique name with no more than 64 characters (the

shorter the better), and don't specify slashes, or other special characters listed in [Naming restrictions in Team Foundation](#).

The wizard supports configuration of the following resources. Some resources can only be configured if the application-tier server that hosts the collection has been previously configured to support the corresponding application.

Data Tier or SQL Server instance

1. In SQL Server Instance, specify the name of the TFS data-tier server. If you want to use a named instance to host the database for this team project collection, you must also specify the name of the instance as in the following example:

ServerName\InstanceName

2. Choose Create a new database for this collection if you want to create a database for the collection. This option requires that the service account for the Visual Studio Team Foundation Background Job Agent uses has permissions to create a database on the instance of SQL Server.

Or, choose Use this existing database if you want to use a database that already exists, and specify the name of the database in the text box. This option requires that an empty database exists on the named SQL Server instance and you have write permissions.

SharePoint web application

1. SharePoint web application appears if you have configured the application-tier with a SharePoint web application, otherwise it is disabled. To configure it later, see [Add SharePoint products to your deployment](#).
2. Choose Next if you want to use the default option to create a site collection. Choose this option unless your business infrastructure requires that you use an existing site collection. This option will create a SharePoint site collection with the name of the collection used as the

name of the sub-site of the root site that is configured in the SharePoint web application.

This option requires the TFS service account to be a member of the Farm Administrators group. If it isn't, you can't create a site collection.

3. Or, to use an existing site collection that a member of the Farm Administrators group created for you, expand Advanced configuration, choose Specify a path to an existing SharePoint site, and specify the relative path of the site collection that was created for you.

Choose Verify Path, and if the path is correct, choose Next.

SQL Server Reporting Services

1. Reports appears if you have configured the application-tier to use SQL Server Reporting Services, otherwise it is disabled. To configure it later, see [Add a report server](#).
2. Review the information for the server and the folder that will host reports, and choose Next. This option requires your user account to have permissions to create a folder on the server that is running Reporting Services.

Unless security restrictions in your business infrastructure prevent the automatic creation of a folder as part of the wizard, you should use the default option to create a folder.

3. If you must use a folder that an administrator created for you on the server that is running Reporting Services, expand Advanced configuration, choose Specify a path to an existing folder, and specify the relative path of the folder that has been created for you.

Choose Verify Path, and if the path is correct, choose Next.

Lab Management

- Lab Management appears if you have configured the application-tier to use Lab Management for the collection, otherwise it is disabled. To

configure it later, see [Configure Lab Management for SCVMM environments](#).

If you do not use Lab Management, skip to Verification process.

If you do use Lab Management, follow the steps provided in To configure Lab Management for each team project collection as described in [Configure Lab Management for SCVMM environments](#).

Verification process

1. In Readiness Checks, review the status of the checks.

A blue underlined Error indicator appears next to any configuration that contains an error. You can choose the indicator for a detailed message about the problem. You must address all errors before you can continue.

After all readiness checks have passed, choose Create.

2. The process of creating a team project collection starts.

After the wizard finishes, choose Close.

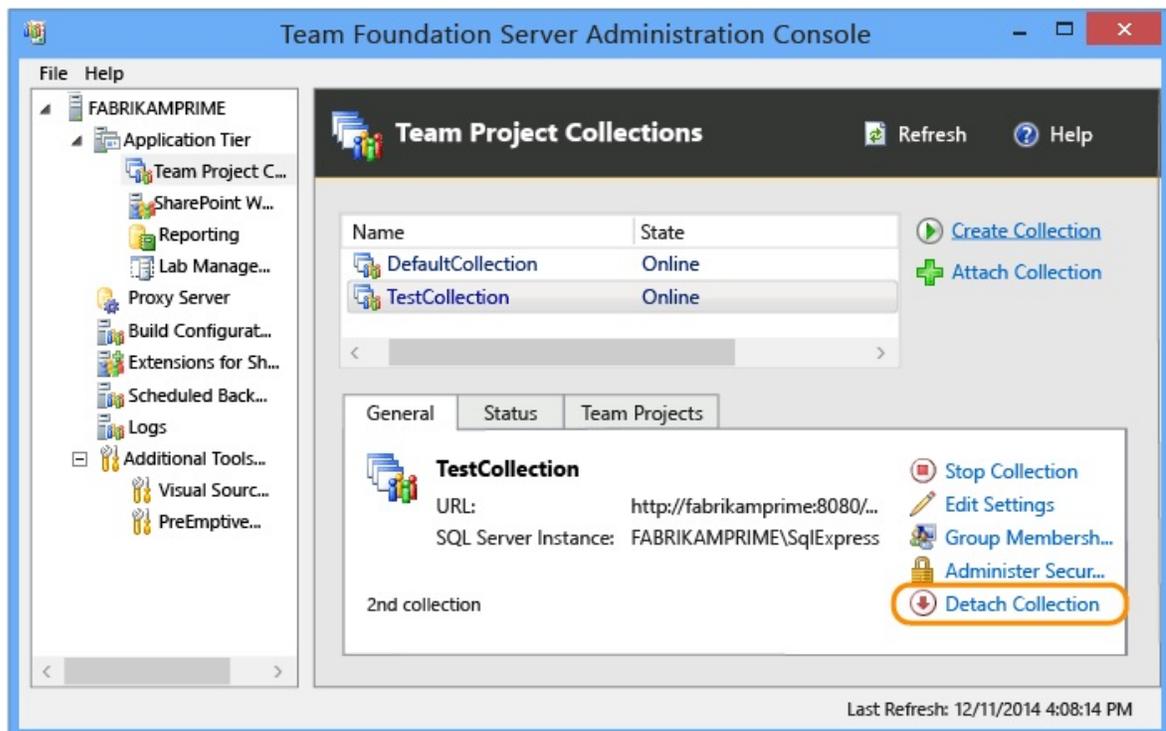
Detach or delete a team project collection

You detach a team project collection when you want to perform a maintenance operation, such as moving or splitting a collection. Teams can't access team projects or source code when you detach the collection.

You delete a collection when you no longer need the data stored in the team projects defined in the collection. The three steps to delete a collection are (1) detach the collection, (2) delete the collection database, and (3) delete the SharePoint site collection that supported the deleted collection.

Detach the collection

1. From the administration console, highlight the name of the collection that you want to delete, and then choose Detach Collection.



2. Follow the guidance provided by the Detach Team Project Collection

Wizard.

(Optional) On the Provide a servicing message for the team project collection page, in Servicing Message, specify a message for users who might try to connect to projects in this collection.

3. When all the readiness checks have completed successfully, choose Detach.

On the Monitor the team project collection detach progress page, when all processes have completed, choose Next.

4. (Optional) On the Review supplemental information for this team project collection page, note the location of the log file.

Delete the database and the SharePoint site collection

1. Open SQL Server Management Studio, connect to the instance of the SQL Server Database Engine that hosts the collection database, and expand the instance.

Highlight the name of the collection database (by default, TFS_CollectionName), and then delete the database.

For more information, see [How to: Delete a Database](#).

2. Open SharePoint Central Administration, and delete the site collection that supported the deleted collection.

For more information, see [Delete a site collection in SharePoint 2013](#).

The team project collection no longer appears in the list of collections in the administration console.

Q & A

Q: What are the pros and cons of creating multiple team project collections?

If your development efforts will benefit from the ability to branch and merge code or you must query the status of work items that relate to the same code, you should consolidate your team projects in the same team project collection.

A: Advantages for creating more than one collection

You can better separate the operational needs for one code base or other grouping of projects from the operational needs for another grouping. Because the data for each collection is stored in its own database, you can independently manage many aspects of each collection separately from other collections in your deployment. For example, you can stop and start each collection individually. Therefore, you can schedule maintenance operations for each collection at different times.

Grouping team projects into more than one collection provides the following advantages:

- Greater flexibility and scalability in managing and distributing databases and resources. A group of related team projects share reports, work items, and process guidance, as well as a code base.

By creating a database for each collection, teams and administrators can perform the following tasks:

- Build, branch, merge, and iterate an autonomous code base according to the needs of the projects within the collection. Code dependencies outside the collection can be formally managed.
- Back up and restore the data for each collection independently of other collections.
- Store all collection databases on a single instance of SQL Server, or

distribute the databases across one or more instances.

- Detach a collection, back it up, and then restore it to a different deployment of TFS.
- Reassign resources to better meet the demands of projects as they increase in size over time.
- Increased operational security. Because each collection has its own set of users and permissions, isolating different code bases can be isolated under different collections. Administrators can add users only to the collection that contains the project or projects that pertain to that particular code base.
- Increased capability to support custom workflow processes. Each collection manages process templates, work item types, link types, global lists, and work item fields separate from other collections. By separating team projects that have different workflow processes into different collections, you only expose those customizations needed to those team projects within a collection.

A: Disadvantages of creating more than one collection

The main disadvantage of creating more than one team project collection is that you increase the complexity of your deployment of TFS.

- You must backup and restore the database for each collection, and other management and maintenance tasks also increase in proportion to the number of collections that you have. For example, you must manage the set of users and permissions for each team project collection individually.
- Teams cannot link work items across collections.
- Teams cannot branch or merge code across collections.
- Teams cannot create queries across collections.

Q: What objects or resources are managed at the collection level?

A: Each team project belongs to a collection. In addition, the following objects are managed at the collection level:

- Source control (TFVC):

[File types](#) and enabling/disabling asynchronous checkout in server workspaces.

- Work item tracking: Process templates, work item types, link types, work item fields, global lists, and global workflow.

All fields defined for all team projects defined within a collection are managed or configured for a collection. You can define no more than 1,024 work item fields in the same team project collection, and you can set no more than 1,024 fields to be reportable in all team project collections.

- Dev-Test-Ops: Build controllers and Lab management resources (host group and library share from SCVMM)
- Security: Collection-level groups and collection level permissions.

Q: How does TFS support reports from data stored for different team collections?

A: A single relational data warehouse contains all reportable data from all team projects that are defined in all project collections for a TFS deployment. Data from that warehouse is then processed and written to the OLAP cube. Because data is collected into a single data warehouse, you can report across multiple team project collections.

To create or customize reports, you must add user accounts to the TfsWarehouseDataReader role. Report authors need read access to both the relational data warehouse and Analysis Services cube. These accounts can view data for all team projects that are hosted in all team project collections in the TFS deployment. There is no way to limit access to a team project or collection.

Q: How do I scale my deployment with collections?

A: See [Move a team project collection](#) or [Split a team project collection](#).

Q: Is there a command line tool for managing collections?

A: You can use the **TFSCONFIG Collection** command to attach, detach, delete, or

clone a team project collection. See [Collection command \[TFSConfig\]](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Move a team project collection

[Send Feedback](#)

You can move a team project collection from one deployment of Team Foundation Server (TFS) to another. For example:

- You have more than one deployment of TFS in your organization, and you want to move a collection to another deployment to better align with your business needs.
- You want to move the collection to a remote office that has its own deployment of TFS.
- You want to expand your deployment of TFS by adding another instance of SQL Server to it, and you want to distribute existing collections between the instances.
- You need to incrementally upgrade your deployment by detaching an individual team project collection from a deployment of TFS running an earlier version, and then move it to a server running the current version of TFS. (One common reason for this is some teams might need to migrate to a newer version of TFS, while others must remain on an older version for tools or projects reasons.) In this scenario, you must also then upgrade each team project within the collection by using the Configure Features wizard. For more information, see [Configure features after a TFS upgrade](#).

The steps to move a collection will depend on the particular configuration of the deployment that hosted the collection and the deployment to which you move it. For example, if you move a collection to a new domain, you will need to add users from that domain to the appropriate groups at the collection level and the project level. Similarly, if you move a collection to a deployment of TFS that uses a different SharePoint Web application, you'll either need to move the site

collection database that supports the collection to that SharePoint Web application, or you must add the SharePoint Web application that supports the collection to the deployment of TFS to which you have moved it.

Here's how to move an entire team project collection. If you want to move part of a collection, see [Split a Team Project Collection](#).

In this topic

1. [Detach the collection](#)
2. [Back up the collection database](#)
3. [Move the collection database](#)
4. [Attach the collection](#)
5. [Configure the moved team project collection](#)
6. [Configure projects](#)

[Q & A](#)

- My TFS deployment uses reporting. How do I move reports when I move a collection?
- How do I move a collection that uses SharePoint Products?
- How do I move a team project collection that includes Lab Management?
- Can I move a team project collection that's integrated with Project Server?
- I moved my collection to a TFS that has users or groups that shouldn't have access to projects or functions in some of the team projects, but need access to other projects in the moved collection. Can I keep them from seeing projects I don't want them to see?

Before you begin

Before you start your move, make sure that you're an administrator on the servers and in the software used by both the deployment you're moving from,

and the one you're moving to. If you're not an administrator, [get added as one](#).

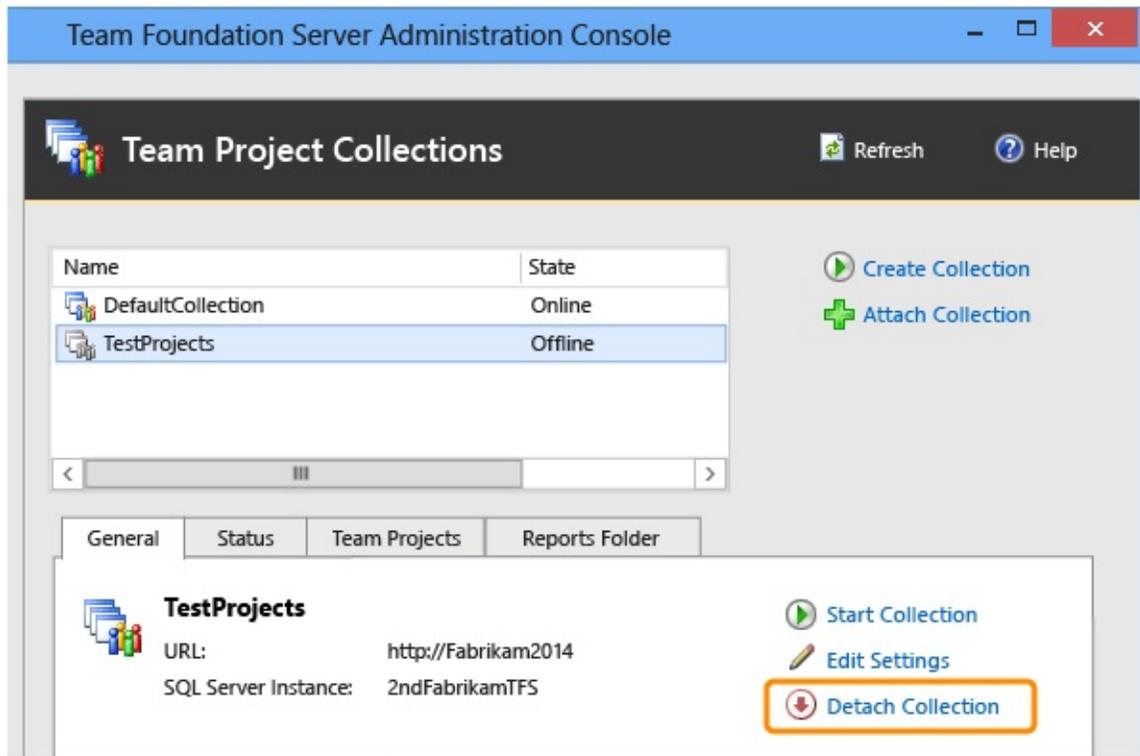
1. Detach the collection

Before you move a collection, you must first detach it from the deployment of TFS on which it is running. It's very important that you not skip this step. When you detach a collection, all jobs and services are stopped, and then the collection database is stopped. In addition, the detach process copies over the collection-specific data from the configuration database and saves it as part of the team project collection database. This configuration data is what allows the collection database to be attached to a different deployment of TFS. If that data is not present, you cannot attach the collection to any deployment of TFS except the one from which it originated.

Note

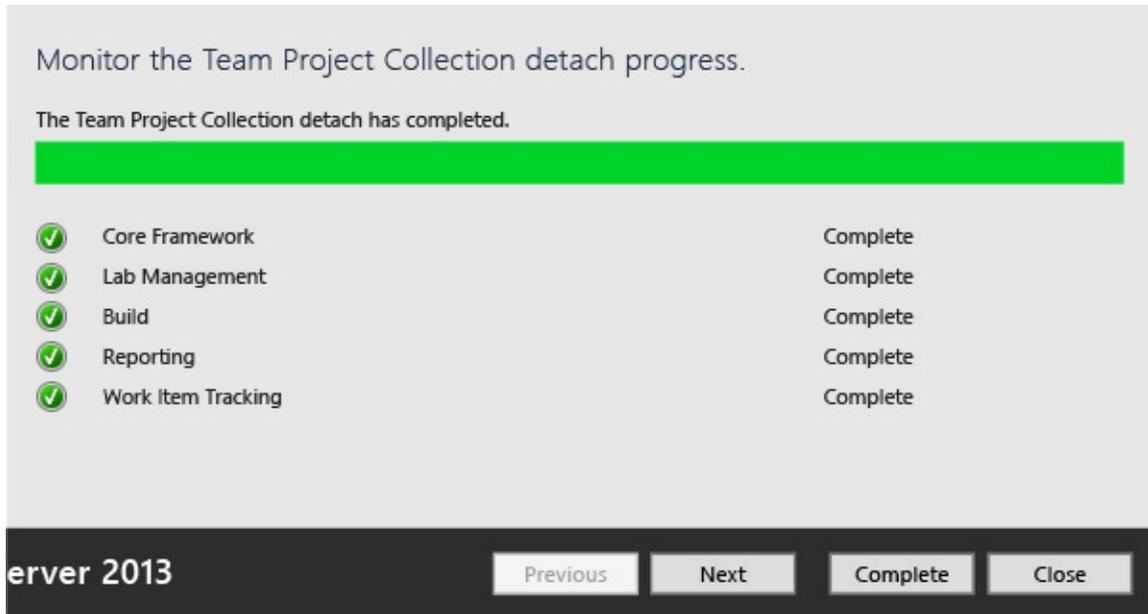
Detaching a collection prevents users from accessing any projects in that collection.

1. Open the administration console for Team Foundation on the server that hosts the collection that you want to move, and in Team Project Collections, highlight the collection that you want to move.
2. On the General tab, choose Detach Collection.



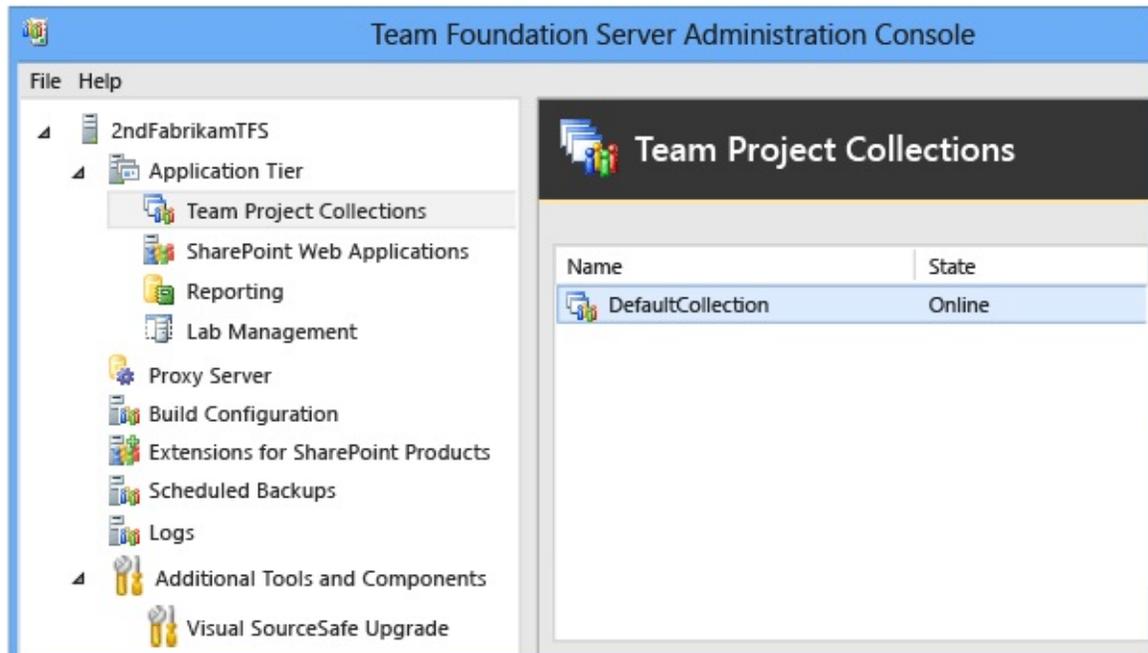
The Detach Team Project Collection Wizard opens.

3. (Optional) On the Provide a servicing message for the team project collection page, in Servicing Message, provide a message for users who might try to connect to projects in this collection.
4. Choose Next.
5. On the Review settings that will be used to detach team project collection page, review the details.
6. If you want to changes any settings, choose Previous. If they appear to be correct, choose Verify.
7. When all the readiness checks have completed successfully, choose Detach.
8. On the Monitor the team project collection detach progress page, when all processes have completed, choose Next.



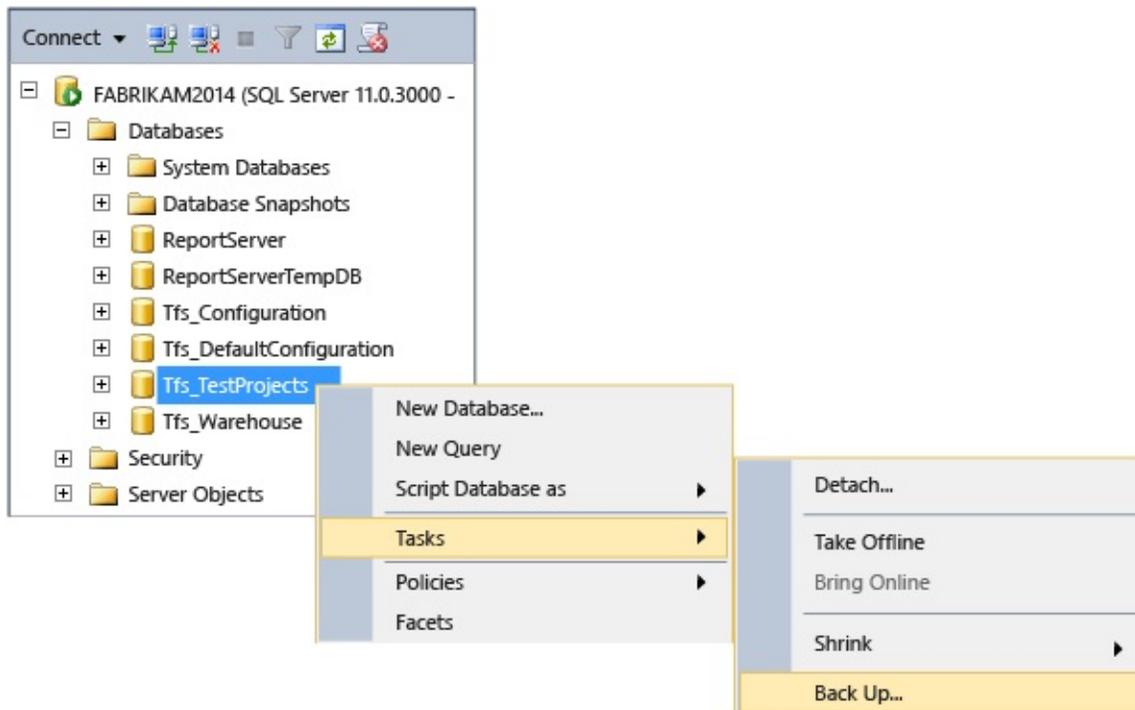
9. (Optional) On the Review supplemental information for this team project collection page, either choose or note the location of the log file.
10. Choose Close.

The team project collection no longer appears in the list of collections in the administration console.



2. Back up the collection database

After you have detached the collection, you must back up its database to move it to the other server. To perform this task, you must use the tools that are provided with SQL Server.



- For information about this task, see the following page on the Microsoft website, and make sure to choose the version of SQL Server that matches your deployment:

[Backing Up and Restoring Databases in SQL Server](#) and [Configure a backup schedule and plan](#).

⚠ Important

You can only restore a database to the same version or a more recent version of SQL Server. You can't restore a SQL Server database to an earlier version of the product.

If your original deployment used the Enterprise or Datacenter editions of SQL

Server, and you want to restore databases to a server running Standard edition, you must use a backup set that was made with SQL Server compression disabled. Unless you disable data compression, you will not be able to successfully restore Enterprise or Datacenter edition databases to a server running Standard edition. To turn off compression, follow the steps in the [Microsoft Knowledge Base article](#).

3. Move the collection database

As part of moving the collection, you must restore, copy, or otherwise move the collection database to an instance of SQL Server that is configured to support the deployment of TFS to which you want to move the collection. You can choose whichever method you prefer to move the database.



For more information about moving a database to another instance of SQL Server, see:

- [Use the Copy Database Wizard](#)
- [Copy Databases with Backup and Restore](#)

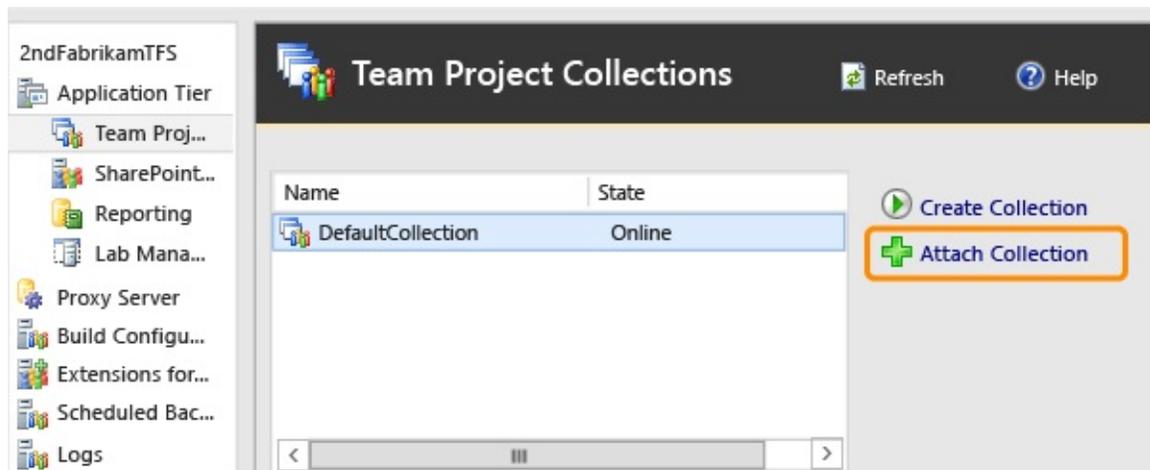
5. Attach the collection

After you restore the collection database, you can attach the collection to the deployment of TFS to which you want to move it. If the deployment you're moving to uses reporting, a reporting folder and default reports will be built for the collection you're attaching as part of the process.

Note

Warnings will appear when you attach the collection if your deployment uses SharePoint Products and the service account for TFS is not a member of the Farm Administrators group, or if your deployment uses reporting and you've already created a reporting folder and path that is identical to the previous deployment's folder and path. This behavior is expected, and you can proceed.

1. Open the administration console for Team Foundation on the server that hosts the application tier for the deployment to which you want to move the collection.
2. Choose Team Project Collections, and then choose Attach Collection.



The Attach Team Project Collection Wizard opens.

3. On the Select the team project collection database to attach page, in SQL

Server Instance, provide the name of the server and the instance that hosts the collection database, if it is not already listed.

Select the Team Project Collection database to attach.

Type in a SQL Server instance path, click List Available Databases, then select a database from the list.

SQL Server Instance:
 ⓘ [List Available Databases](#)

Databases:

Database Name	Version
Tfs_TestProjects	Microsoft Team Foundation Server 2013 Project Collection Database

This operation will update the schema of your collection database to the TFS 2013 format. This format is not usable on previous versions of TFS and is not reversible without a backup. You may create a backup now by launching the backup tool from the link below. You must confirm that you have a current backup before you continue.
[Click here to launch the Database Backup Tool](#)

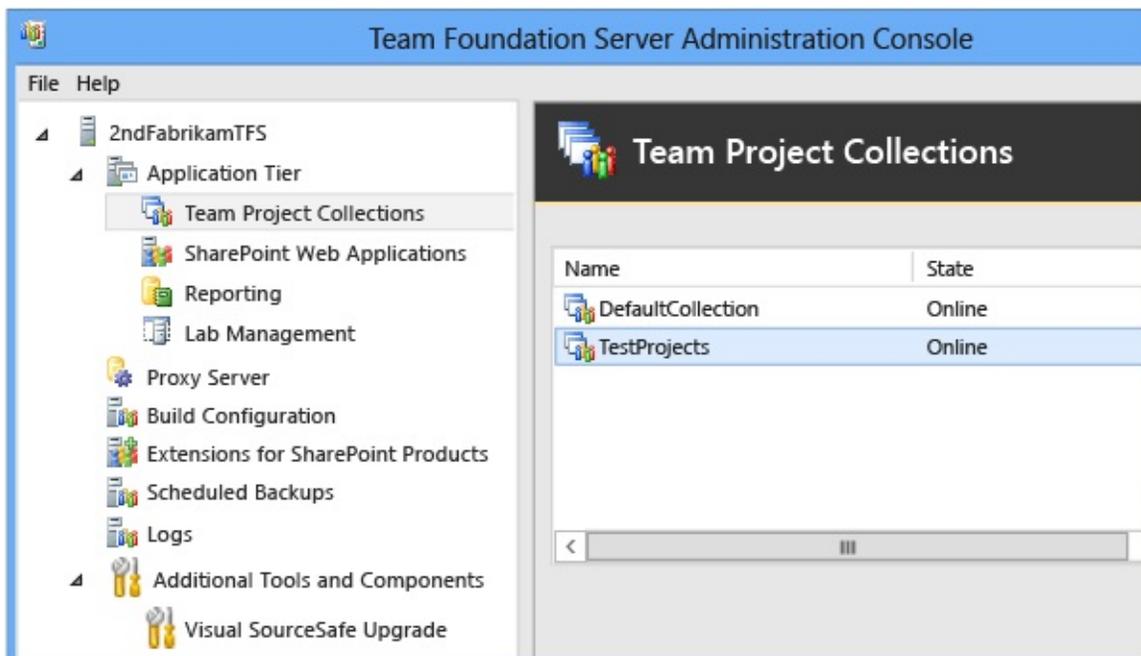
 By checking this box, I confirm that I have a current backup.

erver 2013

4. In the Databases list, choose the collection database that you want to attach, and then choose Next.
5. On the Enter the team project collection information page, provide a name for the collection in Name if one is not already present.
6. (Optional) In Description, provide a description of the collection.
7. Choose Next.
8. On the Review settings that will be used to attach the team project collection page, review the information.
9. If you must change any settings, choose Previous. If all the settings are correct, choose Verify.
10. When all the readiness checks have completed successfully, choose Attach.

11. On the Monitor the team project collection attach progress page, when all processes have completed, choose Next.
12. (Optional) On the Review supplemental information for this team project collection page, choose or note the location of the log file.
13. Choose Close.

The team project collection appears in the list of collections in the administration console. The SharePoint Web application that supported this collection in its original deployment will appear in the list of Web applications.



14. (Optional) Modify or remove the Web application from the list of Web applications. For more information, see [Modify or Remove Access Between a SharePoint Web Application and Team Foundation Server](#).

6. Configure the moved team project collection

You can skip this procedure if you moved the collection in the same domain, intend to use the same Web application that previously supported the collection, **and** you want to allow access for the administrators of that collection to this deployment of TFS.

After you have moved a collection, you must update the Web application and permission groups for that collection with the appropriate settings.

To configure the moved collection

- Open each tab for the team project collection and, if necessary, modify the settings to reflect the services and locations to match the locations of the resources on the current TFS. This is particularly important to do for SharePoint and reporting resources. If you see errors, make sure that your account has the permissions required for administrators of team project collections, and that any other team project collection administrators have been added appropriately.

For information about this task, see

[Manage team project collections.](#)

■ **Configure projects**

You can skip this procedure if you moved the collection in the same domain and you want to allow access for the users of projects in that collection to this deployment of TFS.

After you configure administrators for the moved collection, either you or those administrators must add users and groups to the projects in that collection. Depending on your deployment, you might also need to configure permissions for those users in SharePoint Products and Reporting Services.

To add users to team projects

- For information about this task, see

[Add users to team projects.](#)

To add resources to moved team projects

- For more information about this task, see [Configure and manage TFS resources.](#)

Q & A

Q: My TFS deployment uses reporting. How do I move reports along with moving a collection?

A: First, you'll need to save or export any reports you want to move from the report server that supported the collection in its original deployment. Then you'll need to upload each report manually to the report server that will support the moved collection, which can be a lengthy process. Consider whether you want to upload all reports or whether a subset of reports will meet your business needs. You do not have to upload all reports or any reports, but only those reports that you upload will be available after the move process is completed.

You'll also need to rebuild the warehouse and analysis services cube on the original deployment after you've moved the collection, so that the original deployment doesn't keep trying to build reports for a collection that is no longer there.

To move reports

1. Export or save the reports you want to move from the report server that supported the collection in its original deployment. For information about this task, see

[Exporting Reports](#) and [Saving Reports](#).

- Upload each report that you want to move to the appropriate folder on the report server that supports the collection in its new environment in Report Manager.

For more information, see the following topic on the Microsoft website: [Uploading Files to a Folder](#).

- In Report Manager, edit each report to change the data source to the new report server.

For more information, see the following topic on the Microsoft website: How to:

Configure Data Source Properties for a Report.

Rebuild the data warehouse and Analysis Services

1. Open the administration console for Team Foundation.
2. In the navigation bar, choose Reporting.
3. In Reporting, choose Start Rebuild.
4. In the Rebuild the Warehouse and Analysis Services Databases dialog box, choose OK.

📌Note

The warehouses will finish rebuilding and the data will finish repopulating after the Start Rebuild action completes. Depending on the size of your deployment and the amount of data, the entire process might take several hours to complete.

❓Q: How do I move a collection that uses SharePoint Products?

A: To move a team project collection that uses a SharePoint Web application, you must move both the team project collection itself and the SharePoint site collection that supports the team project collection. The site collection must be moved to the Web application that will support the team project collection in the new deployment. Specifically, you must [back up the site collection database](#) and then [move the site collection database](#). Once you've done that and attached the moved team project collection to its destination TFS, you'll need to repair the connection between that TFS and its SharePoint web application to ensure that the attached collection connects properly to the moved site collection. You'll also need to make sure that the SharePoint tab for the team project collection points to that site collection database.

📌Note

If you are moving the collection between deployments that use SharePoint Products, it is strongly recommended that the service account for TFS be a

member of the Farm Administrators group in SharePoint Products in both deployments. Otherwise, you might experience errors when you attempt to detach or attach the collection.

You can move a team project collection without granting this membership to the service account for TFS. However, errors will appear when you attach the collection, and you will need to perform additional steps to reconnect projects with their portals. Even if your operational requirements generally restrict granting this membership to the service account, you should consider adding the service account to the Farm Administrators group for the duration of the move operation.

1. Open the administration console for Team Foundation, choose SharePoint Web Applications, and in the list of Web applications, choose the Web application that will support the collection that you just attached.

The Repair Connection button appears after you select a Web application in the list.

2. Choose Repair Connection, and in the Repair the connection to a SharePoint Web Application dialog box, choose Repair.
3. In Team Project Collections, select the moved team project collection, choose the SharePoint Products tab, and modify the settings to point to the site collection database.

Q: How do I move a team project collection that includes Lab Management?

A: If you moved the collection to a different domain or intend to use a different System Center Virtual Machine Manager, you must delete the virtual machines, templates, team project host groups, and team project library shares from the collection database before you start the move, and recreate them in Microsoft Test Manager after the move.

To delete the Lab Management resources before moving the collection

- For information about how to remove all group hosts, library shares, and environments from a specified team project collection, see [TFSSConfig Lab /Delete Command](#) with the /External option.

To configure Lab Management resources after moving the collection

1. Configure the application tier for Team Foundation.

For more information, see [Configure Lab Management for SCVMM environments](#).

2. Recreate the golden master virtual machines and templates in the new SCVMM and import virtual machines and templates into the team project collection.

For more information, see [Create and store virtual machines and templates ready for Lab Management](#).

3. Recreate the environments for each team project.

For more information, see [Creating an SCVMM Environment Using Stored Virtual Machines and Templates](#).

Q: Can I move a team project collection that's integrated with Project Server?

A: Yes you can. You'll probably need to unmap the collection from Project Web Access or Project Web App (PWA) before the move, and remap it afterwards. Your exact steps will vary. Each instance of PWA can integrate with only one deployment of TFS. Collections on different deployments of TFS cannot synchronize with the same instance of PWA.

For example, if you have two collections on your deployment of TFS and they are both integrated with Project Server:

- If you intend to move both collections and continue to use the same instance of PWA, you do not need to unmap the collections. However, you must re-register them after the move by following the steps in this section.

- If you intend to move only one collection, you must first unmap one of the collections. Which collection you unmap depends on how you want to support Project Server integration with your TFS deployments.
 - You must unmap a collection if you do not intend to continue integration of it or if you are moving it to a deployment of TFS that is integrated with a different instance of PWA.
 - If you intend to continue integration with the same instance of PWA, you must unmap the other collection or collections that are integrated with that instance before you move the collection.

To unmap a collection before you move it

- Use the **TFSAdmin ProjectServer /UnmapPWAFromCollection** command with the **/force** option.

⚠Important

If you use the **/force** option, you will unlink all work items from Project Server, which can have unintended consequences if this isn't what you meant to do. For more information, see [Map a team project collection to an instance of PWA and Change your deployment configuration](#).

Once you've finished moving the collection, if you want to continue integration with Project Server, you'll need to perform additional steps:

- Register the instance of Project Web Access or Project Web App (PWA) that supports an enterprise project plan with the application-tier server that hosts the moved team project collection.
- Wait for the metadata to synchronize.
- Unregister the application-tier server that formerly hosted the collection, if it does not host other collections that are integrated with Project Server.

To integrate the collection with Project Server after the move

- To register an instance of PWA with the deployment of TFS that hosts the moved collection, use the **TFSAdmin ProjectServer /RegisterPWA** command with the **/tfs**, **/force**, and **/pwa** options. For more information, see Register an instance of PWA to TFS.

After you have registered the instance, you must wait for the metadata to synchronize. This process happens automatically on a predetermined schedule. For more information, see Synchronization process overview for TFS-Project Server integration.

- After synchronization is complete, remove the registered association between the previous instance of PWA, if any, and TFS by using the **TFSAdmin ProjectServer /UnregisterPWA** command with the **/tfs** and **/pwa** options. For more information, see Register an instance of PWA to TFS and Remove a component from participating in data synchronization.

≡**Q: How do I restrict access to select functions in a team project?**

A: Users who have permissions to access one project within a collection can view other projects within that collection, even if they don't have permissions to modify work items or perform other actions in that project. You can restrict individuals or TFS groups from creating or modifying select artifacts, as well as restrict them from seeing projects, by specifically creating groups and configuring restrictions on those groups. For more information, see Restricting access to projects in the deployment.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Split a team project collection

[Send Feedback](#)

As your business changes, you might want to split a single team project collection into multiple team project collections. For example:

- You want the projects in a collection to align with business units in your organization, and the projects in the collection are now owned by separate units.
- You upgraded from an earlier version of TFS, you have only one collection, and you want to organize your projects into separate collections for security or business alignment reasons.
- You want to change ownership of some of the projects in the collection to a remote office that has its own deployment of TFS. This scenario requires that you first split a collection and then move one of the resulting collections to the remote office deployment.

Note

The procedures in this topic support only splitting a team project collection. If you want to move a collection after you split it, see [Move a team project collection](#).

In this topic

To split a team project collection, follow these steps:

1. Prepare to split the collection:

1. [Detach the collection](#)

2. [Back up the collection database](#)
2. Split the collection:
 1. [Restore the collection database with a different Name](#)
 2. [Attach the original collection database](#)
 3. [Attach the renamed collection database](#)
 4. [Delete projects from the split collections](#)
 5. [Start the collections](#)
3. Configure the split collections:
 1. [Configure Users and Groups for the Split Team Project Collections](#)
 2. [Configure users and groups for projects in the collections](#)

[Q & A](#)

- Q: My deployment uses reporting. Are there any additional steps I need to take when splitting collections?
- Q: Can I split a collection that uses SharePoint Products to support one or more team projects in the collection?
- Q: Are there any special considerations or actions to take if I split a team project collection that has projects integrated with Project Server?
- Q: How do I split a collection configured for Lab Management?

Before you begin

Make sure that you're an administrator on the servers and in SQL Server and TFS. If you're not an administrator, [get added as one](#).

■ **1-a. Detach the collection**

First detach the collection from the deployment of TFS on which it is running. Detaching a collection stops all jobs and services as well as the collection database itself. In addition, the detach process copies over the collection-specific data from the configuration database and saves it as part of the team project collection database.

To detach a team project collection

1. Open the administration console for Team Foundation on the server that hosts the collection that you want to split.
2. Choose Team Project Collections, and in the list of collections, choose the collection that you want to split.

In this example, the administrator chooses "TestProjects."

The screenshot shows the Team Foundation Server Administration Console. On the left, a tree view shows the hierarchy: 2ndFabrikamTFS > Application Tier > Team Project Collections. The main content area is titled 'Team Project Collections' and contains a table with the following data:

Name	State
DefaultCollection	Online
TestProjects	Online

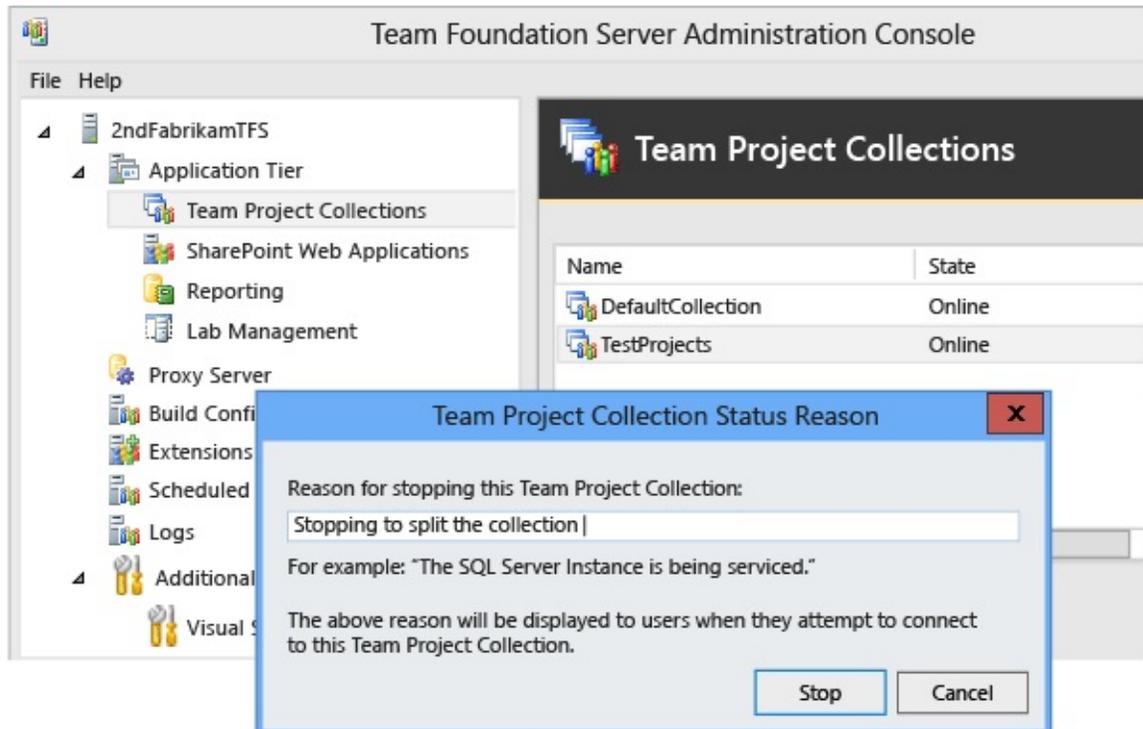
Below this table, there are tabs for 'General', 'Status', 'Team Projects', and 'Reports F...'. The 'Team Projects' tab is selected, showing a sub-table of projects:

Name	State
GlobalPhone	Active
Lorem Ipsum Phone Tiger Test	Active
Phone Antivirus Protection	Active
Phone Green	Active
PhoneSaverNextGen	Active
Pizza Style Phone Experience	Active
SuperPhone	Active

Tip

The default name for a team project collection is "DefaultCollection." If you are splitting this database, make sure to give the second collection a distinctly different name, because this is the default choice at connection.

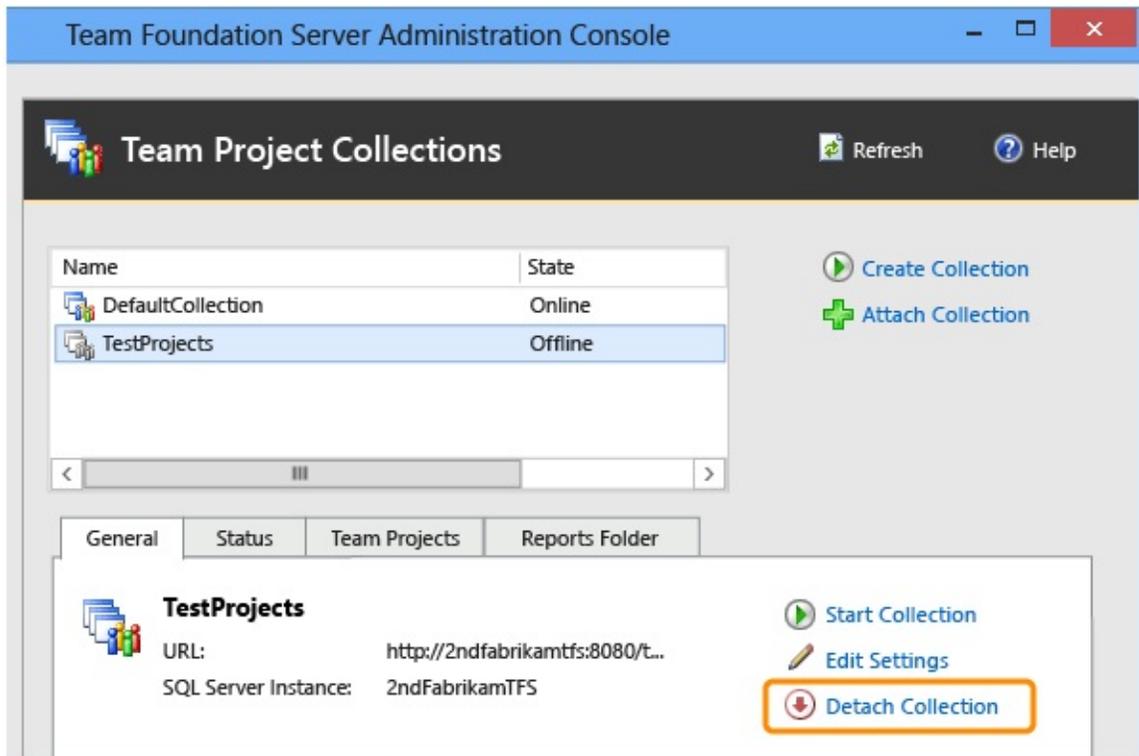
3. On the General tab, choose Stop Collection.



The Team Project Collection Status Reason dialog box opens. The text you enter will be displayed to your users. Choose Stop, and wait for the collection to stop. When it is stopped, its status will show as Offline.

4. On the General tab, choose Detach Collection.

The Detach Team Project Collection Wizard opens.

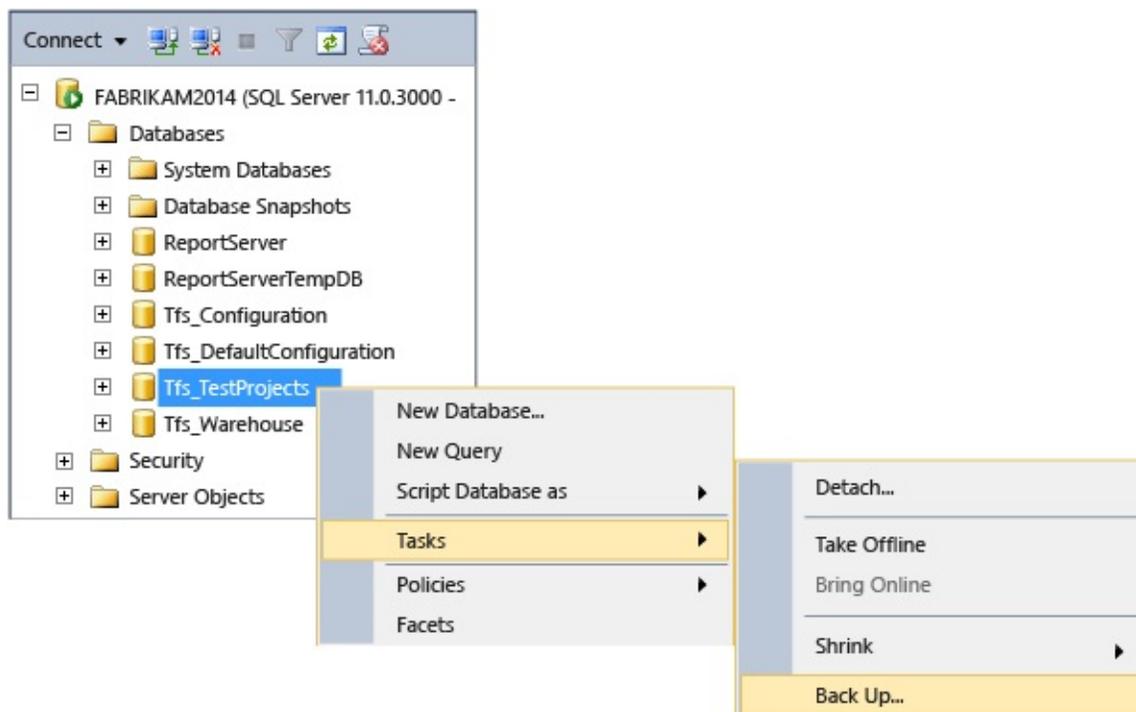


5. (Optional) On the Provide a servicing message for the team project collection page, in Servicing Message, provide a message for users who might try to connect to projects in this collection.
6. On the Review settings that will be used to detach team project collection page, review the details. If you want to changes any settings, choose Previous. If they appear to be correct, choose Verify.
7. When all the readiness checks have completed successfully, choose Detach.
8. On the Monitor the team project collection detach progress page, when all processes have completed, choose Next.
9. (Optional) On the Review supplemental information for this team project collection page, choose or note the location of the log file, and then close the wizard.

The team project collection no longer appears in the list of collections in the administration console.

1-b. Back up the collection database

After you have detached the collection, you must back up its database before you can restore a copy to the server with a different name. That copy will become the database for the part of the original collection that you want to split into another collection. To perform this task, you must use the tools that are provided with SQL Server.



To back up a collection database

- For information about how to manually back up and restore individual databases, see the following pages on the Microsoft Web site, and make sure to choose the version of SQL Server that matches your deployment:

[Backing Up and Restoring Databases in SQL Server](#) and [Configure a backup schedule and plan](#).

⚠ Important

If your original deployment used the Enterprise or Datacenter editions of SQL

Server, and you want to restore the database that you want to split to a server running Standard edition, you must use a backup set that was made with SQL Server compression disabled. Unless you disable data compression, you will not be able to successfully restore Enterprise or Datacenter edition databases to a server running Standard edition. To turn off compression, follow the steps in the [Microsoft Knowledge Base article](#).

2-a. Restore the collection database

When you split a collection, you must restore the backup of the collection database to an instance of SQL Server that is configured to support the deployment of TFS. When you restore the database, you must give it a different name from the name of the original collection database.

Tip

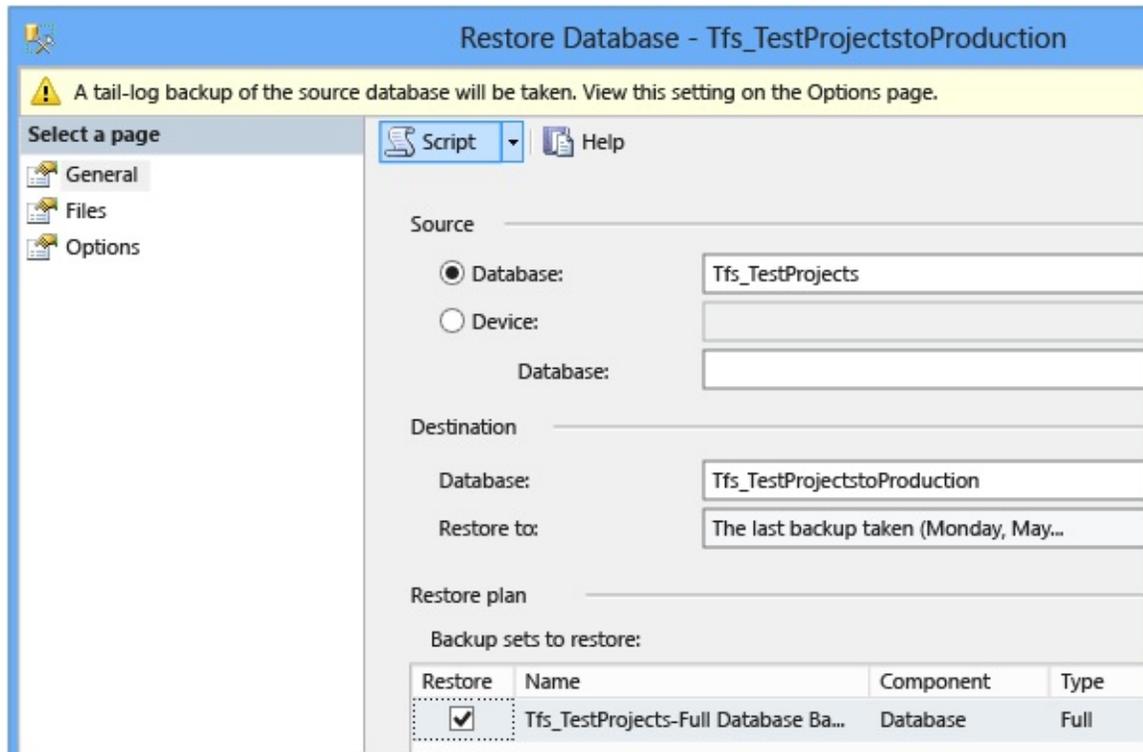
The steps below give a general overview of how to restore a team project collection database in SQL Server 2012 using SQL Server Management Studio. For more information about how to manually back up and restore individual databases, see the following page on the Microsoft Web site, and make sure to choose the version of SQL Server that matches your deployment:

[Backing Up and Restoring Databases in SQL Server.](#)

To restore the collection database with a new name

1. Open SQL Server Management Studio and connect to the instance that hosts the database for the team project collection that you want to split.
2. In Object Explorer, expand Databases, open the sub-menu for the database you want to split, and then choose Tasks, choose Restore, and then choose Database.

The Restore Database window opens on the General page.



3. In Source, make sure that the team project collection database is chosen. In Destination, provide a name for the copy of the database. Keep the Tfs_ prefix, but give it a distinct name after that prefix. Ideally that name will be the name of the split team project collection. In Restore plan, make sure that the backup sets to restore are the ones you want to restore to. To make sure that these are valid sets, choose Verify Backup Media and then, in Select a page, choose Options.
4. In Restore options, leave all the check boxes blank. Make sure that Recovery state is set to RESTORE WITH RECOVERY. In Tail-Log Backup, clear the Leave source database in the restoring state check box, and then choose OK.

Tip

If the restore operation fails with an error message indicating that the database is in use and cannot be overwritten, you might need to manually configure all the logical file names to reflect the new name for the database. In Select a page, choose Files, choose the ellipsis button next to each file being restored, and make sure that the names of the files reflect the new name for the database, not the old one. Then try the restore

operation again.

2-b. Attach the original collection database

After you have restored the database with a different name, you must reattach the original collection database to the deployment of TFS.

Note

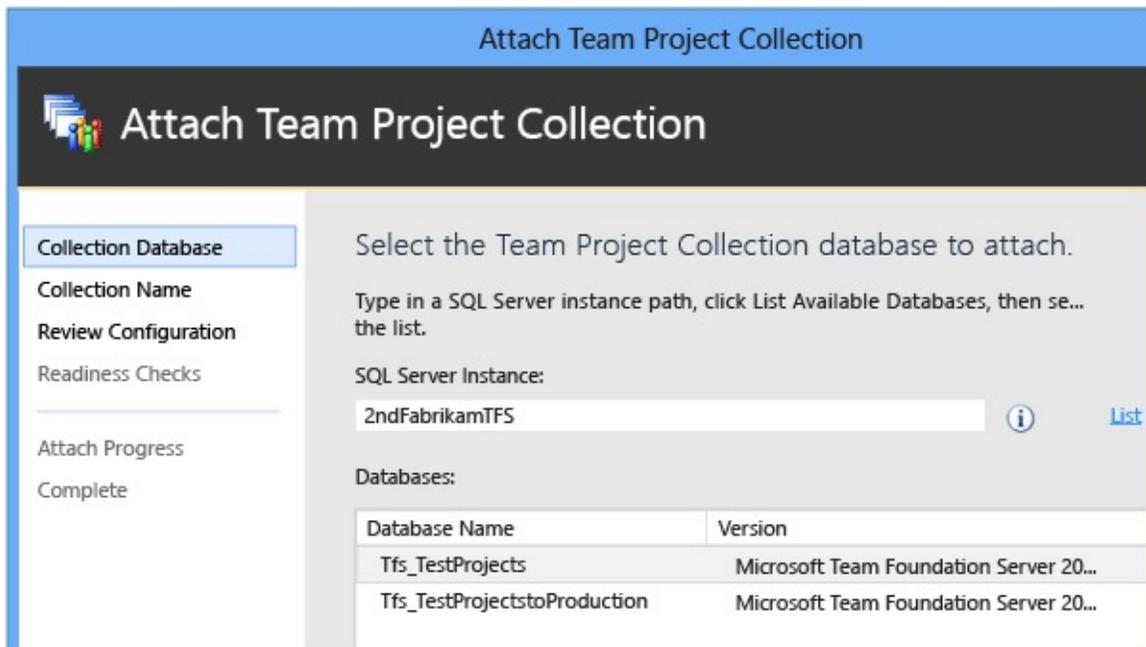
If your deployment uses SharePoint Products and the service account for TFS is not a member of the Farm Administrators group, warnings will appear when you attach the collection. This behavior is expected.

To attach the collection

1. Open the administration console for Team Foundation.
2. Choose Team Project Collections, and then choose Attach Collection.

The Attach Team Project Collection Wizard opens.

3. On the Select the team project collection database to attach page, in SQL Server Instance, provide the name of the server and the instance that hosts the collection database, if it is not already listed.
4. In the Databases list, choose the collection database that you want to attach.



5. On the Enter the team project collection information page, provide a name for the collection in Name if one is not already present. Since this is the original collection, you can choose to leave the name the same as it was before. In Description, optionally provide a description of the collection.
6. On the Review settings that will be used to attach the team project collection page, review the information.
7. If you must change any settings, choose Previous. If all the settings are correct, choose Verify.
8. When all the readiness checks have completed successfully, choose Attach.
9. On the Monitor the team project collection attach progress page, when all processes have completed, choose Next.
10. (Optional) On the Review supplemental information for this team project collection page, choose or note the location of the log file, and close the wizard.
11. The team project collection appears in the list of collections in the administration console. If the collection state is listed as Online, you must stop it before continuing. Choose the collection from the list, and on the General tab, choose Stop Collection.

Team Foundation Server Administration Console

Team Project Collections

Refresh Help

Name	State
DefaultCollection	Online
TestProjects	Online

Create Collection
Attach Collection

General Status Team Projects Reports Folder

TestProjects
URL: http://2ndfabrikamtf:8080/t...
Stop Collection
Edit Settings

2-c. Attach the renamed collection database

After you attach the original collection database, you must attach the renamed collection to the deployment of TFS. When this collection is attached, it will remain stopped. You will not be able to start it until all duplicate projects have been removed.

Note

Warnings will appear when you attach the collection if your deployment uses SharePoint Products and the service account for TFS is not a member of the Farm Administrators group. This behavior is expected.

To attach the renamed collection database

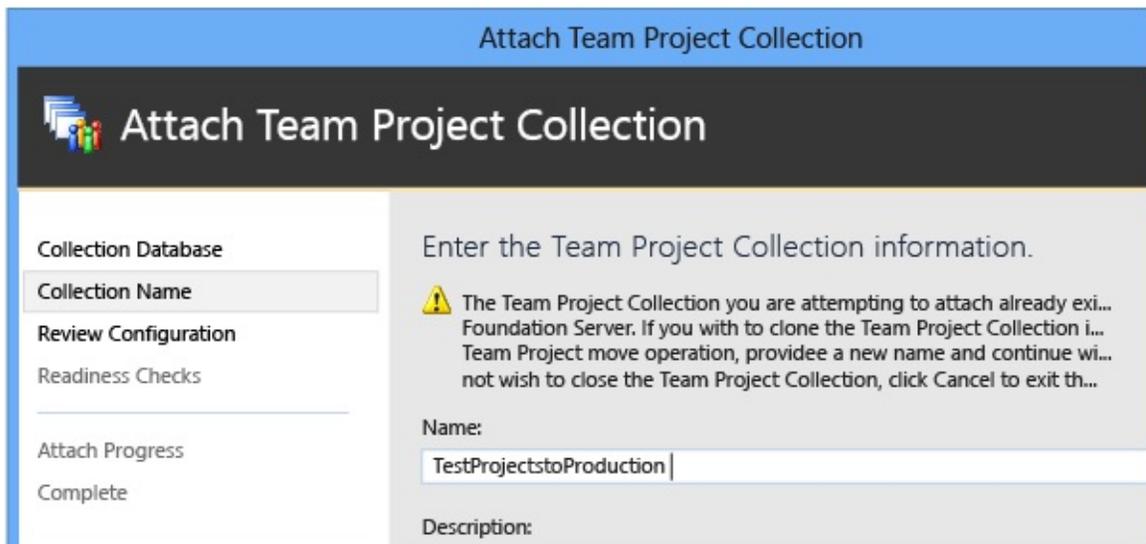
1. Open the administration console for Team Foundation.
2. Choose Team Project Collections, and then choose Attach Collection to open the wizard.
3. On the Select the team project collection database to attach page, in SQL Server Instance, provide the name of the server and the instance that hosts the renamed collection database, if it is not already listed.
4. In the Databases list, choose the renamed collection database.
5. On the Enter the team project collection information page, type a name for the renamed collection in Name that differs from the name of the original name of the collection. Ideally, this should match the name you gave the renamed database without the Tfs_ prefix.

6. (Optional) In Description, type a description of the collection.
7. On the Review settings that will be used to attach the team project collection page, review the information. If you must change any settings, choose Previous. If all the settings are correct, choose Verify.
8. When all the readiness checks have completed successfully, choose Attach.
9. On the Monitor the team project collection attach progress page, when all processes have completed, choose Next.

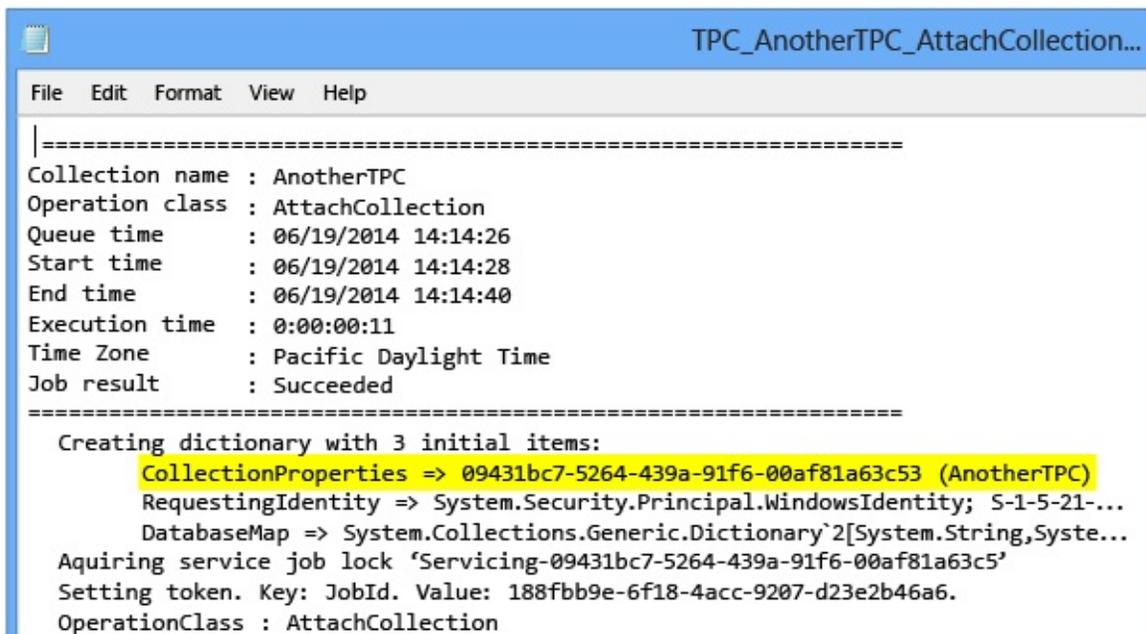
Note

If the collection is supported by a SharePoint Web application, a warning icon will appear for the attach status of the SharePoint Web application. Similarly, if the original collection included reporting, a warning icon will appear for the attach status for reports. This behavior is expected, and you can ignore it.

10. (Optional) On the Review supplemental information for this team project collection page, choose or note the location of the log file, and then close the wizard.
11. The name of the collection appears in the list of collections in the administration console, and its status should display as Offline.



- To ensure that both collections have attached with unique IDs, in the administration console, go to Event Logs and open the log files for both collection attach operations. The GUIDs for CollectionProperties should not match.



In the unlikely event that the CollectionProperties GUIDs do match, you must change the ID to a unique ID before continuing by running the TFSSConfig

[Collection command \[TFSSConfig\]](#) on the second collection with the /clone

parameter..

2-d. Delete projects on the split collections

Now that you have two copies of the collection attached to TFS, you must delete each project from either the original collection or the renamed collection so that no project remains in both collections.

Important

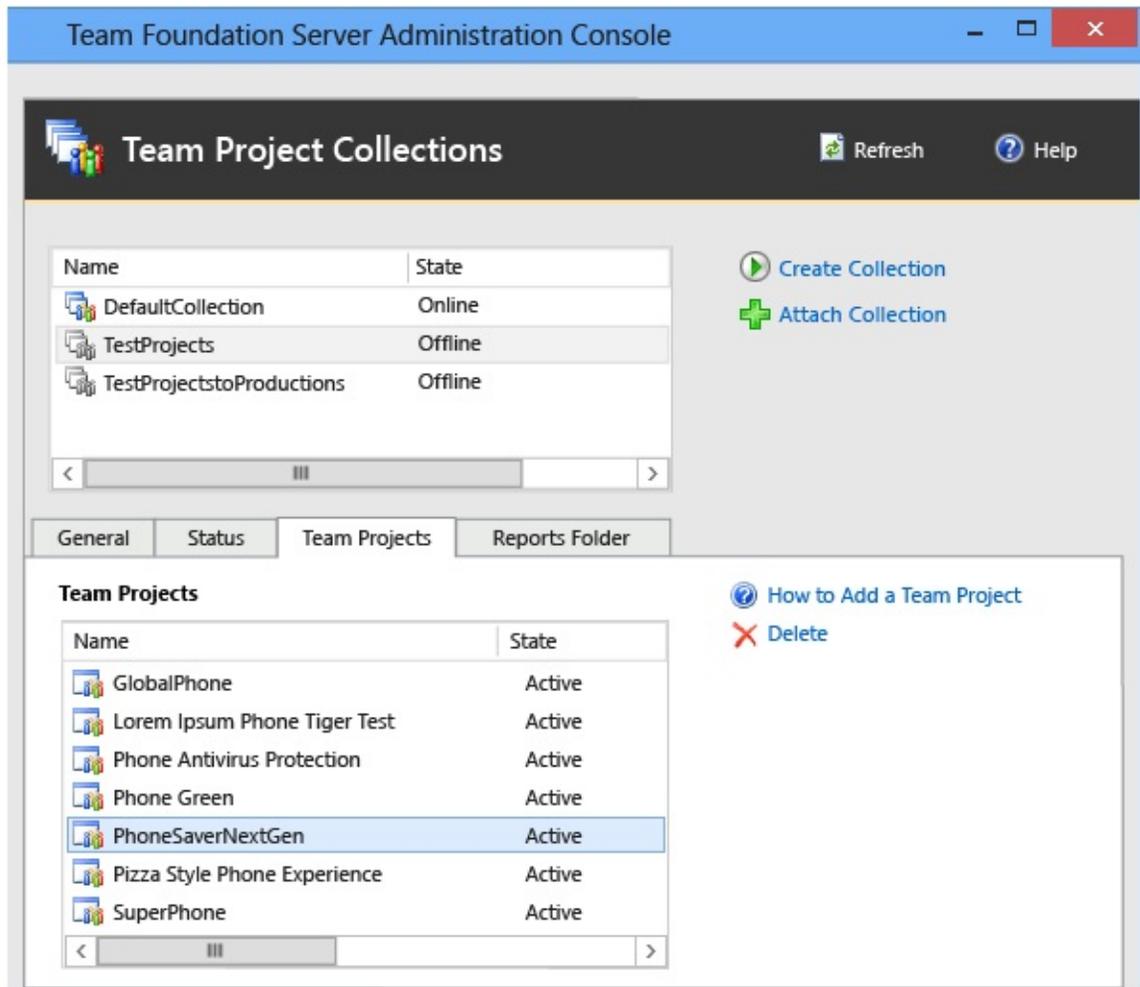
A project cannot exist in more than one collection. Until you delete all duplicated projects between the split collections, you will not be able to start the renamed collection.

To delete projects from the collections

1. Open the administration console for Team Foundation.
2. Choose Team Project Collections, and in the list of collections, choose the original team project collection that you stopped in order to split it.
3. On the Team Projects tab, in the list of team projects, choose a project that you want to delete from the collection, and then choose Delete.

Tip

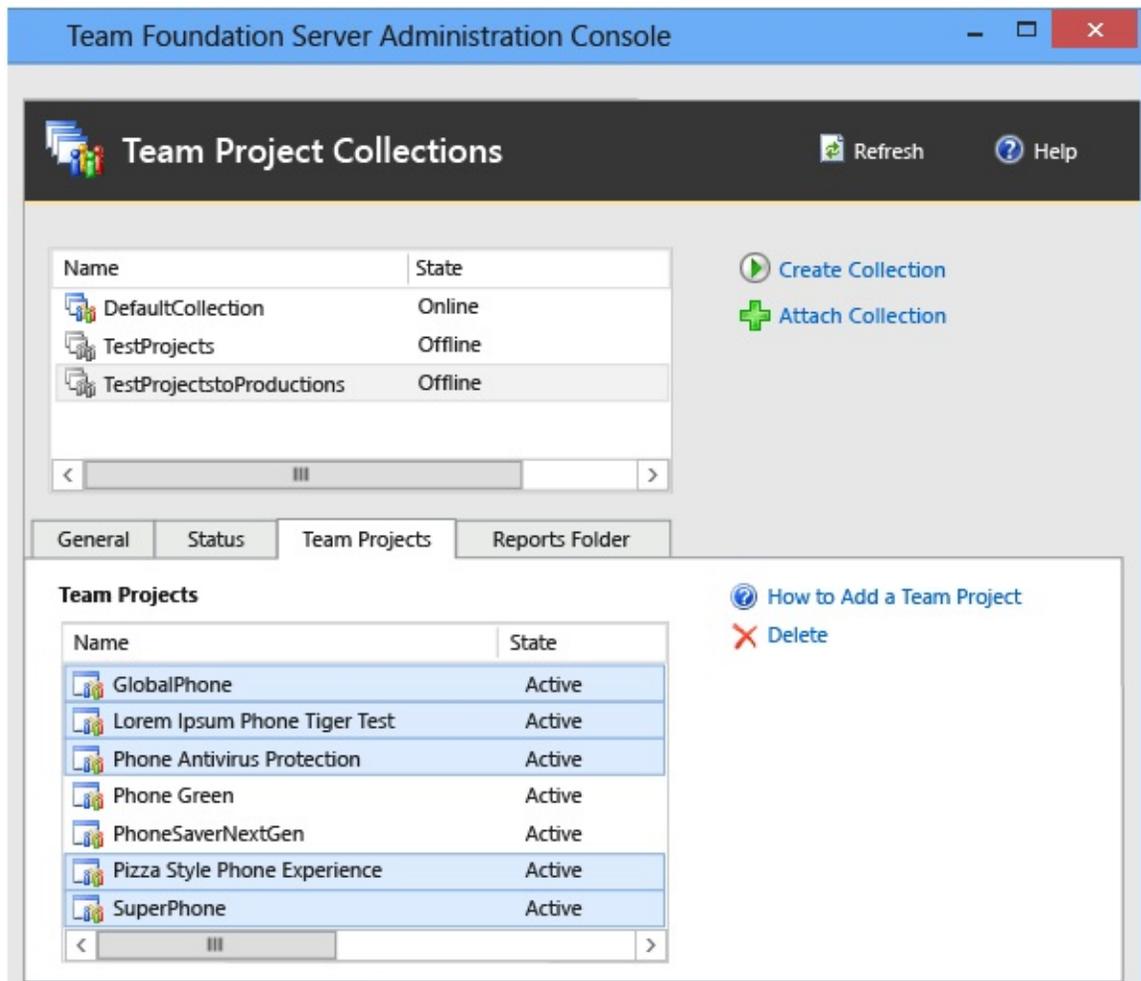
You can select more than one project to delete at a time.



4. Select the Delete workspace data check box, leave the Delete external artifacts check box cleared, and then choose Delete.

If the Delete external artifacts check box is not cleared and your team project is configured to use Lab Management, the virtual machines and templates that are associated with the project will be deleted from System Center Virtual Machine Manager. They will no longer be available to the team project in the renamed collection.

5. When you have finished deleting the projects you do not want hosted in the original team project collection, choose the renamed team project collection from the list of collections. Then, on the Team Projects tab, delete the projects you do not want hosted on the new collection.



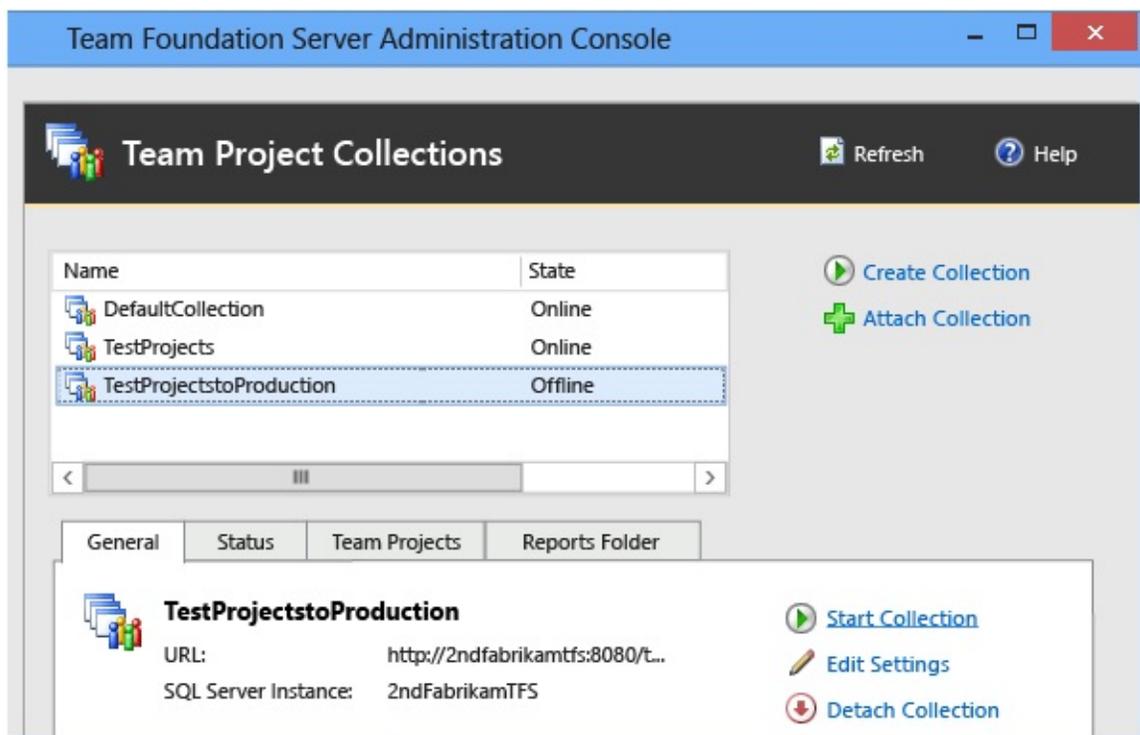
6. Repeat these steps until both collections contain a set of unique projects.

2-e. Start the team project collections

After you delete projects, you must restart both collections.

To start a team project collection

1. Open the administration console for Team Foundation.
2. Choose Team Project Collections, and in the list of collections, choose the collection that you stopped in order to split it.
3. On the General tab, choose Start Collection.
4. Repeat step 2 for the collection that you attached with a new name.



3-a. Configure users and groups for the split collections

You can skip this procedure if both split collections will remain in the same domain and you want to allow access for the administrators of the original collection to both collections.

After you have split a collection, you must update the permission groups for both collections with users and groups that will administer those collections.

To configure administrators for both collections

- For more information, see

[Set administrator permissions for team project collections.](#)

3-b. Configure users and groups for projects

You can skip this procedure if the split collections will remain in the same domain and you want to allow access for the users of projects in the original collection to both collections.

After you configure administrators for both collections, either you or those administrators must configure access for users and groups to the projects in each collection. Depending on your deployment, you might also need to configure permissions for those users in SharePoint Products and Reporting Services.

To configure access for users to team projects

- For more information, see

[Add users to team projects](#) and Add team members.

Q & A

Q: My deployment uses reporting. Are there any additional steps I need to take when splitting collections?

A: Yes, you'll need to split reports after you've finished deleting team projects so that both collections have a unique set of projects. You'll also need to rebuild your data warehouse.

After you delete projects, you must move the reports that the split collection uses into a different folder, and you must delete them from the original folder.

⚠️ Important

The report folders exist in both locations. Make sure that you move all reports appropriately before you delete any report folders.

To split reports into separate folders

1. In Report Manager, move the reports that support the split collection into the appropriate folders for that collection.

For more information, see the following topic on the Microsoft Web site:

[Move Items Page](#).

- If your deployment utilizes a SharePoint Web application, you might need to repair the connection again after you move the reports before they will appear correctly. If reports do not appear correctly, follow the steps in the previous procedure to repair the connection.

Once you've split the reports and started both collections, you must rebuild the warehouse for Team Foundation and the database for Analysis Services. You must perform this step to ensure that reports and dashboards work correctly for the deployment after you split the collection and that no conflicts occur with

other collections in the deployment.

To rebuild the data warehouse and the Analysis Services database

1. Open the administration console for Team Foundation.
2. In the navigation bar, choose Reporting.
3. In Reporting, choose Start Rebuild.
4. In the Rebuild the Warehouse and Analysis Services Databases dialog box, choose OK.

☑Note

The warehouses will continue to be rebuilt and the data will continue to be repopulated after the Start Rebuild action finishes. Depending on the size of your deployment and the amount of data, the whole process might take several hours to complete.

☐Q: Can I split a collection that uses SharePoint Products to support one or more team projects in the collection?

A: Yes, but you'll need to perform additional steps for the split collection.

After you attach the renamed collection and remove all duplicate projects, you must repair the connection to the SharePoint Web application. Repairing the connection ensures that all connections are correctly set between the Web application and the original and renamed collections.

If your deployment uses SharePoint Products, it is strongly recommended that the service account for TFS be a member of the Farm Administrators group.

☑Note

You can split a team project collection without granting this membership to the service account for TFS. However, you will see errors when you attach the collection, and you will need to perform additional steps to reconnect projects

with their portals. Even if your operational requirements generally restrict granting this membership to the service account, you should consider adding the service account to the Farm Administrators group for the duration of the split operation.

To repair the connection to a SharePoint Web application

1. Open the administration console for Team Foundation on the server that hosts the application tier for the deployment to which you want to move the collection.
2. Choose SharePoint Web Applications, and in the list of Web applications, choose the Web application that supports the collections that you just attached.

The Repair Connection button appears after you select a Web application in the list.

3. Choose Repair Connection, and in the Repair the connection to a SharePoint Web Application dialog box, choose Repair.
4. When the Status window reports Reconnect operation succeeded, choose Close. This might take a few minutes. In addition, you might see some errors as part of this process, since the two collections are still using the same SharePoint default site location for their team project portals. This is expected behavior.

After you have repaired the connection and started both collections, you must reconfigure the team project portals for projects in each collection so that those portals reflect the correct data for those projects.

To reconfigure team project portals

- Open Team Explorer, connect to each team project collection, and for each team project, configure the URL for the SharePoint site. For each team project, choose Settings, choose Portal Settings, and make sure that the Reports and dashboards refer to data for this team project check box is selected.

You can continue to use the same site collection in SharePoint Products to support both split collections. Team projects in both collections will use the same team project portals as before. All portals are hosted on the site collection that supported the original team project collection. However, this configuration not only complicates the one-to-one relationship between a team project collection and a site collection but also makes restoring your deployment potentially more difficult. To avoid this complexity, you can split the site collection that supported the original team project collection to reflect the split that you made for the team project collections.

To split the site collection and redirect the split team project collections to use the split site collections

1. For information about how to split a site collection, see [Move site collections between databases](#) or the latest guidance for your version of SharePoint Products.

Tip

Make sure that you configure user permissions and access to the site collections to match the user access to the team project collections, as detailed earlier in this topic.

2. Configure any affected team project collection to utilize the split site collection by opening the administration console, choosing the collection from the list of the team project collections, and on the SharePoint Site tab, choosing Edit Default Site Location.
3. Reconfigure the team project portals for team projects in each collection so that those portals reflect the correct data for those projects.

For more information, see Reconfigure Team Project Portals earlier in this topic.

Q: Are there any special considerations or actions to take if I split a team project collection that has projects integrated with Project Server?

A: After you split your team project collection, you must wait for the metadata to update as part of the synchronization between TFS and Microsoft Project Server. After synchronization, your data should appear correctly in Project Server. For more information, see [Change your deployment configuration](#).

To verify synchronization

- After you have split a collection, wait for the data to synchronize. This process happens automatically on a predetermined schedule. For more information, see [Synchronization process overview for TFS-Project Server integration](#).
- If one of the split collections will no longer synchronize with Project Server, unmap it by using the **TFSAdmin ProjectServer /UnmapPWAFromCollection** command with the **/force** option.

For more information, see [Map a team project collection to an instance of PWA](#), [Change your deployment configuration](#), and [Remove a component from participating in data synchronization](#).

You do not need to take any steps to continue synchronizing data between the collection and Project Server after you have split the collection unless you also plan to move the collection to a different server. In that case, you must follow the steps for moving a collection after you split the collection. For more information, see [Move a team project collection](#).

≡Q: How do I split a collection configured for Lab Management?

A: You'll need to perform several additional steps to split the collection. Before you start the split, you'll need to delete the Lab Management resources from the collection, and then, after the split, you'll have to individually configure Lab Management resources for each of the split collections.

Before you start the split, delete the resources that Lab Management uses from the collection database. These resources include virtual machines, templates, team project host groups, and team project library shares. You will need to re-create the Lab Management assets after you restore and attach the collection.

To delete the Lab Management resources

- For information about how to remove all group hosts, library shares, and environments from a specified team project collection, see [TFSSConfig Lab /Delete Command](#) with the /External option.

Once you've completed the split, you must recreate team project host groups. You must also recreate team project library shares in TFS and the virtual machines, templates, and environments in Microsoft Test Manager.

To configure Lab Management resources

1. Configure the application tier for Team Foundation.

For more information, see [Configure Lab Management for SCVMM environments](#).

2. Recreate the golden master virtual machines and templates in the new SCVMM and import virtual machines and templates into the team project collection.

For more information, see [Create and store virtual machines and templates ready for Lab Management](#).

3. Recreate the environments for each team project.

For more information, see [Creating an SCVMM Environment Using Stored Virtual Machines and Templates](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Delete a team project

[Send Feedback](#)

If you find that you have a team project you no longer use, you can delete it. Deleting a team project helps simplify the navigation to team projects that are in use.

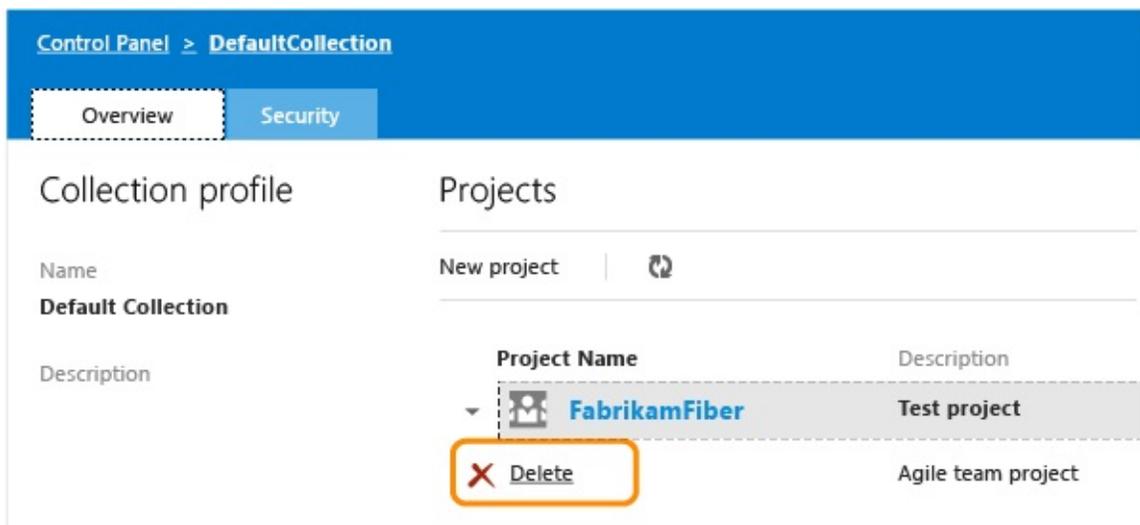
Caution

Deleting a team project permanently removes data associated with that project from the database. You cannot recover it later. Therefore, you should [save team project data](#) that you might want to access later.

You can delete a team project from Visual Studio Online or from an on-premises deployment of Team Foundation Server (TFS).

Delete a team project from Visual Studio Online

1. Open the administration context for the team project collection by choosing the ⚙ gear icon. Open the ▾ context menu for the team project that you want to delete.



If you don't see the context icon (▾), either you're not accessing Visual Studio Online or you're not a member of the Project Collection Administrators group. For on-premises team projects,

[go to the next section](#). Otherwise, go here to learn how to get added to the Project Collection Administrators group.

- You must select the check box in order to initiate the delete operation.

DELETE PROJECT

× Close



Warning: Please read the following carefully before proceeding!

All data in the team project "FabrikamFiber" will be permanently deleted. This includes any work items and source code.

Backup any data that is important first!! To learn more, see deleting a team project.

I understand that all data in the team project "Fabrikam Fiber" will be deleted.

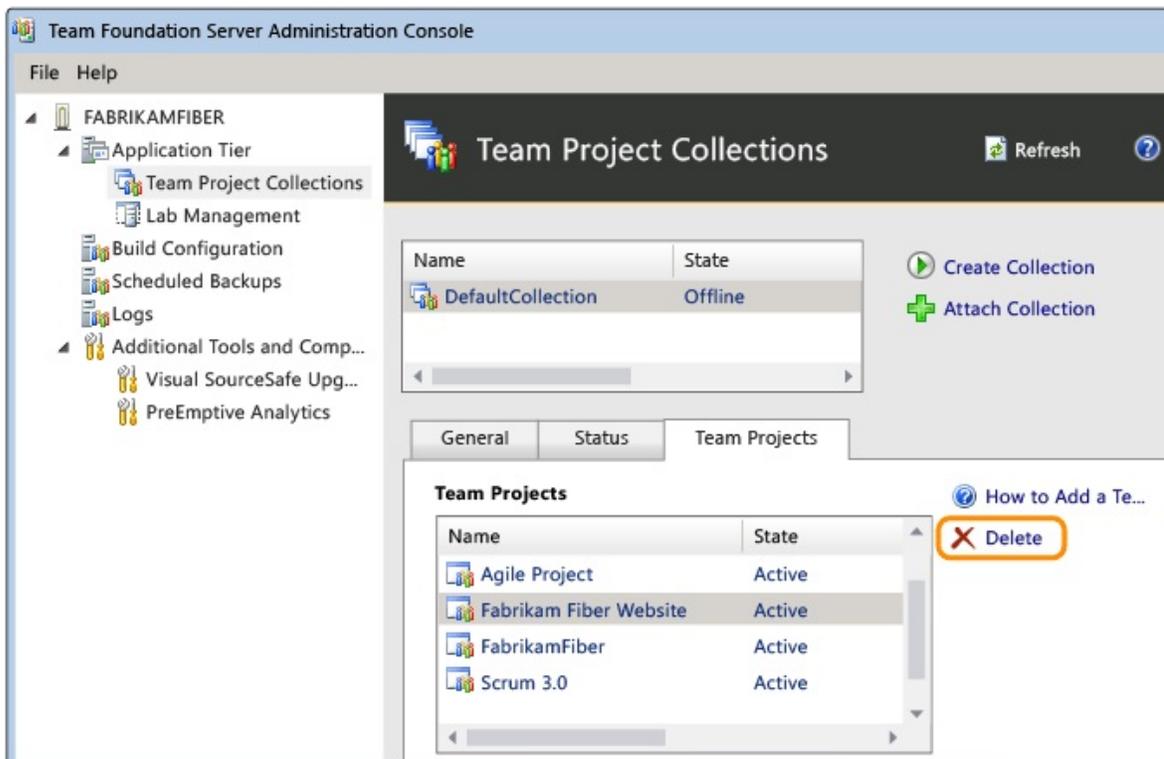
Delete Project

Cancel

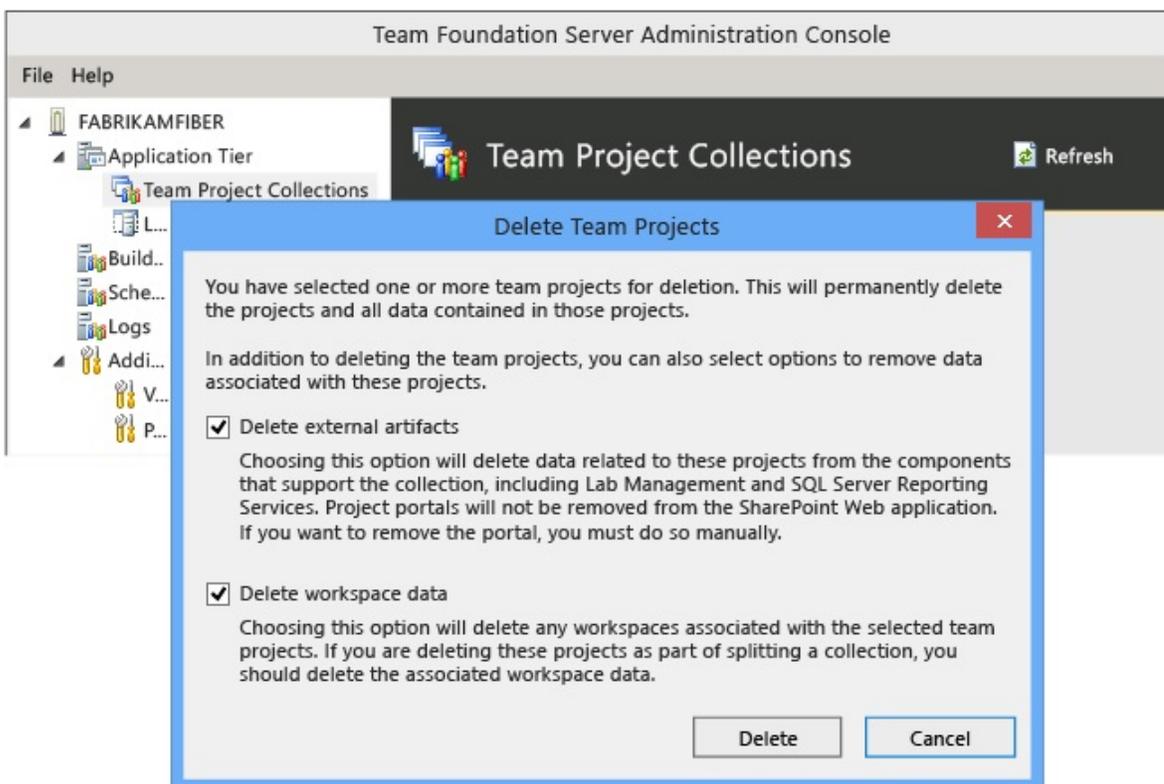
Delete a team project from Team Foundation Server (on-premises)

Using the administration console, you can delete a team project from a team project collection. Afterwards, you'll need to manually delete any associated reports and SharePoint project portal. Or, you can use the `TFSDelProject` command line tool to delete all artifacts.

1. If you're not a member of one or more of the following administrator groups, get those permissions now:
 - Team Foundation Administrators group (required).
 - SQL Server System Administrators group (required).
 - Farm Administrators group for SharePoint Products (required when your deployment uses SharePoint Products).
2. Open the administration console for TFS and delete the team project from its team project collection.



3. Choose whether to delete external data associated with the team project and then initiate the delete action.



4. (Optional) To review the status of the delete action, open the Status tab.

To review the details of the delete action, you can open the log file from either the Status tab or Logs tab.

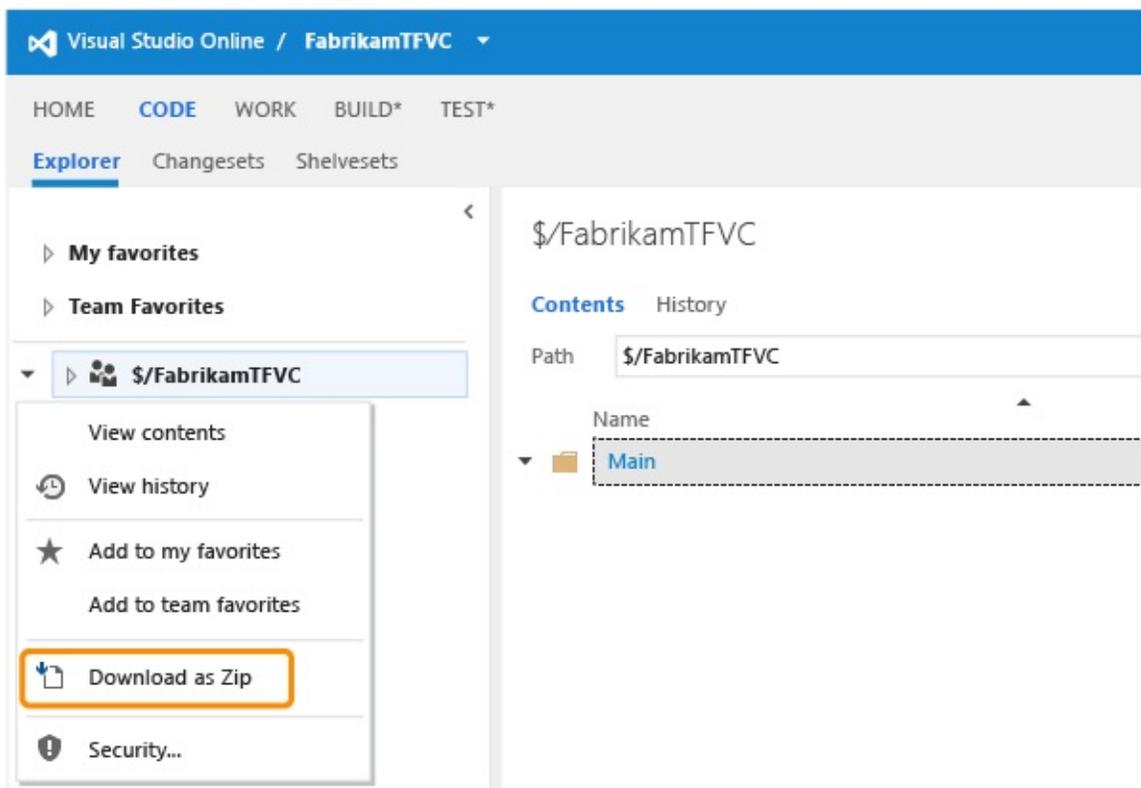
Save team project data

Make sure that any project and its data are not needed before you delete it. If you have any concerns, save data when there is a possibility that you will need later.

Save data stored on Visual Studio Online

You can use the following procedures to save data that users most care about, such as source code, build data, and work items.

- Source code and custom build templates: You can download your files as a zip file.



This process doesn't save any change history or links to other artifacts.

If you use Git, clone your repositories to retain the full project history and

all the branches.

- Build data: To save logs and data in your drop build folders, see View build results.
- Work item tracking data: Create a work item query and open it using Excel. Save the Excel spreadsheet.

This process doesn't save any attachments, change history, or links to other artifacts.

■ **Save data stored on Team Foundation Server**

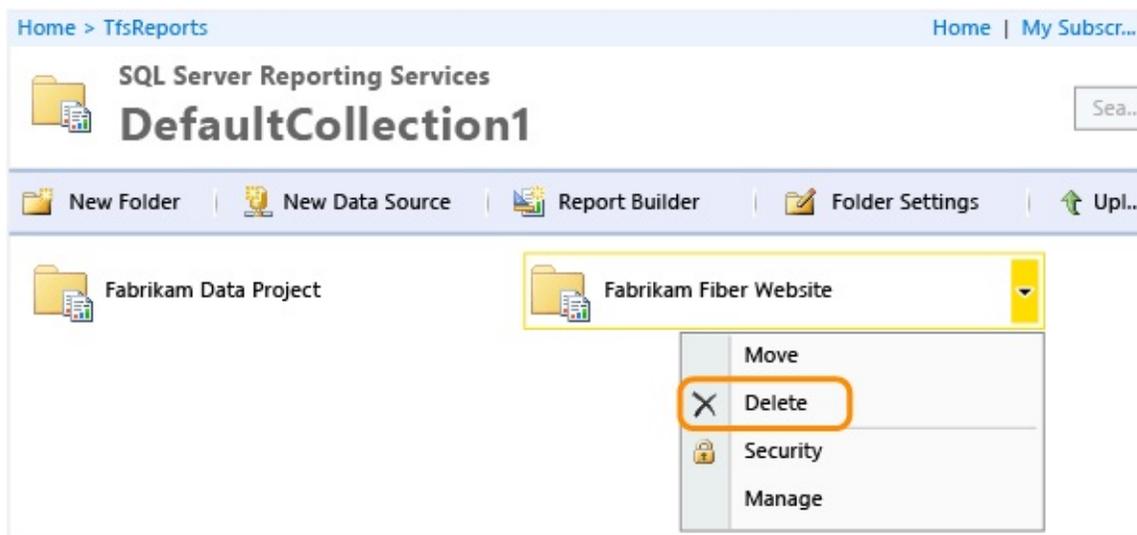
You can easily save data stored for a team project collection by making a backup of the database. Or, you can use the same operations described earlier in

[Save data stored on Visual Studio Online.](#)

Q & A (for on-premises deployments only)

Q: How do I manually delete reports that remain after the project is deleted?

A: If your on-premises team project used reporting, and you didn't choose to delete external artifacts, you can delete the reports using SQL Server Report Manager. From the team project collection page, delete the folder that corresponds to the deleted team project.



Q: How do I remove the team project portal?

A: If your on-premises team project had a team project portal, all links to that portal will be removed from TWA and Team Explorer, but the SharePoint site or website that acted as the portal will not be deleted. If you want to delete the portal, you must do so manually after the project has been deleted. See [How to: Create, Edit, and Delete Windows SharePoint Services Sites](#).

Q: What should I do if the delete action doesn't finish?

A: For on-premises deployments, review the status and log files for the delete action. Open the Status tab and for Deleted, review the additional information in parentheses, and take the indicated action.

- (Processing) means that the process has started and is in progress.
- (Pending) means that the deletion process has started from a client application. The deletion might be in progress or might have failed. Because the process was started from a client application, the server cannot accurately report the status of the deletion.

If a project deletion remains pending for a long time, try to delete the project again from the administration console.

- (Failed) means that the deletion process started but did not successfully finish. The log file will contain specific information about the failure.

Review the information about the failure, and then try to delete the project again.

If partial data remains, you can also use the `TFSDelProject` command line tool.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Stop and start services, application pools, and websites

[Send Feedback](#)

For Visual Studio Team Foundation Server (TFS) to operate correctly, all required services, application pools, and Web sites must be running on the appropriate server. In single-server deployments, each component must run on the server that runs TFS. In multiple-server deployments, each component must run on the appropriate server. In addition, you might need to stop an element to perform a particular task, such as moving your deployment to a different set of hardware.

For operations such as backing up or restoring databases, you can run the [TFSServiceControl Command](#) to start or stop all TFS services and application pools.

■ Stop or start a service, application pool, or Web site

1. If you're not a member of the Administrators group on the server that hosts the service, application pool, or Web site that you want to manage, get added now. See [Set administrator permissions for Team Foundation Server](#).
2. Log on to the server that hosts the service, application pool, or Web site.
3. Open Computer Management.
4. In the navigation pane, expand Services and Applications.
5. Perform one of the following steps based on what element you want to stop or start:
 - For a service, open the navigation menu for the service, and then choose Stop or Start.
 - For an application pool, open Internet Information Services (IIS) Manager, expand the local computer and open Application Pools. Open the navigation menu and choose Stop or Start.
 - For a web site, open Internet Information Services (IIS) Manager, expand the local computer, and open Web Sites or Sites. Open the navigation menu and then choose Stop or Start.

Location of services, application pools, and web sites

The following table lists the server on which each service, application pool, and web site must be running. The Name column lists the display name for each element with service names in parentheses. Which services you need will vary based on which features of Team Foundation you have installed.

Element	Location	Name
Services	Application-tier server	<ul style="list-style-type: none">• Code Coverage Analysis Service• Internet Information Services Administration Service (IISADMIN)• HTTP SSL (HTTPFilter)• Visual Studio Team Foundation Build (VSTFBUILD) (only when Team Foundation Build is installed)• Visual Studio Team Foundation Background Job Agent (TFSJobAgent)• World Wide Web Publishing Service (W3SVC)

Server that hosts the databases for Team Foundation

- SQL Server (TFSINSTANCE)
- SQL Server Agent (TFSINSTANCE) (SQLSERVERAGENT)

Server that hosts SQL Server Reporting Services

- IIS Admin Service (IISADMIN)
- HTTP SSL (HTTPFilter)
- SQL Server Reporting Services (TFSINSTANCE) (ReportServer)
- World Wide Web Publishing Service (W3SVC)

Server that hosts SQL Server Analysis Services

- SQL Server Analysis Services

Server that hosts

- Internet Information Services Administration (IISADMIN)
- HTTP SSL (HTTPFilter)

	SharePoint Products	<ul style="list-style-type: none"> • Windows SharePoint Services Timer (SPTimer) • World Wide Web Publishing Service (W3SVC) • Microsoft Team Foundation Server Application Pool
Application pools	Application-tier server	<ul style="list-style-type: none"> • Microsoft Team Foundation Server Proxy Application Pool (only when Team Foundation Server Proxy is installed) • DefaultAppPool (used by the Team Project portal)
	Server that hosts SharePoint Products	<p> Note</p> <p>The name might vary based on how SharePoint Products was installed.</p> <ul style="list-style-type: none"> • SharePoint Central Administration v3 • Team Foundation

Web sites

Application-
tier server

Server

- Team Foundation Server Proxy (only if Team Foundation Server Proxy is installed)
- Default Web Site or Team Web site

Note

Server that
hosts
SharePoint
Products

The name might vary based on how SharePoint Products was installed.

- SharePoint Central Administration v3

Q & A

Q: Which service account supports each service?

A: See [Service accounts and dependencies in Team Foundation Server](#).

Q: Are there additional services that TFS supports?

A: Yes, TFS includes a set of Web services and application-level services See [Team Foundation Server architecture](#).

Q: What services depend on service accounts?

A: See [Service accounts and dependencies in Team Foundation Server](#).

Q: How do I change the TFS service account or password?

A: See [Change the service account or password for Team Foundation Server](#).

Q: How do I change the service account or password for SQL Server Reporting Service?

A: See [Change the service account or password for SQL Server Reporting Services](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Service accounts and dependencies in Team Foundation Server

[Send Feedback](#)

You can better manage Visual Studio Team Foundation Server (TFS) if you understand the services and several service accounts that every deployment of TFS includes and on which every deployment depends. Depending on how you have installed and configured TFS, these services and service accounts might all run on one computer, or they might run on many computers. This changes certain aspects of managing your deployment. For example, if the server-side components of your deployment run on more than one computer, you must ensure that the service accounts your deployment uses have the access and permissions they require to function correctly.

Team Foundation Server has services and service accounts that run on the following computers in a deployment:

- any server that hosts one or more databases for Team Foundation Server
- any server that hosts components of the application tier for Team Foundation
- any computer that is running Team Foundation Server Proxy
- any build computer
- any test machine
- any computer that is running one or more components of Visual Studio Lab Management

You can install and deploy different features of TFS in various ways. The distribution of features in your deployment determines what services and service accounts run on which physical computers. In addition, you might need to

manage the service accounts for software programs that are configured to work with TFS, such as the service accounts for SharePoint Products and SQL Server.

Service accounts for Team Foundation Server

Although TFS uses several service accounts, you can use the same domain or workgroup account for most or all of them. For example, you can use the same domain account "Contoso\Example" as both the service account for Team Foundation Server (TFSService) and the data sources account for SQL Server Reporting Services (TFSReports). However, different service accounts can require different permission levels. For example, TFSService must have the Log on as a service permission, and TFSReports must have the Allow log on locally permission. If you use the same account "Contoso\Example" for both, you must grant both of these permissions to it. In addition, TFSService requires significantly more permissions to operate correctly than those that TFSReports requires, as the table later in this topic shows. For security purposes, you should consider using separate accounts for these two service accounts.

Important

You must not use the account that was used to install Team Foundation Server as the account for either of these service accounts.

If you have deployed Team Foundation Server in an Active Directory domain, you should set the Account is sensitive and cannot be delegated option for service accounts. For example, in the following table, you should set that option for TFSService. For more information about required service accounts and placeholder names used in documentation for Team Foundation Server, see the topic "Accounts required for installation of Team Foundation Server" in the installation guide for Team Foundation. For more information about how to restrict account delegation in Active Directory, see the following page on the Microsoft Web site:

[Enabling Delegated Authentication.](#)

Because you must manage several service accounts, each service account is

referred to by a placeholder name that identifies its function, as listed in the table later in this topic. The placeholder name is not the actual name of the account that you use for each service account. The actual name of the account varies depending on your deployment. In the previous example, the account used for both TFSService and TFSReports was "Contoso\Example." In your own deployment, you might create domain accounts with the specific names of "TFSService" and "TFSReports," or you might use the system account Network Service as the service account for Team Foundation Server.

⚠ Important

Unless specifically stated otherwise, no groups or accounts in the following table should be members of the Administrators group on any of the servers in your deployment of Team Foundation Server.

The following table shows all the service accounts that you might use in a deployment of TFS:

Service account	Placeholder name and usable account type	Required permission and group membership	Notes
Service account for Team Foundation Server	TFSService, which can be a local account, a domain account, Local Service in a workgroup, or Network Service in a domain	<ul style="list-style-type: none"> • Log on as a service on the application-tier server • Farm Administrators group for any SharePoint Web applications that Team Foundation Server uses¹ • TFSExecRole, or if this role does not exist for the database, a combination of the following roles for any databases that Team Foundation Server uses: 	This service account is used for all of the Web services for Team Foundation Server. If you use a domain account for this account, must be a member of a domain that contains the computers

		<ul style="list-style-type: none"> ◦ db_owner ◦ db_create 	throughout the deployment fully trust.
Data sources account for SQL Server Reporting Services	TFSReports, which can be a local account, a domain account, or Local Service in a workgroup	<ul style="list-style-type: none"> • Allow log on locally on the application-tier server and on the server that is running SQL Server Reporting Services • TFSWarehouseDataReader on the report server 	This service account retrieves data for reports from Reporting Services.
Service account for Team Foundation Build	TFSBuild, which can be a local account, a domain account, or Local Service in a workgroup	Log on as a service	This service account is used when builds are configured and when build status information is communicated between the build controller and the build agents.
Service account for Lab Management	TFSLab, which can be a local account, a domain account, or Local Service in a workgroup, or	Log on as a service	This service account is used when information about Lab Management is communicated between Team Foundation

	Network Service in a domain		Server and the lab agent that is running on virtual machine.
Service account for Team Foundation Server Proxy	TFSPProxy, which can be a local account, a domain account, Local Service in a workgroup, or Network Service in a domain	Log on as a service	This service account is used for all of the proxy services. If you use a domain account for this account, must be a member of a domain that all computers throughout the deployment fully trust.
Service account for Test Agent and Test Agent Controller	TFSTest, which can be a local account, a domain account, or Network Service in a domain.	Log on as a service	This service account is used when information about tests is communicated between the test agent controller and the test agent.
			You must add at least one service

<p>Service accounts for SharePoint Web applications</p>	<p>WebAppService</p>	<p>Allow log on locally</p>	<p>account for each SharePoint Web application that you configure for use with Team Foundation Server. This service account is used to create team project portals and to enable dashboard functionality.</p>
<p>Service account for Visual Studio Online</p>	<p>Account Service (CollectionName)</p>	<p>None. This account is only used if you are using a hosted deployment of TFS. It is automatically created for you, and can be viewed through the administration page of Team Web Access.</p>	<p>This service account is created automatically when you create a collection in Visual Studio Online, and is used when clients communicate with the host service.</p>

¹ You can integrate your deployment with SharePoint Products without this permission, but you must perform additional steps if the service account is not a member of the Farm Administrators group. For more information, see [Integrate with SharePoint Products Without Administrative Permissions](#).

Services that run under service accounts

The following services run under service accounts in a deployment of Team Foundation Server:

Service name	Service account	Logical Tier
Code Coverage Service	TFSService	application tier
Team Foundation Server Web Services	TFSService	application tier
SQL Server Reporting Services (MSSQLSERVER or InstanceName if using a named instance)	Local System or a domain account	application tier
Report Web Service	Local System, Network Service, or a domain account	application tier

SharePoint Administration (if SharePoint Products is installed and configured for use with Team Foundation Server) Local System, Network Service, or a domain account application tier

SharePoint Timer (if SharePoint Products is installed and configured for use with Team Foundation Server) Domain account application tier

Visual Studio Team Foundation Build Service Host (if Team Foundation Build is installed) TFSSBuild build computer

Visual Studio Team Foundation Background Job Agent TFSService application tier

Visual Studio Test Controller TFSTest any computer

Visual Studio Test Agent	TFSTest	test computer
--------------------------	---------	---------------

Analysis Server (MSSQLSERVER or InstanceName if you are using a named instance)	Local System or a domain account	data tier
---	----------------------------------	-----------

SQL Server Browser	Local Service or a domain account	data tier
--------------------	-----------------------------------	-----------

SQL Server (MSSQLSERVER or InstanceName if using a named instance)	Local System, Network Service, or a domain account	data tier
--	--	-----------

SQL Server Agent (MSSQLSERVER or InstanceName if using a named instance)	Local System, Network Service, or a domain account	data tier
--	--	-----------

Account Service (CollectionName)	Automatic	web tier (Visual Studio Online only)
----------------------------------	-----------	--------------------------------------

|

For more information about service accounts for SQL Server, see the following page on the Microsoft Web site:

[SQL Server Books Online](#). For the most recent information about service accounts in Team Foundation, see Team Foundation Server install guide.

Note

If you change the service account for Team Foundation Build, you must make sure that the new service account is a member of the Build Services group. You must also make sure that the account has read/write permissions to the temporary folders and the ASP.NET temporary folder. Similarly, if you change the service account for the Team Foundation Server Proxy service, you must make sure that the account is a member of the appropriate groups. For more information, see [Setting Up a Build Computer](#).

Q & A

Q: Are service accounts assigned to an access level group?

A: By default service accounts are added to the default access level. If you make Stakeholder the default access level, you must add the TFS service account to the Basic or Advanced group.

Q: Do service accounts require a license?

A: No. Service accounts don't require a separate license.

Q: How do I change the password or account for a service account?

A: See the following topics:

- [Change the service account or password for Team Foundation Server](#)
 - [Change the service account or password for SQL Server Reporting Services](#)
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Change the service account or password for Team Foundation Server

[See Also](#) [Send Feedback](#)

You can help improve the security of Team Foundation Server (TFS) by changing its service account or the password used for that account. TFS runs services such as its Web services and the Team Foundation Background Job Agent in the context of a service account.

For simplicity, the TFS documentation refers to this account as TFSService, although that is not the actual name of the account unless you specifically create an account with that name. TFS stores a record of the name of the actual account that is used as its service account. By changing the record, you can assign a different account to act as the service account. You can also change the password for that account. Whether you change the account, the password, or both, you stay synchronized with other components in your deployment. For example, if an Active Directory domain policy requires that all passwords expire periodically, you can update the password information for the service account in TFS when that password changes.

Note

TFS and its utilities cannot create a new local or domain account to use as TFSService, and they cannot update the password for that account in the workgroup or the domain. Instead, the utilities update the records to match the new credentials. If your deployment includes more than one application-tier server, you must manually update each server with any changes to the service account or its password.

For more information about TFS service accounts, see [Service accounts and dependencies in Team Foundation Server](#) or Accounts required for installation of

Team Foundation Server.

Requirements

- To perform these procedures, you must be a member of the Administrators group on the Team Foundation application-tier server and a member of the sysadmin group on the server and instance of SQL Server that hosts the TFS configuration database. For more information, see [Team Foundation Server architecture](#) and [Permission reference for Team Foundation Server](#).
- To follow a command-line procedure, you might need to open an elevated Command Prompt window. To perform this task, open the context menu for Command Prompt, and choose Run as Administrator. For more information, see this page on the Microsoft Web site: [User Account Control](#).

Change the password of the service account

You must log on to the TFS application-tier server and either use the administration console for Team Foundation, or open a Command Prompt window and use the **TFSConfig** command-line tool. If your deployment includes more than one application-tier server, you must perform this task on each server, or the account information will be out of synchronization.

Note

Depending on your deployment configuration, you might need to restart Internet Information Services (IIS) after you complete the procedure before the changes will take effect.

To use the administration console to change the password

1. Open the administration console for Team Foundation on the server that hosts the application tier.

For more information, see [How to: Open the Team Foundation Administration Console](#).

2. In the console, expand the server name and choose Application Tier.
3. In the Application Tier pane, choose Update Account Password.

The Update Account Password window opens.

Note

If you used a system account as the service account, you will see an error message when you choose Update Account Password. You do not need to change the password of that account. System accounts do not have user-

managed passwords.

4. Type the new password in Password, and then choose OK.

The Change Service Account window opens.

5. Wait for all the status messages to complete in Status, and then choose Close.

Note

This process might take a few minutes.

To use TFSSConfig to change the password

1. On the application-tier server, open a Command Prompt window and change directories to the directory that contains the **TFSSConfig** utility.

By default, this utility is located in Drive:\Program Files\Microsoft Team Foundation Server 12.0\Tools.

2. At the command line, type **TFSSConfig Accounts /UpdatePassword /accountType:ApplicationTier /account:AccountName /password:NewPassword**, and then choose ENTER.
3. You must specify both the name of the account you want to use as TFSService (AccountName) and the password of the account (NewPassword).

Assign a different account as the service account

You can use either the administration console or the **TFSConfig** command-line tool. If your deployment includes more than one application-tier server, you must perform this task on each server, or the account information will be out of synchronization. Before you use either tool to make the change, consider these issues:

- You must choose a new account that is either a system account or a member of a workgroup or domain that is trusted by every computer in this deployment of TFS.
- The configuration utilities grant the Log on as a service permission to the new service account, but the utilities do not revoke this permission from the account previously used as the service account if another service still uses that account. If the old account no longer needs that permission for the service for which it is still in use, you might want to manually remove that permission from the old account.

For more information, see this page on the Microsoft Web site: [Add the Log on as a service right to an account](#).

- You might need to restart IIS after you complete the procedure before the changes will take effect.
- The **TFSConfig** utility changes only those services that run under the old account.

To use the administration console to change the service account

1. Open the administration console for Team Foundation on the server that hosts the application tier.
2. In the console, expand the server name and choose Application Tier.

3. In the Application Tier pane, choose Change Account.

The Update Service Account window opens.

4. Perform one of the following steps:

1. If you want to use a system account, choose Use a system account, and then choose a system account from the drop-down list.

If your server is a member of an Active Directory domain, the default choice for the system account to use is Network Service. If your server is a member of a workgroup, the default choice is Local Service. Depending on the details of your deployment, the default choice might be the only available choice.

Note

System accounts do not have user-managed passwords. If you choose to use a system account as TFSService, you should not type a password in the password field.

2. If you want to use a domain or workgroup account, choose Use a user account, type the name of the account in Account Name, and then type the password for that account in Password.

5. Choose OK.

The Change Service Account window opens.

6. Wait for all the status messages to complete in Status, and then choose Close.

Note

This process might take a few minutes.

To use TFSSConfig to change the service account

1. On the application-tier server, open a Command Prompt window and change directories to the directory that contains the **TFSTConfig** command-line tool.

By default, this tool is located in Drive:\Program Files\Microsoft Team Foundation Server 12.0\Tools.

2. At the command line, type **TFSTConfig Accounts /change /accountType:ApplicationTier /account:AccountName /password:NewPassword**, and then choose ENTER.

For more information, see [Accounts Command](#).

Q & A

Q: How do I change my account password that I use to log into TFS?

A: User accounts for logging into an on-premises TFS deployment are managed by Active Directory or a Windows group. You change your password through the user interfaces provided by those respective systems.

See Also

Tasks

[Change the service account or password for SQL Server Reporting Services](#)

Reference

[Accounts Command](#)

Concepts

[Service accounts and dependencies in Team Foundation Server](#)

Other Resources

[TFSSConfig: Manage TFS server configuration](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Change the service account or password for SQL Server Reporting Services

[See Also](#) [Send Feedback](#)

You can help improve the security of Team Foundation Server (TFS) by changing the service account that it uses for the data sources for SQL Server Reporting Services or by changing the password that is used for that account. TFS acts in the security context of a service account when it retrieves project data from the data sources in SQL Server Reporting Services. TFS documentation refers to this service account by the placeholder TFSReports. The actual account name depends on your installation. You might need to change the password of that account, or designate a different account. For example, if the password of the underlying account expires, and you assign a new password, you must change the password of the TFSReports account in TFS to match.

The **TFSConfig** utility does not create a new account to use as the data sources account, nor does the utility change the account password. Instead, the utility updates TFS to use a different set of credentials.

You change the password or account used as the TFSReports account by using the TFSConfig command-line utility with the Accounts option.

Important

The **TFSConfig** utility changes only those services that run under the old account.

You can use the same utility to assign a different account to be the TFSReports account, but you might need to perform one or more of the following additional actions:

- Before you assign an account to use as the TFSReports account, you must verify that it is a member of a workgroup or domain that is trusted by every computer in the deployment of Team Foundation.
- You must manually grant the account that you will use as the TFSReports account the Allow log on locally permission. The TFSService utility does not grant this permission when it assigns the account.
- Optionally, after you use TFSService to specify an account to use as the TFSReports account, you can revoke its Log on as a service permission, which TFSService automatically grants to the TFSReports account. TFSReports does not need this permission, but the TFSService account does. Therefore, you should not remove this permission if you use the same domain or workgroup account for both service accounts.

For more information about the Log on as a service permission, see this page on the Microsoft website: [Add the Log on as a service right to an account](#). For more information about the Allow log on locally permission, see this page on the Microsoft website: [Allow log on locally](#).

For more information about required service accounts, see the topic [Service accounts and dependencies in Team Foundation Server](#) and also the topic "Accounts required for installation of Team Foundation Server" in the installation guide for Team Foundation.

Requirements

To perform these procedures, you must be a member of the Administrators group on the server where TFSService is installed. You must also be a member of the sysadmin group on the server that hosts the configuration database. For more information about permissions, see [Permission reference for Team Foundation Server](#).

In addition to these permissions, you might need to address the following requirements:

- To follow a command-line procedure, you might need to open an elevated Command Prompt.
- To access Report Manager, reports, or websites for SQL Server Reporting

Services, you might need to add these sites to the list of trusted sites in Internet Explorer or start Internet Explorer as an administrator.

Change the Reporting Services account name or password

To change the password of the TFSReports account or to assign a different account, you must log on to a server that hosts the application services for Team Foundation and use the **TfsConfig Accounts** utility.

Note

Depending on your deployment configuration, you might need to restart Internet Information Services (IIS) after you complete this procedure for the changes to take effect.

To change the password using the TFSSConfig utility

1. Open a Command Prompt window and change to the directory that contains the **TFSSConfig** utility.

By default, the utility is located in Drive:\Program Files\Microsoft Team Foundation Server 12.0\Tools.

2. At the command line, type **TFSSConfig Accounts /UpdatePassword /accountType:ReportingDatasource /account:AccountName /password:newPassword**, and then press ENTER.

Replace AccountName with the name of the current TFSReports account. Replace newPassword with the new password of the account.

To use the administration console to change the password

1. Open the administration console for Team Foundation on the server that hosts the application tier.

For more information, see [Configure and manage TFS resources](#).

2. In the console, expand the server name and choose Application Tier.
3. In the Application Tier pane, navigate to Reporting Services Summary choose Update Account Password.

The Update Account Password window opens.

 **Note**

If you used a system account as the service account, you will see an error message when you choose Update Account Password. You do not need to change the password of that account. System accounts do not have user-managed passwords.

4. Type the new password in Password, and then choose OK.

The Change Report Reader Account window opens.

5. Wait for all the status messages to complete in Status, and then choose Close.

 **Note**

This process might take a few minutes.

To assign a new reporting services service account to all Team Foundation Server services using the TFSSConfig utility

1. Open a Command Prompt window and change to the directory that contains the **TFSSConfig** utility.

By default, the utility is located in Drive:\Program Files\Microsoft Team Foundation Server 12.0\Tools.

2. At the command line, type **TFSSConfig Accounts /change /accountType:ReportingDatasource /account:NewAccountName /password:newPassword**, and then press ENTER.

Replace NewAccountName with the name of the new TFSReports account.
Replace newPassword with the password of the account.

To use the administration console to change the account

1. Open the administration console for Team Foundation on the server that hosts the application tier.
2. In the console, expand the server name and choose Application Tier.
3. In the Application Tier pane, navigate to Reporting Services Summary, and then choose Change Account.

The Change Report Reader Account window opens.

4. Choose one of the following steps:
 1. If you want to use a system account, choose Use a system account, and then choose a system account from the drop-down list.

Note

System accounts do not have user-managed passwords. If you choose to use a system account as TFSReports, you should not type a password in the password field.

2. If you want to use a domain or workgroup account, choose Use a user account, type the name of the account in Account Name, and then type the password for that account in Password.
5. Choose OK.

The Change Report Reader Account window opens.

6. Wait for all the status messages to complete in Status, and then choose Close.

Note

This process might take a few minutes.

See Also

Reference

[Accounts Command](#)

Concepts

[Service accounts and dependencies in Team Foundation Server](#)

Other Resources

[TFSSConfig: Manage TFS server configuration](#)

[Change the service account or password for Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Team Foundation Background Job Agent

[See Also](#) [Send Feedback](#)

The Visual Studio Team Foundation Background Job Agent service provides a general scheduling mechanism for Web services and jobs for Team Foundation. This Windows service is also used to run the tasks spawned by various wizards, such as the New Team Project wizard and Create a Team Project Collection wizard. The service uses the service account for Team Foundation Server (TFS), referred to as TFSService. The service runs on any server that is running a Web service or Web application in the logical application tier for Team Foundation. To operate correctly, the service account for the Team Foundation Background Job Agent service must have the permissions required for the tasks that it performs.

Some Team Foundation services have tasks that recur at regular intervals. For example, administrators might want to schedule builds on a nightly basis. To accomplish this, build services must be able to set up an automatically scheduled event in the registration database. The Team Foundation Background Job Agent service provides a single Windows-based service to schedule repeating tasks on servers that are running Team Foundation. The service runs through the registration database, identifies all Team Foundation Server Web services that have scheduled events, and schedules these tasks.

⚠ Important

To perform tasks such as creating team project collections, the service account that the Team Foundation Background Job Agent uses must have certain permissions granted to it. For more information, see [Service accounts and dependencies in Team Foundation Server](#).

Instances

Only one instance of the Team Foundation Background Job Agent service should be running on any application-tier server for Team Foundation. By default, the service runs under the service account that you specified when you installed Team Foundation Server. To view the status of this service on an application-tier server, open Services and browse to find the service.

Permissions

The Team Foundation Background Job Agent service uses the same service account as TFS does, TFSService. To operate correctly, this account requires the following permissions:

- Log on as a service
- Farm Administrators group for any SharePoint Web applications that Team Foundation Server uses
- TFSExecRole or both of the following for any databases that Team Foundation Server uses:
 - db_owner
 - db_create

Assumptions and limitations

The Team Foundation Background Job Agent service runs continuously on all application-tier servers. Administrators should not need to manually stop or start this service except during system recovery. For example, you must stop this service before you restore databases. The service should restart automatically when a server is restarted.

Administrators will not directly configure the Team Foundation Background Job Agent service. Tasks that need to be scheduled are configured directly in individual components of Team Foundation, such as Team Foundation Build. When an event is added or deleted, the service automatically reconfigures the tasks scheduled in the registration database.

The Team Foundation Background Job Agent service will log only one instance of any given error until that error is resolved and a success message is recorded in the Event Log, or until the service is manually restarted. If you want to monitor the Event Log for that error message, you must first stop and restart the service.

The Team Foundation Background Job Agent service is not designed to be an all-purpose scheduling mechanism. It is not designed to provide scheduling precision beyond day of the week, hour of the day, and minute of the day. Most administrators will not need to schedule tasks beyond this level of granularity.

See Also

Tasks

[Change the service account or password for SQL Server Reporting Services](#)

Other Resources

[Change the service account or password for Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Move or clone Team Foundation Server (hardware move)

[Send Feedback](#)

You can move or clone your deployment of Team Foundation Server (TFS) software. You move TFS from one machine to another by restoring it to new hardware (also known as a restoration-based move). For example, you might want to move TFS to a server with greater capacity or improved processing speed, or you have concerns about the reliability of the hardware on which TFS is currently running. Moving TFS to a new server allows you to change the hardware that runs TFS without losing any of your project history.

To clone your TFS deployment, you perform the same steps as a move plus a few additional ones. You perform a move when you plan to discontinue use of the original hardware and TFS deployment. You perform a clone when you intend to continue using the original TFS instance after moving it.

Important

In some situations you might want to change the domain of a TFS deployment as well as its hardware. Changing the domain is an environment-based move, and you should never combine the two move types. First complete the hardware move, and then change the environment.

To move or clone TFS, perform these steps:

1. [Check your permissions](#)
2. Back up databases and install software
 1. [Back up the databases and the encryption key](#)

2. [Install and configure SQL Server on the new data-tier server](#)
3. [Install and configure software on the new application-tier server](#)
3. Restore TFS databases to the new hardware
 1. [Restore the TFS databases](#)
 2. [Redirect SharePoint Products to the new location of the content database](#)
 3. [\(Clone option\) Reconfigure server IDs and remap databases](#)
4. Update the configuration of the new application-tier server
 1. [Configure the new application tier](#)
 2. [Update Team Foundation Server URLs](#)
 3. [Update all service accounts](#)
 4. [Update build servers](#)
 5. [Configure SharePoint Web Applications in TFS](#)
 6. [Configure Reporting and SQL Server Analysis Services](#)
5. Verify permissions, notify users, and configure backups
 1. [Verify permissions for users, groups, teams, and service accounts](#)
 2. [Refresh the caches on client computers](#)
 3. [Notify users](#)
 4. [Configure backups](#)

1. Check your permissions

You need to be an administrator on both sets of hardware (the old and the new). In addition, you need to be an administrator (or have the equivalent permissions) for TFS and all of the software on which your deployment depends: SQL Server, reporting, SharePoint Products (if your deployment uses reporting or SharePoint), and any other software with which your deployment interoperates, such as Project Server.

- Make sure you are a member of the following groups:
 - Servers: Administrators (local Administrators group or equivalent)
 - TFS: Team Foundation Administrators and Admin Console Users
 - SQL Server: sysadmin
 - SharePoint Products: Farm Administrators (if your TFS deployment integrates with SharePoint Products)

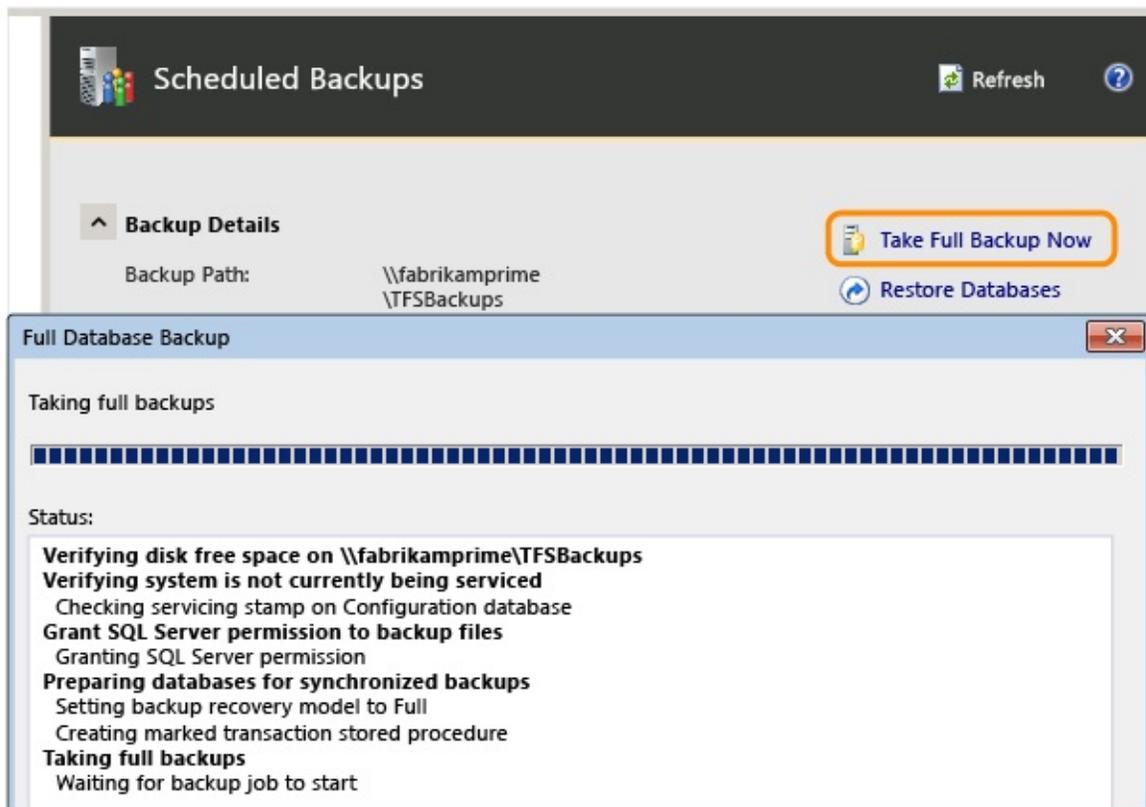
If you aren't a member of one or more of these groups, get permissions now.

2. Back up and install

First, make a backup of your TFS databases, and then install software on the new hardware.

2-a. Back up the databases and the SQL Server Reporting Services encryption key

1. Open the administration console for TFS and on the Scheduled Backups page, take a full backup. The backup will back up everything you configured for backup in your backup plan, but it will do so immediately, not according to the time scheduled in the plan. If your deployment uses reporting, you can back up the encryption key as part of this backup set.

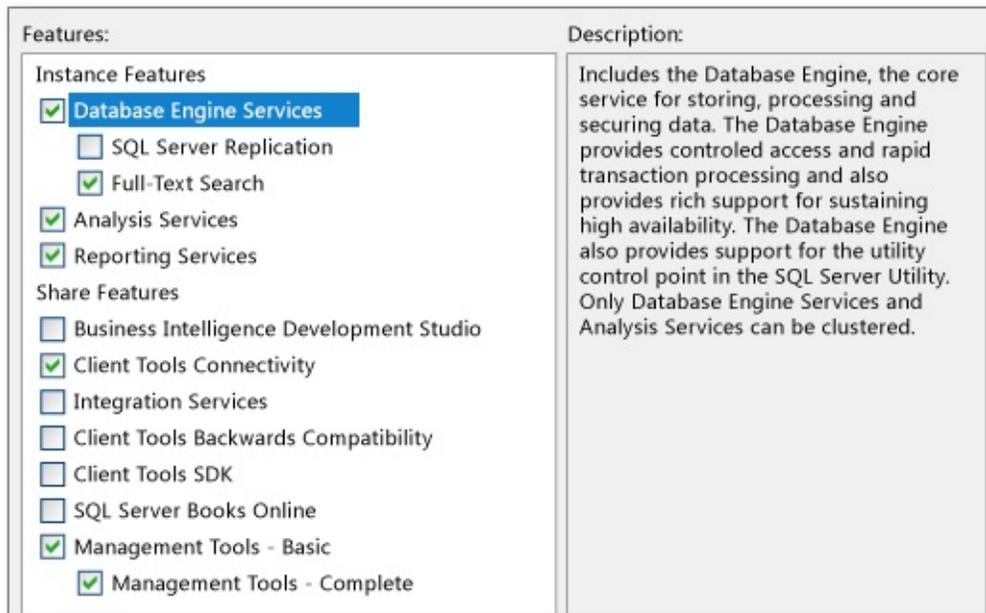


(If you don't have backups configured, you'll have to create a plan before you can take a full backup.)

2. Once the backup completes, verify that the backup is available on the storage device or network share, and that you can access this backup from the new hardware.

2-b. Install and configure SQL Server on the new data-tier server

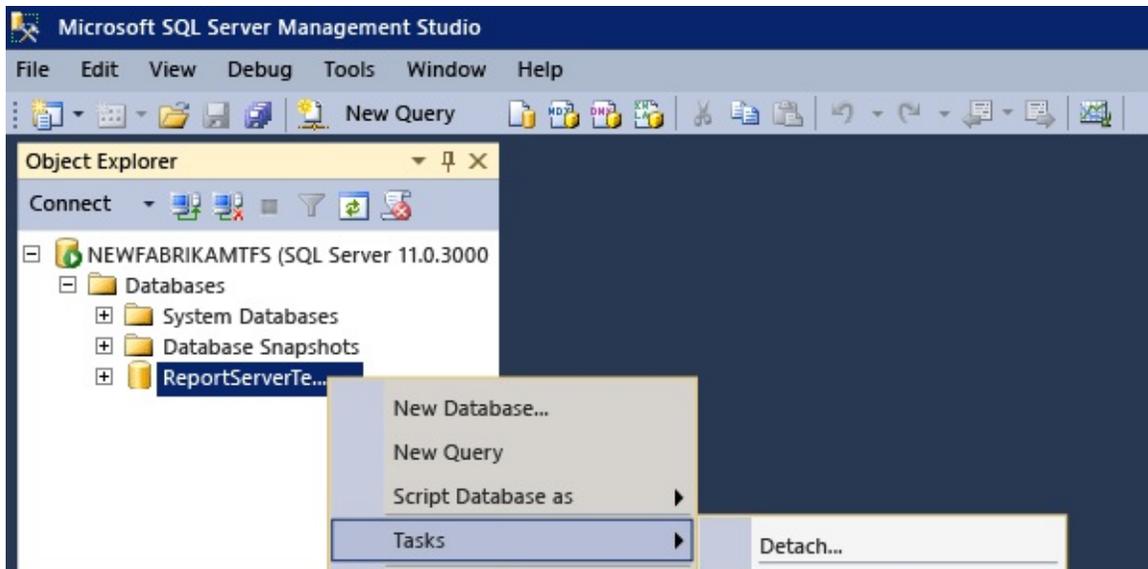
- Install SQL Server on the new server and make sure that it is operational. If your previous deployment used reporting, make sure that you include the reporting and analysis services components. You must install the same version and edition that you used previously, including service pack and cumulative update levels.



As an alternative, you can create an instance of SQL Server on a server that already has a matching version installed and restore the TFS databases to that instance, but that will require more post-restoration configuration.

For more information about options for installing and configuring SQL Server, go [here](#).

After installing SQL Server, if your deployment includes reporting, open SQL Server Management Studio and detach the ReportServer and ReportServerTempDB databases. Otherwise you might not be able to restore these databases with the backup you created for the TFS databases.



2-c. Install and configure software on the new application-tier server

To configure a new server or servers for TFS, you must first install and configure the software that is required to support it. This software includes the following components:

- a supported operating system for your deployment configuration
- a supported version of SharePoint Products (if your deployment is integrated with SharePoint Products and you want to move it to the same server as TFS)

Note

Unlike installing a new deployment of TFS, you will not be able to install SharePoint Products as part of the Standard Single-Server or Advanced options when moving to a new server. You must manually install the same version and edition of SharePoint Products that you used in your previous deployment, or follow the guidance for your version and edition of SharePoint Products to move its deployment to the new hardware separately.

Install and configure Windows, IIS (if not configured by default), and SharePoint (if using) in the new environment, and make sure that the server and

its software are operational.

For more information, see the system requirements for Team Foundation Server and Move SharePoint to new hardware for TFS.

3. Restore TFS databases to the new hardware

In order to restore the TFS databases using the restore tool, you must install but not configure TFS on the new data-tier server, and then use the restore function in the Scheduled Backups node.

If you want to restore TFS databases manually using SQL Server restoration tools, you can, but that is a more difficult procedure. In addition, you will have to manually unquiesce the databases in the new deployment. The restore wizard in TFS automatically does this for you as part of its restoration process, but that functionality is not part of SQL Server's restoration tools.

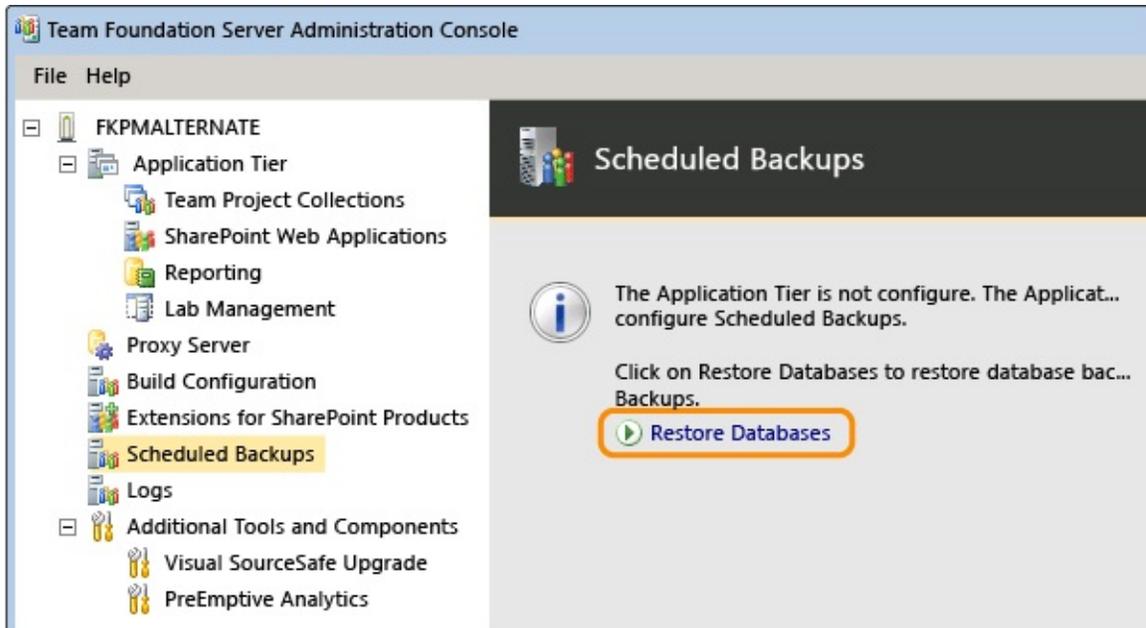
If you installed SharePoint Foundation 2013 using the steps in Move SharePoint to new hardware for TFS, and you plan on using that server as the server for TFS, the installation bits and administration console will already be present on the server and you can skip the first two steps in this next procedure.

3-a. Restore the TFS databases

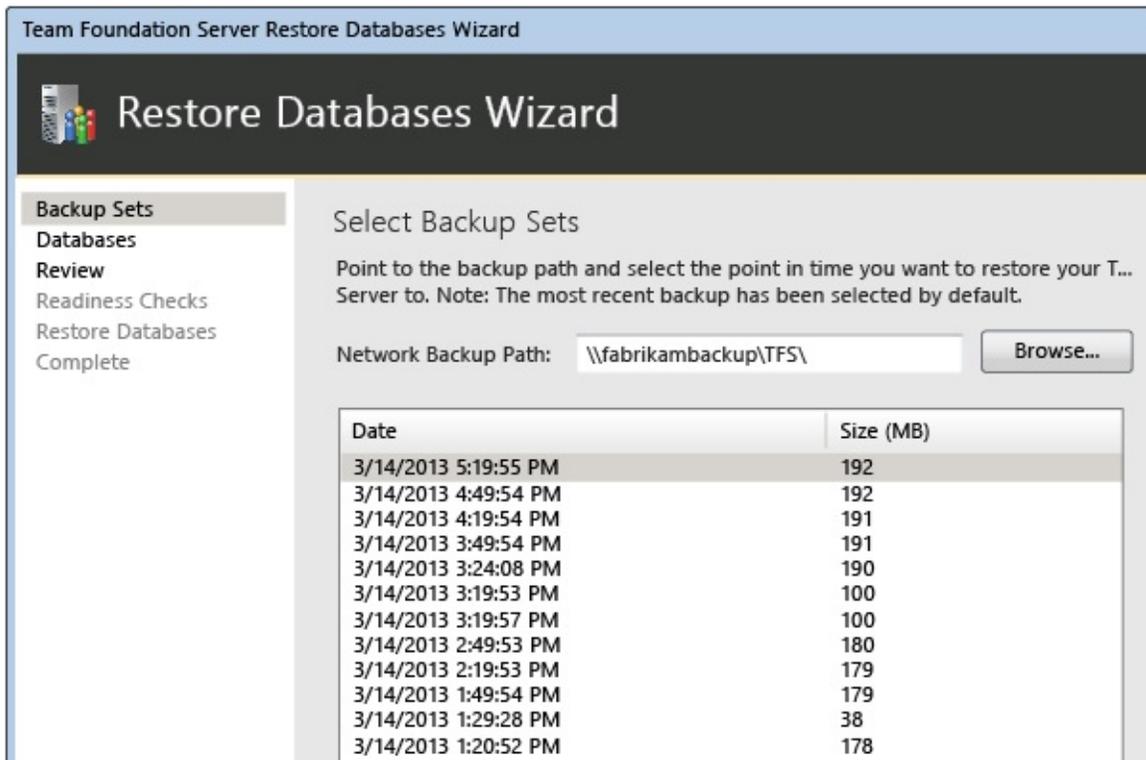
1. Launch the TFS installation media. On the Team Foundation Server Setup page, choose Install.
2. When the installation completes, the Team Foundation Server Configuration Center opens. Close it.

The administration console opens automatically in an unconfigured state. This is expected.

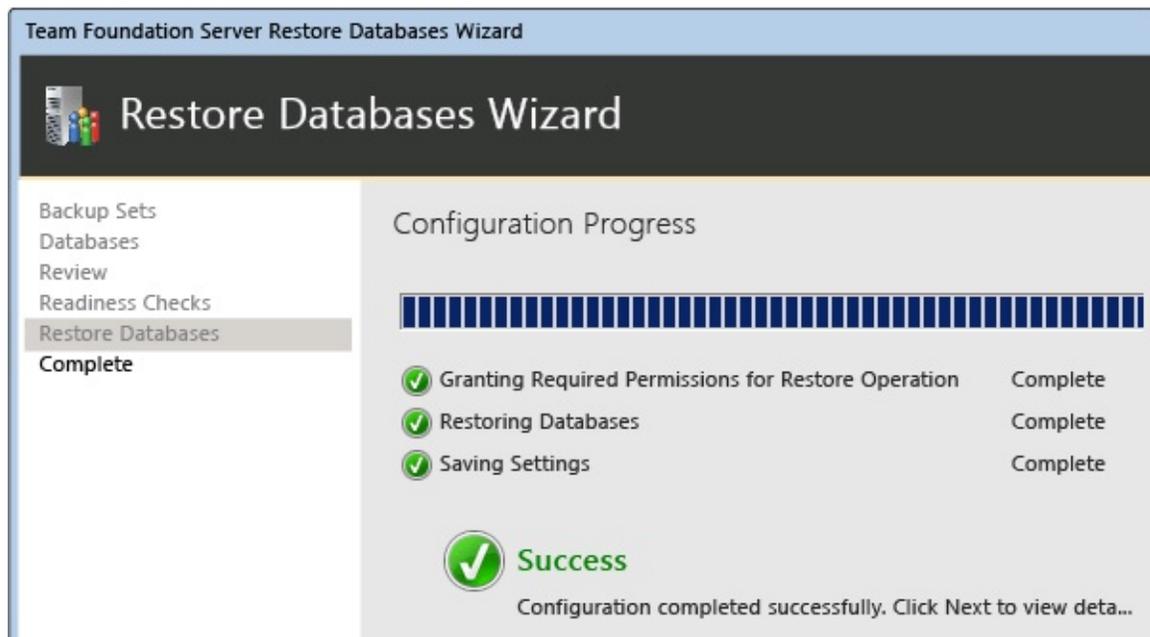
3. To start the Restore wizard, open the administration console for TFS and open Scheduled Backups.



- Specify the path to the backup set and choose the set you created after quiescing the old deployment.



- Complete the wizard and restore the databases to the new instance of SQL Server.



3-b. Redirect SharePoint Products to the new location of the content database

If your deployment uses SharePoint Products, you've already installed the same version and edition of SharePoint Products that you used in your previous deployment by following the steps in Move SharePoint to new hardware for TFS, as mentioned above. Now, after you have restored the old deployment's content database (WSS_Content) to the new server as part of the restoration set, you must redirect the server that is running SharePoint Products to the new location of that database. This database must be operational before you can reconfigure Team Foundation Server with the new locations of its databases.

1. Open a Command Prompt as an administrator on the new hardware that is running SharePoint Foundation.
2. Change directories to *Drive:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\15\bin* and run *stsadm.exe* with the following parameters, where *SharePointFoundationServerName* is the name of the server where you installed SharePoint Foundation 2013, and *SQLServerName* is the name of the server where you restored the WSS_Content database as part of restoring TFS databases:

```
stsadm.exe -o addcontentdb -url http://SharePointFoundationServ
```

3. After that command completes successfully, type the following command, where Domain\UserName is the account you used to install and configure SharePoint Foundation 2013 for use with TFS:

```
stsadm.exe -o addpermissionpolicy -url http://SharePointFoundat:
```

3-c. (Clone option) Reconfigure server IDs and remap databases

Perform the next set of steps on the new application-tier server if you intend to continue using the original TFS instance. These steps are necessary to avoid the risk of corruption of one or both deployments. If both servers are live, you could end up with corruption, particularly if they are pointing to the same SharePoint or reporting resources.

1. Open a Command Prompt window as an administrator and change directories to Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools. Open a Command Prompt window and enter:
2. Run the TFSSConfig PrepareClone command to remove information about scheduled backups, SharePoint, and reporting resources.

```
TFSSConfig PrepareClone /SQLInstance:ServerName  
/DatabaseName:DatabaseName /notificationURL: ApplicationTierURL
```

3. Run the TFSSConfig ChangeServerID command to change the server GUIDs that are associated with the databases. GUIDs must be unique within TFS deployment.

```
TFSSConfig ChangeServerID /SQLInstance:ServerName]  
/DatabaseName:ConfigurationDatabaseName [/ProjectCollectionsOnly]  
[/ConfigDBOnly] [/usesqlalwayson]
```

4. Run the TFSSConfig RemapDBs command to redirect the cloned TFS to its databases.

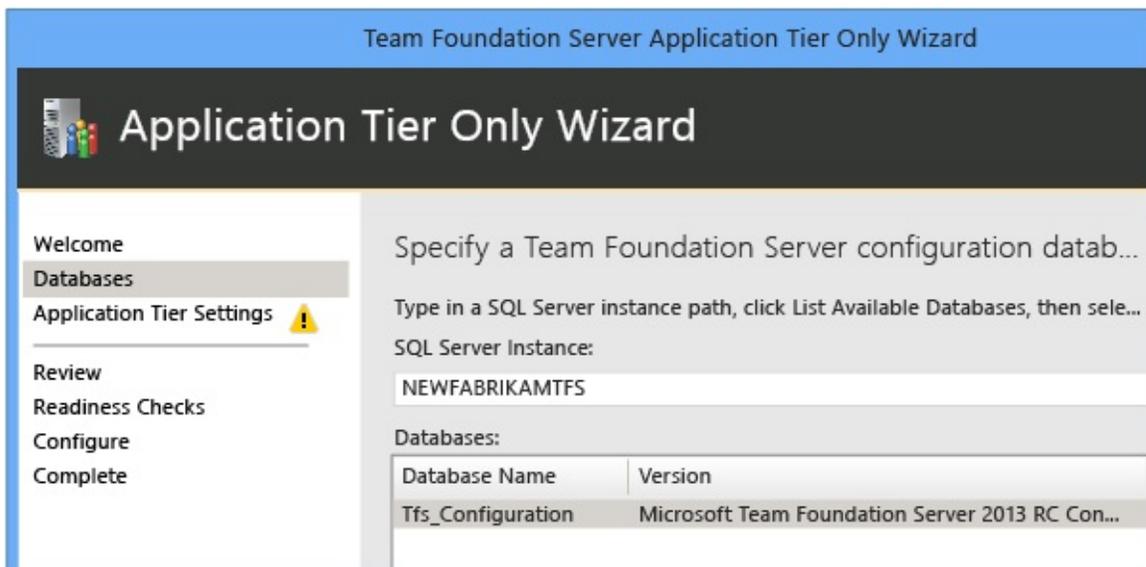
```
TFSSConfig RemapDBs /DatabaseName:ServerName;DatabaseName  
/SQLInstances:ServerName1,ServerName2  
[/AnalysisInstance:ServerName] [/AnalysisDatabaseName:DatabaseName]  
[/preview] [/continue] [/usesqlalwayson]
```

4. Update the configuration of the new application-tier server

Perform these steps on the new application-tier server.

4-a. Configure the new application-tier server

1. From the administration console for TFS, choose Configure Installed Features to launch the configuration center.
2. Launch the Application-Tier Only wizard, and in Databases, specify the new SQL Server instance where you restored the TFS databases. Choose the Tfs_Configuration database from the list.



3. Before you close the final page of the wizard, look for the "i" symbol. It signifies information that you might want for future reference. The final page also includes the location of the configuration log.

Review the Results



Success

Your Team Foundation Server configuration completed successfully.

Connection Details:

Team Foundation Server: <http://newfabrikamtfs:8080/tfs>

Detailed Results:

-  As part of TFS configuration IIS dynamic compression was enabled to improve performance. This is a system-wide setting for IIS. See this link for additional details
<http://go.microsoft.com/fwlink/?LinkId=219608>
-  Firewall exception added for port 8080
-  The time allowed for Windows services to start was increased from 30 seconds to 600 seconds. This affects all Windows services on this server. (The registry value set is HKLM\SYSTEM\CurrentControlSet\Control\ServicesPipeTimeout.)
-  TF255450: The notification URL for this instance of Team Foundation Server is <http://fabrikam2008:8080/tfs> and might need to be updated. You can modify the notification URL in the Team Foundation Administration Console by using the Change URLs command.

[Click here to open the complete configuration log](#)

4-b. Update TFS URLs

1. Go to the application-tier node and look at the notification and Team Web Access URLs. Note that they still point to the location of the old deployment. Update them.

Application Tier Refresh Help

Application Tier Summary

Service Account:	NT AUTHORITY\NETWORK SERVICE	Update Password
Web Site:	Team Foundation Server	Change Account
Application Pool:	Microsoft Team Foundation Server Application Pool	Reapply Account
Authentication:	NTLM	Authentication Settings
Notification URL:	http://fabrikam2008:8080/tfs	Group Membership
Server URL:	http://localhost:8080/tfs	Administer Security
Web Access URL:	http://fabrikam2008:8080/tfs	Change URLs
Machine Name:	NEWFABRIKAMTFS	
Ports:	8080	
Virtual Directory:	/tfs	
Version:	12.0.20812.0 (Tfs2013.RC)	

2. After updating the URLs with the name of the new server, review the information to make sure that it is correct.

Team Foundation Server Administration Console

File Help

NEWFABRIKAMTFS

- Application Tier
 - Team Project Collections
 - SharePoint Web Applications
 - Reporting
 - Lab Management
- Proxy Server
- Build Configuration
- Extensions for SharePoint Products
- Scheduled Backups
- Logs
- Additional Tools and Components
 - Visual SourceSafe Upgrade
 - PreEmptive Analytics

Application Tier

Application Tier Summary

Service Account:	NT AUTHORITY\NETWORK SER...
Web Site:	Team Foundation Server
Application Pool:	Microsoft Team Foundation Se... Pool
Authentication:	NTLM
Notification URL:	http://newfabrikamtfms:8080/tfs
Server URL:	http://localhost:8080/tfs
Web Access URL:	http://newfabrikamtfms:8080/tfs
Machine Name:	NEWFABRIKAMTFS
Ports:	8080
Virtual Directory:	/tfs

4-c. Update all service accounts

You must update the TFS service account (TFSService) and the data sources account (TFSReports). Even if these accounts have not changed, you must update the information to help ensure that the identity and the format of the accounts are appropriate for the new server.

1. Open a Command Prompt window as an administrator and change directories to Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools.
2. At the command prompt, type the following command to add the service account for Team Foundation, where DatabaseName is the name of the configuration database (by default, TFS_Configuration):

```
TfsConfig Accounts /add /AccountType:ApplicationTier  
/account:AccountName /SQLInstance:ServerName  
/DatabaseName:DatabaseName
```

3. At the command prompt, type the following command to add the data sources account:

```
TfsConfig Accounts /add /AccountType:ReportingDataSource  
/account:AccountName /SQLInstance:ServerName  
/DatabaseName:DatabaseName
```

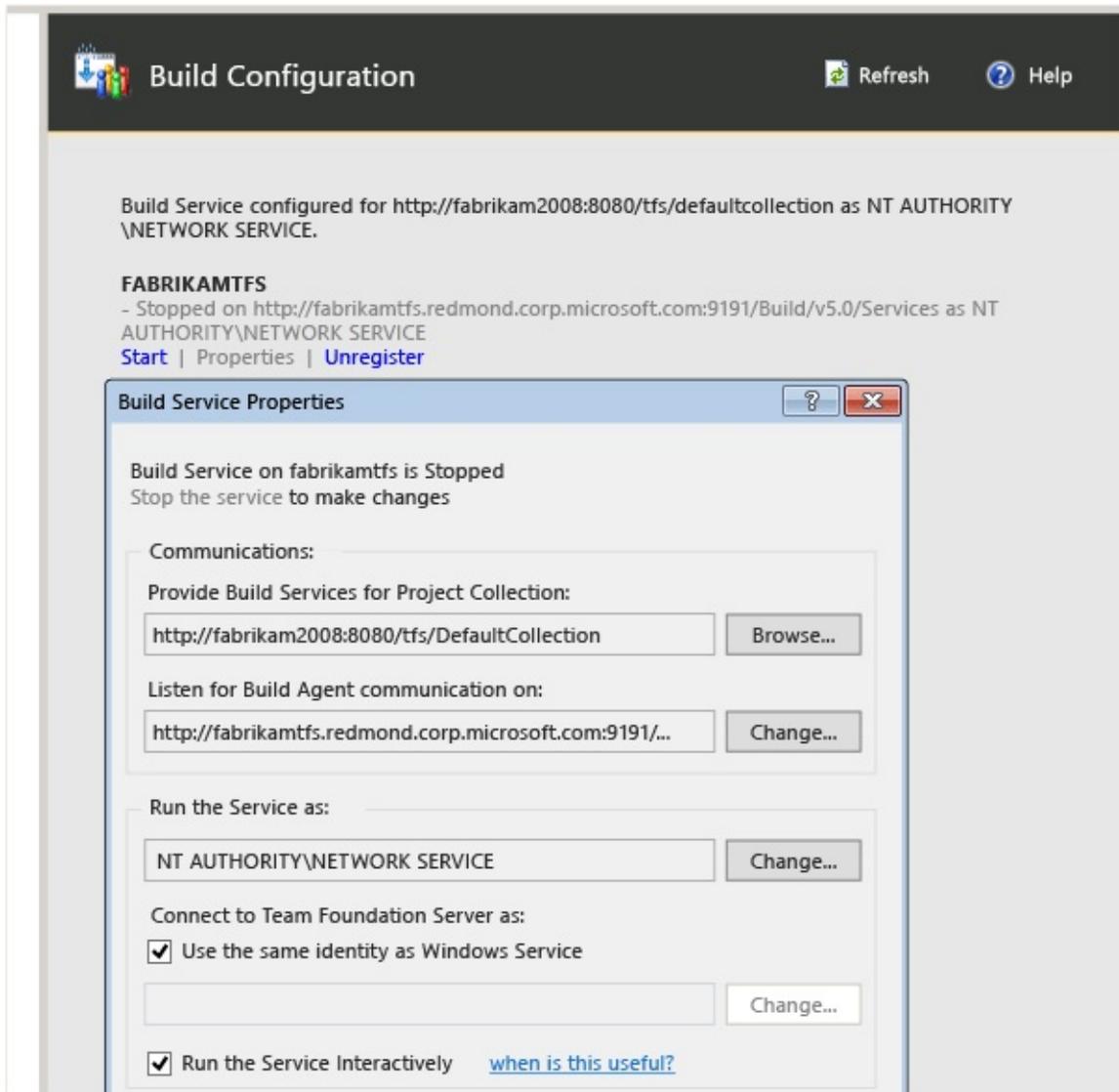
For more information, see

[Accounts Command](#).

4-d. Update build servers

Now you'll need to redirect your build servers to point to the moved TFS deployment.

1. On each build server, open the administration console and stop the build service.
2. In the properties for the build service, update the communications properties.



4-d. Configure SharePoint Web Applications

If your deployment uses SharePoint Products and you moved SharePoint Foundation 2013 as part of the TFS move, you might need to redirect TFS to the new web application. Even if you don't, you should still repair the connection to help ensure proper performance.

If you're not using SharePoint Products as part of your deployment, or if your deployment will continue to use the old SharePoint server, you can skip this procedure.

- Open the administration console and navigate to SharePoint Web

Applications. If the web application still refers to the old site, or if your new deployment uses a different web application than the one listed, choose Change and update the settings.

SharePoint Web Applications Refresh

SharePoint Web Applications

You can use SharePoint Web applications to provide collaboration resources for members of a team project, such as team project portals and document libraries. Before you can specify a SharePoint Web application, you must install the Team Foundation Server Extensions for SharePoint Products and configure them for at least one Web application. [More Information](#)

Name	URL
http://fabrikam2008	http://fabrikam2008

Service Accounts for SharePoint Web Applications

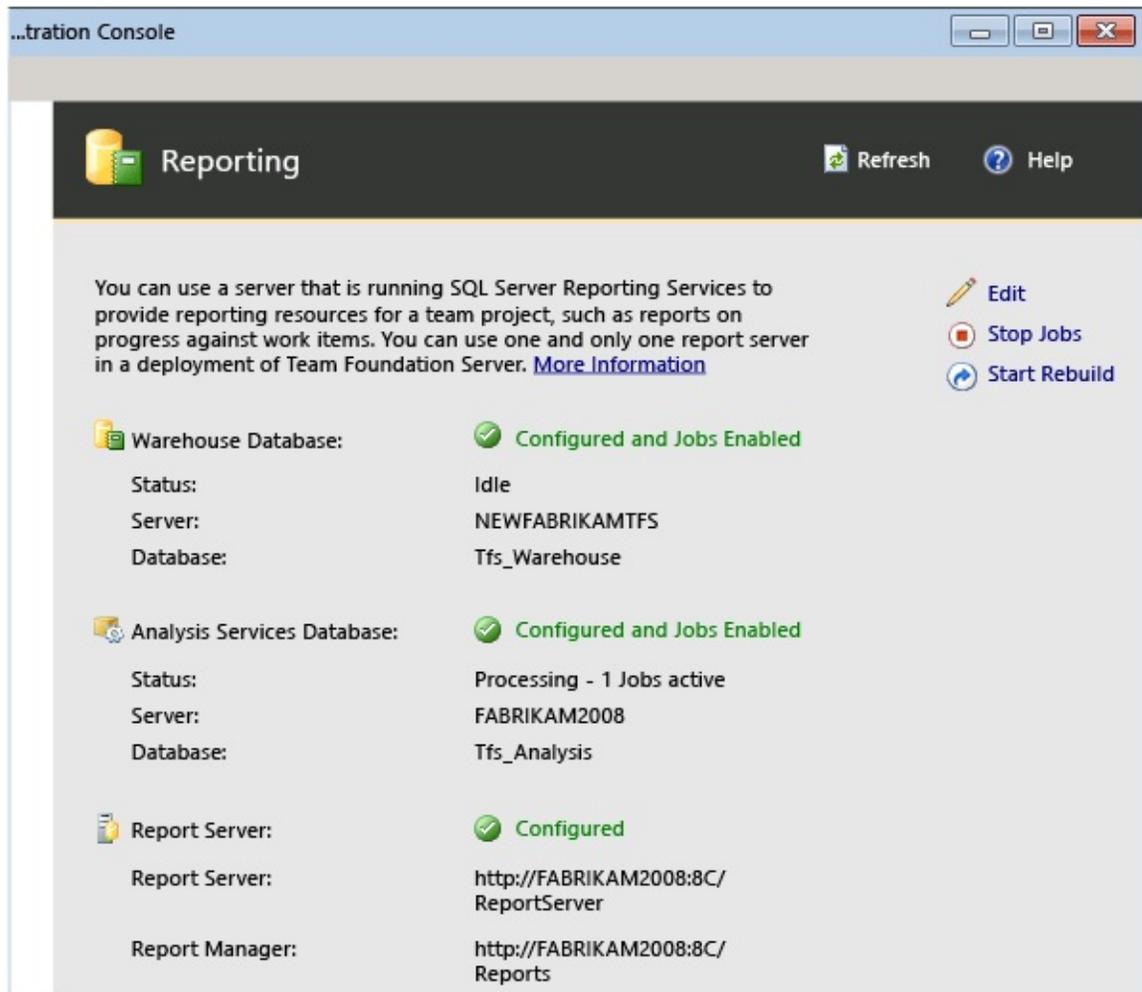
You must specify the service accounts for SharePoint Web applications to support team project portal functionality for Team Foundation Server. The service account for both the Web application and Central Administration site are required.

If the information is correct, or after you have corrected it, choose Repair Connection. This helps ensure that everything is working properly.

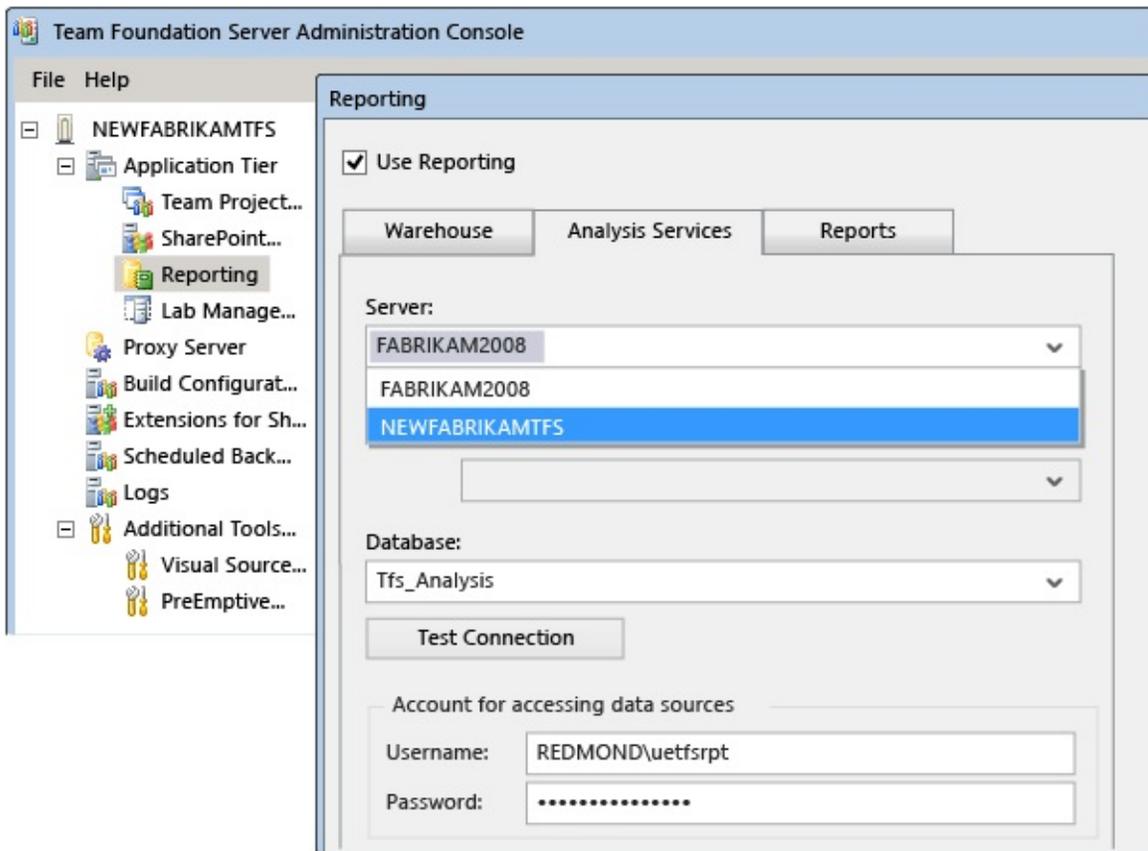
4-e. Configure Reporting and Analysis Services

If your deployment uses a report server, you must redirect Team Foundation Server to its location, restart the warehouse, and manually rebuild the database for Analysis Services. If you don't use reporting, skip this procedure.

1. Go to the Reporting node. The listed report server values are the old ones, not the new, so edit them.



2. Change the values on all three tabs to point to the new server. Make sure that you provide the correct information for the data sources account in the new deployment.



3. Choose Start Jobs to restart reporting.
4. Choose Start Rebuild to rebuild the warehouse.

5. Verify permissions, notify users, and configure backups

After you move to new hardware, make sure that all users, groups, and service accounts for your deployment are configured with the permissions that they require to function correctly on each server. Some permissions, such as additional permissions in SQL Server or on the local computer, cannot be automatically migrated. For example, Team Foundation administrators must be members of the local Administrators group on the application-tier server to open the administration console, so you must add manually them to that group.

5-a. Verify permissions for users, groups, and service accounts

- Log on to the server and make sure that users, groups, and service accounts are configured with the permissions required for operation. Manually spot-check membership in project groups and teams, and verify that those groups and teams have the permissions you expect.
- Browse to a team project collection and make sure that all projects in that collection appear as expected, and that users in those projects can appropriately access their work items.
- Open Team Web Access and verify that team sites and teams appear as expected.

Not sure what groups and permissions to expect? For more information, see

[Add users to team projects](#), [Set administrator permissions for team project collections](#), [Set administrator permissions for Team Foundation Server](#), and [Service accounts and dependencies in Team Foundation Server](#).

5-b. Refresh the data cache on client computers

- Log on to the server and use the ClientService Web service to force clients to update the cache for tracking work items and for Team Foundation

version control.

Copy Code

```
http://ServerName:8080/tfs/WorkItemTracking/v3.0/ClientService.
```

For more information, see

[Refresh the data caches on client computers.](#)

If you want to refresh the entire cache for all users the next time they log on, use the witadmin rebuildcache command.

Note

If you restored your databases to a different point in time, you will also need to refresh the version control cache as documented in [Refresh the data caches on client computers](#).

5-c. Notify users

Now that you've moved TFS, you'll need to tell your users how to connect to the moved deployment. Specifically, you'll need to give them the following information:

- The name of the new server and the URL for Team Web Access, so that they can reconnect to their projects
- The new database names for reporting, if reporting is part of your deployment
- The new URL for SharePoint, if SharePoint is part of your deployment
- If they are members of a team project that uses Git, instructions for how to update every clone they have locally for every repository for that project. Specifically, they will have to run the following command for every clone:

Copy Code

```
git remote set-url <remote name> <new URL>
```

Users can see what the URL is for each clone by browsing the team project from the Explorer tab.



Learn more about Git here.

5-d. Configure backups on the new TFS instance

In the administration console, go to the Scheduled Backups node and reconfigure the scheduled backups to back up the TFS databases on the new server.

For more information, see

[Configure a backup schedule and plan.](#)

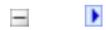
Q & A

Q: I want to change domains, not physical servers. Can I do that?

A: Yes. That's called an environment-based move, and the steps can be found here. You should not try to combine an environment-based move with a hardware-based move. First complete the hardware move, and then change the environment.

Q: What steps are needed to integrate with Project Server after a hardware move?

A: Yes, after you complete the hardware move you'll need to use the **TFSAdmin ProjectServer /RegisterPWA** command with the **/tfs**, **/force**, and **/pwa** options to re-register TFS with Project Server. You can read more about TFS integration with Project Server [here](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Move Team Foundation Server from one environment to another

[Send Feedback](#)

The most common environment-based move scenario is changing the domain of the TFS deployment, whether it's a domain name change or going from a workgroup to a domain.

⚠ Important

In some situations you might want to change the domain of a TFS deployment as well as its hardware. Changing the hardware is a restoration-based move, and you should never combine the two move types. First complete the hardware move, and then change the environment.

Additionally, changing identities in TFS as part of an environmental move is the aspect that most often causes conflicts or problems. The [Identities Command](#) is a powerful tool, but it has certain limitations. Read up about it as part of planning your move. To help ensure a successful move, make sure that you understand the following requirements:

- Once a user account is present in TFS, it cannot be removed or have another account mapped to it. For example, if you are moving DomainA/UserA to DomainB/UserB, the Identities command would only work to migrate the user if DomainB/UserB is not already present in TFS.
- Because the members of the local Administrators group are automatically added to TFS, make sure to remove any accounts that you want migrated from that group before you change the domain or environment.

For further background information, [go here](#) for a detailed description of how identity changes in TFS work, including limitations of the tool.

We'll walk through the steps to change the environment of your TFS deployment in the following sections:

1. [Check permissions and accounts](#)
2. [Stop TFS services](#)
3. [Back Up Data](#)
4. [Join TFS to its new domain](#)
5. [Configure SharePoint Products for the new environment](#)
6. [Move TFS user and service accounts](#)
7. [Configure Reporting and Analysis Services](#)
8. [Restart TFS services](#)

■ **Check permissions and accounts**

In order to successfully change the environment for TFS, you'll need to be an administrator on the local computer as well as for TFS and all of the software on which your deployment depends: SQL Server, reporting, SharePoint Products (if your deployment uses reporting or SharePoint), and any other software with which your deployment interoperates, such as Project Server. However, all members of the local Administrators group are automatically included in TFS, which can cause problems when trying to migrate accounts. Therefore, you should use an account that you do not intend to migrate as part of the environmental move. You might consider adding a special administrative account just for the move, and using that account to perform the migration.

To verify administrator-level permissions

- Make sure the account you're using is a member of the following groups:
 - Servers: Administrators (local Administrators group or equivalent)
 - TFS: Team Foundation Administrators and Admin Console Users
 - SQL Server: sysadmin
 - SharePoint Products: Farm Administrators (if your TFS deployment integrates with SharePoint Products)

If you aren't a member of one or more of these groups, get permissions now.

Now that you're sure you're using an account that has all the permissions needed, it's time to start checking accounts to see if there might be any conflicts with names or groups in the environment to which you'll be moving. We already know that accounts that are members of the local Administrators group can't be migrated, so let's remove those first.

Remove accounts to be migrated from local Administrators group

- Open the local Administrators group and remove any accounts that you

wish to migrate to the new environment. Repeat this step for any other groups that might be affected.

Now check the list of identities in the current TFS environment and look for any potential problems with groups or individual user accounts that might exist in the new environment.

Tip

Consider creating a table or migration map of identities to be moved as part of the environmental move, including details of which accounts might not be able to be migrated automatically.

Check identities

1. On the application-tier server for Team Foundation, open a Command Prompt window with administrative permissions, navigate to `%ProgramFiles%\Microsoft Visual Studio 12.0 Team Foundation Server\Tools`, and run the following command to view the identities currently in the system:

 **Copy Code**

```
TFSSConfig Identities
```

2. A list of identities will display. Check these users and groups to ensure that there are no potential duplicates or problems with identities in the environment to which you'll move TFS, and take steps to mitigate any potential conflicts.

■ Stop TFS services

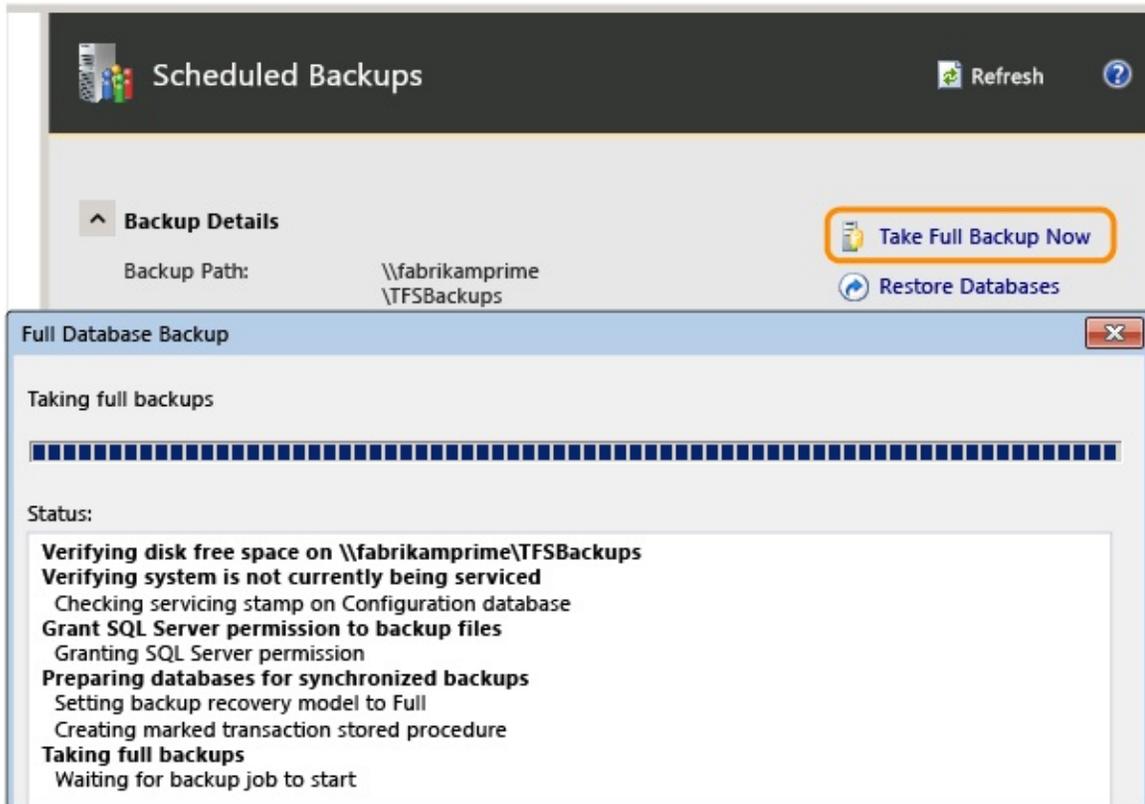
Stopping the services helps ensure that users cannot make changes to work items or check in source code to the original deployment during or after the move process.

1. On the TFS application-tier computer, open a Command Prompt window, and change directories to Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools.
2. Type the following TFSServiceControl command:

```
TFSServiceControl quiesce
```

Back up the databases and the SQL Server Reporting Services encryption key

1. Open the administration console for TFS and on the Scheduled Backups page, take a full backup. The backup will back up everything you configured for backup in your backup plan, but it will do so immediately, not according to the time scheduled in the plan. If your deployment uses reporting, you can back up the encryption key as part of this backup set.



(If you don't have backups configured, you'll have to create a plan before you can take a full backup.)

2. Once the backup completes, verify that the backup is available on the storage device or network share, and that you can access this backup from

the new hardware.

Join TFS to its new domain

1. On each server, open the properties for the computer.
2. Change the settings for the computer to the domain or workgroup to which you want to join the server.

If you are prompted to provide the user name and password of an account that has permissions to join this computer to the domain, provide the appropriate credentials.

3. Restart the computer for the domain change to take effect.

Note

After you restart the computer, a warning might appear that services or drivers could not be started. Continue with the next procedure.

■ **Configure SharePoint Products for the new environment**

If you are changing the environment to one where there is no trust with your previous environment, you might need to configure SharePoint Products before it will operate correctly. Information about users imported from directory services is available on SharePoint sites from the People Picker Web control. Site administrators and other users use the People Picker to select people and groups when assigning permissions. When information about users is located on multiple forests or on a forest without a trust relationship for all users, additional steps might be necessary to ensure that all people and groups are available from this Web control.

Skip this procedure if you are not using SharePoint Products in your deployment, if your new environment has a two-way trust to the old environment, or if no errors for your SharePoint Web application appear in the administration console for Team Foundation.

1. On every server that is part of the SharePoint farm that supports your deployment of Team Foundation Server, open a Command Prompt window with administrative permissions, and change directories to %programfiles%\Common Files\Microsoft Shared\Web Server Extensions\15\BIN.
2. Type the following command, where Key is the encryption key you want to use in your deployment of SharePoint Products:

stsadm.exe -o setapppassword -password Key

■ **Note**

This key is an encryption string that is used to encrypt the password for the account that is used to access the forest or domain. The encryption string must be the same for every server in the farm, but a unique string should be used for each farm.

3. Type the following command, where domain:DNSName is the target forest or domain and its DNS name, user,password is the username and password for an account that has access to the target forest or domain, and WebApp is the name of the Web application that supports your deployment of Team Foundation Server:

```
stsadm.exe -o setproperty -pn peoplepicker-searchadforests -pv  
domain:DnsName,user,password -url http://WebApp
```

4. Type the following command, where URL is the URL for a site collection that supports a team project collection, Port is the port number that is assigned to that site collection, and UserName is the user name of the account that will act as the owner for that site collection:

```
stsadm.exe -o siteowner -url http://URL:Port -ownerlogin UserName
```

5. Repeat the previous step for each site collection that your deployment of Team Foundation Server uses.

Move TFS user and service accounts

As mentioned at the beginning of this topic, moving accounts is when you're most likely to encounter difficulties, particularly if you haven't carefully planned for user migration. The TFSSConfig Identities command cannot migrate any account to an account that already exists in TFS.

If account names are the same in both domains, and the only difference is the domain name, then you can use the batch mode of TFSSConfig Identities to change all the identities at once. Otherwise you must change identities individually and specify a different target account name, as detailed below.

1. On the application-tier server for Team Foundation, open a Command Prompt window with administrative permissions, navigate to `%ProgramFiles%\Microsoft Visual Studio 12.0 Team Foundation Server\Tools`, and run the following command to change the service IDs (SIDs) for the service account to the new domain:

```
TFSSConfig identities /change /fromdomain:OldComputerorDomainName
```

⚠Caution

If your service account was a system account such as Network Service, you cannot directly migrate the service account, because a system account with the same name exists in the new environment. You'll have to perform a two-stage process change. See the example in

[Identities Command](#).

- To migrate all accounts that have the same name in the new environment, type the following command:

 **Copy Code**

```
TFSSConfig Identities /change /fromdomain:OldDomainName /todomain:New
```

This will batch process the accounts.

- If your new domain contains one or more identities where the name changes between environments, you'll need to manually update the SIDs for each of those identities. For example, if Christie Church's user account was Fabrikam\CChurch in the previous environment, but is NewFabrikam\ChristieC in the new environment, you would have to manually update her SID. For every account that has this requirement, type the following command:

 **Copy**

```
TFSSConfig Identities /change /fromdomain:OldDomainName /todomain:New
```

- Now run the following command to update the service account:

 **Copy Code**

```
TFSSConfig Accounts /change /AccountType:ApplicationTier /account:AccountName
```

- If your deployment uses reporting, run the following command to update the data source account used for reporting:

 **Copy Code**

```
TFSSConfig Accounts /change /AccountType:ReportingDataSource /account:AccountName
```

- If your deployment uses Team Foundation Server Proxy, run the following command to update the service account used for the proxy:

 **Copy Code**

```
TFSSConfig Accounts /change /AccountType:Proxy /account:AccountName /
```

 **Note**

If you are moving to a non-trusted domain, you might also need to manually add users and groups to teams, projects, collections, and Team Foundation Server itself. For more information, see [Add users to team projects](#), [Set administrator permissions for team project collections](#), and [Set administrator permissions for Team Foundation Server](#).

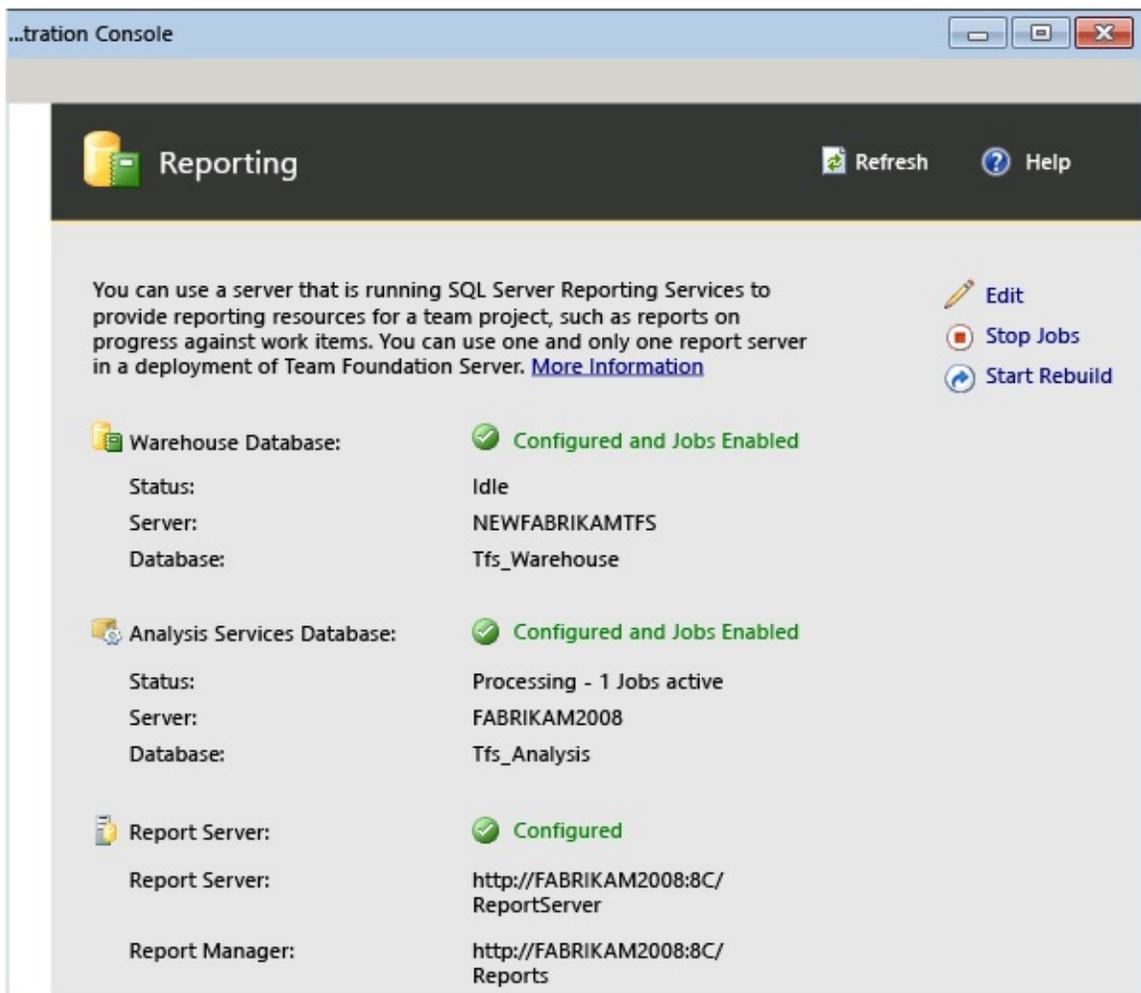
- If your deployment is integrated with Project Server, you might need to perform additional steps to configure the service accounts with the permissions required for operation. For more information, see [Assign permissions to support TFS-Project Server integration](#) and [Configure TFS-Project Server integration](#).

Configure Reporting and Analysis Services

You can skip this procedure if you are not using reporting as part of your deployment.

If you renamed a report server as part of this type of move, you must redirect Team Foundation Server to the report server at its new location. You must also restart the warehouse and manually rebuild the database for Analysis Services.

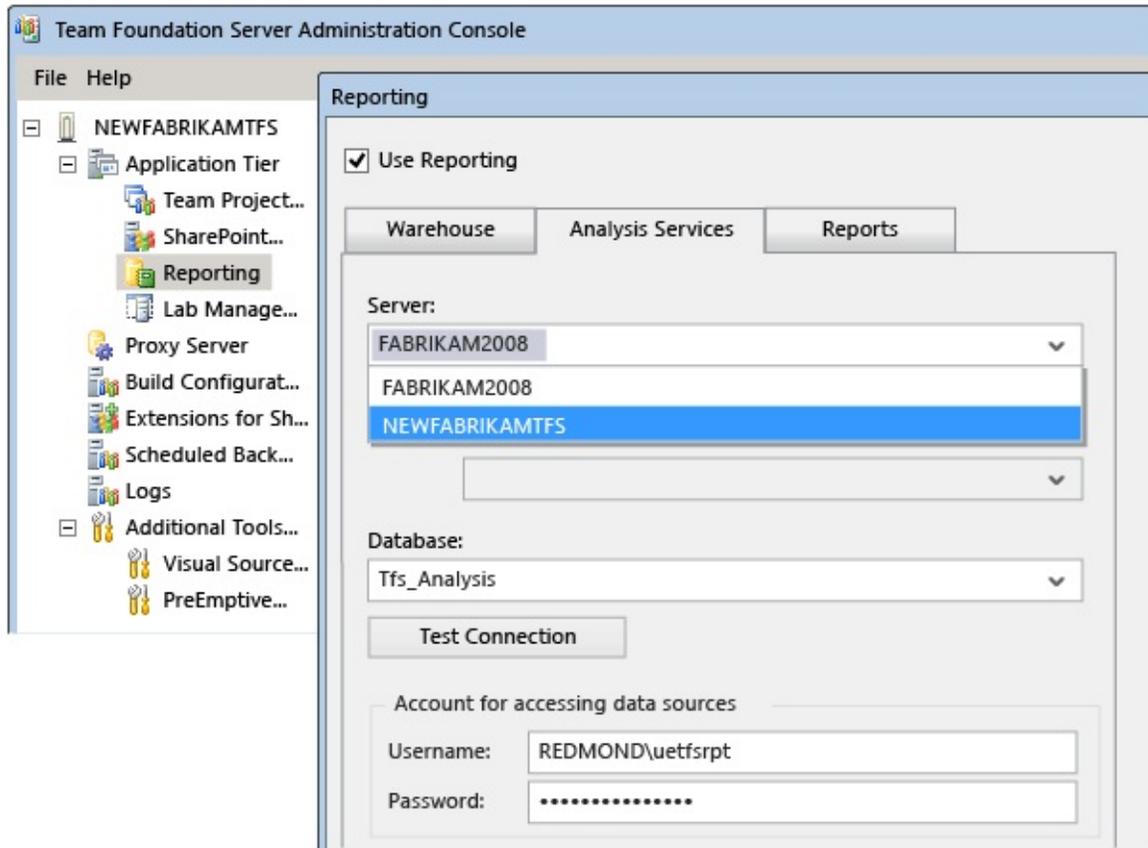
1. Open the administration console for Team Foundation, go to the Reporting node, and edit the settings.



The screenshot shows the Reporting administration console. The title bar reads "...tration Console". The main header is "Reporting" with "Refresh" and "Help" buttons. Below the header, there is a descriptive paragraph and three action buttons: "Edit", "Stop Jobs", and "Start Rebuild". The main content area displays configuration details for three components:

Warehouse Database:	Configured and Jobs Enabled
Status:	Idle
Server:	NEWFABRIKAMTFS
Database:	Tfs_Warehouse
Analysis Services Database:	Configured and Jobs Enabled
Status:	Processing - 1 Jobs active
Server:	FABRIKAM2008
Database:	Tfs_Analysis
Report Server:	Configured
Report Server:	http://FABRIKAM2008:8C/ReportServer
Report Manager:	http://FABRIKAM2008:8C/Reports

2. Change the values on all three tabs so that they include the new name of the server. Make sure that you provide the correct information for the data sources account in the new environment.



3. Choose Start Jobs to restart reporting.
4. Choose Start Rebuild to rebuild the warehouse.

■ **Configure backups**

If the network share name or storage device changed with the domain name change, you'll need to update the scheduled backup plan to point to those renamed resources.

- In the administration console, go to the Scheduled Backups node and reconfigure the scheduled backups to back up the TFS databases on the new server. For more information, see

[Configure a backup schedule and plan.](#)

Restart TFS services

Now that you've updated TFS with all the information for the new environment, restart the services.

1. On the TFS application-tier computer, open a Command Prompt window with administrative permissions and change directories to Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools.
2. Type the following TFSServiceControl command:

```
TFSServiceControl unquiesce
```

Q & A

•

Q: I want to change the physical server or servers for my deployment, not domains. Can I do that?

A: Yes. That's called a hardware-based move, and the steps can be found here. You should not try to combine an environment-based move with a hardware-based move. First complete the hardware move, and then change the environment.

Q: I have a deployment that integrates with Project Server. Do I have to perform any extra steps to get it to work with my moved TFS?

A: Yes, after you complete the environment move you'll need to use the **TFSAdmin ProjectServer /RegisterPWA** command with the **/tfs**, **/force**, and **/pwa** options to re-register TFS with Project Server. You can read more about TFS integration with Project Server

[here](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Add SharePoint products to your deployment

[See Also](#) [Send Feedback](#)

To integrate team projects with SharePoint sites, you must add one or more SharePoint Web applications to standard and advanced installations of Team Foundation Server (TFS). The version of SharePoint Products must also be compatible with TFS. For more information, see [SharePoint Products requirements for Team Foundation Server](#).

For guidance about how to install SharePoint Products for TFS, see [Manually Install SharePoint Products](#). For guidance about how to add an existing deployment of SharePoint Products to TFS, see [Verify SharePoint products for Team Foundation Server](#).

1. Install and configure the Extensions for SharePoint Products on the server or servers hosting the SharePoint deployment you want to integrate with TFS.
2. On the server where you installed the extensions, open the administration console for Team Foundation.
3. Expand the tree, and choose Extensions for SharePoint Products.
4. In the Extensions pane, choose Grant access.

The Access for Team Foundation Server window opens.

5. In URL for Team Foundation Server, the public URL for Team Foundation Server.

You can look up this information in the Application Tier node of the administration console where you have installed the application tier for Team Foundation Server.

6. In SharePoint Web Application, choose the URL for the SharePoint web application that you want to use from the drop-down list.

If you have installed the Team Foundation Server Extensions for SharePoint Products on the server that is hosting the SharePoint Web applications, all SharePoint web applications on that server appear in the list. If no web applications appear, either the extensions were not installed correctly, or no SharePoint web applications have been configured.

7. (optional) In Restrict site creation to the following path, specify the path of the site collection where you want to create all sites that Team Foundation Server will use.

This step is recommended for most deployments, because it helps ensure that the deployment can be backed up and restored correctly.

8. In Enterprise Application Definition, specify the name of the definition that you created for Team Foundation Server.

This step is required only if you are configuring a web application that is hosted on SharePoint Server and you want reports and dashboards to operate correctly. For more information, see [Configure the enterprise application definition for Team Foundation Server](#).

9. When you have provided the required information, choose OK.

If all values are correct, access will be granted. This process might take several moments.

To grant access between Team Foundation Server and a SharePoint web application

1. On the server where you have installed the application tier for Team Foundation Server, open the administration console for Team Foundation.
2. Expand the tree, and choose SharePoint Web Applications.
3. In the SharePoint Web Applications pane, choose Add.

The SharePoint Web Application Settings window opens.

4. On the General tab, in Friendly Name, specify a name for this SharePoint web application.

This name will appear in the list of web applications. If you intend to use more than one SharePoint web application in your deployment, consider specifying a name that will help users distinguish this access grant from the access grants for other SharePoint web applications. You can also add a description in the Description box to help identify this access grant, but the description will not appear in the list.

5. In Web Application URL, specify the URL of the SharePoint web application for which you want to grant access.

This name should match the name of the web application in SharePoint Products. If the SharePoint web application is configured to use a port number that is not the standard default (80), you must also specify the port number.

6. In Central Administration URL, specify the URL and the port number for SharePoint Central Administration.

Note

If you have administrative permissions on the server that is running SharePoint Products, you can view this URL and port number by opening SharePoint Central Administration. If not, you must obtain this information from an administrator for that server.

7. In Default location for team project collection sites, specify the relative path for the site collection that you want to use as the root for team project collections that use this SharePoint web application.

Note

By default, the location name is /sites, but you can use any named path for site collections as configured in SharePoint Central Administration. If you are deploying Team Foundation Server in an environment where another administrator manages SharePoint Products, consider contacting that

administrator for guidance on which path to use.

8. (optional) If you want to make sure that the relative path that you provided is valid, choose Verify Path.
9. Choose OK.

If all the values are correct, the SharePoint Web application appears in the list of Web applications that are available for use with Team Foundation Server. This process might take several moments.

Note

Team Foundation Server will not only grant access but also attempt to add the service accounts for the SharePoint web application to the appropriate group in Team Foundation. If any one of these attempts fails, the configuration information for the web application is saved, but an error message appears. You must then manually configure the settings that the message indicates.

■ Add the Service Account for the SharePoint web Application

When you grant access between a SharePoint web application and Team Foundation Server, you must add the service account for the web application to the SharePoint Web Application Services group in Team Foundation Server.

■ Note

You do not have to perform the procedure to grant access between a SharePoint web application and Team Foundation Server if your account is a member of the Farm Administrators group. If your account is a member of that group, these settings are configured for you automatically when you perform the procedure to grant access between Team Foundation Server and the SharePoint web application.

To add a service account for a SharePoint web application to SharePoint Web Application Services

1. On the server where you have installed the application tier for Team Foundation Server, open the administration console for Team Foundation.
2. Expand the tree, and choose SharePoint Web Applications.
3. In the SharePoint Web Applications pane, in the Service Accounts for SharePoint Web Applications section, choose Add Members.

The Select Users, Computers, or Groups dialog box opens.

4. In Enter the object names to select, specify the account that you want to add, and then choose OK.

See Also

Concepts

[Manage team project collections](#)

[Configure and manage TFS resources](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Set up HTTPS with Secure Sockets Layer (SSL) for Team Foundation Server

[See Also](#) [Send Feedback](#)

You can strengthen the security of your deployment of Visual Studio Team Foundation Server (TFS) by configuring it to use Hypertext Transfer Protocol Secure (HTTPS) with Secure Sockets Layer (SSL). You can choose either to require this protocol, which maximizes the security of your deployment, or you can choose to support HTTPS with SSL in addition to the default protocol, HTTP. If you use Release Management for Visual Studio 2013, you can also configure that to use HTTPS with SSL, although you cannot configure it to support both HTTP and HTTPS with SSL.

Before you choose a configuration, review the advantages and disadvantages described here. After you identify the configuration that best meets the security needs of your organization, follow the steps in this topic to configure your deployment.

In this topic

- Conceptual information
 - [Advantages of Supporting HTTPS with SSL in Addition to HTTP](#)
 - [Advantages of Requiring HTTPS with SSL for All Connections](#)
 - [Disadvantages of Supporting or Requiring HTTPS with SSL](#)
 - [Prerequisites](#)
 - [Assumptions](#)
- Server configuration
 - [Obtaining a Certificate](#)
 - [Requesting, installing, and configuring websites with a certificate](#)
 - [Configuring Your Firewall](#)
 - [Configuring SQL Server Reporting Services](#)
 - [Configuring HTTPS for TFS](#)
- Optional configuration tasks
 - [Testing Access to Your Deployment \(Optional\)](#)
 - [Configuring Your Deployment to Require HTTPS with SSL \(Optional\)](#)
- Build configuration
 - [Installing the Certificate on Build Servers](#)
 - [Updating the Build Configuration](#)
- Release Management configuration

- [Release Management and TFS](#)
- [Configure Release Management Server to use HTTPS](#)
- [Making all the Release Management connections with HTTPS](#)
- Client configuration
 1. [Configuring Client Computers](#)
 2. Git repository configuration
 1. [Configuring Git for certificates](#)

Advantages of Supporting HTTPS with SSL in Addition to HTTP

If you configure your deployment of TFS to support both protocols, users whose computers have been configured for HTTPS with SSL will connect by using that protocol, which makes your deployment more secure. In addition, users whose computers are configured for HTTP only can still connect to your deployment. Although you should not deploy this configuration over public networks, you can gain the following advantages by continuing to support HTTP connections in a controlled network environment:

- You can increase the security of your deployment over time by configuring client computers for HTTPS with SSL as your schedule permits. If you take a phased approach, you do not need to upgrade all computers at the same time, and users whose computers have not yet been upgraded can still connect to the deployment.
- You can more easily configure and maintain Team Foundation Server.
- Calls from one Web service to another are faster over HTTP than over HTTPS with SSL. Therefore, you can continue to support HTTP connections from client computers for which the performance requirements outweigh the security risks.

Advantages of Requiring HTTPS with SSL for All Connections

If you require HTTPS with SSL for all connections, you gain the following advantages:

- All web connections between the application tier, the data tier, and the client tier for Team Foundation are more secure because they require certificates.
- You can control access more easily by configuring certificates to expire when a project phase is expected to end.

Disadvantages of Supporting or Requiring HTTPS with SSL

Before you configure TFS to support or require HTTPS with SSL, you should consider the following disadvantages:

- You might complicate ongoing administration tasks. For example, you might have to reconfigure your deployment to stop supporting HTTPS with SSL before you can apply service packs or other updates.
- You must not only configure but also manage a certification authority (CA) and certificate trusts. You can use Certificate Services in Windows Server 2003 and Windows Server 2008, but you might not want to invest the time and resources that deploying a secure public key infrastructure (PKI) requires.
- You must spend significant time setting up and testing either of these configurations, and troubleshooting your deployment will become more difficult.
- If you continue to support both protocols, external connections might not be encrypted if the application tier for Team Foundation is not appropriately secured.
- If you require HTTPS with SSL, your deployment's performance will be slower.

■ **Configuring Your Deployment to Support or Require HTTPS with SSL**

The procedures in this topic describe one process for requesting, issuing, and assigning certificates that are required for SSL connections in TFS. If you are using different software than what this topic describes, you might need to perform different steps. To support external connections to your TFS deployment, you must also enable Basic authentication, Digest authentication, or both in Internet Information Services (IIS).

By following the procedures in this topic, you will accomplish the following tasks:

1. Obtain certificates for your deployment of Team Foundation Server and the websites that it uses.
2. Install and assign the certificates.
3. Configure Team Foundation Server.
4. Configure Team Foundation Build.
5. Configure Release Management for Visual Studio 2013
6. Configure client computers.

Prerequisites

To perform the procedures in this topic, you must first meet the following requirements:

- The logical components in the data and application tiers of Team Foundation must be installed, although in the case of TFS itself, not necessarily configured. These tiers include IIS, SQL Server, and any additional components you might have integrated, such as SharePoint Products, Team Foundation Build, Release Management and SQL Server Reporting Services.

The procedures in this topic refer to the server or servers that are running the logical components in the application tier for Team Foundation and the data tier for Team Foundation. The application and data tiers might be running on the same server or multiple servers, as described in Team Foundation Server install guide.

- You must have a certification authority (CA) from which you can issue certificates, or have subscribed to a third-party certifying authority with a trusted chain. This topic assumes that you are using Certificate Services as your CA, but you can use any CA that you have configured for your deployment, or certificates from a trusted third-party certification authority. If you do not have a certification authority, you can install Certificate Services and configure one. For more information, see the one of the following sets of documentation on the Microsoft website:

- For Windows Server 2012:

[Active Directory Certificate Services](#)

- For Windows Server 2008: [Active Directory Certificate Services and Public Key Management](#)

Required Permissions

You need to be an administrator to configure all the components of your

deployment for HTTPS and SSL. If you work in a distributed deployment where different people have administrative permissions for individual components, such as SharePoint, you'll need to coordinate with those people to complete configuration.

Specifically, you must belong to the Team Foundation Administrators group, and you must belong to the Administrators group on the application-tier, data-tier, and TFS Proxy server or servers for Team Foundation. To configure a build server, you must belong to the Administrators group on that server. To configure Release Management, you must belong to the Administrators group on the server that hosts Release Management Server and be a member of the Release Manager role in Release Management. If your deployment uses SharePoint Products, you must belong to the Administrators group on the server that hosts SharePoint Central Administration. You must also belong to the Farm Administrators group. If your deployment uses reporting, you must be a member of an administrative security group or have equivalent permissions individually set for configuring reporting services. For more information about permissions, see [Permission reference for Team Foundation Server](#).

Assumptions

The procedures in this topic assume that the following conditions are true:

- The data-tier and application-tier server or servers have been installed and deployed in a secure environment and configured according to security best practices.
- You have installed Release Management for Visual Studio 2013.
- You are familiar with how to configure and manage PKIs and requesting, issuing, and assigning certificates.
- You have a working knowledge of the network topology of the development environment, and you are familiar with configuring network settings, IIS, and SQL Server.

■ Obtaining a certificate

Before you configure TFS to use HTTPS with SSL, you must obtain and install a server certificate for the servers in your deployment. To obtain a server certificate, you must install and configure your own certification authority, or you must use a certification authority from an external organization that you trust (third-party certificates).

For more information about how to install a certification authority, see the following topics on the Microsoft website:

- For Windows Server 2012:

[Deploying an AD CS Two-Tier PKI Hierarchy](#)

- For Windows Server 2008: [Active Directory Certificate Services and Public Key Management](#)

Requesting, installing, and configuring websites with a certificate

After you enlist in a certification authority, you must either request a certificate by using IIS Manager, or you must manually install the certificate on each of the following servers in your deployment:

- Each application-tier server.
- Each server that is running Team Foundation Server Proxy, if any are configured for your deployment.
- Each server that is running Team Foundation Build Service as either a build controller or a build agent, if any are configured for your deployment.
- Each server that is running Release Management Server, or any servers¹ in a release environment that are running the deployment agent, if Release Management is part of your deployment.
- Each server that is running SharePoint Products, if SharePoint Products is configured for your deployment.

Note

Configuring a SharePoint site to use HTTPS and certificates often requires additional steps, such as

[configuring alternate access mappings](#) and [configuring authentication infrastructure](#). For more information, consult the latest SharePoint documentation for your version of the product.

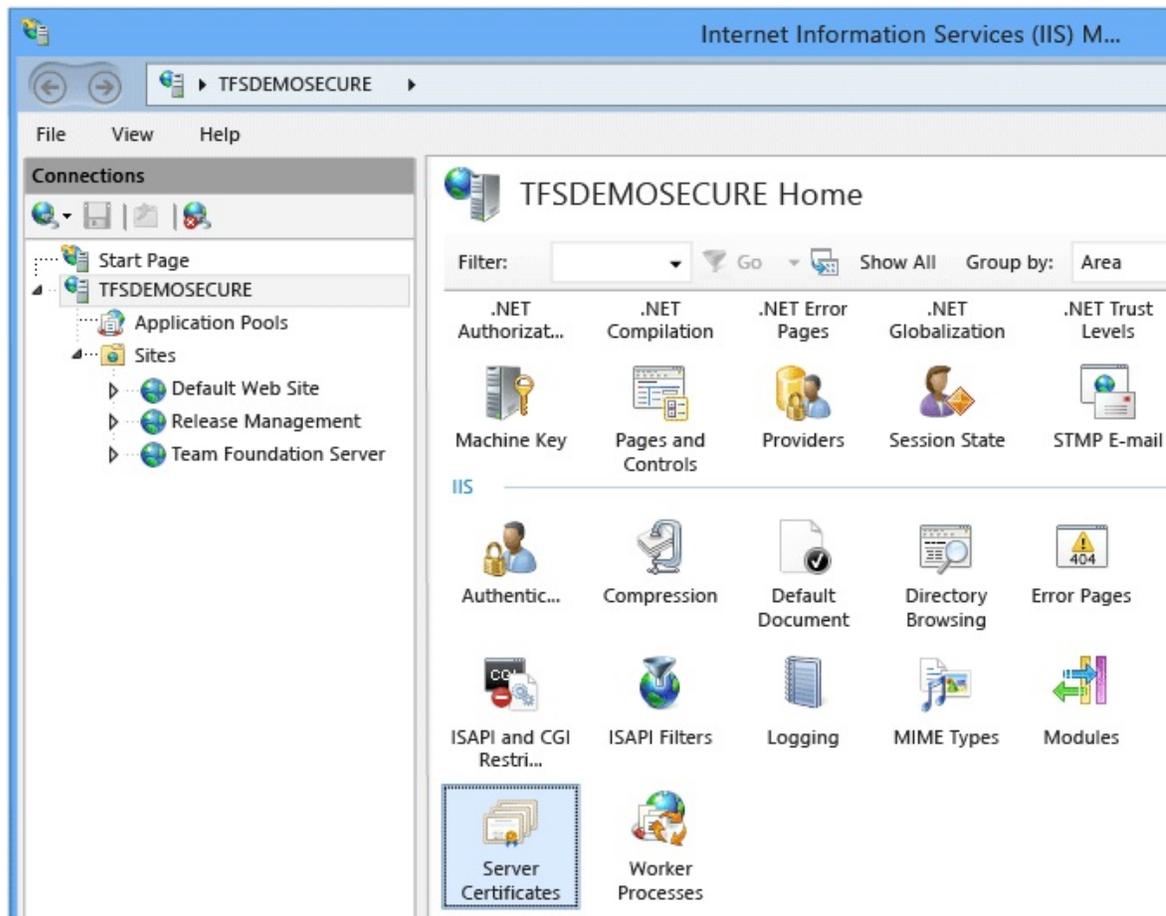
- The server that is running Reporting Services, if one is configured for your deployment.

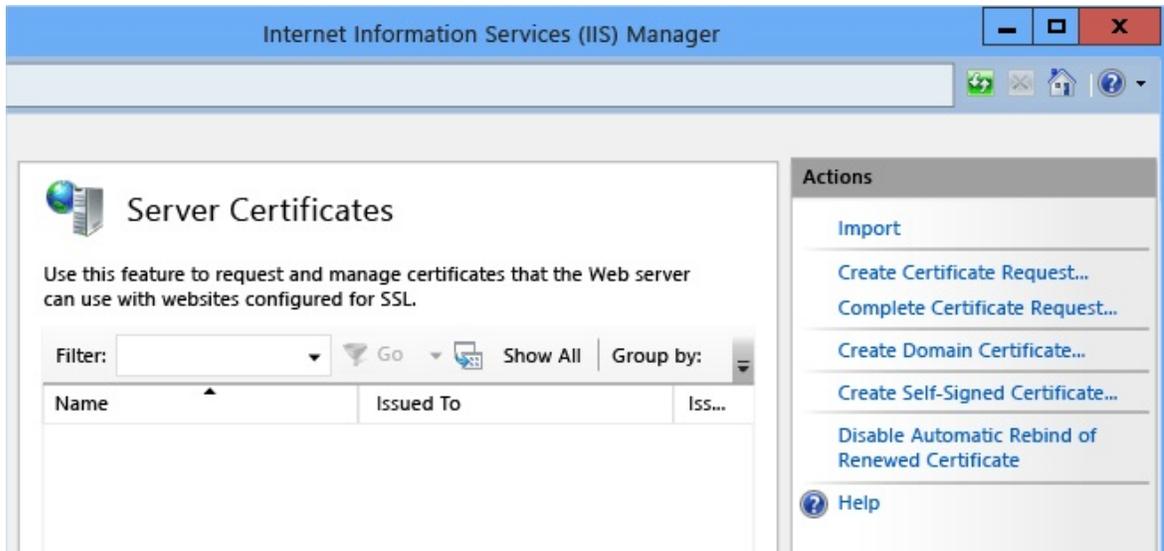
In addition, the client computers in your deployment will need to be enrolled in

the certificate chain and request the needed certificate. If you're using Release Management, this includes any computers running the Release Management client, as well as any clients¹ running the deployment agent in your release environments. If one or more of your projects uses Git for version control, users in those projects will also have to configure Git on their computers to recognize and use the client certificate. For information about how to request a client certificate from a specific CA, see the documentation for that certification authority.

¹ Clients and servers are called out separately here, but that's just a convention of this document. Any computer running the deployment agent needs the certificate installed.

1. Open Internet Information Services (IIS) Manager.
2. Expand your server, navigate to Server Certificates, and create and complete your certificate request.



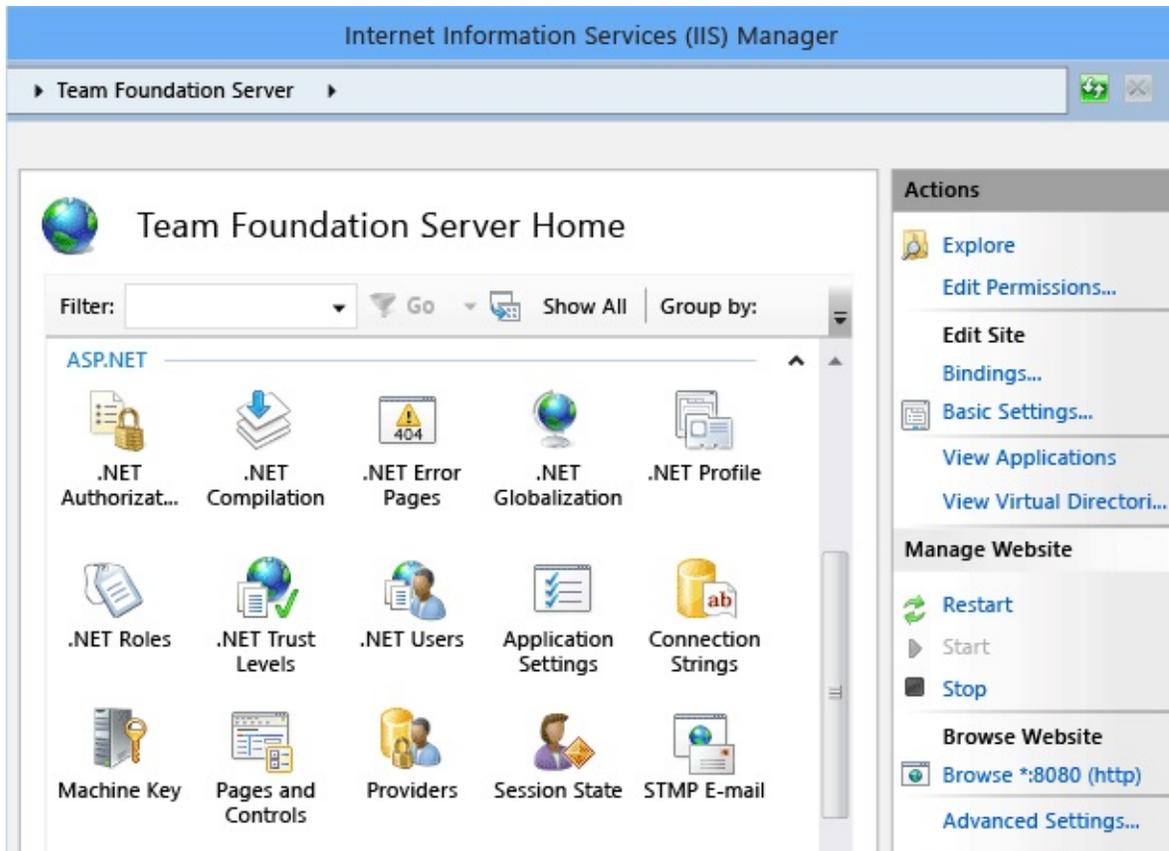


For more information, see [Configuring Server Certificates in IIS](#).

3. Import the certificate.
4. Now you need to configure each website that will require this certificate with the appropriate settings, (with the exception of the Release Management website, which you will configure later). Specifically, you'll need to do this for each of the following websites:
 - Default Website
 - Team Foundation Server
 - Microsoft Team Foundation Server Proxy (if your deployment uses it)
 - SharePoint Central Administration (if your deployment uses SharePoint)

On each server that hosts a website that you want to configure, open Internet Information Services (IIS) Manager.

5. Expand ComputerName, expand Sites, open the submenu for the website that you want to configure (for example, Team Foundation Server), and then choose Bindings from the Actions pane.



6. In Site Bindings, choose Add.

The Add Site Binding dialog box appears.

7. In the Type list, choose https.

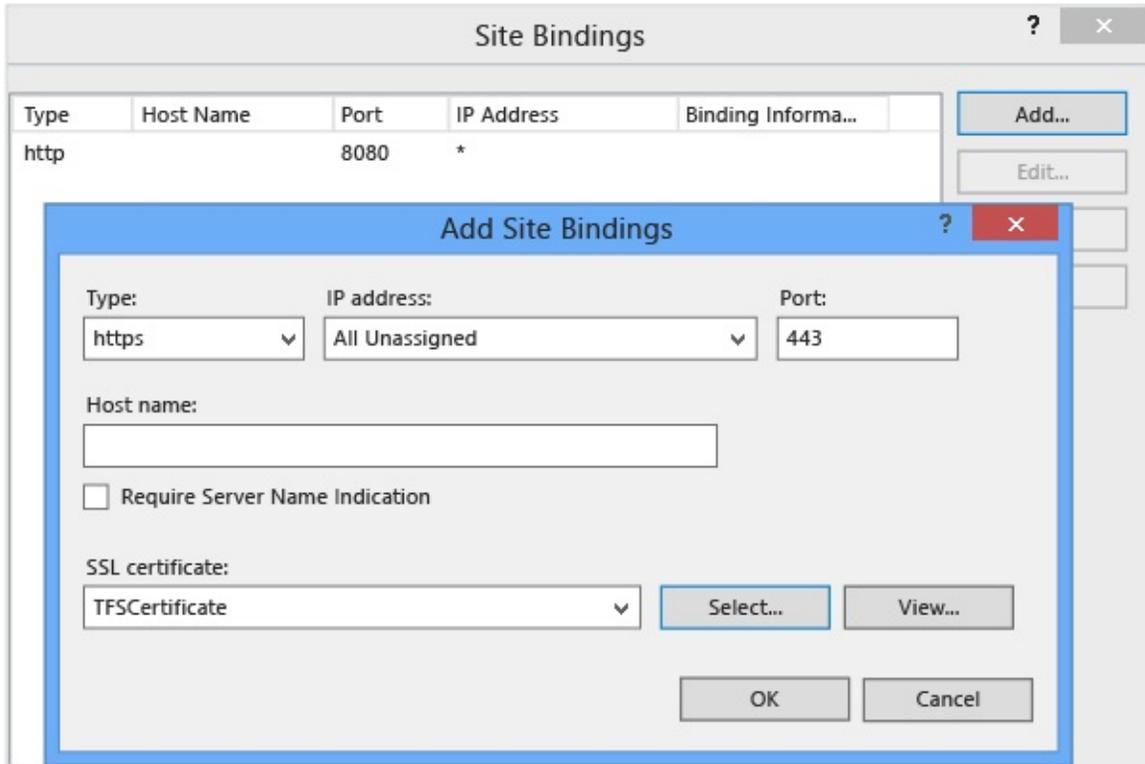
In Port, type a different port number.

⚠ Important

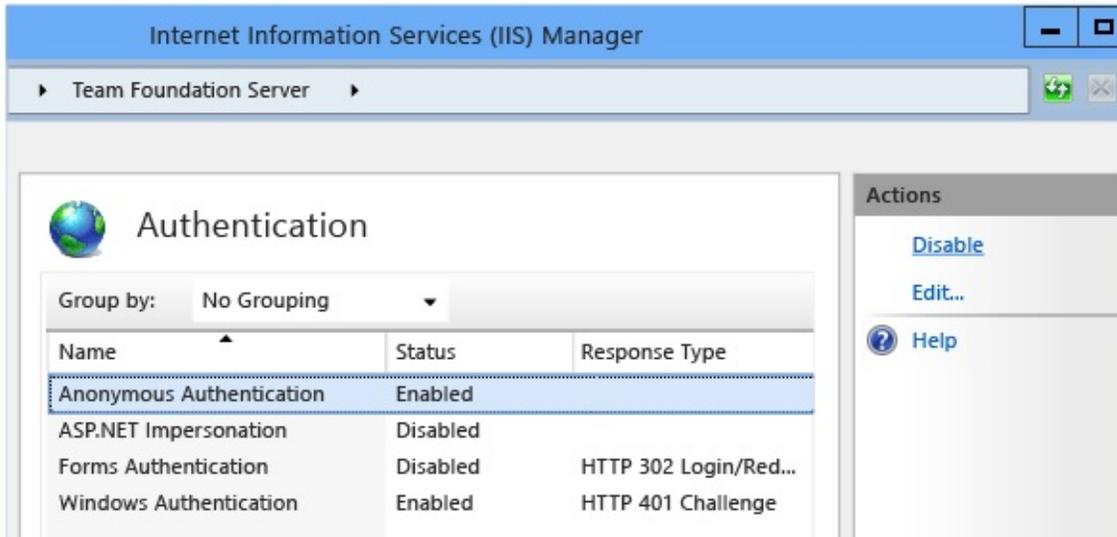
The default port number for SSL connections is 443, but you must assign a unique port number for each of the following sites: Default Website, Team Foundation Server, Microsoft Team Foundation Server Proxy (if your deployment uses it), and SharePoint Central Administration (if your deployment uses SharePoint).

You should record the SSL port number for each website that you configure. You will need to specify these numbers in the administration console for Team Foundation.

In SSL Certificate, choose the certificate that you imported, and then choose OK and close the Bindings page.



8. On the Home page for the website that you are configuring, open the Features view.
9. Under IIS, choose Authentication.
10. Choose an authentication method that you want to configure, open its submenu, and then enable, disable, or perform additional configuration on the method, as best meets your security needs. For example, if you wanted to disable anonymous authentication, you would choose the Anonymous Authentication method and the choose Disable from the Actions menu.



11. Once you have finished configuration, restart web services.

▣ **Configuring Your Firewall**

You must configure your firewall to allow traffic through the SSL ports that you just specified in IIS. For more information, see the documentation for your firewall.

⚠ Important

Make sure to test traffic on the ports you specified from another computer. If you cannot access the default website or Team Web Access, double-check the port settings you specified for these websites in IIS, and make sure that the firewall is configured appropriately to allow traffic on those ports.

■ **Configure SQL Server Reporting Services**

If your deployment uses reporting, you must configure SQL Server Reporting Services to support HTTPS with SSL and to use the port that you specified in IIS for Team Foundation Server. Otherwise, the report server will not function correctly for your deployment. For more information, see

[Configuring a Report Server for Secure Sockets Layer \(SSL\) Connections.](#)

■ **Tip**

If your deployment does not use reporting, you can skip this procedure.

Configuring HTTPS for TFS

Follow these steps to configure your TFS deployment with the HTTPS ports and values that you configured in IIS for the default and Team Foundation Server websites.

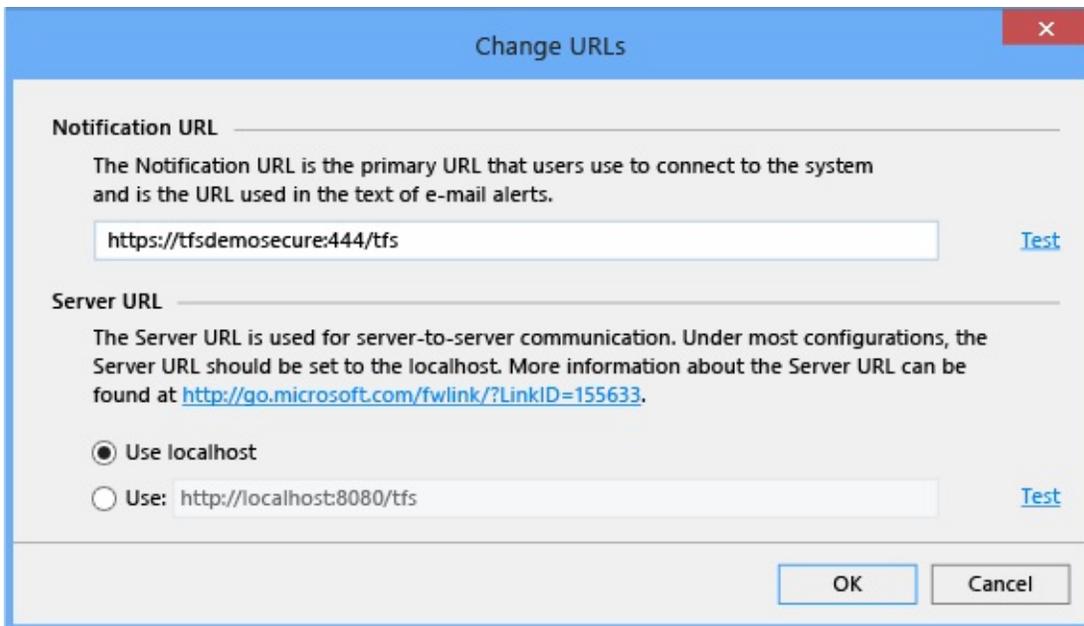
To reconfigure Team Foundation Server to use or require HTTPS

1. Open the administration console for Team Foundation and browse to the application tier node.
2. In Application Tier Summary, choose Change URLs.

The Change URLs window opens.

3. In Notification URL, type the HTTPS URL that you configured for the Team Foundation Server website in IIS.

For example, you might have configured the website to use port 444. In this case, you type `https://ServerName:444/tfs`. Make sure that you use the fully qualified domain name of the server instead of localhost.



Change URLs

Notification URL

The Notification URL is the primary URL that users use to connect to the system and is the URL used in the text of e-mail alerts.

[Test](#)

Server URL

The Server URL is used for server-to-server communication. Under most configurations, the Server URL should be set to the localhost. More information about the Server URL can be found at <http://go.microsoft.com/fwlink/?LinkID=155633>.

Use localhost

Use: [Test](#)

4. Choose Test. Don't choose OK if the test doesn't pass. Go back and make sure that you entered the correct URL and port information, that all firewalls are configured to allow traffic on those ports, and that the site is available and running in IIS Manager.
5. To require HTTPS, choose Use in Server URL, and then type the HTTPS URL that you configured for the Team Foundation Server website.

Make sure that you use the fully qualified domain name of the server instead of localhost.

6. Choose Test, and then choose OK if the test passes.
7. If your deployment uses SharePoint Products, choose SharePoint Web Applications in the administration console. Otherwise, skip the next six steps.
8. In SharePoint Web Applications, in the Name list, choose a web application, and then choose Change.

The SharePoint Web Application Settings page opens.

9. In Web Application URL, change the URL to the HTTPS value for the application.
10. In Central Administration URL, change the URL to the HTTPS value for the Central Administration website.
11. (Optional) In Friendly Name, change the value to reflect the HTTPS address of this application.
12. Choose OK.
13. Repeat the previous five steps for every SharePoint web application in your deployment.
14. If your deployment uses Reporting Services, in the administration console, choose Reporting. Otherwise, skip the rest of this procedure.
15. In Reporting, choose Edit.

If the Take Offline dialog box opens, choose OK.

The Reporting window opens.

16. Choose the Reports tab. In URLs for Report Server, type the HTTPS URLs for Web Service and Report Manager, and then choose OK.

■ **Test Access to Your Deployment**

You should test whether your changes are functioning as you expect. This step is optional but strongly recommended.

To test access to your deployment

1. On a computer that does not host the application tier, open a web browser and navigate to a team home page.
2. Verify whether you can access your teams and projects from Team Web Access, including the administration pages.
3. If you cannot access your deployment through Team Web Access, review the steps that you just completed, and make sure that you have made all configuration changes correctly.

▣ **Configuring Your Deployment to Require HTTPS with SSL (Optional)**

You can require all connections to the TFS application tier to use HTTPS with SSL. This additional security is optional but recommended.

To require SSL connections

1. On the server that hosts the website that you want to configure, choose Start, choose Administrative Tools, and then choose Internet Information Services (IIS) Manager.
2. Follow the appropriate steps for your version of IIS:

For deployments that use IIS 7.0:

1. Expand ComputerName, expand Web Sites, and then choose the website that you want to configure.
2. On the home page for that website, choose SSL Settings.
3. In the SSL Settings pane, select the Require SSL check box.

(Optional) Select the Require 128-bit SSL check box.
4. In Client Certificates, choose Ignore, Accept, or Require, depending on the security requirements of your deployment.
5. In Actions, choose Apply.
6. Repeat these steps for each website for which you want to require SSL.

■ Installing the Certificate on Build Servers

If you installed Team Foundation Build Service on one or more servers, you must install the certificate in the Trusted Root Certification Authorities store of each server. For more information, see

[Obtaining a Certificate](#) and [Requesting, installing, and configuring websites with a certificate](#) earlier in this topic. Both the controller and the agent require a certificate with a private key with which to identify themselves in HTTPS connections.

■ Note

To perform builds over SSL, the certificate must be installed in the trusted root store on both the build controller and the build agent.

■ Updating Build Configurations

To configure Team Foundation Build for SSL connections, you must configure the build service to use the HTTPS URL that you configured for the application tier and the collection that the build configuration supports. You must configure this URL for each build configuration in your deployment.

To change a build configuration to use HTTPS

1. On the server that hosts the build configuration that you want to configure, open the administration console for Team Foundation.
2. Under Team Foundation, expand the name of the server, and then choose Build Configuration.

The Build Configuration pane appears.

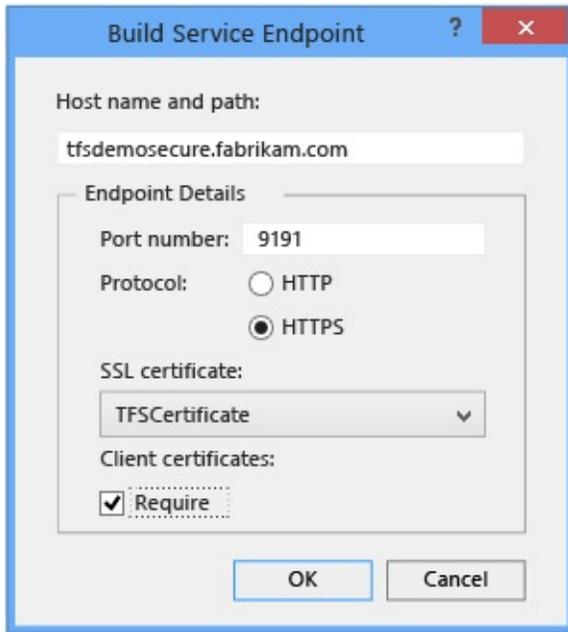
3. Under the service configuration, choose Stop, and then choose Properties.

The Build Service Properties dialog box opens.

4. In Communications, make sure that the URL for the team project collection is using the correct HTTPS address and full server name.
5. In Local Build Service Endpoint (incoming), choose Change.

The Build Service Endpoint dialog box opens.

6. In Endpoint Details, verify that the port number matches your configuration details.
7. In Protocol, choose HTTPS.
8. In the SSL Certificates list, choose the certificate that you installed and configured for use with this deployment, and then choose OK.



9. In the Build Service Properties dialog box, choose Start.

■ Release Management and TFS

You can deploy Release Management with HTTPS completely separate from TFS, regardless of the protocol you're using for TFS, or if you're using TFS at all. However you decide to deploy Release Management, the instructions for creating a secure deployment for Release Management are very similar to what's set down here for TFS. The big difference is the procedure for binding the HTTPS protocol to the Release Management website, which is covered below.

To deploy Release Management with HTTPS, use the task list below. If you're configuring Release Management with TFS, skip over any tasks you might have already completed for TFS configuration.

1. Obtain a certificate. For more information, see

[Obtaining a Certificate](#).

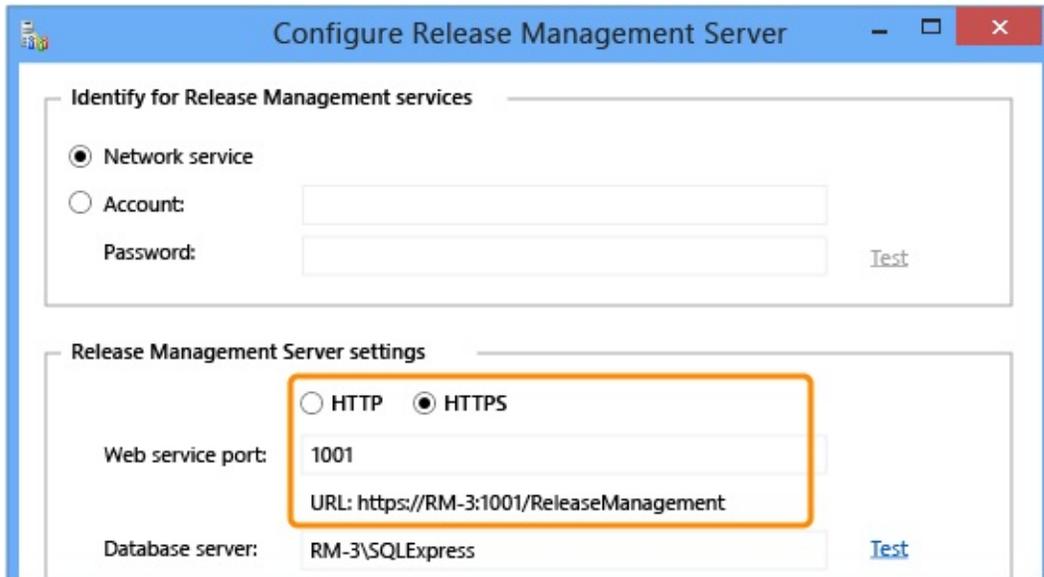
- Configure Release Management Server to use HTTPS. See the next section, [Configure Release Management Server to use HTTPS](#).
- Install the certificate in the trusted root store of any computer running Release Management Client or the Microsoft Deployment Agent. For more information, see [Configuring Client Computers](#) below.
- Open any firewalls. After you install the certificates, make sure to open any ports you used for SSL traffic. For more information, see [Configure Your Firewall](#).
- Test. The website for Release Management Server is not configured for browsing, so to test that it's available, you should connect to it with Release Management Client, connect the agents in your environment and then make a release. For more information, see [Making all the Release Management connections with HTTPS](#) and [Manage your release](#).

■ Configure Release Management Server to use HTTPS

Release Management supports either the HTTPS or HTTP protocol but not both

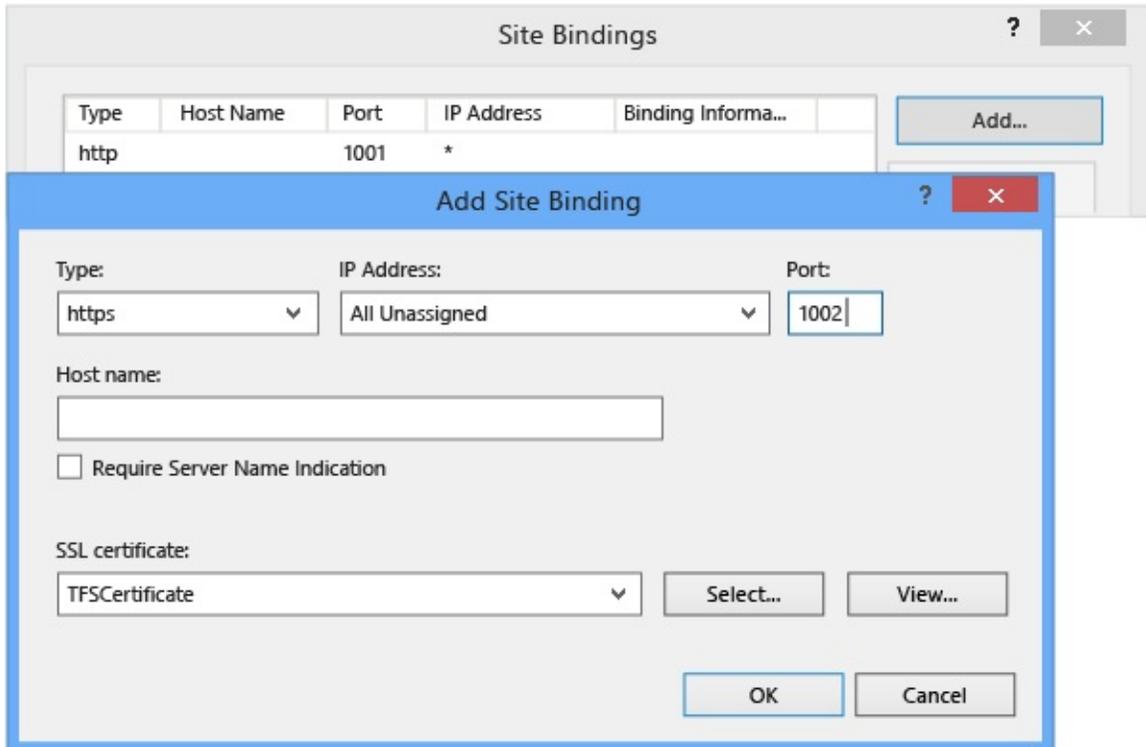
protocols at the same time. Use this procedure to bind the HTTPS protocol to the Release Management website.

1. In Release Management Server, select HTTPS, enter the port number you want to use for HTTPS traffic in Web service port, and then choose Apply settings.



The screenshot shows the 'Configure Release Management Server' dialog box. It is divided into two main sections: 'Identify for Release Management services' and 'Release Management Server settings'. In the first section, the 'Network service' radio button is selected. In the second section, the 'HTTPS' radio button is selected, and the 'Web service port' is set to 1001. The 'Database server' is set to 'RM-3\SQLExpress'. A blue box highlights the 'HTTPS' radio button, the 'Web service port' field, and the 'URL' field, which contains 'https://RM-3:1001/ReleaseManagement'. There are 'Test' buttons next to the 'Account/Password' and 'Database server' fields.

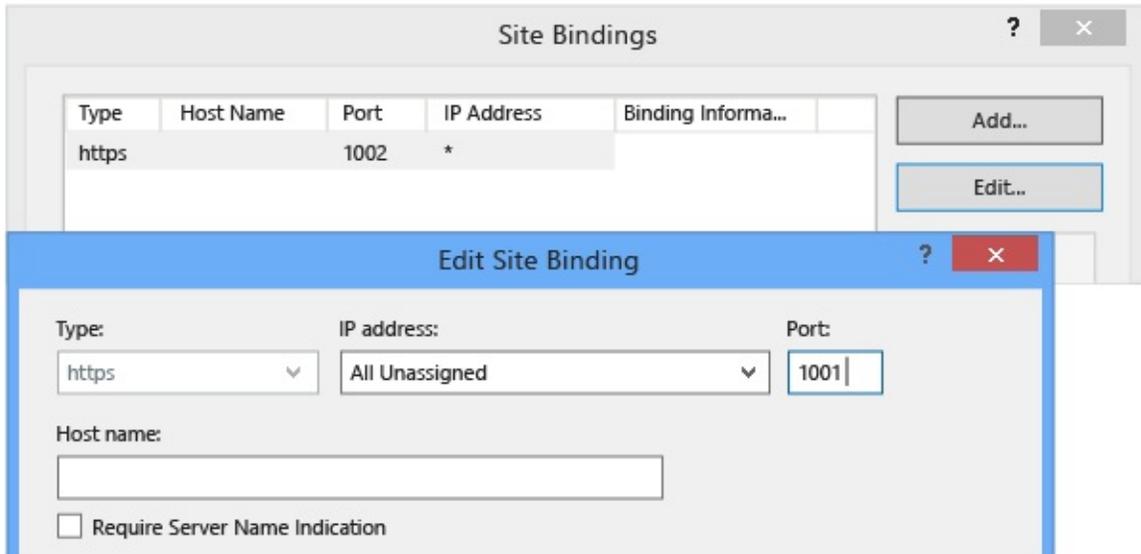
2. Open Internet Information Services (IIS) Manager.
3. Expand ComputerName, expand Sites, open the submenu for the Release Management website, and then choose Bindings from the Actions pane.
4. In Site Bindings, choose Add.
5. In the Type list, choose https.
6. In Port, type a different port number. This is a just temporary port number necessary to complete the configuration.



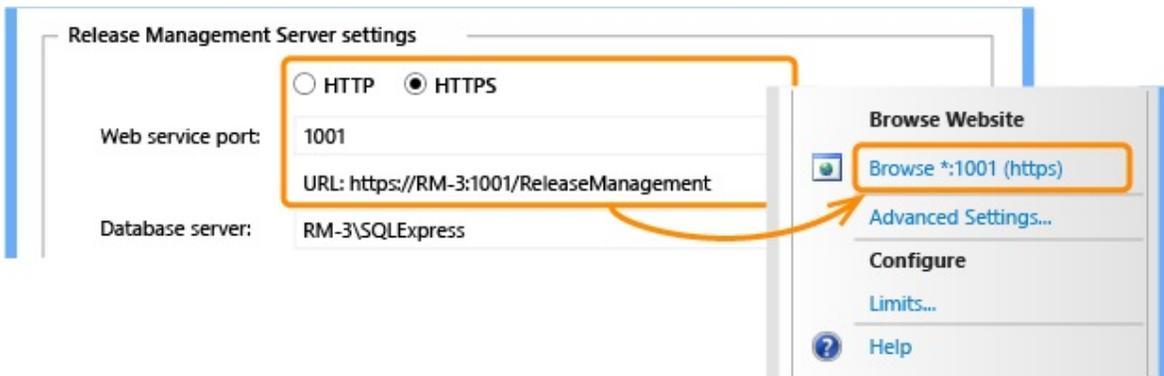
7. In SSL Certificate, choose the certificate you will use, and then choose OK and close the Bindings page.

You see the original HTTP binding, and the HTTPS binding you just created.

8. Select the original HTTP binding and choose Remove.
9. Select the HTTPS binding and choose Edit.
10. Change Port from the temporary value you added in step 6, to the port number you used in the Release Management Server in step 1, and then choose OK and Close.



The HTTPS port binding on the Release Management website in IIS matches the port that you originally entered in Release Management Server configuration tool.



■ Making all the Release Management connections with HTTPS

Once the certificates have been installed on all the computers running Release Management Client and Microsoft Deployment Agent, you can connect the computers to the Release Management Server over SSL. If TFS is running HTTPS with SSL, you must configure the TFS connection to use HTTPS.

Setting up a TFS connection for the first time? There are some additional steps and some account permission requirements. For more information, see

[Connect Release Management to TFS](#)

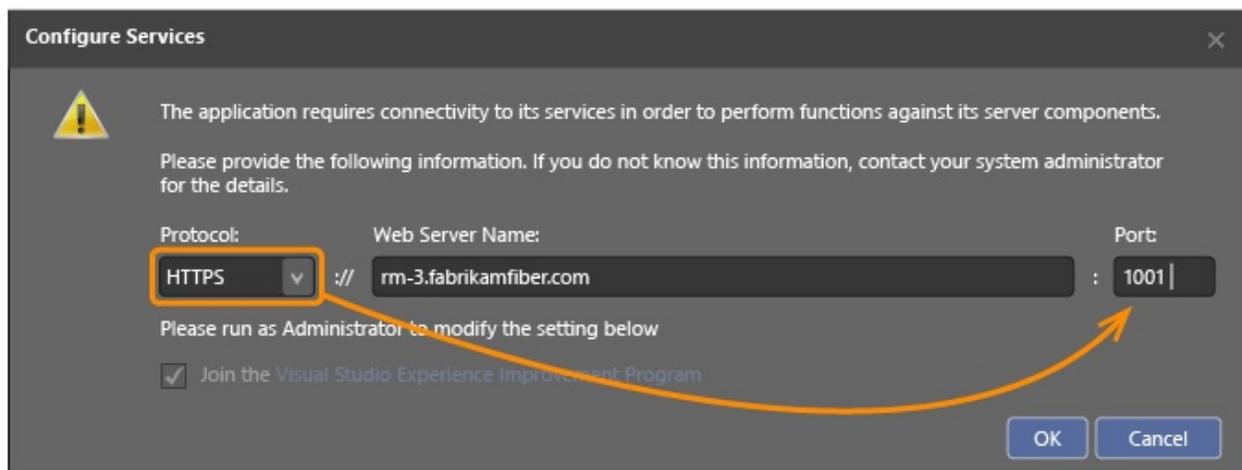
Connect Release Management Client to Release Management Server using HTTPS

1. Launch the Release Management Client.

Tip

If you get an error message saying you no longer have access to the server, you can reinstall the Release Management client, or you can use a command line tool to point the client to the server using the new port and protocol. For more information, see this [blog post](#).

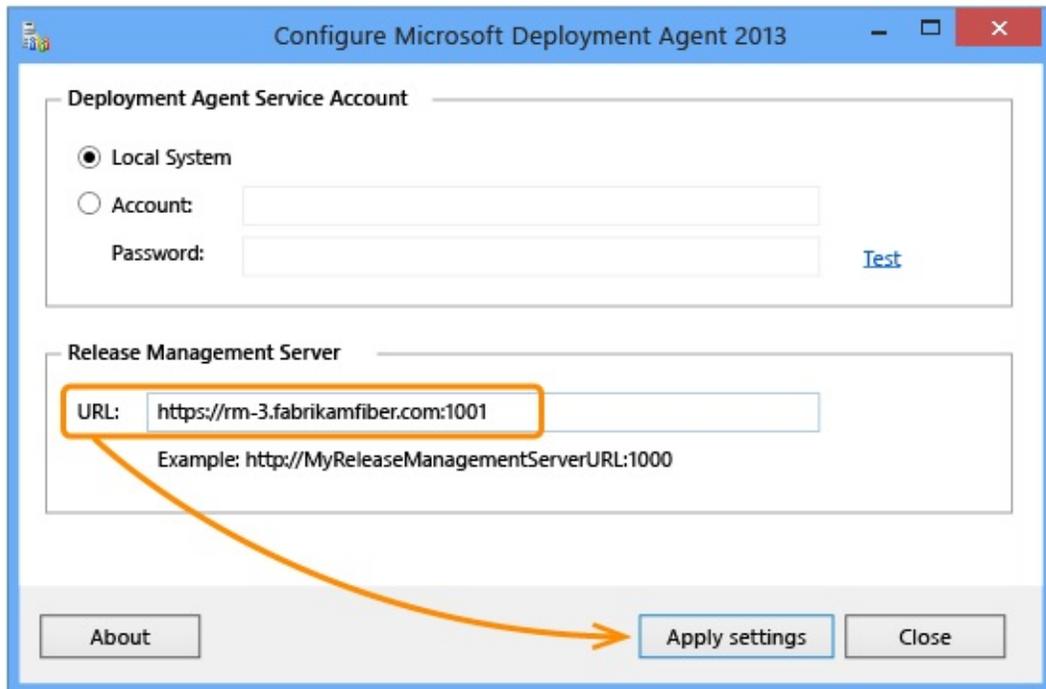
2. Choose Administration, and then choose Settings.
3. In Release Management Server URL, choose Edit.
4. In the Configure Services dialog box, select HTTPS and enter the fully qualified domain name and the SSL port of the Release Management Server, and then choose OK. You'll be prompted to restart the application.



Connect Microsoft Deployment Agent to Release Management Server using HTTPS

1. Launch the Microsoft Deployment Agent.
2. In Release Management Server, enter the URL for the Release Management

Server and choose Apply Settings. Remember to use HTTPS protocol, the fully qualified domain name for the server, and the port you set up in IIS.



Connect Release Management Server to TFS using HTTPS

1. Launch the Release Management Client.
2. Choose Administration, and then choose Manage TFS.
3. Change the protocol of the connection to HTTPS, update the port (if necessary) and choose Verify.

TFS Connection* ▶ https://rm-3.fabrikamfiber.com...

General Information

Name or URL of the TFS rm-3.fabrikamfiber.com

Project collection name DefaultCollection

Connection Details

Path tfs

Port number 443

Protocol HTTP HTTPS

URL https://rm-3.fabrikamfiber.com:443/tfs/defaultcoll...

■ **Configuring Client Computers**

On every client computer from which users access Team Foundation, you must install the certificate locally and clear the client cache for any user who has accessed Team Foundation from that computer. Otherwise, users will not be able to connect to Team Foundation from that computer. For more information, see

[Manage Trusted Root Certificates](#).

⚠ Important

Do not follow this procedure for computers that are running both Team Foundation Server and one or more clients of Team Foundation.

To install the certificate on a client computer

1. Log on to the computer by using an account that belongs to the Administrators group on that computer.
2. Install the certificate into the Trusted Root Certification Authorities folder for the local computer.

To clear the cache on a client computer

1. Log on to the computer by using the credentials of the user whose cache you want to clear.
2. Close any open instances of Visual Studio.
3. In a browser window, open the following folder:

Drive:\Users\UserName\AppData\Local\Microsoft\Team Foundation\4.0\Cache

4. Delete the contents of the Cache directory. Make sure that you delete all subfolders.

5. Choose Start, choose Run, type **devenv /resetuserdata**, and then choose OK.
6. Repeat these steps for the account of every user who has accessed Team Foundation from that computer.

☑ **Note**

You might want to distribute instructions for clearing the cache to all of your Team Foundation users so that they can clear the caches for themselves.

To connect client computers to the reconfigured deployment

- In Visual Studio, connect to Team Foundation Server by using the new HTTPS URL.

For more information, see [Connect to team projects in Team Foundation Server](#).

■ Configuring Git

By default, projects that use Git for version control will fail to validate the SSL certificate you have configured for TFS. This is because unlike TFS and Visual Studio, Git does not recognize the Windows certificate store. Instead, it uses OpenSSL for its certificate store. In order to use a Git repository for projects configured with SSL, you'll need to configure Git with the certificate at the root of the certification chain for your TFS 2013 deployment. This is a client configuration task that only applies to Git repository projects.

For more information about how Git network operations work in Visual Studio 2013, see this

[blog post](#).

📌 Tip

For other Git credential management tasks, such as Windows authentication, consider downloading and installing [Windows Credential Store for Git](#).

To configure the certificate store for Git

- Log on to the computer by using an account that belongs to the Administrators group on that computer.
- Make sure that the required certificate has been installed and configured on the computer, as per above.
- In your supported web browser, extract the TFS root certificate as a base64-encoded X.509 CER/PEM file.
- Create a private copy of the Git root certificate store and add that to your private user copy of the store.

For a full walkthrough of this procedure, including screenshots, see [Philip Kelley's blog](#).

See Also

Other Resources

[Securing Team Foundation Server with HTTPS and Secure Sockets Layer \(SSL\)](#)
[Team Foundation Server, HTTPS, and Secure Sockets Layer \(SSL\)](#)

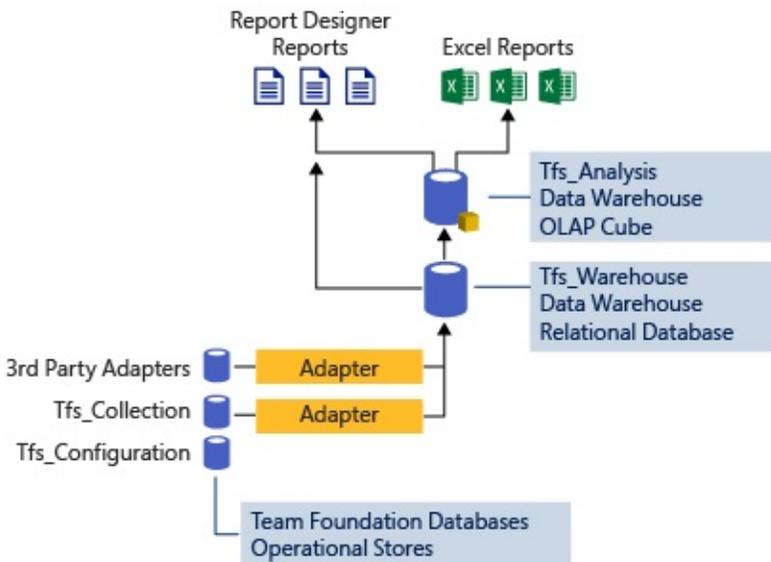
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Manage TFS reports, data warehouse, and analysis services cube

[Send Feedback](#)

If you added SQL Server Reporting Services at installation, then your on-premises Team Foundation Server (TFS) deployment is configured with a data warehouse and SQL Server Analysis Services cube. If you didn't add these services previously and want to add them now, see [Add reports to a team project](#).

The reporting warehouse is a traditional data warehouse that consists of a relational database and an Analysis Services database.



All reportable data from all team projects that are defined in all project collections for a TFS deployment is written to a single relational database (Tfs_Warehouse). Data from that warehouse is then processed and written to the Analysis Services cube (Tfs_Analysis). Collecting data into a single data warehouse supports reporting across team projects and project collections. To learn more, see [Components of the TFS data warehouse](#).

With SQL Server Reporting Services, you gain access to many default Excel and SQL Server Reporting Services reports. These reports aggregate metrics from work items, version control, test results, and builds. See [Create, customize, and manage reports for Visual Studio ALM](#).

Without these services, you can create status and trend charts from work item queries directly from the operational data stores.

▣ **View, add, update, or customize reports or report functionality**

You use work item fields to track data for a work item type, to define the filter criteria for queries, and to design reports. To support reporting, you can add fields or change the attributes of existing fields. When you add or modify fields, you will want to apply systematic naming conventions to make sure that data is logically grouped into folders in the cube. To learn more, see [Add or modify work item fields to support reporting](#).

To bulk add or update reports for a team project, see [Upload reports to a team project](#).

To manage Reporting Services Reports, see [Reporting Services Reports \(SSRS\) Upload reports to a team project](#).

Depending on the process template that you use to create your team project, you may have several reports already defined. You can customize these reports additionally or create new reports. These reports may contain new data fields that you added to work item types. See [Create, customize, and manage reports for Visual Studio ALM](#).

■ **Manage permissions to view and create reports**

To create reports that access data in the cube, you must add team members to the TFSWarehouseDataReader role. To view or refresh data within a report, you must add team members to one or more of the Report Server roles. See [Grant permissions to view or create reports in TFS](#).

A user who has permission to view the data in the warehouse can view data for all team projects that are hosted in all team project collections in the TFS deployment. There is no way to limit access to a team project or collection.

Manage and troubleshoot the data warehouse and analysis services cube

You can manage the warehouse and analysis services cube to address the following scenarios:

- Correct errors that block processing of the data warehouse. Schema conflicts occur when a set of attributes for reportable fields differs across team project collections. Schema conflicts block updated data from being moved into the warehouse and the cube. You must correct all schema conflicts to unblock processing and to update reports with current data.

See [Resolve schema conflicts that are occurring in the data warehouse](#).

- Update the data warehouse or cube on demand. Resolve a problem with reports that are not up to date or contain missing data. To resolve a problem with reports that do not update or contain missing data, you may have to process the data warehouse manually. Also, you can troubleshoot errors that appear in the event viewer for an application-tier server that are related to warehouse processing jobs.

See [Manually process the TFS data warehouse and analysis services cube](#).

- Re-create the schemas, and rebuild the data warehouse databases. Rebuild the warehouse and cube after you move, restore, rename, or fail over the data-tier server for Team Foundation. To access high-level reports, you must rebuild the data warehouse whenever you move, restore, rename, or fail over the data-tier server.

See [Rebuild the TFS data warehouse and cube](#).

- Modify the refresh frequency of the data warehouse or cube. Change the refresh frequency or other process control setting for the warehouse or cube. The default properties for the warehouse are set when TFS is installed, but you can later change the default values to respond to changing requirements. Two properties that you might want to change are the

frequency with which the data is updated in the data warehouse and the security settings that control user access to the data warehouse.

See [Change a process control setting for the data warehouse or Analysis Services cube](#).

Q & A

Q: Where can I learn more about reportable attributes for each field?

A: A default set of fields appears in the relational warehouse database or the cube. These fields have a reportable attribute value of **Detail**, **Dimension**, or **Measure**. To look up the definition for a reportable field, see [Reportable fields reference for Visual Studio ALM](#).

Q: What are the relationships and dependencies between SQL Server and TFS?

A: See Understanding SQL Server and SQL Server Reporting Services.

Q: Can I add or remove data from a team project collection from the data warehouse?

A: You can enable and disable data that flows into the data warehouse by editing the reporting configuration for your team project collections. After you add a report server to your deployment, you can configure reporting resources for your team project collections and the projects in those collections. See [Manage team project collections](#).

Q: How do I add new data types to the data warehouse?

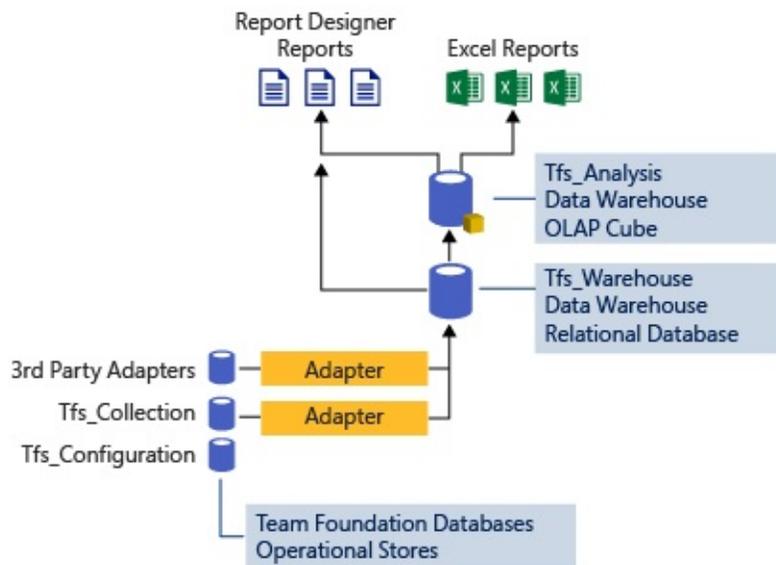
A: You can add new data types to the data warehouse by implementing a warehouse adaptor. To learn more, see [Data Warehouse extensibility](#).

- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Components of the TFS data warehouse

[Send Feedback](#)

The Team Foundation reporting warehouse is a traditional data warehouse consisting of a relational database organized in an approximate star schema and a SQL Server Analysis Services cube built on top of the relational database. The following diagram shows the high-level architecture of the Team Foundation data warehouse and the relationships between the operational stores, the data warehouse, and the team reports.



Operational stores

Each tool or plug-in in Team Foundation uses a relational database in SQL Server 2008 to store the data used by the tool in its day-to-day operations. This relational database is often referred to as the operational store. The operational stores for Team Foundation include:

- Common structure databases (Tfs_Configuration)
- Team project collection databases (Tfs_Collection)

You might also have operational stores created for third-party tools.

Like most operational stores, the schema of the relational database is designed and optimized for the online transactional processing of data. As the tool or plug-in performs an activity, it writes the latest information to the operational store. Therefore, data in the operational store is constantly changing and being updated, and all data is current.

Warehouse adapters

Because each tool or plug-in has its own schema requirements and data is stored in the operational store to optimize transactional processing, the purpose of the warehouse adapter is to put the operational data into a form usable by the data warehouse. The warehouse adapter is a managed assembly that extracts the data from the operational store, transforms the data to a standardized format compatible with the warehouse, and writes the transformed data into the warehouse relational database. There is a separate adapter for each operational data store.

The warehouse adapter copies and transforms those data fields specified in either the basic warehouse configuration or in the process template used at the time a new team project is created. If you subsequently change the process template to add or delete which data fields are written to the data warehouse, these changes are detected the next time the adapter is run. The adapter runs periodically with a frequency set by the `RunIntervalSeconds` property. The default setting for the refresh frequency is two hours (7,200 seconds), so give careful consideration to the appropriate refresh frequency for your installation. For more information about changing the refresh frequency, see

[Change a process control setting for the data warehouse or Analysis Services cube.](#)

It is important that data is not written from the relational database to the data cube while the relational database is itself being updated from the operational store. To avoid conflicts reading and writing data, the warehouse adapters that push and pull the data are synchronized. After the adapters have completed their calls, the cube is reprocessed.

■ The relational database or data warehouse

Each tool describes its contribution to the data warehouse in an XML schema. The schema specifies the fields that are written to the relational database as dimensions, measures, and details. The schema is also mapped directly into the cube.

The data in the warehouse are stored in a set of tables organized in a star schema. The central table of the star schema is called the fact table, and the related tables represent dimensions. Dimensions provide the means for disaggregating reports into smaller parts. A row in a fact table usually contains either the value of a measure or a foreign key reference to a dimension table. The row represents the current state of every item covered by the fact table. For example, the Work Item fact table has one row for every work item stored in Work Item operational store.

A dimension table stores the set of values that exist for a given dimension. Dimensions may be shared between different fact tables and cubes, and they may be referenced by a single fact table or data cube. A Person dimension, for example, will be referenced by the Work Items fact table for Assigned To, Opened By, Resolved By, and Closed By properties, and it will be referenced by the Code Churn fact table for the Checked In By property.

Measures are values taken from the operational data. For example, Total Churn is a measure that indicates the number of source code changes in the selected changesets. Count is a special measure in that it can be implicit, as long as there is one record for every item that is counted. The measures defined in a fact table form a measure group in the cube.

For more information about the facts, dimensions, and measures in the data warehouse, see

[Perspectives and measure groups provided in the Analysis Services cube for Visual Studio.](#)

■ **The Analysis Services cube**

Fact tables are a good source of information for reports that show the current state of affairs. However, to report on trends for data that changes over time, you need to duplicate the same data for each of the time increments that you want to report on. For example, to report on daily trends for work items or test results, the warehouse needs to retain the state of every item for each day. This allows the data cube to aggregate the measures by day. The cube aggregates both data from the underlying star schema and time data into multidimensional structures.

Each time the data cube is processed, the data stored in the star schemas in the relational database are pulled into the cube, aggregated, and stored. The data in the cube is aggregated so that high-level reports, which would otherwise require complex processing using the star schema, are simple select statements. The cube provides a central place to obtain data for reports without having to know the schema for each operational store and without having to access each store separately.

Report Designer reports

Report Designer is a component of Visual Studio that allows you to define the Team Foundation data warehouse as a data source and then design a report interactively. Report Designer provides tabbed windows for Data, Layout, and Preview, and you can add datasets to accommodate a new report design idea, or adjust report layout based on preview results. In addition to the Data, Layout, and Preview design surfaces, Report Designer provides query builders, an Expression editor, and wizards to help you place images or step you through the process of creating a simple report. For more information about using Report Designer, see [Create, customize, and manage reports for Visual Studio ALM](#).

Excel reports

Team Foundation integrates with Microsoft Excel to allow you to use Microsoft Excel to manage your project and produce reports. Microsoft Excel provides pivot tables and charts for viewing and analyzing multi-dimensional data. You can bind these pivot tables directly to the Team Foundation cube, so you can interact with the data in the cube. For more information about using Microsoft Excel for reporting, see [Create Excel reports from a work item query](#).

Security

Security for the Team Foundation data warehouse is defined at the database level, while security for team reports is at the team project level. The Team Foundation Server administrator determines who has access to the data in the data warehouse by granting or revoking permissions on the user's account. By default, write access to the warehouse is restricted to a service account under which the warehouse service runs. Each tool adapter has write access to the data warehouse because it runs in this security context. Read-only access is granted by the administrator to individual users or groups of users. A user who has permission to view the data in the warehouse has full access to all of the data for all team projects in all team project collections. For more information about granting or denying read-only access to the data warehouse, see

[Grant permissions to view or create reports in TFS.](#)

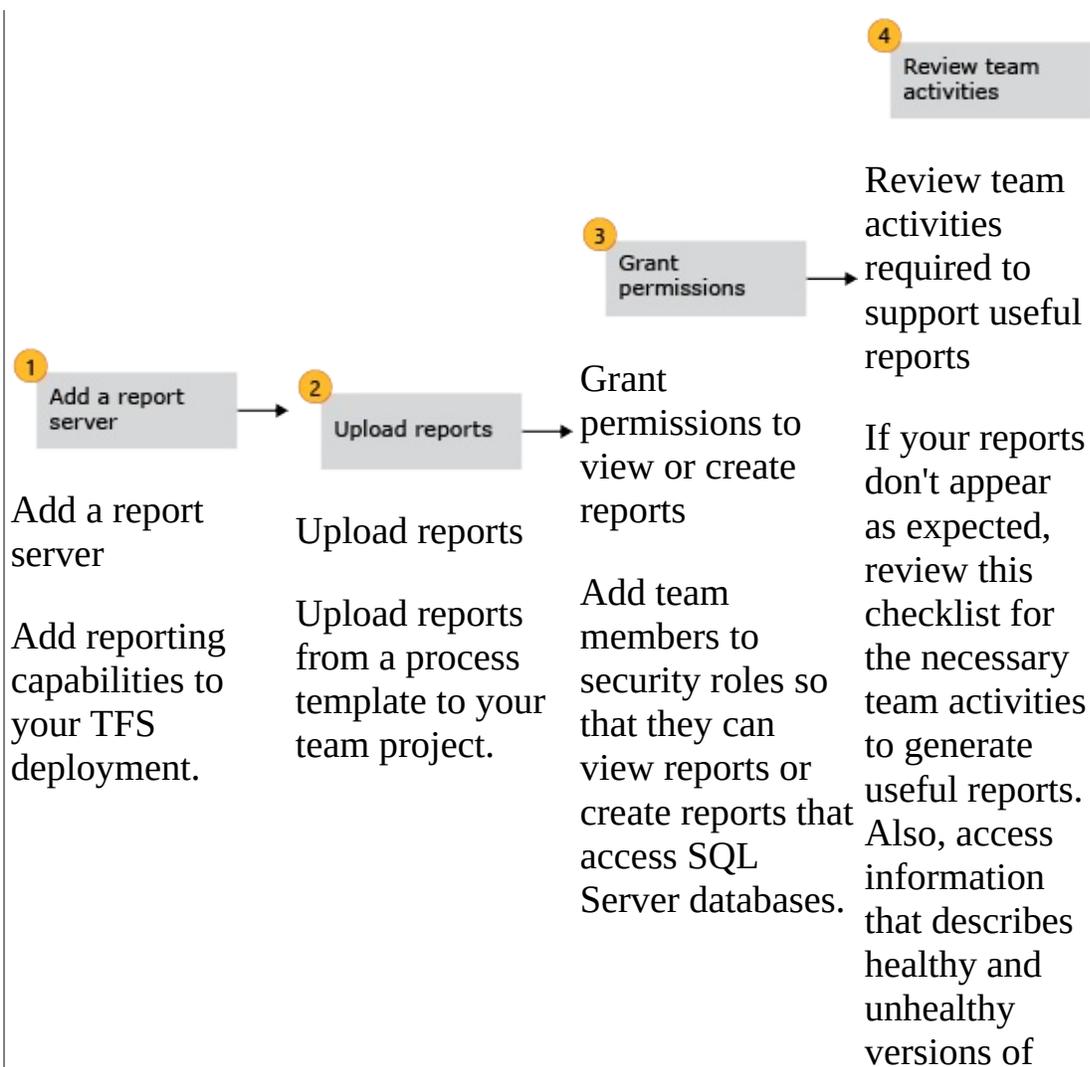


- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Add reports to a team project

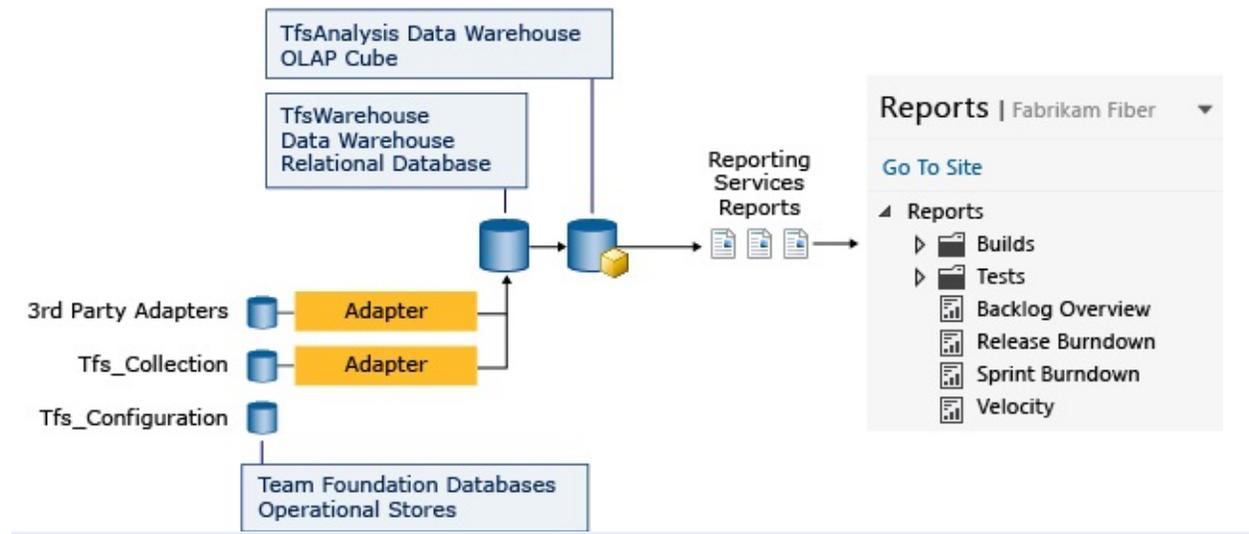
[Send Feedback](#)

By adding a report server to your TFS (on-premises) deployment, you can access a wealth of data about your team's projects, such as build quality, bug trends, burndown, and test progress. SQL Server Reporting Services (SSRS) reports provide insight to help teams manage work and improve processes.



each report.

After completing the sequence of tasks, you'll be able to access the default reports provided with the process template used to create your team project.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

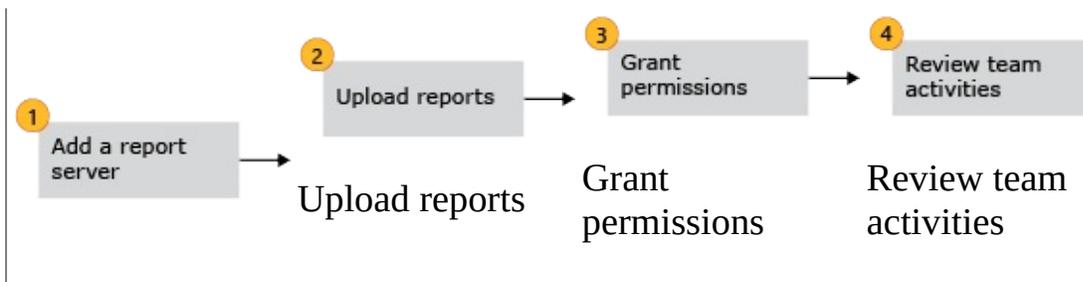
Add a report server

[Send Feedback](#)

On-premises installations of TFS can include reports to help you manage your software development projects. However, you need a report server as part of your deployment in order to use them. If you don't have one, you can add SQL Server Reporting Services to your deployment.

Tip

If you're using Visual Studio Online, adding a report server isn't an option for you at this time. Additionally, if you installed TFS on a client operating system, such as Windows 7, you can't add reporting as described here, because it isn't supported on client operating systems.



■ What do I need to know before adding a report server?

Q: What kind of report server can I add?

A: You'll need to add SQL Server Reporting Services to provide a report server for TFS. This can be added to the SQL Server instance that is supporting TFS, or to a different instance.

Q: What permissions do I need?

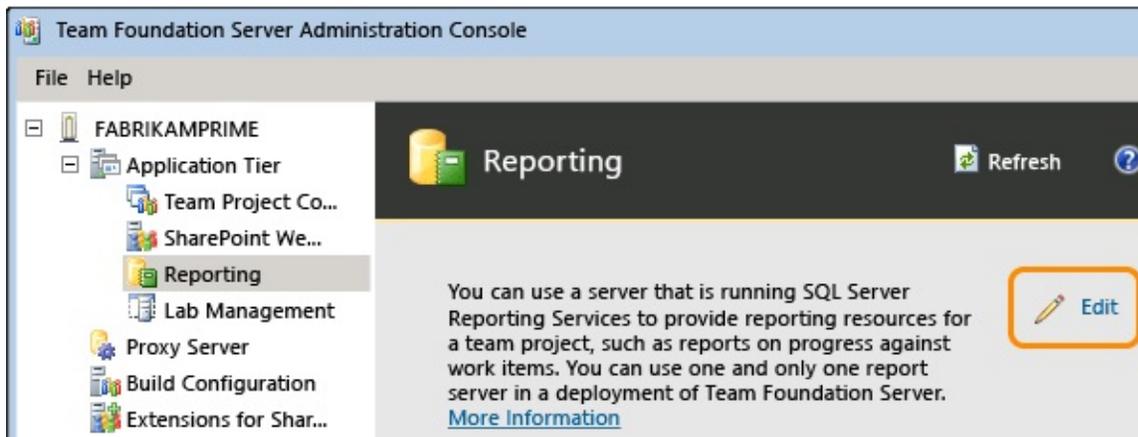
A: You need to be a member of the local administrators group on the server, the Team Foundation Administrators group in TFS, and the sysadmin group in SQL Server, or have the equivalent permissions. If you don't have them, [get those permissions now](#).

Q: Can I add reporting if TFS is installed on a client operating system?

A: If it's a client operating system, you can't add reporting and analysis services directly. Your choices aren't easy. If you still want to add reporting, either [move the team project collection](#) to a deployment of TFS that supports reporting, or [restore your entire deployment to a different server](#) that is running a server operating system and a version of SQL Server that supports reporting.

Q: How do I find out if I already have reporting available for TFS?

A: Look in the administration console for Team Foundation Server. If you see a reporting node and it shows that reporting is configured and enabled, it's already there.



Jump to the bottom of this topic to confirm that reports are available for the team project collection, and then move on to [Upload reports to a team project](#).

Q: Do I need to back up my data first?

A: You should have a recent backup available. If you don't, you can make a backup using Scheduled Backups.

▣ **Verify your options and upgrade if necessary**

1. Find out what version and edition of SQL Server you have [by following the instructions here](#). If the answer is SQL Server Express, keep following these steps. If it's any other edition, but you don't see any version and edition information about reporting or analysis services, jump ahead to [Add Reporting Services](#). If you do see information about these services, then your SQL Server already has them installed, and you can jump ahead to [Add Reporting to TFS](#).
2. If your deployment is using SQL Server Express, [upgrade to a different edition](#). For more information about the editions that support reporting in TFS, [go here](#).

Add Reporting Services and Analysis Services to SQL Server

After you confirm that you have an edition of SQL Server that supports reporting (or you upgrade to one), add the reporting and analysis services features if they aren't already configured.

1. Open the SQL Server Installation Center to add features to an existing installation.



You might have to restart your computer during this process, particularly if you've just finished upgrading from SQL Server Express.

2. Once your server has passed all the setup rules checks, pick the instance to add features to. Tip: If you upgraded from SQL Server Express, you'll have a named instance called SQLEXPRESS.

Installation Type

Perform a new installation or add features to an existing instance of SQL Server 2012.

Setup Support Rules

Installation Type

Feature Selection

Installation Rules

Disk Space Requirements

Error Reporting

Installation Configuration Rules

Ready to Install

Installation Progress

Complete

Perform a new installation of SQL Server 2012
 Select this option if you want to install a new instance of SQL Serv... components such as SQL Server Management Studio or Integratio...

Add features to an existing instance of SQL Server 2012
 Select this option if you want to add features to an existing instan... want to add the Analysis Services features to the instance that con... within an instance must be the same edition.

Installed instances:

Instance Name	Instance ID	Features	Edition
SQLEXPRESS	MSSQL11.SQLEXP...	SQLEngine,SQLEn...	Enterp...

3. In Feature Selection, add these features:

- Analysis Services
- Reporting Services - Native
- Management Tools - Complete (not pictured)

Feature Selection

Select the Enterprise features to install.

Setup Support Rules

Installation Type

Feature Selection

Installation Rules

Disk Space Requirements

Server Configuration

Analysis Services Configuration

Reporting Services Configura...

Error Reporting

Installation Configuration Rules

Features:

Instance Features

Database Engine Services

- SQL Server Replication
- Full-Text and Semantic Extractions for Search...
- Data Quality Services

Analysis Services

Reporting Services - Native

Shared Features

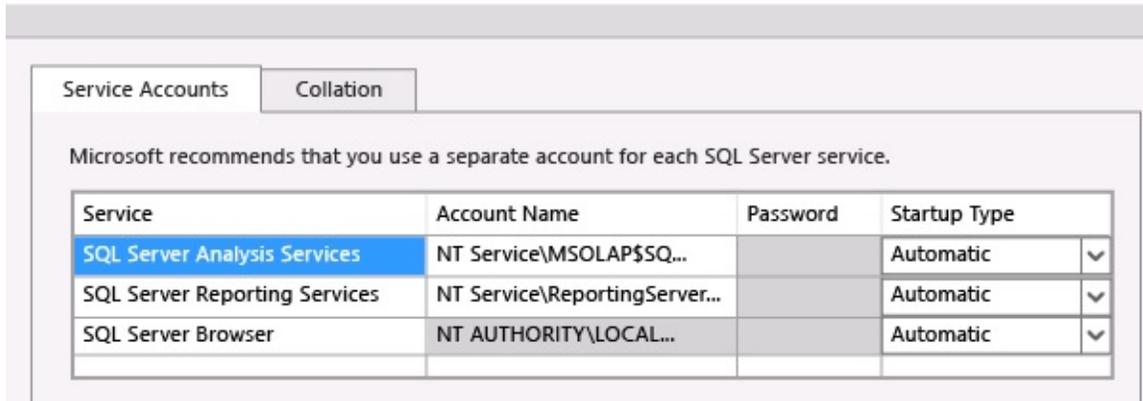
Reporting Services - SharePoint

Reporting Services Add-in for SharePoint Prod...

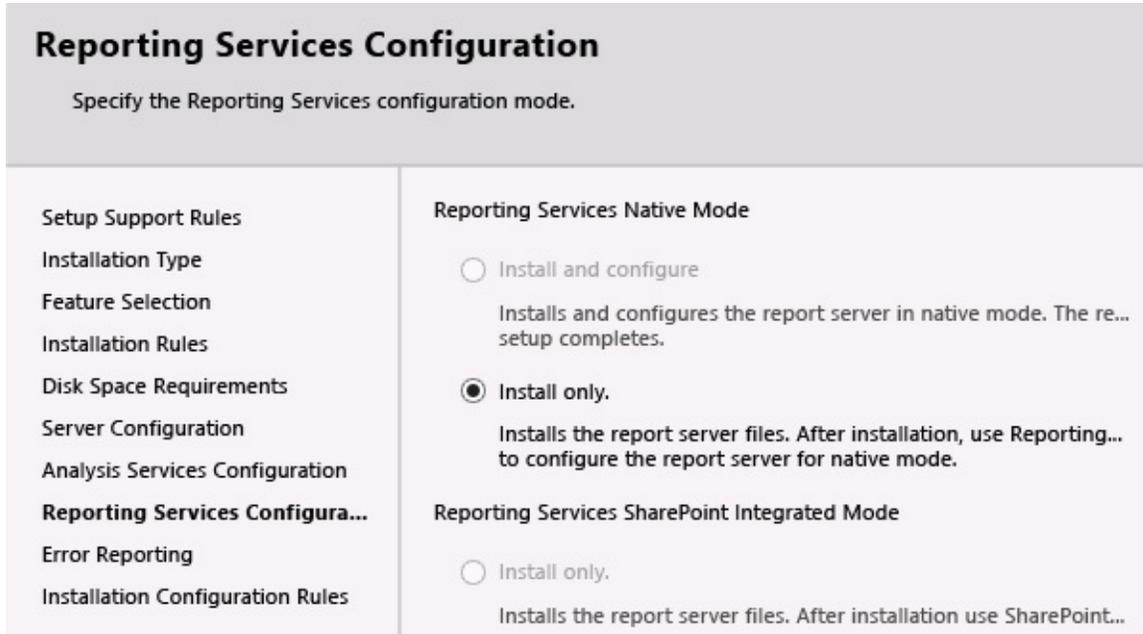
Fe...

Pr...

- Accept the defaults for service accounts in Server Configuration, unless your deployment has specific service account needs. Make sure all services are set to start automatically.



- Add your user account in Analysis Services Configuration.
- Choose Install and Configure for reporting services if you can; if not, choose Install Only. If you're adding features after upgrading from SQL Server Express, Install Only might be the only option available.



- Finish the wizard, including any server restarts that might be required.

If you had to choose Install Only when you added reporting, you'll need to open Reporting Services Configuration Manager and do some additional

configuration. If not, jump ahead to

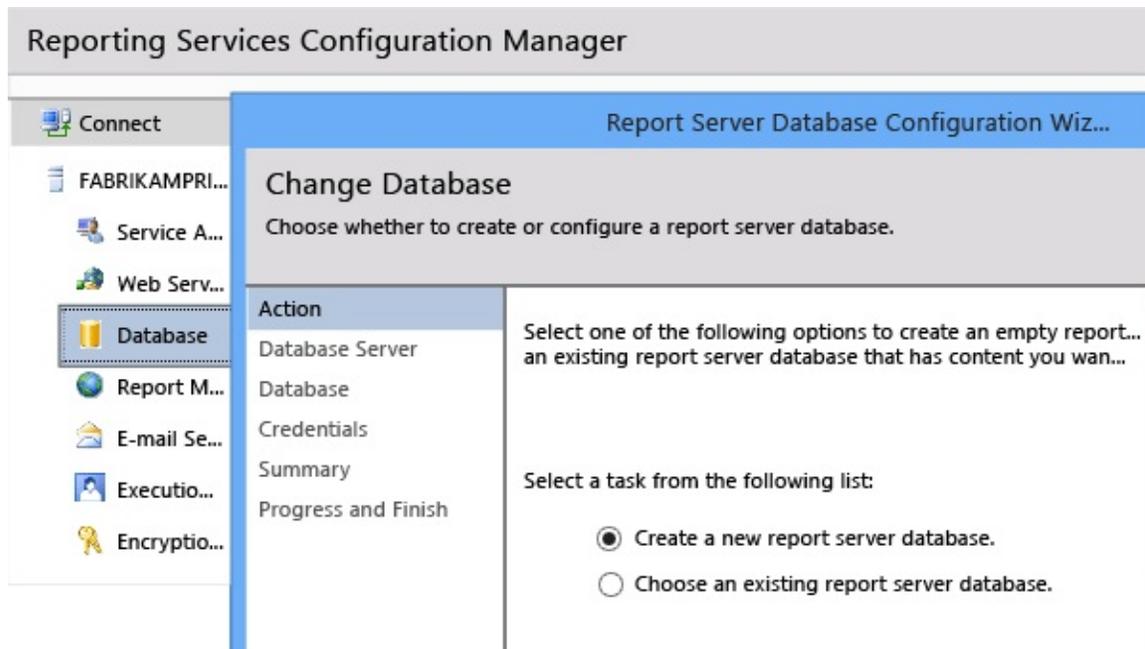
[Add Reporting to TFS.](#)

Configure Reporting Services Manually

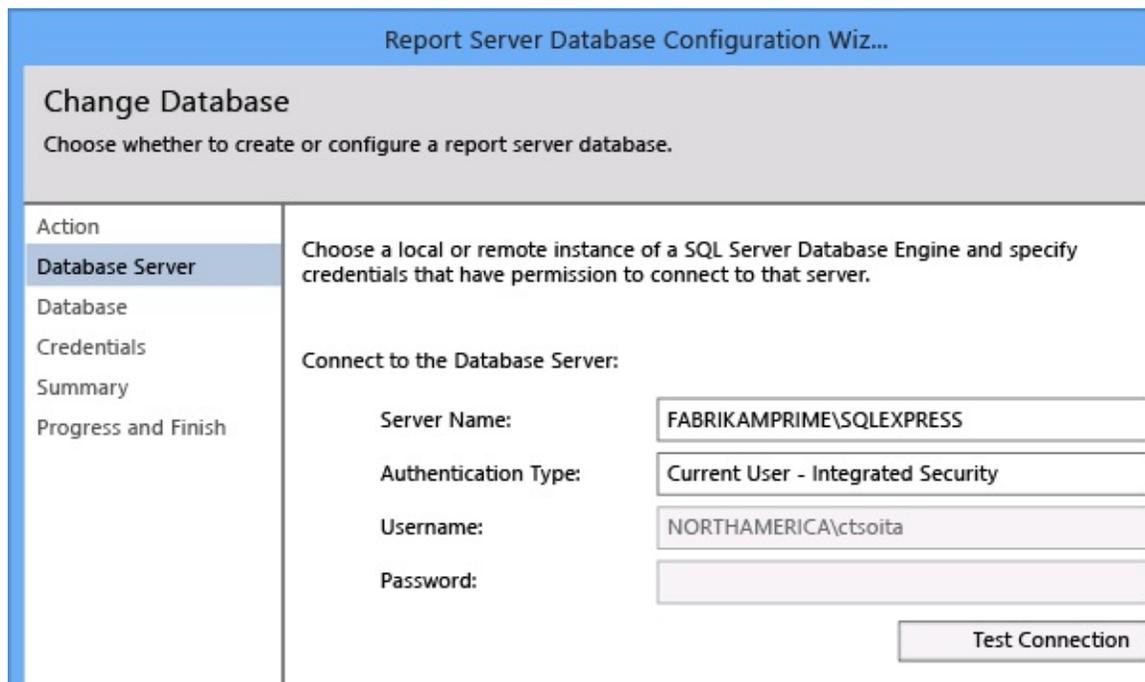
1. Open Reporting Services Configuration Manager and connect to your report server.



2. Once you've connected, make sure the service is started, then go to the Web Service URL page and apply all the defaults.
3. Now you'll need to create a database for reporting. On the Report Server Database page, the Change Database option will open the Report Server Database Configuration Wizard assist you in creating that database.



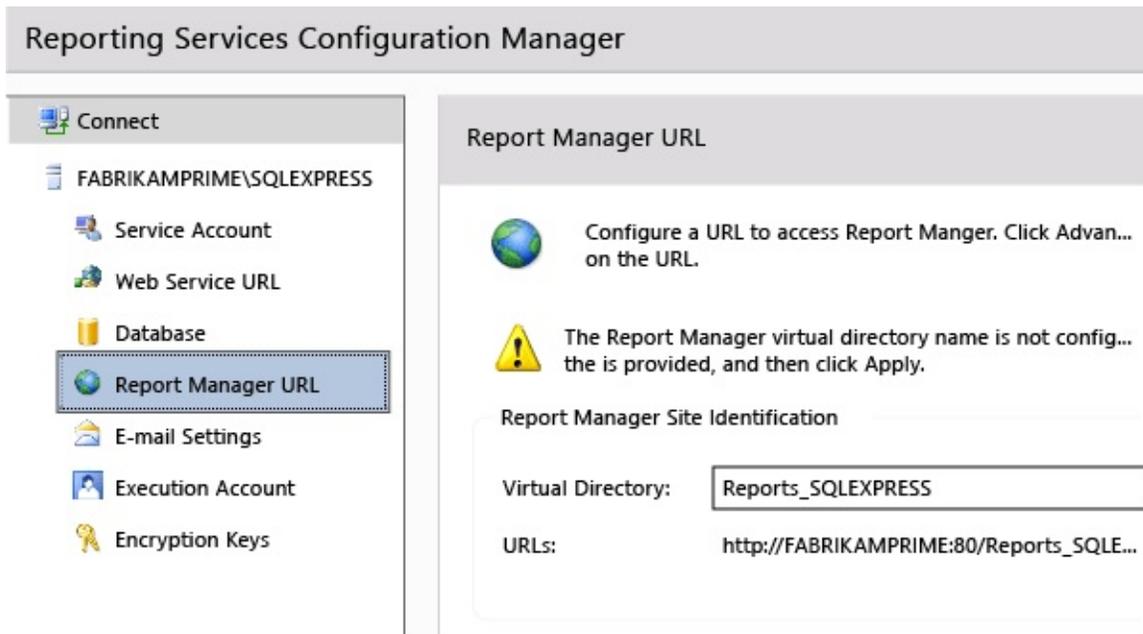
4. When you specify the database server connection, include the instance name as well as the server name, separated by a slash (\).



If you aren't using a named instance, you can just provide the server name.

5. Accept the default values on the rest of the pages of the wizard and wait for it to finish. This can take a few minutes.

6. Accept all the defaults on the Report Manager URL page.

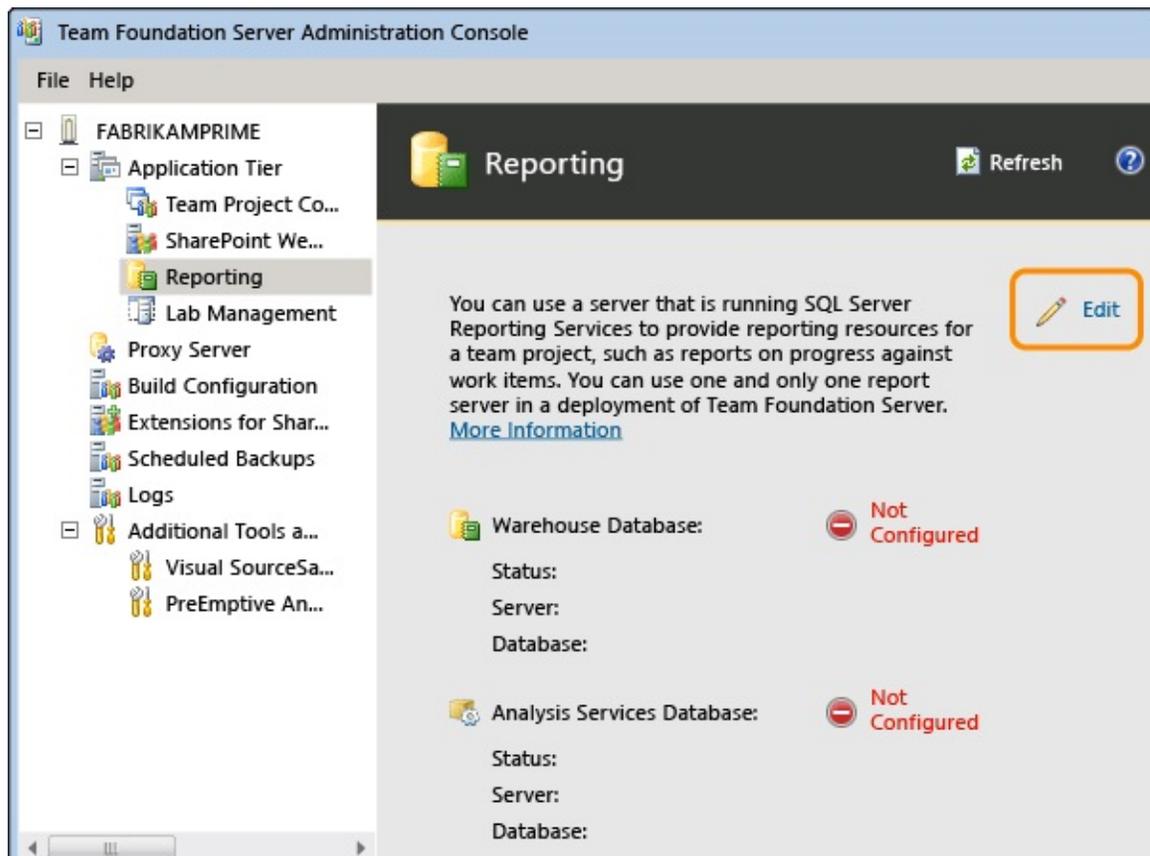


You've now fully configured reporting on the SQL Server instance that supports TFS. It's time to add the reporting features to TFS!

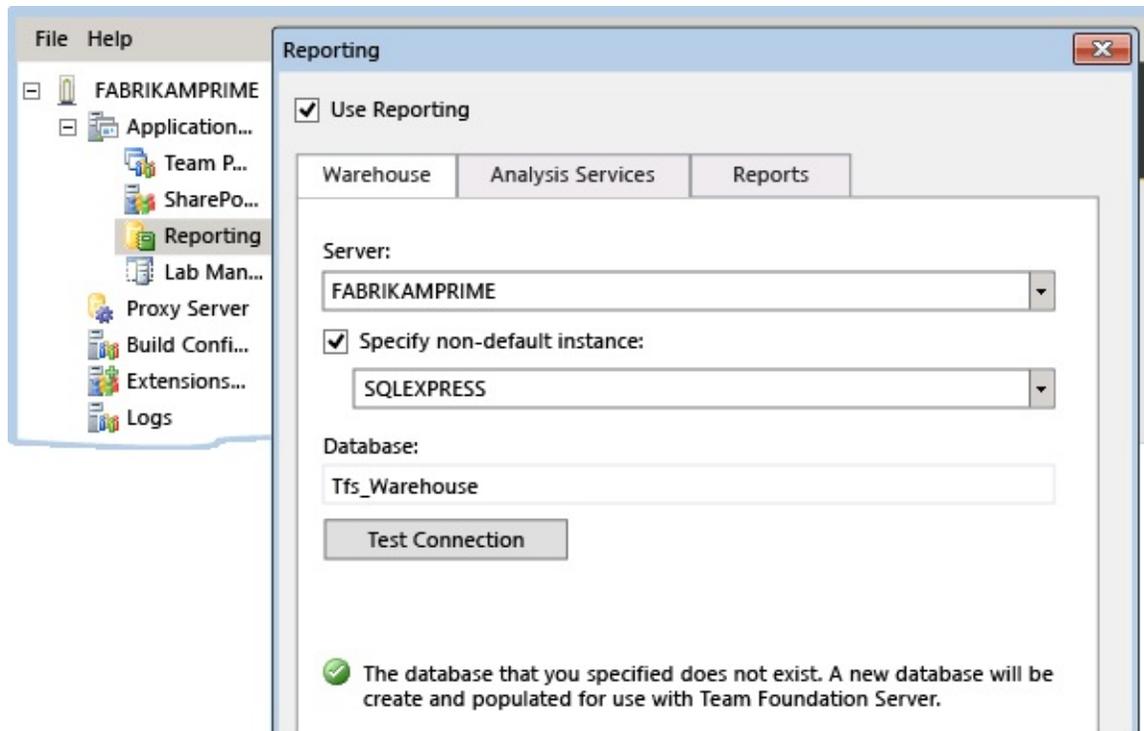
Add Reporting to TFS

You'll need to add reporting in two places: to TFS itself, and then to your team project collection.

1. Open the administration console for TFS and go to the Reporting node to start the configuration process.



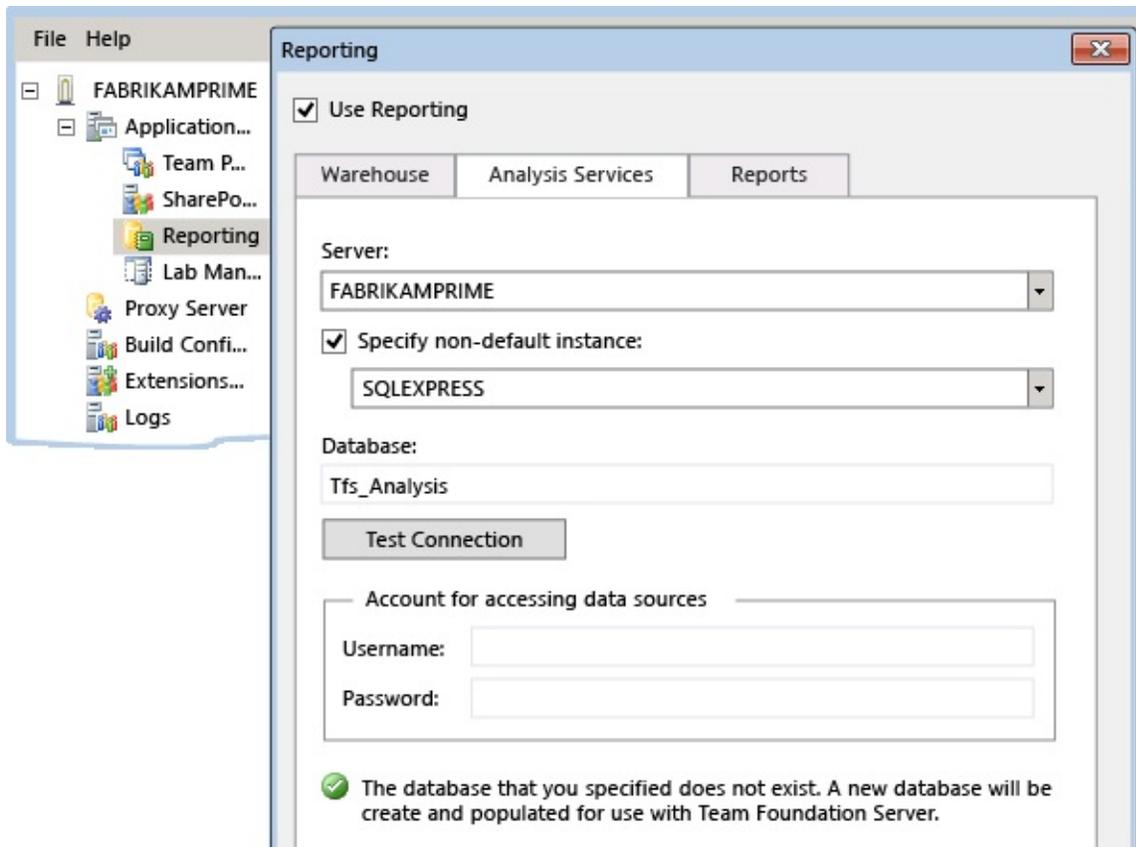
2. Select the Use Reporting check box. On the Warehouse tab, provide the name of the report server and instance for the warehouse database. Use the default name for the database (TFS_Warehouse), and test the connection.



Tip: Because you're configuring reporting for the first time, the databases don't exist yet. They will be created when you finish the configuration process.

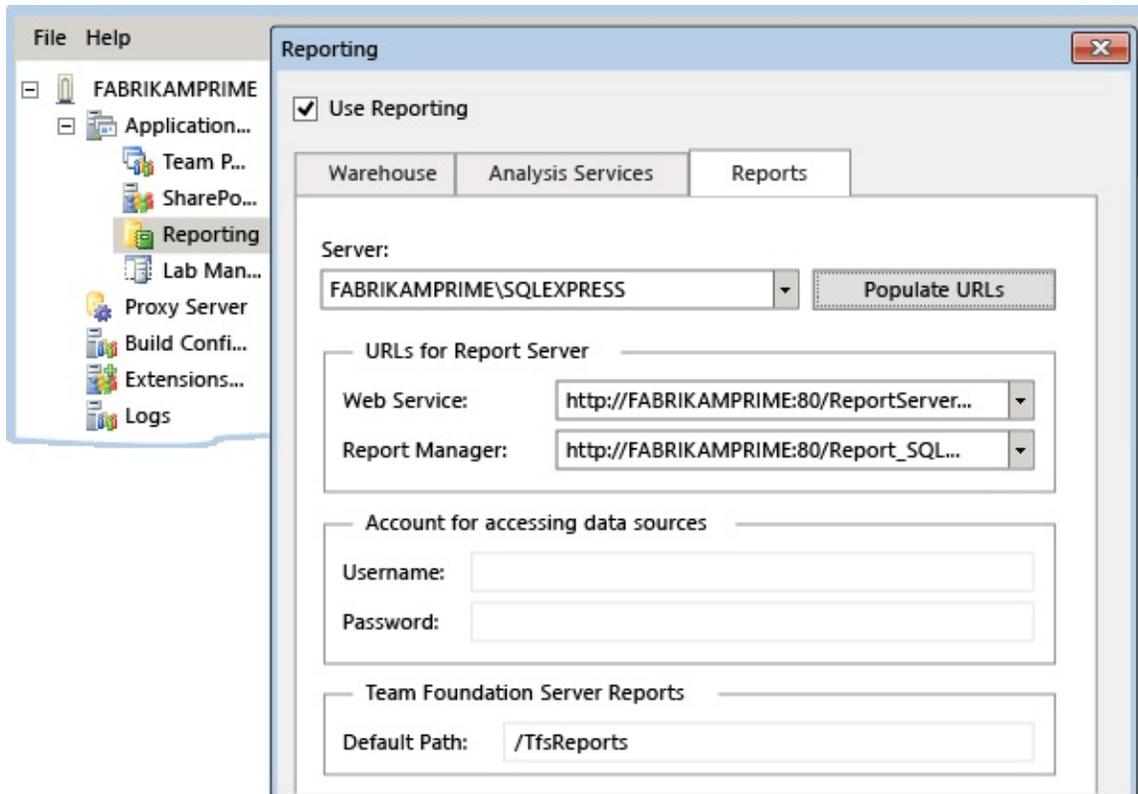
3. Provide the same server and instance information for Analysis Services, but use the default name for the analysis database (TFS_Analysis). Provide an account name and password for a data sources account, a special account you've created just for this purpose. (You can read more about these accounts

[here.](#))

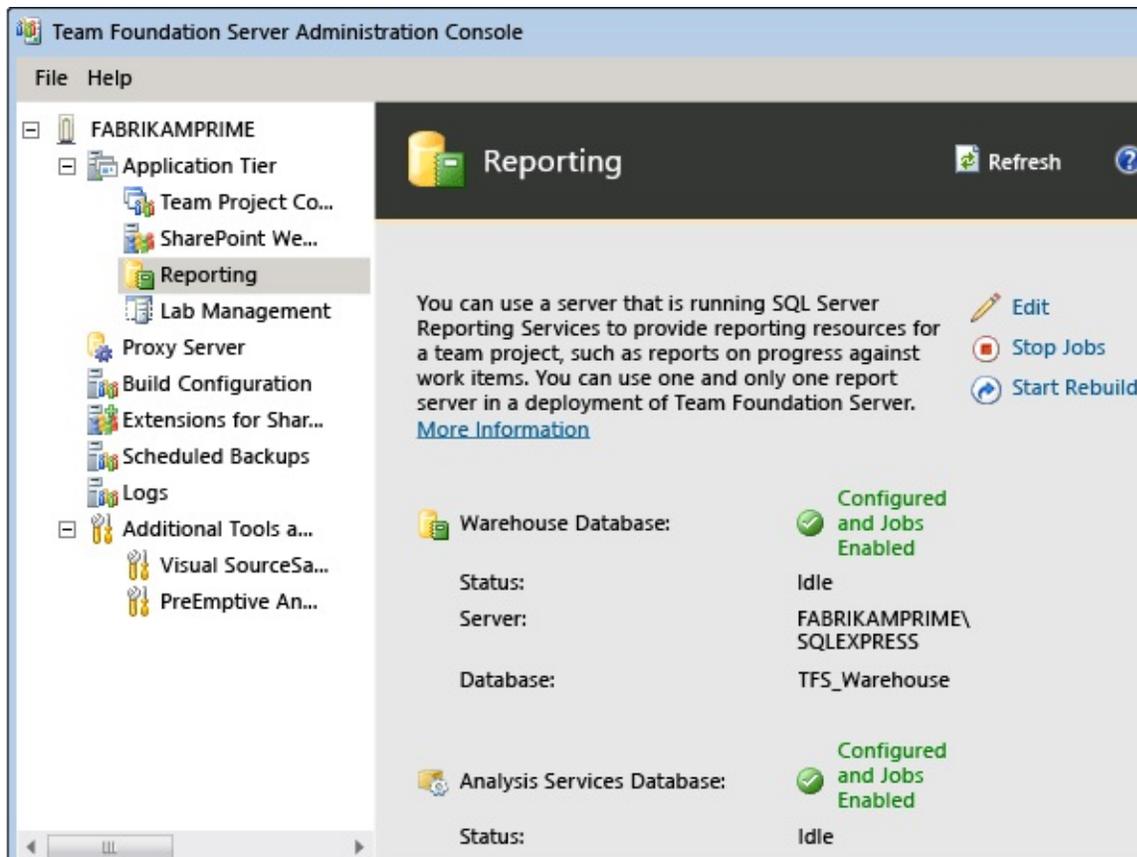


Tip: Test the connection before continuing.

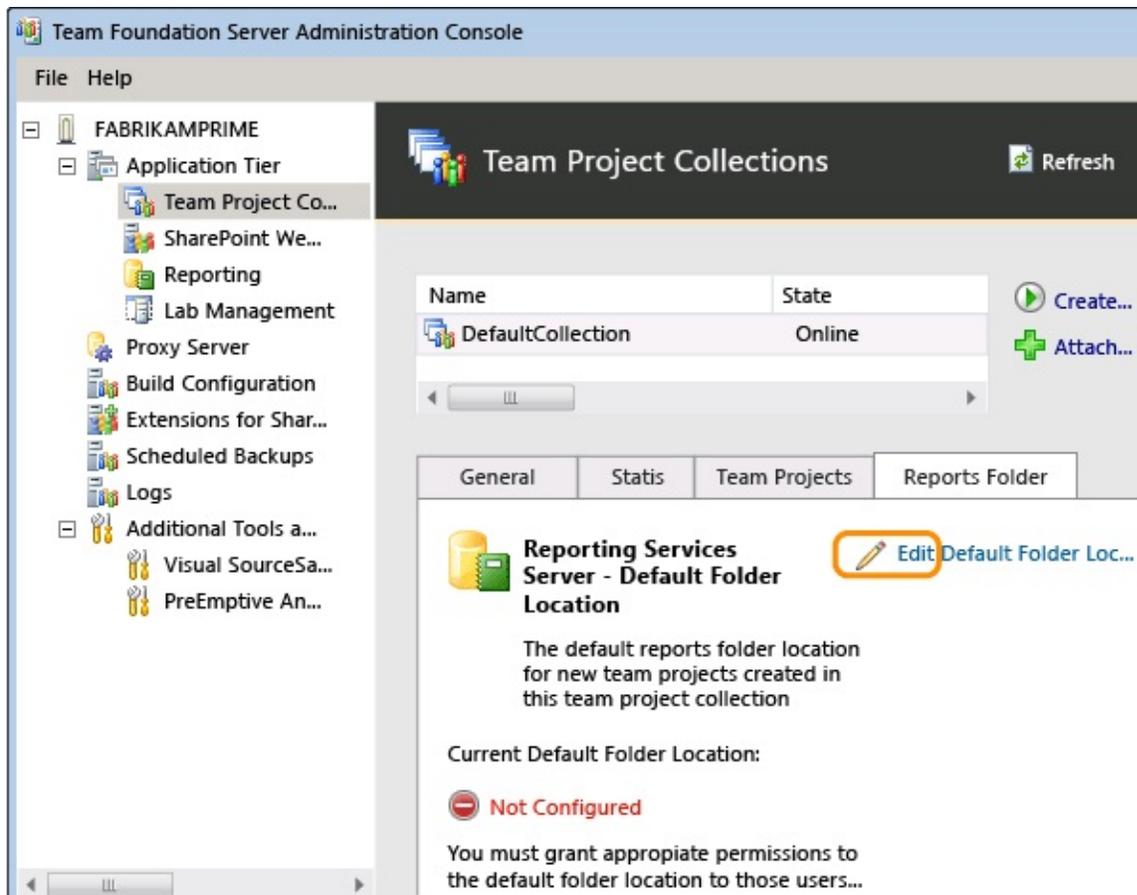
- For Reports, provide the server name and the instance name in Server, separated by a slash. Use Populate URLs to automatically populate the next two boxes. Add the same data sources account information you provided on the previous tab.



Once you finish, databases will be created, and the warehouse will start building.



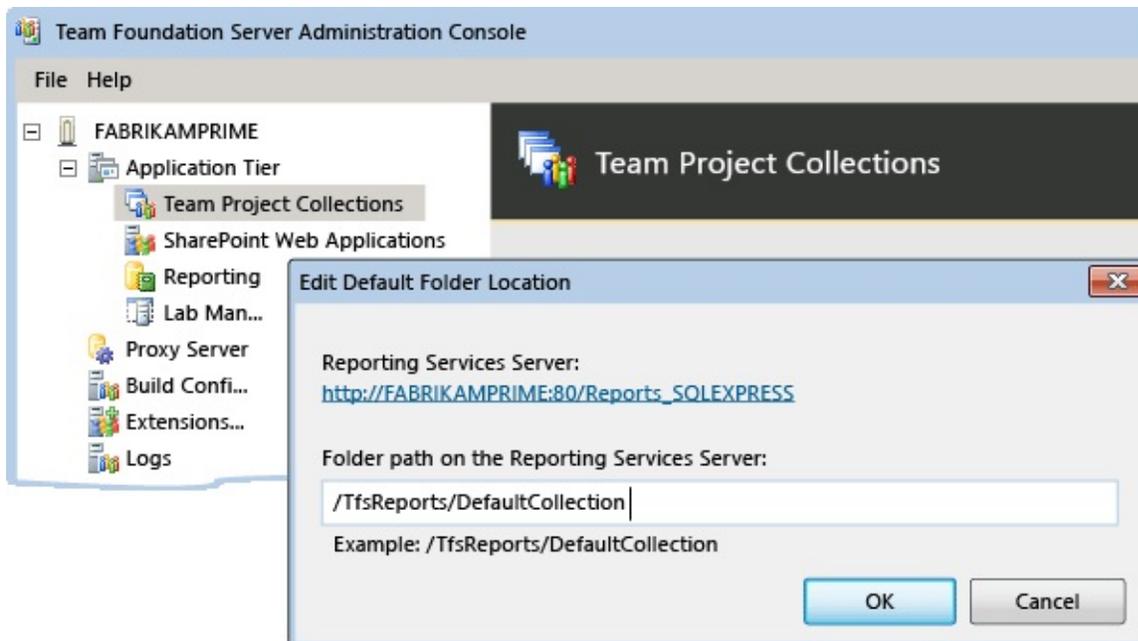
- Now add the report server to the team project collection. Edit the information in Reports Folder.



Tip

If you don't see the Reports Folder tab, you might need to refresh or restart the administration console.

- Unless you have reasons otherwise, use the example provided, and make sure you include the collection name.



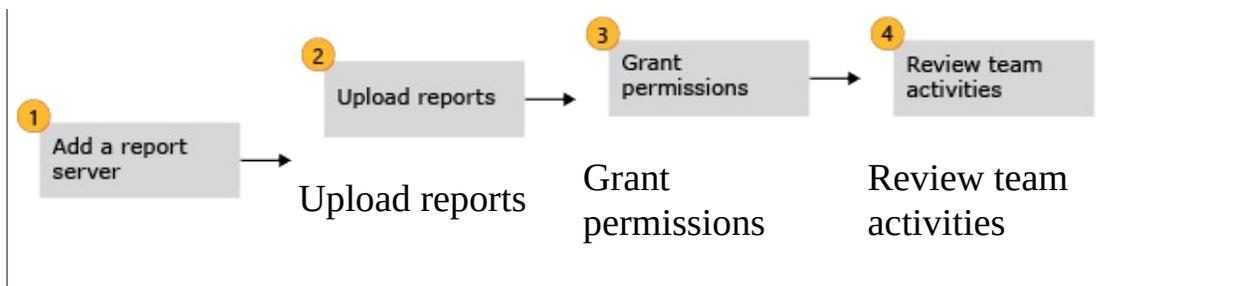
Now that you've added reporting to the server and to the team project collection, you're ready to start adding reports to your projects.

▣ **Try this next**

[Upload reports to a team project](#)

Dig deeper

- Do you want to know more about installing and configuring SQL Server for use with TFS? Read about [supported versions and editions](#), [working with named instances](#), [using SQL Server 2012 AlwaysOn](#), as well as [understanding how TFS, SQL Server, and Reporting all work together](#).
- Do you want to add someone as an administrator of a team project collection or for all of TFS? Learn what you have to do in [Set administrator permissions for team project collections](#) and [Set administrator permissions for Team Foundation Server](#).
- Do you need to change the data reader account you chose? Learn how to in [Change the service account or password for SQL Server Reporting Services](#), or read about [Service accounts and dependencies in Team Foundation Server](#) to better understand its requirements and what it does.



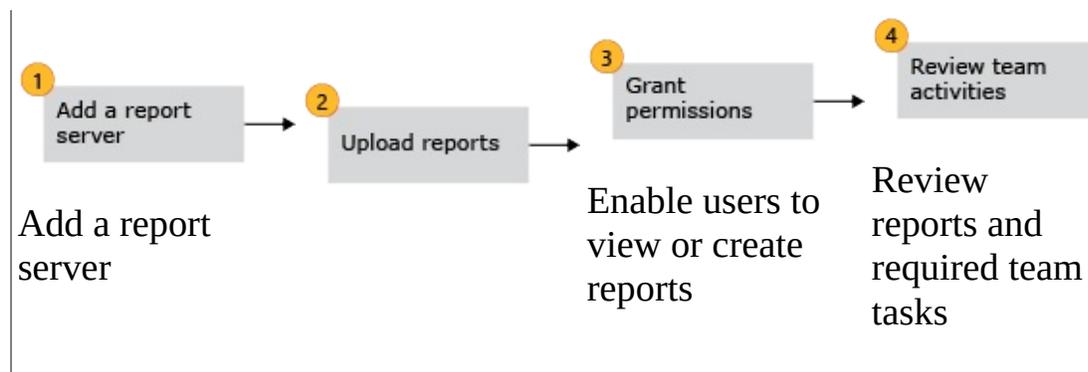


- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Upload reports to a team project

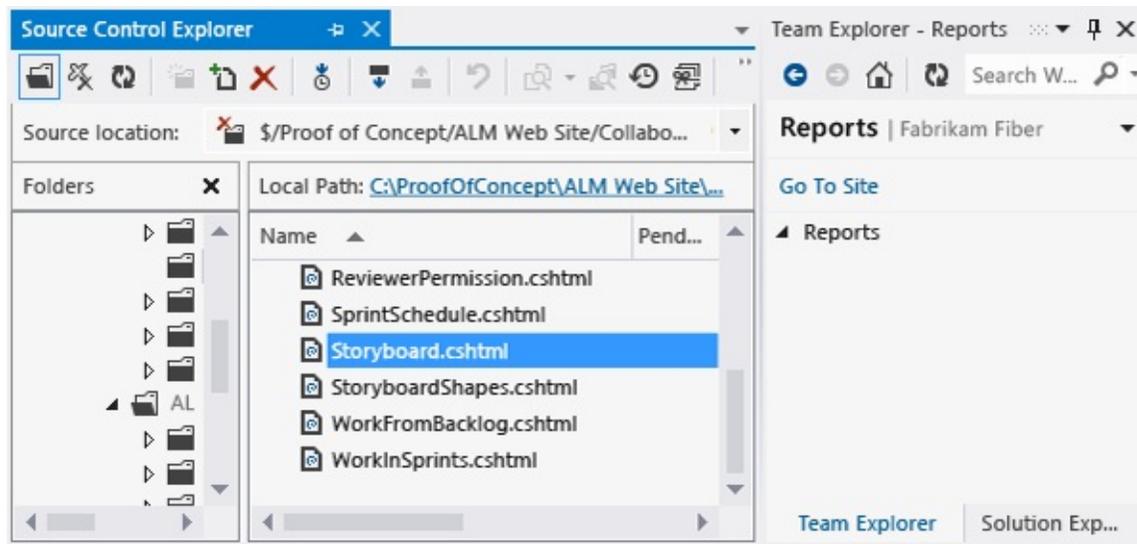
[Send Feedback](#)

By adding a report server to your TFS (on-premises) deployment, you can access a wealth of data about your team's projects, such as build quality, bug trends, burndown, and test progress. SQL Server Reporting Services (SSRS) reports provide insight to help teams manage work and improve processes.



After you've added a report server, you'll want to add reports to your team project. You can upload reports when connecting to an on-premises deployment of TFS 2010, TFS 2012, or TFS 2013.

If the Reports page in Team Explorer appears empty, then you'll want to upload reports.



If the Reports page doesn't appear at all, then you'll want to add a report server.

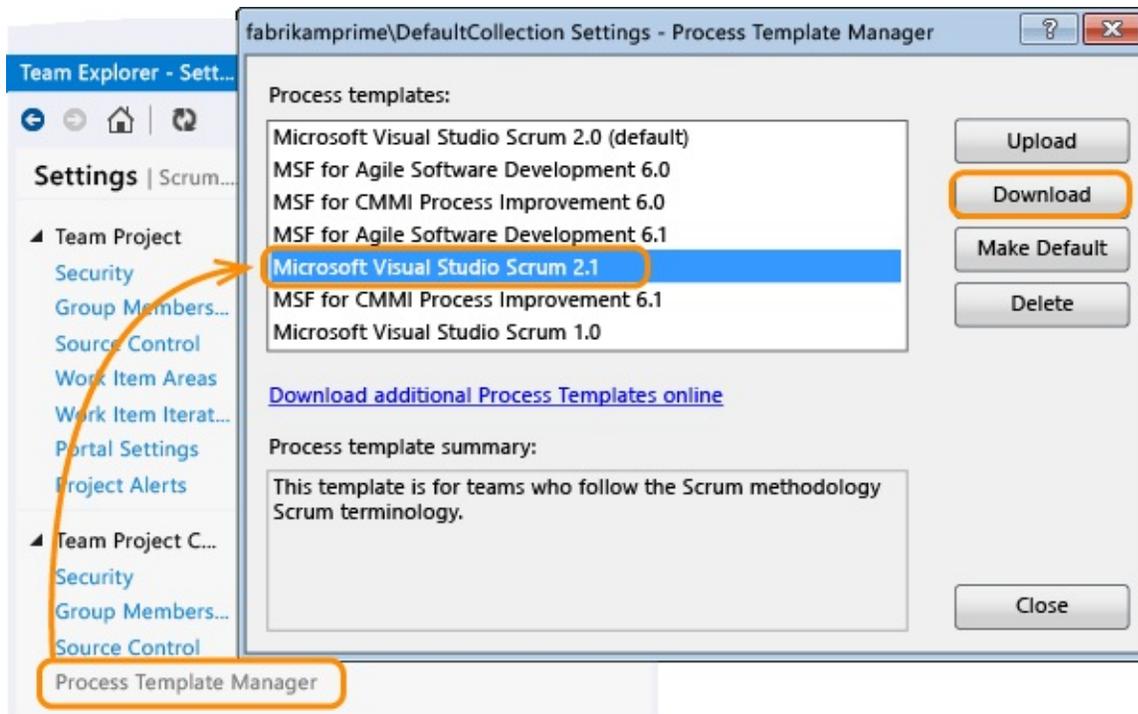
■ Upload reports to your team project

Use the **tfpt** command line tool that TFS Power Tools provides.

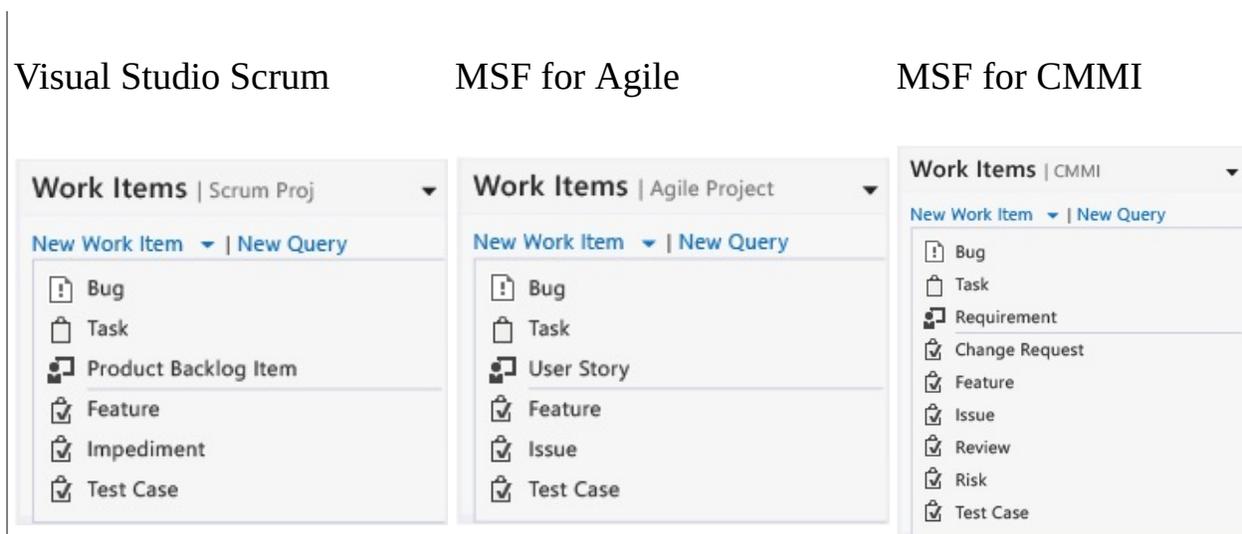
1. Verify that you have the following tools, configurations, and permissions.
 - If you haven't installed TFS power tools,

[install them now](#).

- If you haven't installed Visual Studio or Team Explorer, [install one of them now](#). You can download Team Explorer for free. Team Explorer must be present on the same machine where you run the **tfpt** command line tool.
- If you aren't an administrator for the team project, get added now.
- If you haven't been added to the Content Manager or Team Foundation Content Manager role for SQL Server Reporting Services, get added now.
- If you haven't installed [Microsoft Visual Studio Team Foundation Server 2013](#), install it now. This is the only way to obtain the latest version of the default process templates, which contains the latest version of the reports.
- From Team Explorer, download the latest process template that is compatible with the one used to create your team project.



To determine the type of process template that was used to create your team project, review the work item types that appear in the New Work Item menu for Team Explorer and then compare them with the work item types in the following chart. If your work item types differ from those shown listed, then a custom process template might have been used.



Note

- In general, you can use the most recent version of the process template that was used to create your team project. For example, you can upload reports from Microsoft Visual Studio Scrum 2013 version, even if your team project was created with Microsoft Visual Studio Scrum 2.0.
 - To download a process template, you need to be an administrator for the team project collection, and you must connect to TFS using the same version of Visual Studio or Team Explorer. For example, connect to TFS 2013 from Team Explorer 2013.
 - To learn more about process templates and work item types, go here.
- Open a Command Prompt in administrative mode and change to the directory where you installed the power tools.

Copy Code

```
cd %programfiles%\Microsoft Team Foundation Server 2013 Power Tools
```

On a 64-bit edition of Windows, replace %programfiles% with %programfiles(x86)%.

- Upload the reports. The process template you specify must be compatible with the one used to create the team project. And, it must have been uploaded to the team project collection that hosts your team project.



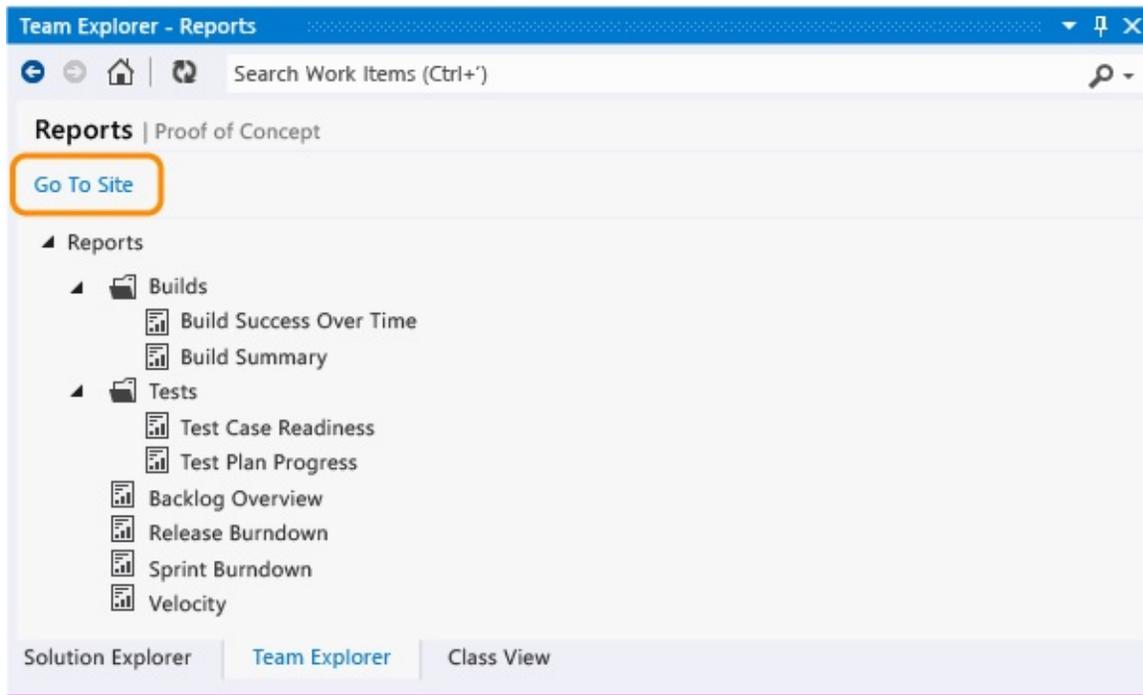
```
tfpt addprojectreports /collection:"http://MyServer:8080/tfs/Default
```

These are the names of the process templates uploaded with TFS 2013:

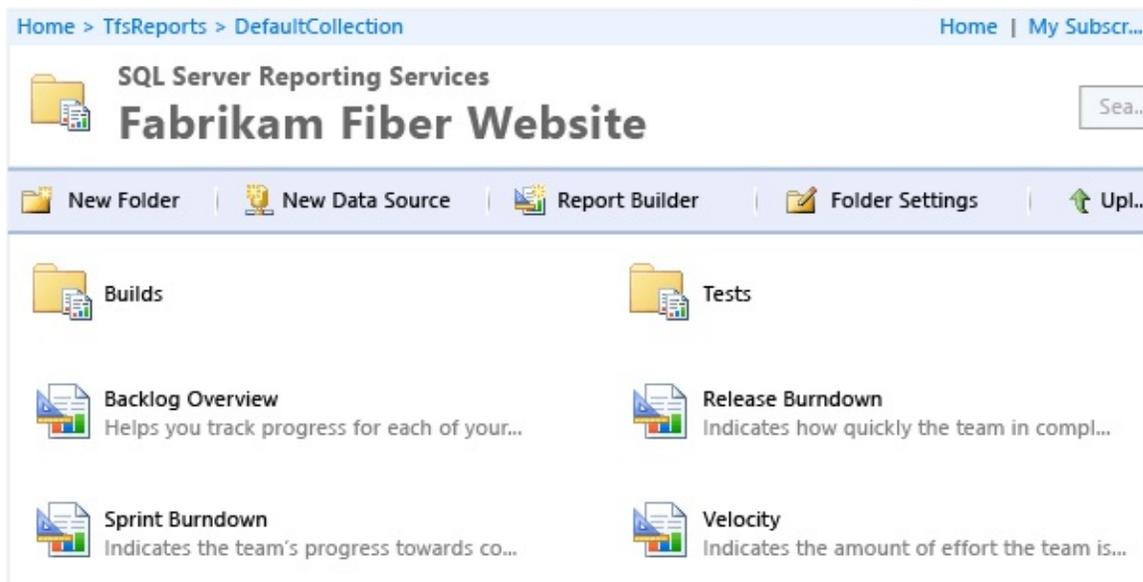
- Microsoft Visual Studio Scrum 2013
- MSF for Agile Software Development 2013
- MSF for CMMI Process Improvement 2013

To overwrite reports that were previously uploaded, specify the /force option.

- Open the report site from the  Reports page in Team Explorer. You might need to  refresh the page to see the newly uploaded reports.



And, here's a view of uploaded reports for a Scrum project in Report Manager.



Depending upon the amount of data that has been collected for your team

project, it can take several minutes to several hours for the data warehouse and cube to build. Once they do, however, you can view progress for your team project since TFS was first deployed.

▣ **Try this next**

Grant permissions to view or create reports.

Q & A

Q: How can I determine if my report is up to date?

A: To see when a report was last updated, check the Date Last Updated time stamp, which appears in the lower-right corner of each report.

Q: How often are reports refreshed?

A: Reports access data from the relational data warehouse (Tfs_Warehouse), which is updated every two minutes, and the cube (Tfs_Analysis), which is updated every two hours.

Q: How do I resolve issues causing reports to be out of date?

A: Reports will be out of date when all or some data stops flowing into the data warehouse. To verify the data flow, run the GetProcessStatus of the Warehouse Control Web service. Also, fix any schema conflicts that may be occurring.

Q: What tasks does tfpt addprojectreports perform?

A: The **tfpt addprojectreports** command performs the instructions contained in the ReportTasks file of the process template. This file is located in the Reports folder of the process template. The tasks include creating report folders, setting report parameters and data sources for each report, and uploading reports to the report server.

Q: What if I just want to upload a single report?

A: You can upload a single report using Upload file in Report Manager. You'll need to define the report's data source and possibly other parameters.

Q: What are the names of the data sources?

A: The names of the reporting services data sources are Tfs2010ReportsDS and

Tfs2010OlapReportsDS. Even though you may have installed or upgraded to TFS 2012, these names, which were assigned to the data sources for a TFS 2010 installation, are in use.

You can always check the names by opening the context menu for any report, choose Manage, and then choose Data Sources.

Q: Can I upload reports from another process template?

A: Most reports are customized based on the work item types and workflow states used in each process template. You might need to customize the reports you copy from another process template to fit the type definitions for your team project.

To learn about which reports are provided with which process templates, see [Review team activities to support useful reports](#).

Q: What about uploading Excel reports?

A: You can use **tfpt addprojectportal** command to add a project portal and Excel reports once you have added a SharePoint web application to your team project collection.

Q: Can I get these same reports using Visual Studio Online?

A: Not at this time. Uploading reports is only supported for TFS on-premises deployments. You can, however, [chart the results of a flat-list query](#).

How do I customize a report?

A: You can customize the reports you uploaded using [SQL Server Report Builder 2012](#). To learn more, see [Creating Reports for Team Foundation Server 2010](#) and [Customizing Reports for Team Foundation Server 2010](#). Even though these articles are based on TFS 2010, they are valid for TFS 2013.

How can I learn more about the TFS cube?

A: See [Perspectives and measure groups provided in the Analysis Services cube](#)

[for Visual Studio.](#)

☐ **Where can I go if I have more questions?**

A: Search for an answer or post a question in the [Team Foundation Server - Reporting & Warehouse forum](#).

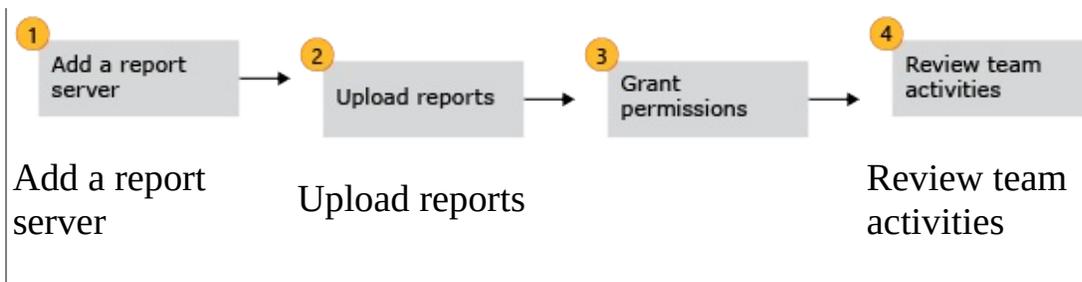


- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Grant permissions to view or create reports in TFS

[Send Feedback](#)

By adding a report server to your TFS (on-premises) deployment, you can access a wealth of data about your team's projects, such as build quality, bug trends, burndown, and test progress. SQL Server Reporting Services (SSRS) reports provide insight to help teams manage work and improve processes.



Now that you've uploaded reports, you'll want to enable members of your team to view or manage them. Also, to create or modify reports, you'll need to grant them access to read databases.

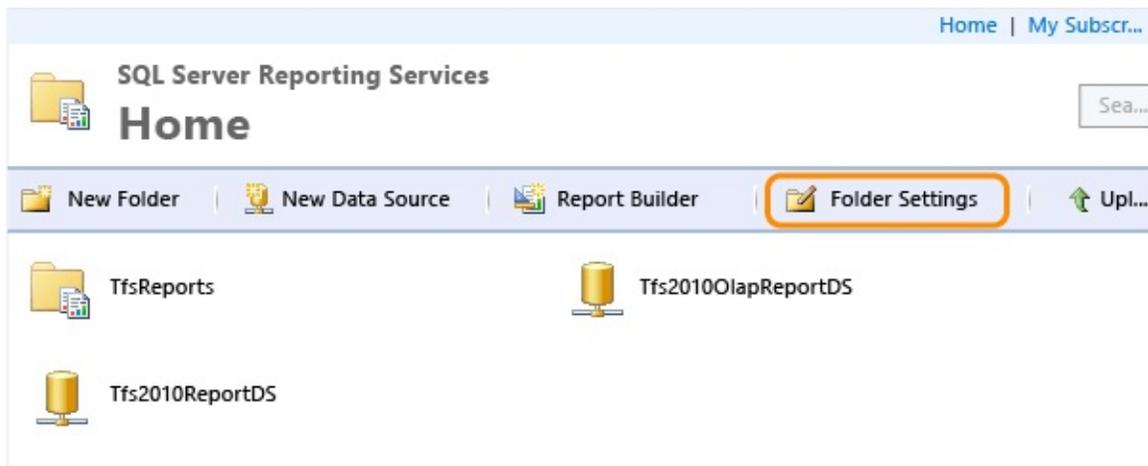
Add accounts to predefined roles to view or manage reports

Add report viewers to the Browser role. Add TFS report authors to the Team Foundation Content Manager role.

Tip

Permissions to access Report Manager are managed separately from TFS permissions. Even if you have added team members to a TFS group, you will still have to add them to a Report Manager role.

1. If you haven't been added to the Content Manager role for Reporting Services, get added by someone who has been added to this role.
2. From the Report Manager home page, open Folder Settings.



The URL is <http://ReportServer/Reports/Pages/Folder.aspx>, or if using a named instance, http://ReportServer/Reports_InstanceName/Pages/Folder.aspx.

3. Open New Role Assignment.

Home

SQL Server Reporting Services
Home

Security

Delete **New Role Assignment**

<input type="checkbox"/>	Group or User ↓	Role(s)
<input type="checkbox"/>	Edit BUILTIN\Administrators	Content Manager
<input type="checkbox"/>	Edit NORTHAMERICA\ctsodev1	Content Manager
<input type="checkbox"/>	Edit NORTHAMERICA\ctsodev2	Content Manager
<input type="checkbox"/>	Edit NT AUTHORITY\NETWORK SERVICE	Team Foundation Content Manager

 **Tip**

To limit access to reports defined for a team project or team project collection, first navigate to the corresponding folder and then open New Role Assignment.

4. Add the account name and select their role.

Home Home | My Subscr...

SQL Server Reporting Services
New Role Assignment

Use this page to define role-based security for Home.

Group or user name:

Select one or more roles to assign to the group or user.

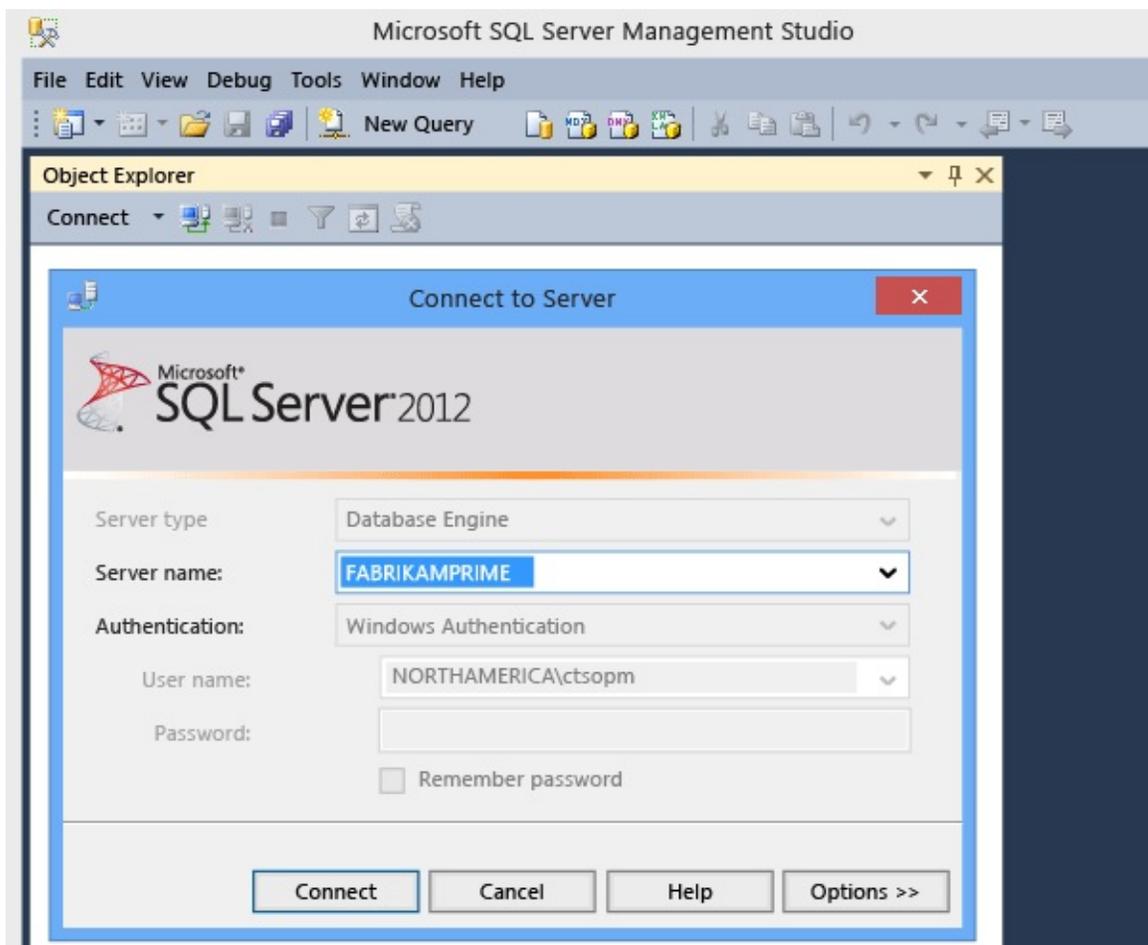
<input type="checkbox"/>	Role	Description
<input type="checkbox"/>	Browser	May view folders, reports and subscribe to reports.
<input type="checkbox"/>	Content Manager	May manage content in the Report Server. This includes... reports and resources
<input type="checkbox"/>	My Reports	May publish reports and linked reports; manage folders, r... resources in a users My Reports folder.
<input type="checkbox"/>	Publisher	May publish reports and linked reports to the Report Serv...
<input type="checkbox"/>	Report Builder	May view report definitions.
<input checked="" type="checkbox"/>	Team Foundation Content Manager	May manage Team Foundation Server related content in th... Server. This includes folders, reports and resources.

OK Cancel

■ Add report authors to database roles

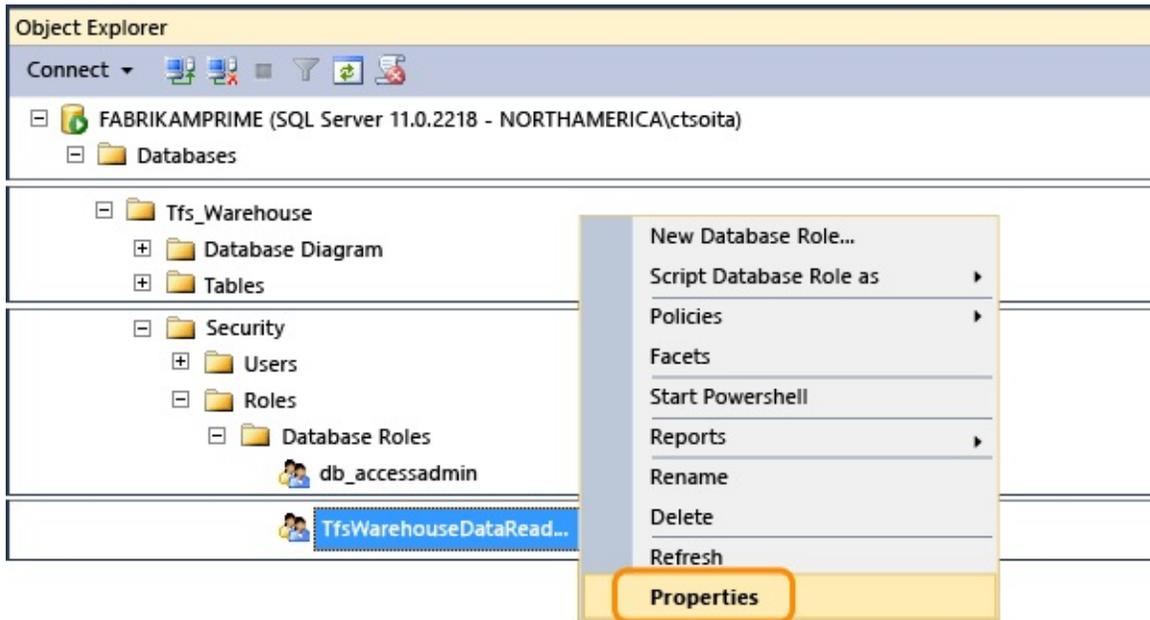
If members need to create or customize reports, add their accounts to the TfsWarehouseDataReader role. Report authors need read access to both the relational data warehouse and Analysis Services cube. Team members who create Excel reports from work item queries or by connecting to the cube need only read access to the cube.

1. If you aren't an administrator for the TFS database, get added as one.
2. Connect to the Database Engine for TFS using SQL Server Management Studio.

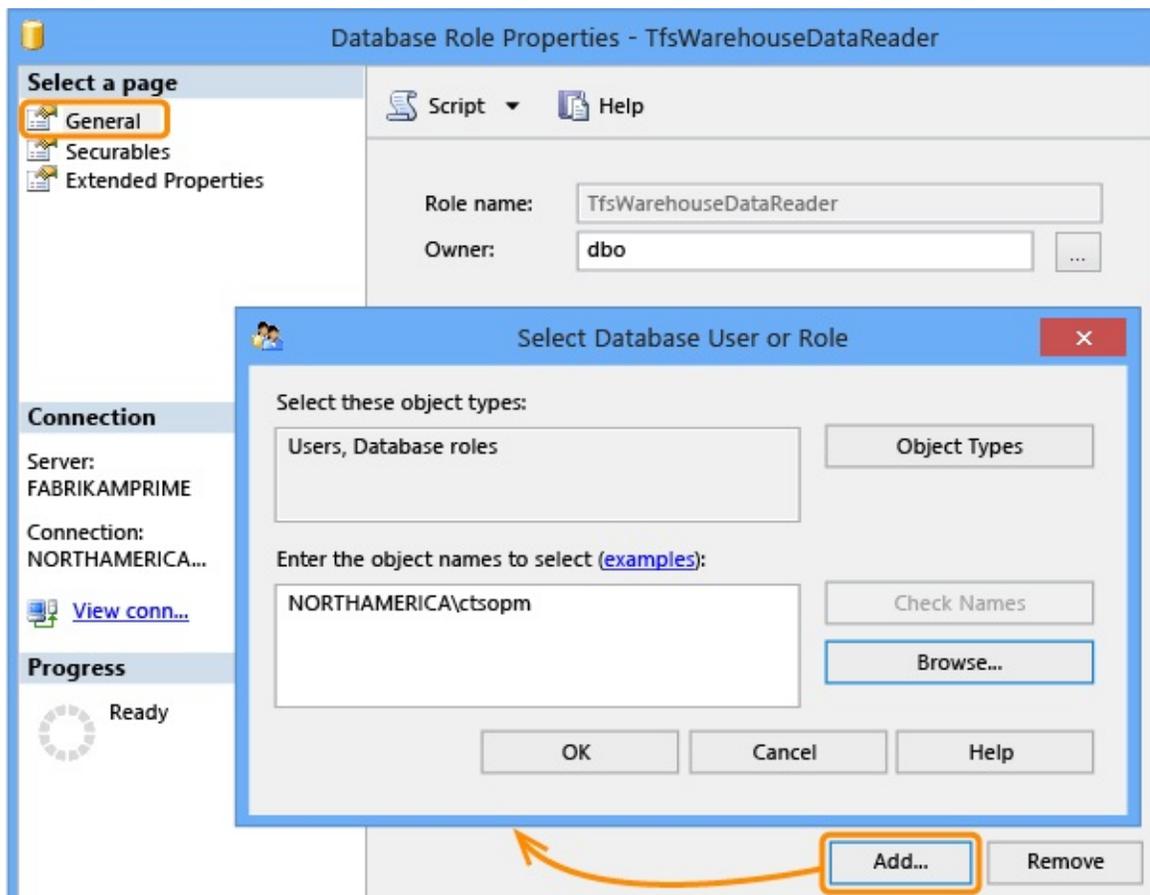


3. Open the properties page for the TfsWarehouseDataReader role under the

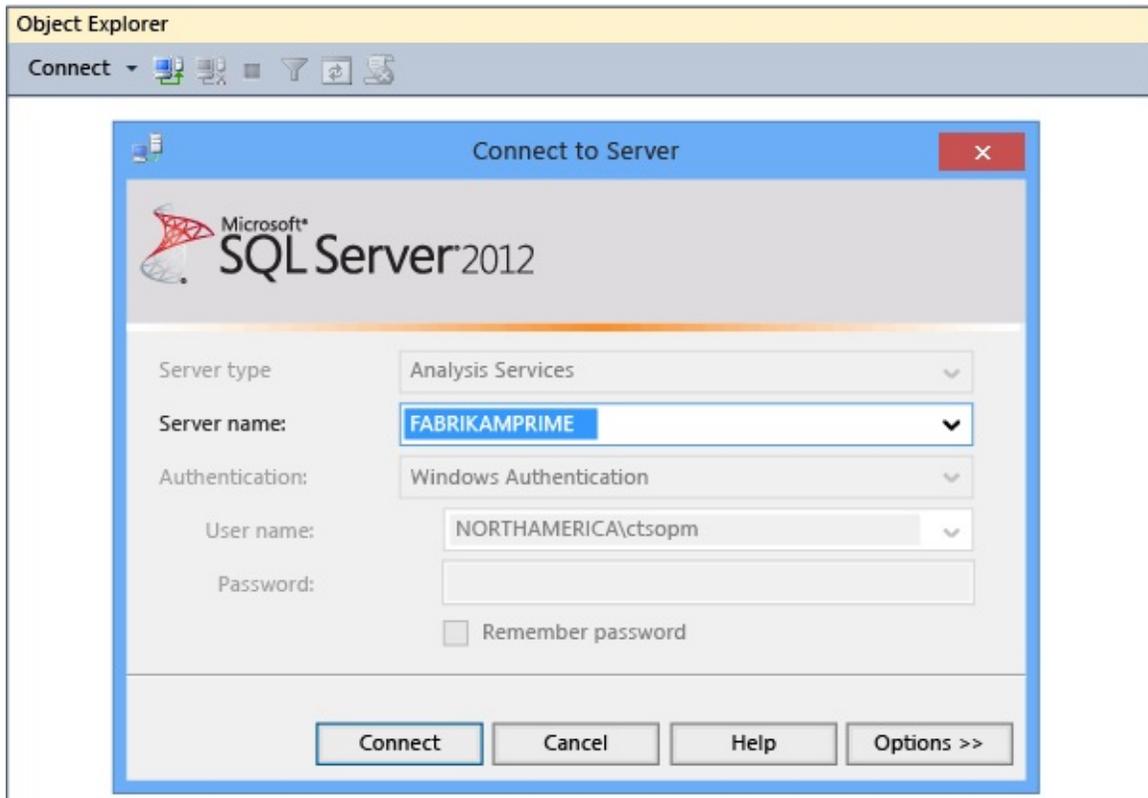
Databases/Tfs_Warehouse/Security/Roles/Database Roles folder.



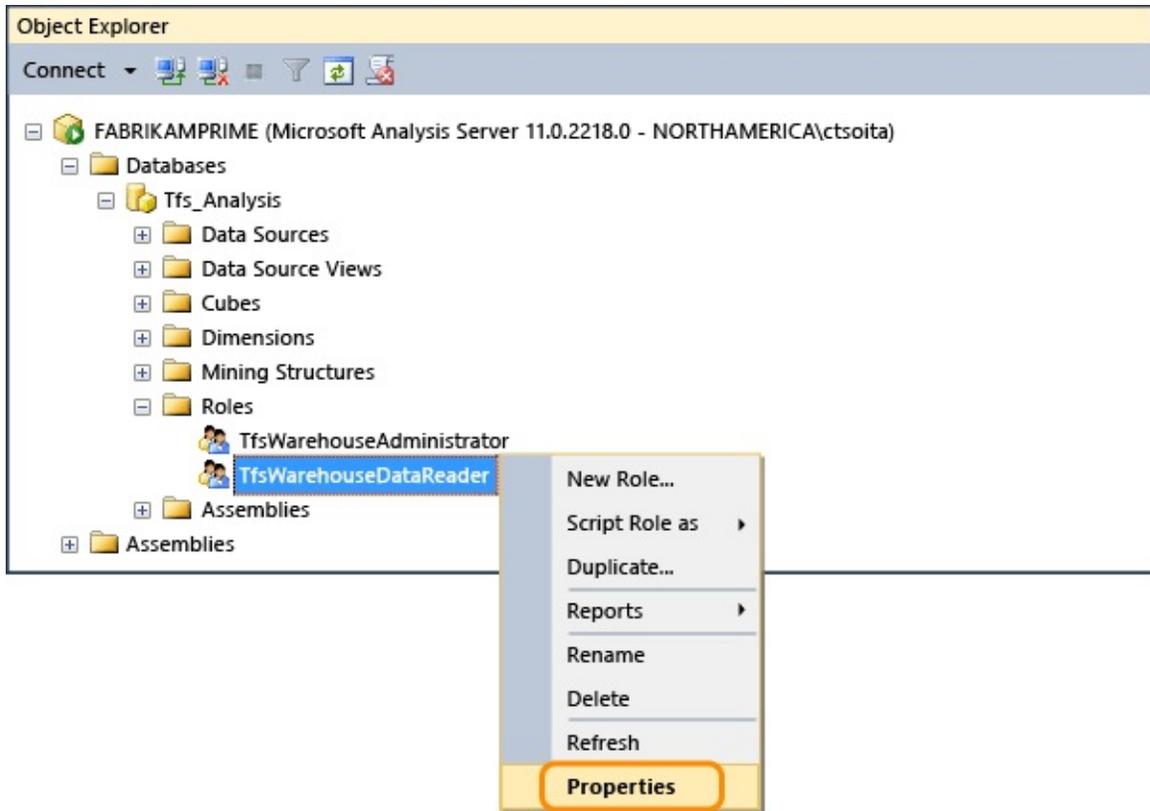
4. Add the account.



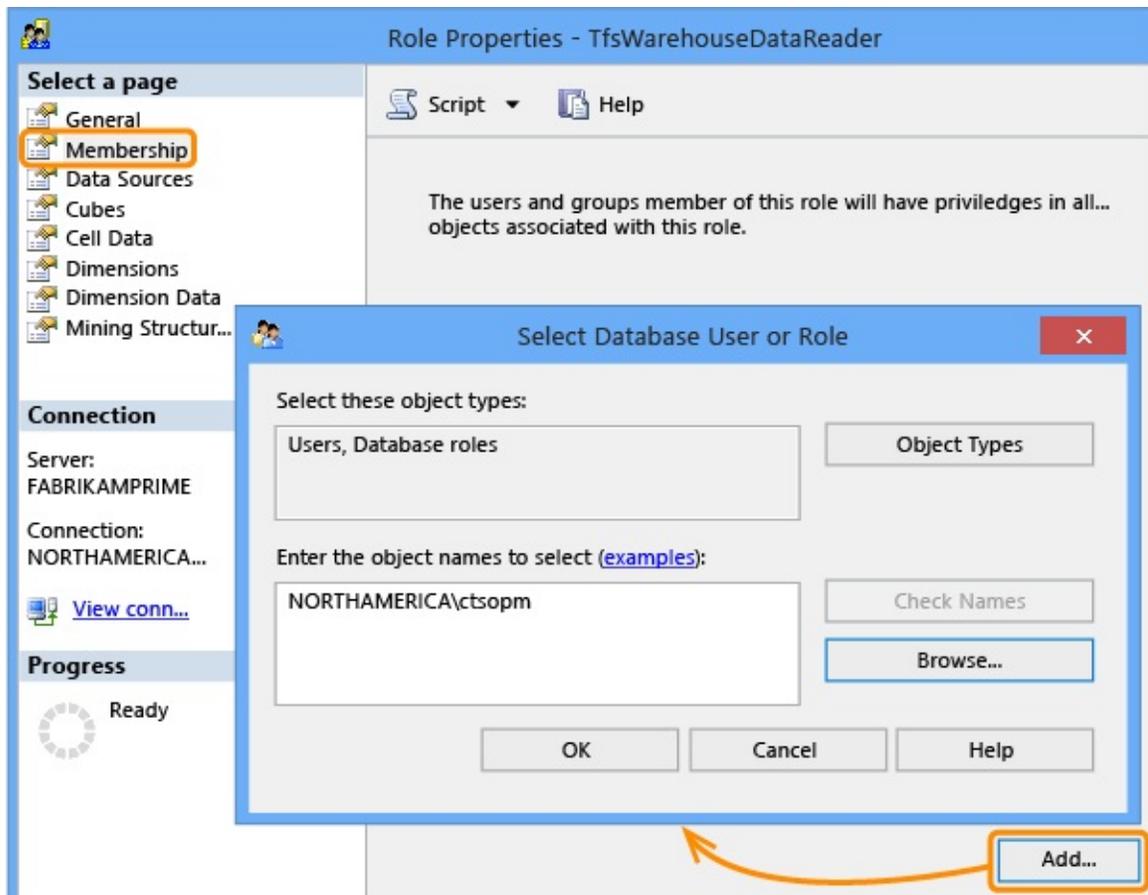
5. Next, connect to the Analysis Services database.



6. Open the properties page for the TfsWarehouseDataReader role under the Databases/Tfs_Analysis/Roles folder.



7. Add the account.



⚠️ Important

Accounts that you add to the TfsWarehouseDataReader roles can view data for all team projects that are hosted in all team project collections in the TFS deployment. There is no way to limit access to a team project or collection.

▣ **Try this next**

Review team activities to support useful reports.

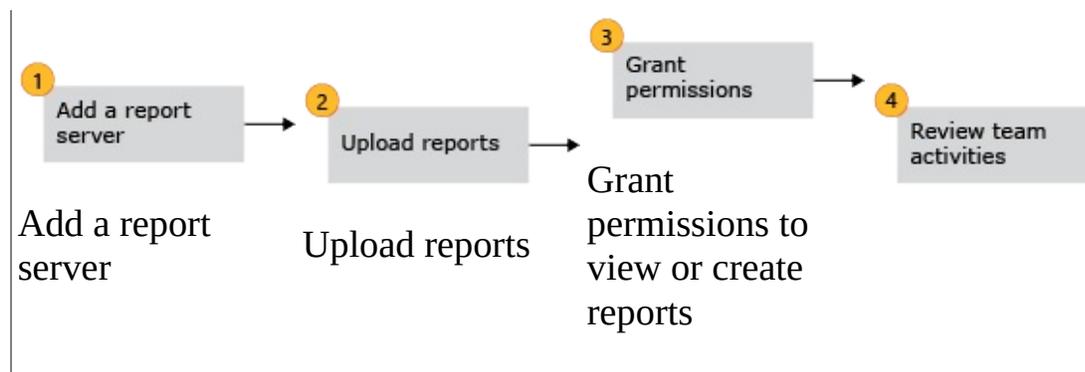


- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Review team activities to support useful reports

[Send Feedback](#)

By adding a report server to your TFS (on-premises) deployment, you can access a wealth of data about your team's projects, such as build quality, bug trends, burndown, and test progress. SQL Server Reporting Services (SSRS) reports provide insight to help teams manage work and improve processes.



Now that you've uploaded reports, how do you use them to track progress, gain insight, and improve processes?

First, make sure your team is performing the activities that create the data that these reports use. Your team is probably performing most of these activities already.

Here's a summary of the reports that TFS provides and the team activities that are associated with them. Over time, you can use these reports to see trends and identify which practices and processes require more attention to deliver desired results.

Monitor code quality

Build reports track the quality of software under development. By defining tests to run automatically as part of each build definition and instrumenting tests to gather code coverage data, you can gain insight about the quality of the builds, tests, and code.

Build and test activities	Build reports
1. Configure a build system.	• Build Quality Indicators (Agile and CMMI only)
2. Define your build process.	
3. Run tests in your build process.	• Build Success Over Time (pictured)
4. Rate completed builds to populate the Build Quality dimension.	• Build Summary

For a free downloadable guide to testing and monitoring builds, see [Testing for Continuous Delivery with Visual Studio 2012](#).

Sample build success over time report

			No Build	Build Failed	Build Succeeded, No Tests	Tests Failed	Tests Passed, Low Coverage	Passed						
														
Build Definition	Platform	Configuration	6/2	6/3	6/4	6/5	6/6	6/9	6/10	6/11	6/12	6/13	6/14	6/15
AllDebug	x86	Debug												
DebugAnyCPU	Any CPU	Debug												
Debugx86	x86	Debug												
ReleaseAnyCPU	Any CPU	Release												
Releasex86	x86	Release												

Monitor progress

Project management reports provide insight into how much work the team is tackling within a sprint or release, and the rate of their progress. By linking work items and updating specific fields as work is performed, you can track the progress of individual stories and be able to more accurately estimate future activities.

Work item tracking activities

1. Create the backlog.

- Create product backlog items and specify the Effort (Scrum).

Project management (Scrum) reports

- Create user stories and specify the Story Points (Agile).

- Backlog Overview
- Release Burndown

- Create requirements and specify the Size (CMMI).

- Sprint Burndown
- Velocity

2. Work in sprints. Assign backlog

Project management (Agile and CMMI) reports

items to sprints, create tasks and link them to parent backlog items, and assign to a team member.

3. Update Remaining Work for tasks. For Agile and CMMI team projects, update Completed Work as well.

 **Tip**

The only report that references Original Estimate is Status of All Iterations.

4. Create test cases and bugs, link them to their parent backlog item, and update their State.
5. (Optional) Assign work items to areas to filter reports.

- Burndown and Burn Rate
- Remaining Work
- Requirements Overview (CMMI)
- Requirements Progress(CMMI)
- Status of All Iterations (similar to Velocity)
- Stories Overview (Agile) (pictured)
- Stories Progress (Agile)
- Unplanned Work

Sample stories overview report

Title	% Hours Completed	Hours Remaining
As a new customer, I want to order a meal.	62 %	31
Enable selection based on strength, intelligence, etc	39 %	134
Regular members can search for heroes	19 %	144
Gold member can search for villians	60 %	2

Test Points	Test Results	Bugs
3	35 % 53 %	1 2
0		6
2	54 % 46 %	1 2
0		

Monitor test plans and bug tracking

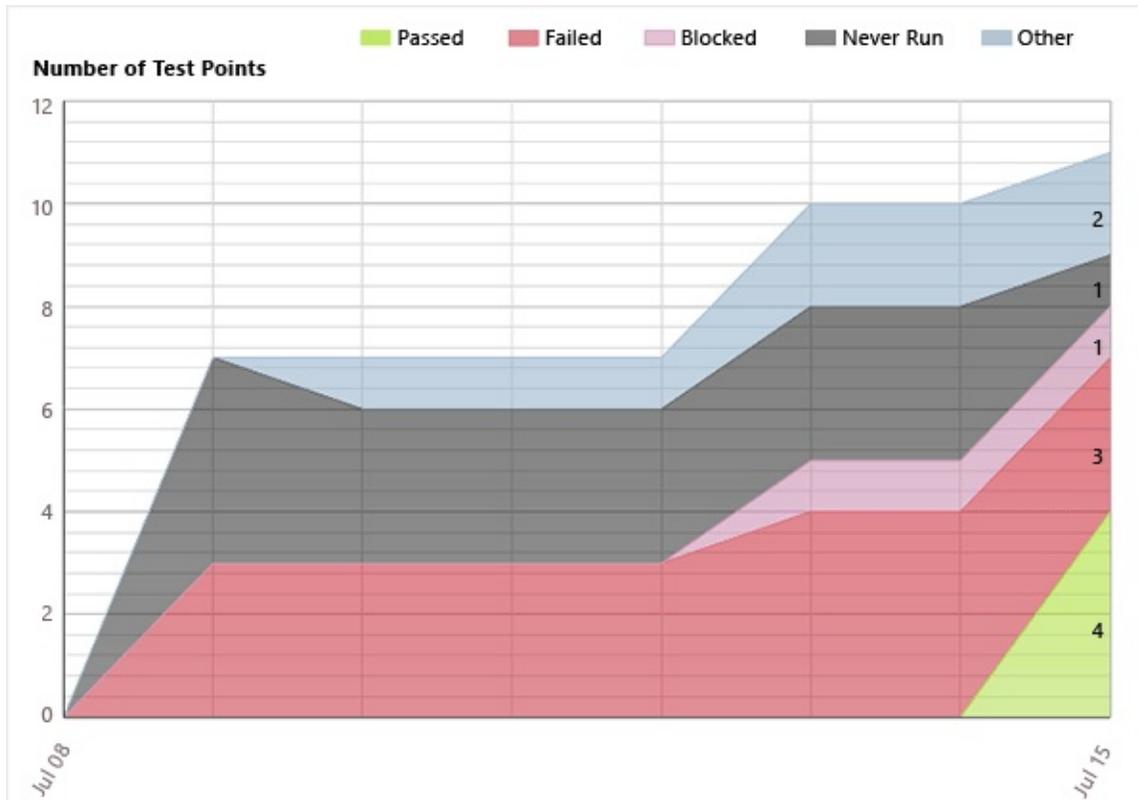
Test planning reports support monitoring the test progress and coverage of backlog items or user stories. Bug tracking reports illustrate the team's capacity to find and resolve bugs.

Test planning and bug tracking activities

- | | |
|---|--|
| 1. Define test plans and test cases, and update their State as work progresses. | Test and bug reports |
| 2. Mark the results of each validation step in manual tests as either passed or failed. | <ul style="list-style-type: none">• Test Case Readiness• Test Plan Progress (pictured)• Bug Status (Agile and CMMI only) |
| 3. Create bugs, specify the Priority and Severity, assign to a team member, and update the State. | <ul style="list-style-type: none">• Bug Trends (Agile and CMMI only)• Reactivations (Agile and CMMI only) |
| 4. (Optional) Assign test | |

cases and bugs
to areas and
iterations to
filter reports.

Sample test plan progress report



Q & A

Q: Do reports handle stories and substories or tasks and subtasks?

A: Yes, you can subdivide stories or backlog items as well as tasks, creating a nested hierarchy of both backlog items and tasks. You can nest items several levels deep. If you subdivide a task into subtasks, specify hours only for the subtasks. These hours are rolled up as summary values for the parent task and their parent backlog item. To correct reports you believe are in error, see Address inaccuracies published for summary values.

Q: Which reports depend on linking work items?

A: The overview and progress reports depend on linking tasks, test cases, and bugs to backlog items. You must link these items using the parent-child link for tasks and bugs and the Tested By link for test cases.

Q: Which reports depend on Microsoft Test Manager?

A: Test Case Readiness and Test Plan Progress reports are designed to work with Test Manager. Also, the test points and test progress in project management overview reports depend on linking test cases to backlog items.

Q: Are these reports the same as the agile planning charts that appear in Team Web Access (TWA)?

A: While some reports do display similar information, such as sprint burndown and velocity or status on all iterations, these reports are formatted differently and support additional filters. Other reports, such as the build and test planning reports, are not available through TWA at this time.

Q: Do you want to create additional product areas or release milestones?

A: See Create areas or iterations.

Q: Do you want to bulk edit work items to assign them to an area, iteration, team member, or priority?

A: See Bulk modify work items.

Q: Do you want to add a field to track additional data?

A: See Add or modify a work item field to support reporting.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Manually process the TFS data warehouse and analysis services cube

[Send Feedback](#)

When you need the freshest data in your reports, when errors have occurred, or after you've resolved schema conflicts, you can manually process the Team Foundation Server (TFS) relational database (Tfs_Warehouse) or SQL Server Analysis Services cube (Tfs_Analysis).

During typical operations, the warehouse is processed within two minutes of changes made to an operational store, and the cube is processed every two hours. By manually processing the warehouse, you help ensure that queries and reports are up-to-date with data that depends on the warehouse.

You use the Warehouse Control Web Service to process the warehouse or cube or perform other maintenance operations. If you know that you want to perform a full rebuild of both databases, then use the Administration console or the TFSSConfig RebuildWarehouse command.

Team Foundation Warehouse Control web service

The following operations are supported. For a formal definition, please review the [Service Description](#).

- [BringAnalysisProcessingOnline](#)
- [BringWarehouseProcessingOnline](#)
- [ChangeSetting](#)
- [GetJobProperties](#)
- [GetProcessingStatus](#)
- [GetSettings](#)
- [ProcessAnalysisDatabase](#)
- [ProcessDefaultDimensionsForExpandedCapacity](#)
- [ProcessWarehouse](#)
- [SetAnalysisJobEnabledState](#)
- [SetWarehouseJobEnabledState](#)
- [TakeAnalysisProcessingOffline](#)
- [TakeWarehouseProcessingOffline](#)

Note

Do not use SQL Server Management Studio (SSMS) to manually process the cube. Processing the cube using that tool is not supported.

Process the warehouse or cube

Processing the warehouse or cube depends on how much data is involved; it can take minutes or hours. Before you process either database, determine the processing status for the synchronization job or jobs that you want to run. Make sure that the status returns Idle

To access the web services

1. If you aren't a member of the Administrators security group on the application-tier server for TFS, get added now.

Also, make sure that your server-level **Administer warehouse** permission must be set to **Allow**.

2. Log on to the application-tier server and open the Warehouse Control Web Service by entering the following URL in a supported web browser:

`http://localhost:8080/tfs/TeamFoundation/Administration/v3.0/WarehouseControlWebService`

If another name was used other than tfs for the virtual directory, then type the IIS Virtual Directory that was specified when Team Foundation Server was installed.

3. The WarehouseControlWebService page opens.

Note

If the page doesn't open, verify that the Microsoft Team Foundation Server Application Pool is running.

To check the process status

- Choose GetProcessingStatus.

WarehouseControlWebService

Click [here](#) for a complete list of operations.

GetProcessingStatus

Test

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

Parameter	Value
serviceHostName:	<input type="text"/>
jobName:	<input type="text"/>
includeOnlineHostsOnly:	<input type="text" value="TRUE"/>

A new browser window opens. It indicates the following job's processing status:

- Common Structures Warehouse Sync
- Full Analysis Database Sync
- Incremental Analysis Database Sync

And, the status for the following jobs for each team project collection are provided:

- Build Warehouse Sync
- Common Structures Warehouse Sync
- Test Management Warehouse Sync
- Version Control Warehouse Sync
- Work Item Tracking Warehouse Sync

A value of Idle indicates that the synchronization job is currently not running. You should process the data warehouse or the cube only when the processing status for these jobs is Idle. If a different value is returned,

repeat this step until Idle is returned for the job that you want to process.

To process the data warehouse

1. Choose ProcessWarehouse, and optionally specify the team project collection to process. If you leave collectionName blank, all collections are processed.

The service returns True when it successfully starts the processing of the warehouse and False if it is not successful. A value of False indicates that the warehouse is currently being processed.

2. To determine the processing status of the warehouse choose GetProcessingStatus as described earlier in [to check the process status](#).

Warehouse processing is completed when the GetProcessingStatus service returns a value of Idle, and a value of Succeeded for Common Structures Warehouse Sync and for each warehouse for each team project collection.

To process the Analysis Services cube

1. Choose ProcessAnalysisDatabase.
2. For processingType, type either Incremental or Full.

If you specify Incremental, data is processed only if it has been added since the most recent processing.

If you specify Full, all data is processed as if the warehouse were being rebuilt. Full processes take longer to perform and should be performed only when required (for example, when a team project or project collection has been removed or deleted).

The service returns True when it successfully starts the processing of the cube and False if it is not successful. A value of False indicates that the cube is currently being processed.

3. To determine the processing status of the warehouse choose GetProcessingStatus as described earlier in [to check the process status](#).

Cube processing is completed when the GetProcessingStatus service returns a value of Idle, and a value of Succeeded for the following jobs.

- Full Analysis Database Sync
- Incremental Analysis Database Sync

Process default dimensions for expanded capacity

You should run the `ProcessDefaultDimensionsForExpandedCapacity` web service when you receive the following error message:

```
Microsoft.TeamFoundation.Warehouse.WarehouseException: TF221122: An error occurred running job Full Analysis Database Sync for team project collection or Team Foundation server TEAM FOUNDATION.
```

```
Microsoft.TeamFoundation.Warehouse.WarehouseException: Failed to Process Analysis Database 'Tfs_Analysis'.
```

```
Microsoft.TeamFoundation.Warehouse.WarehouseException: File system error: A string store or binary store with a compatibility level of '1050' is at the maximum file size of 4 gigabytes. To store additional strings, you can change the StringStoresCompatibilityLevel property of the associated dimension or distinct count measure to '1100' and reprocess. This option is only available on databases with a compatibility level of '1100' or higher.
```

This service changes the `StringStoresCompatibilityLevel` to 1100 for the Version Control File dimension and performs a full cube reprocess.

1. From the Warehouse Control Web Service, choose `ProcessDefaultDimensionsForExpandedCapacity`.
2. The service returns `True` when it successfully resets the value to 1100.

Wait until the cube processing has succeeded.

Q & A

Q: When should I process a database versus rebuild?

A: Process a database manually for one of the following reasons:

- Incrementally process the cube when reports don't show the latest data and you need them up-to-date for an upcoming meeting.
- Process each team project collection within the data warehouse to verify resolution of all schema conflicts.
- Process each team project collection within the data warehouse to collect information to diagnose issues appearing in the event viewer related to warehouse jobs.

Perform a full rebuild of the warehouse and cube when you move, restore, rename, or fail over the TFS data-tier server.

Q: How do I change the refresh processing interval or other processing parameters?

A: See [Manage TFS reports, data warehouse, and analysis services cube](#).

Q: Where can I learn more about how to resolve schema conflicts?

A: See [Resolve schema conflicts that are occurring in the data warehouse](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Rebuild the TFS data warehouse and cube

[See Also](#) [Send Feedback](#)

Whenever you move, restore, rename, or fail over the data-tier server for Team Foundation Server (TFS), you must rebuild the warehouse and cube to access high-level reports. Also, if you move, attach, detach, or delete a team project collection, you must rebuild the warehouse and cube.

The data warehouse aggregates all operational data, such as version control, work item tracking, build, and test. The warehouse corresponds to the relational database, Tfs_Warehouse, and the cube corresponds to Tfs_Analysis, the SQL Server Analysis Services database.

You should not have to rebuild the data warehouse during normal operations. If you want to refresh the data warehouse data, you can manually process the warehouse and cube on demand. See [Manually process the TFS data warehouse and analysis services cube](#). Depending on the amount of data in the data warehouse, the rebuild operation can take several hours to finish, during which time reports are not available.

To rebuild the data warehouse for Team Foundation, you must perform this sequence of steps:

1. [Verify that services and application pools are running and that TCP/IP is enabled for SQL Server](#)
2. [Rebuild the data warehouse](#)
3. [Verify that reports can be accessed](#)

Requirements

You must be a member of these security groups or have the corresponding

permissions:

- **sysadmin** security group on the server or servers that are running the instance of SQL Server that hosts the databases for Team Foundation Server
- **sysadmin** security group on the server for the Analysis Services database for Team Foundation
- Tfs_Warehouse relational database and **TFSEXCROLE** database role
- **Team Foundation Administrators** security group and the Administrators security group on the server or servers that are running the administration console for Team Foundation
- Your server-level **Administer warehouse** permission must be set to **Allow**
- Your View project-level information permission on the team project must be set to Allow to run and view a report. Also, you must be a member of the Browser role in SQL Server Reporting Services

For more information, see [Add accounts to administer TFS](#).

In addition to these permissions, you might need to address these requirements on a computer that is running Windows Server 2008 or Windows Vista:

- To follow a command-line procedure, you might need to open an elevated Command Prompt by choosing Start, opening the context menu for Command Prompt, and choosing Run as Administrator.
- To follow a procedure that requires Internet Explorer, you might need to start it as an administrator by choosing Start, All Programs, opening the context menu for Internet Explorer, and then choosing Run as administrator.
- To access Report Manager, reports, or websites for Reporting Services, you might need to add these sites to the list of trusted sites in Internet Explorer or start Internet Explorer as an administrator.

For more information, see [User Account Control](#).

With the rebuild operation, you re-create both the relational database and the cube. TFS rebuilds the relational database from the operational data stores. It then re-creates the SQL Server Analysis Services cube and processes the cube to populate it by using data from the relational database. If you want to rebuild just the cube, you can use the **TFSConfig RebuildWarehouse** command-line tool. See [RebuildWarehouse Command](#).

1. Verify that services and application pools are running and that TCP/IP is enabled for SQL Server

To complete the steps in this procedure, the services that SQL Server requires must be running. You stop Reporting Services so that users do not access reports while you are rebuilding the data warehouse. Also, for a dual-server deployment, the TCP/IP protocol must be enabled for each instance of a SQL Server database.

By default, TCP/IP is disabled when you install SQL Server.

1. Log on to the appropriate server, open Computer Manager, and then verify that the services and application pools in the following table are running:

Log on to the server that hosts this program	Component
SQL Server Analysis Services	<ul style="list-style-type: none">◦ SQL Server Analysis Services (MSSQLSERVER or TFSInstance).
Team Foundation databases	<ul style="list-style-type: none">◦ SQL Server (MSSQLSERVER or TFSInstance).◦ SQL Server Agent (MSSQLSERVER

Application tier

or TFSInstance).

- Microsoft Team Foundation Server Application Pool.

For more information, see

[Stop and start services, application pools, and websites.](#)

- Log on to the server that hosts Reporting Services, and stop the SQL Server Reporting Services (TFSINSTANCE) service.
- Make sure that TCP/IP has been enabled for SQL Server on the data-tier server. For more information, see [Enable the TCP/IP Protocol for a Database Instance.](#)

2. Rebuild the data warehouse and restart services

1. Open the Team Foundation administration console for Team Foundation.
2. Under Application Tier>Reporting, choose Start Rebuild.

The Take Offline dialog box opens.

3. Choose OK to take the data warehouse and Analysis Services cube offline.

The Rebuild the Warehouse and Analysis Services Databases dialog box opens.

4. Choose OK to initiate the rebuild process.

When you rebuild the warehouse or cube, TFS performs this sequence of actions:

- Takes the databases offline.
- Drops the schema for both databases.
- Re-creates the schema for both databases.
- Brings both databases back online.
- Restarts the warehouse adapter jobs.
- Processes the cube according to its regular processing interval.

The rebuild process starts first for the warehouse and then for the cube. The time that is required to rebuild the databases is based on the amount of data that is stored in the data warehouse. When the databases have been rebuilt, the jobs for processing the databases are automatically started.

 **Note**

While the warehouse and the cube are being rebuilt, the administration console for Team Foundation may stop responding. You might need to refresh the administration console after the warehouse and the cube have been rebuilt.

5. (Optional) To check the status of the rebuild process, you can use the Warehouse Control Web service. For more information, see

[Manually process the TFS data warehouse and analysis services cube.](#)

- Log on to the server that hosts Reporting Services, open Computer Manager, and then start the SQL Server Reporting Services (TFSINSTANCE) service.

3. Verify that reports can be accessed

1. Log on to the server that hosts Reporting Services, open Internet Explorer, type the following string in the Address bar, and then press ENTER:

http://localhost/Reports

Note

You may need to start Internet Explorer as an administrator by choosing Start, All Programs, open the context menu of Internet Explorer, and then choosing Run as administrator.

If you have deployed a named instance on the data-tier server, type this string instead:

http://localhost/Reports_TFSInstance

2. In Contents, choose TFSReports, choose the folder of the team project collection that stores your team project, choose the folder that corresponds to your team project, and then choose the folder that contains a report that you want to view.
3. Choose the folder that contains a report that you want to view.
4. Choose a report, and verify that the report appears correctly.
5. Check the date when the report was last updated. This information appears in the lower-left corner of the report.

Q & A

Q: How do I resolve schema conflicts?

A: If you encounter schema conflicts, you cannot resolve this by rebuilding the data warehouse. Instead, you must resolve the conflicts first and then rebuild the data warehouse. See [Resolve schema conflicts that are occurring in the data warehouse](#)

Q: How can I resolve failure errors that occur when rebuilding or processing the data warehouse?

A: The following actions can cause failure errors to occur.

- You manually modified a TFS database or edited a SQL table. You should not manually modify any of the TFS databases unless you're either instructed to do so by Microsoft Support or when you're following the procedures described for manually backing up the databases (

[Manually back up Team Foundation Server](#)). Any other modifications can invalidate your service agreement, block upgrades and patches, and result in data loss or corruption.

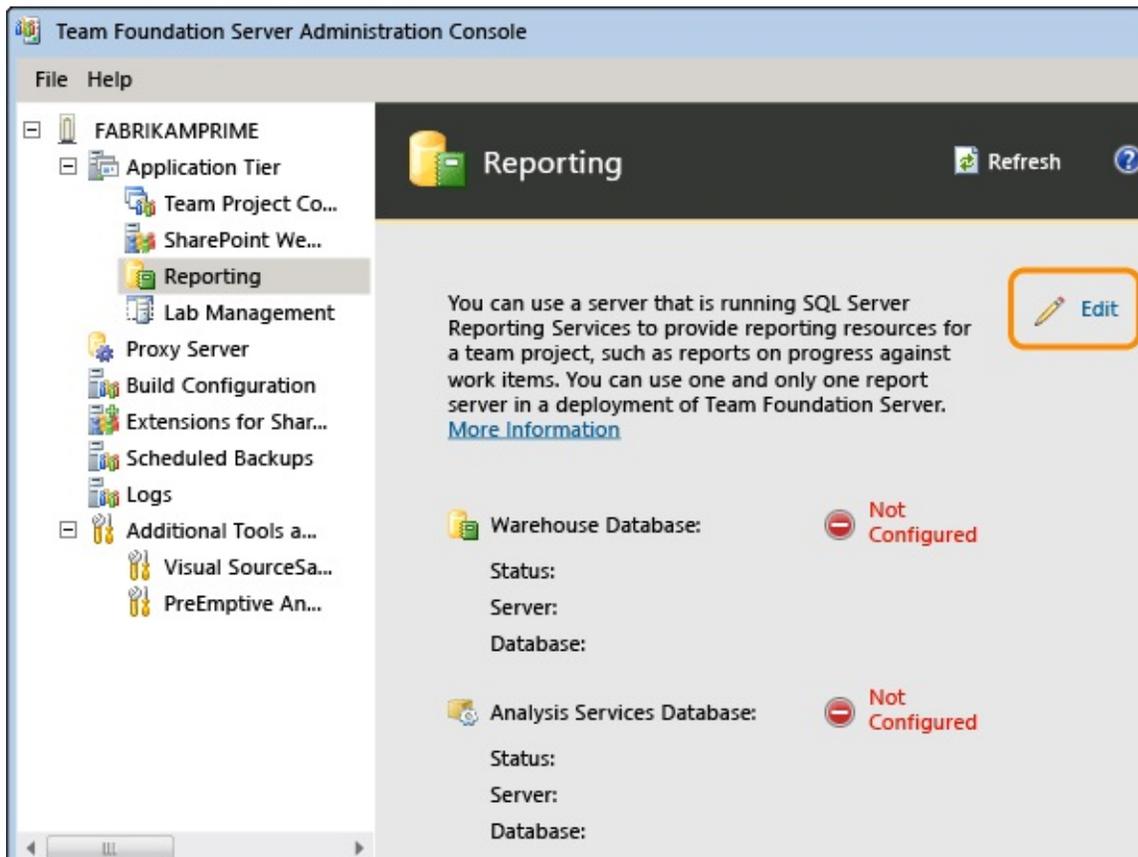
- A detach/attach operation resulted in a misconfiguration of a team project collection. For example, a collection has become attached to two different data-tier servers, which is an unsupported scenario. Correcting the configuration and then rebuilding the data warehouse should resolve the errors.
- You've performed an unsupported backup or restore operation as described in [Back up and restore TFS](#).

Q: How do I add reporting services?

A: If your on-premises TFS isn't configured to support reporting, you can add reports. See [Add reports to a team project](#).

Q: How do I modify the reporting configuration?

A: To modify the reporting configuration for team project collections, use the Edit function that is provided on the Reporting page of the administration console for Team Foundation.



Q: What happens to data that has been purged or destroyed before a rebuild?

A: Data associated with builds or work items that have been permanently deleted from the database will be permanently removed from the data warehouse when you rebuild it.

Also, data in the warehouse or cube that originates from third-party sources might also be lost. Even though most third-party tools are capable of republishing data, that capability depends on the individual vendor. Contact your vendor to determine what (if any) data might be lost.

For details of what data doesn't get deleted from the database when you delete builds, see [this blog post](#).

Deleting builds doesn't remove all associated data from the database. To do that, you must destroy the builds. You use the **TFSBuild destroy** command to destroy builds and to save space in the data warehouse.

See Also

Reference

[RebuildWarehouse Command](#)

Concepts

[Components of the TFS data warehouse](#)

[Manage TFS reports, data warehouse, and analysis services cube](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Change a process control setting for the data warehouse or Analysis Services cube

[See Also](#) [Send Feedback](#)

You can change the refresh frequency and time-out values that control the processing of the data warehouse. You change a process control setting only if you want to tune it for your specific deployment.

During processing, the warehouse adapters pull data from the operational store, transform it as necessary, and write it to the warehouse through the warehouse object model. During typical operations, the relational database (Tfs_Warehouse) is processed within two minutes of changes made to an operational store, and the SQL Server Analysis Services cube (Tfs_Analysis) is processed every two hours.

You might want to decrease or increase the length of time between refreshes to reduce the processing time or reduce the resource demands on the server. You can view the current settings or change a process control setting by using the Warehouse Control Web service.

You can determine the current settings and change a setting by using the Warehouse Control Web service.

Requirements

- SQL Server Reporting Services must be configured for the TFS deployment. If it isn't, see [Add a report server](#).
- You must be a member of the Team Foundation Administrators security group, or you must have the server-level Administer warehouse permission set to Allow. See [Set administrator permissions for Team Foundation Server](#).

- The Microsoft Team Foundation Server Application Pool must be running for the Warehouse Control Web service to be available.

Access the Warehouse Control Web Service

1. Log on to the application-tier server.
2. Open a web browser, type the following string in the Address bar, and then press ENTER:

`http://localhost:8080/VirtualDirectory/TeamFoundation/Administration/v3.0`

For VirtualDirectory, type the IIS Virtual Directory that was specified when TFS was installed. By default, the virtual directory is tfs.

The WarehouseControlWebService page opens.

■ **Change a process control setting**

You can change the refresh frequency of the data warehouse or other process control setting. For a list of all settings, go to

[Process control settings](#).

To change the refresh frequency of the data warehouse

1. From the WarehouseControlWebService page, click ChangeSetting.
2. In the settingID box, type:

RunIntervalSeconds
3. In the newValue box, type the new number in seconds, and then click Invoke.

A confirmation Web page appears and indicates that the RunIntervalSeconds setting has been changed.

To change the refresh frequency of the cube

1. From the WarehouseControlWebService page, click ChangeSetting.
2. In the settingID box, type:

IncrementalProcessIntervalSeconds
3. In the newValue box, type the new number in seconds, and then click Invoke.

⚠ Important

If you reduce the interval to less than the default of two hours (7200 seconds), processing of the data warehouse will consume server resources more frequently. Depending on the volume of data that your deployment

has to process, you may want to reduce the interval to one hour (3600 seconds) or increase it to more than two hours.

A confirmation Web page appears and indicates that the IncrementalProcessIntervalSeconds setting has been changed.

To change another process control setting

1. From the WarehouseControlWebService page, click ChangeSetting.
2. On the ChangeSetting page, type an entry for the settingID and newValue, and then click Invoke.

For a description of each setting and the default values and measures that are assigned to each setting, see the table under [Process control settings](#).

A browser window will open. The service indicates that the setting has been changed to the new value.

Process control settings

All reportable data from all team projects that are defined in all team project collections for an on-premises TFS deployment is written to a single relational database (Tfs_Warehouse). Data from that warehouse is then processed and written to the SQL Server Analysis Services cube (Tfs_Analysis).

You should leave most of these settings at their default assignments. However, on occasion, you may have to modify a setting to meet your specific deployment requirements.

The following table describes each process control setting and provides the SettingID, default value, and unit of measure.

SettingID	Default value	Description
AnalysisSchemaUpdateWaitSeconds	3600 (seconds)	Specifies the time-out that determines how long a job will wait in a running state to perform an update to the cube. If the time-out expires before the schema change finishes, the job quits and restarts later. This time-out interval supports jobs that require exclusive access to the data warehouse resources during the schema change stage.

AnalysisServicesProcessingTimeout	86400 (seconds)	Specifies the time-out, measured in seconds, that determines how long a job will wait for a processing call to Analysis Services to finish.
DailyFullProcessingTime	02:00:00.0000000-08:00	Specifies the time of day when the full processing of the Analysis Services cube is started. By default, it is set to 2 AM.
DataUpdateWaitSeconds	30 (seconds)	Specifies the time-out that determines how long a job will wait in a running state for another adapter that is making schema changes to finish, or for the analysis processing job that changes the cube schema to finish. This time-out is used by jobs to acquire shared access to the warehouse resources during the data change stage. If the time-out expires before the schema

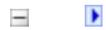
		change process finishes, the job quits and restarts later.
FullProcessIntervalSeconds	86400 (seconds)	Specifies the frequency at which the cube is fully processed. The default value corresponds to 24 hours.
IncrementalProcessIntervalSeconds	7200 (seconds)	Specifies the frequency at which the cube is incrementally updated. The default value corresponds to two hours.
MaxParallelASProcessingCommands	0	Indicates the maximum number of commands that can run in parallel when Team Foundation processes calls to SQL Server Analysis Services. If set to 0, the instance of SQL Server determines the optimal number based on the number of processors that are available on the computer.

RunIntervalSeconds	120 (seconds)	Specifies the frequency at which the relational database is updated.
SchemaUpdateWaitSeconds	120 (seconds)	Specifies the time-out that determines how long a job will wait in a running state to acquire exclusive access to the schema change method. If the time-out expires before the schema change finishes, the job quits and restarts later.
WarehouseCommandSqlTimeout	3600 (seconds)	Specifies the time-out that determines how long a job will wait to acquire an exclusive access to a data warehouse resource.

See Also

Other Resources

[Manually process the TFS data warehouse and analysis services cube](#)



Visual Basic C#

Visual C++

F#

HLSL

JScript

Visual Studio Application Lifecycle Management

Back up and restore TFS

[Send Feedback](#)

The Scheduled Backups Wizard makes it easy to back up your Team Foundation Server (TFS) databases. If you don't regularly back up your databases, you increase the risk of losing productivity or data because of equipment failure or other unexpected events.

All of the information required for restoring a TFS deployment is stored in the TFS data-tier databases on SQL Server. Consequently, you don't have to backup Team Foundation client computers or application-tier servers.

Team Foundation Server Scheduled Backups

Scheduled Backups Wizard

Details
Alerts
Schedules
Review
Readiness Checks
Configure
Complete

Select a Backup Schedule

Nightly Full Backups (i)
 Manual Backups Only (i)
 Custom Schedule (i)

Full Backup Schedule: 02:00 AM

Sunday Thursday
 Monday Friday
 Tuesday Saturday
 Wednesday

Differential Backup Schedule: 02:00 AM

Sunday Thursday
 Monday Friday
 Tuesday Saturday
 Wednesday

Transactional Backup Interval: 30 Minutes

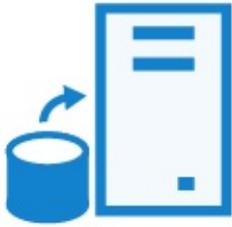
For an overview of TFS databases, see [Understand backing up Team Foundation Server](#). The following topics provide detail procedures for backing up and restoring TFS databases.



Back up your data

- [Configure a backup schedule and plan](#)
- [Manually back up Team Foundation Server](#)

Restore databases



from backup

- [Restore a deployment to new hardware](#)
- [Restore data to the same location](#)

Recover from a hardware failure on the application tier



- [Restore an application-tier server](#)
- [Refresh the data caches on client computers](#)



Back up and restore Lab Management components

[Restore Lab Management components](#)

Q & A

Q: What backup and restore scenarios are and aren't supported?

A: Manual backups must follow the procedures provided here. Manually modifying any TFS database can invalidate your support agreement. It can cause data loss, make it impossible to upgrade or patch TFS, or cause other severe problems.

The following backup scenarios are explicitly not supported:

- Restore a TFS deployment to new hardware, but skip one of the team project collection databases.
- Restore a single team project collection database to an older point in time while leaving either the configuration database or other collection databases at their current point in time.
- Move a backed up team project collection database from one TFS deployment to another without doing a detach/attach.
- Restore the TFS configuration database to an older point in time while leaving collection databases at their current point in time.
- Backup or restore TFS databases using software tools that generate non-standard SQL backups.

Q: Are there situations where I wouldn't want to use the Scheduled Backups wizard?

A: The Scheduled Backups wizard is designed to meet the needs of most deployments. You might need to configure backups manually if your deployment has security restrictions that prevent the use of the tool or has other requirements for backing up databases (for example, for auditing purposes). You can learn how to back up TFS manually [here](#).

≡Q: I deployed TFS across multiple servers. How do I restore it?

A: The steps for restoring TFS in a multiple-server deployment are essentially the same as described in the tutorial for restoring data to a single server. The process is also the same as the process described in a restoration-based move.

≡Q: Can I move TFS?

A: Yes, you can move TFS to new hardware. You can also change its environment, such as its domain.

≡Q: Where can I learn more about TFS architecture and data-tier and application-tier components?

A: See [Team Foundation Server architecture](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Understand backing up Team Foundation Server

[See Also](#) [Send Feedback](#)

You can help protect your deployment from data loss by creating a regular schedule of backups for the databases that Visual Studio Team Foundation Server (TFS) depends on. To restore your deployment of Team Foundation in its entirety, you must first back up all databases for TFS. If your deployment includes SharePoint Products or SQL Server Reporting Services, you must also back up the databases that TFS uses within those components. To prevent synchronization errors or data mismatch errors, you must synchronize all backups to the same time stamp. The easiest way to ensure successful synchronization is by using marked transactions. By routinely marking related transactions in every Team Foundation database, you establish a series of common recovery points in the databases. For step-by-step guidance for backing up a single-server deployment that uses SharePoint Foundation and also uses reporting, see [Configure a backup schedule and plan](#).

Backing up databases

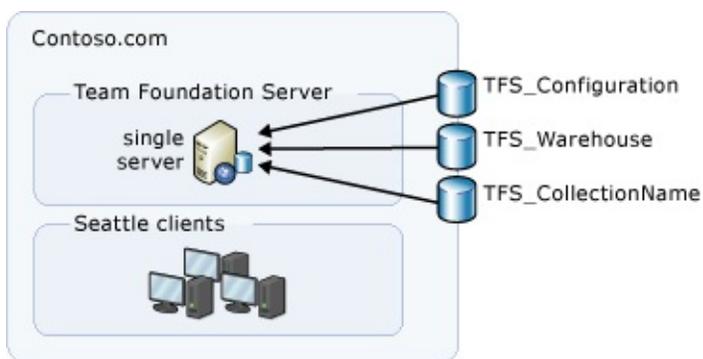
You can create backups of the databases that TFS uses as a key aspect of protecting your deployment of Team Foundation against loss. The following table and accompanying illustrations show which databases you must back up and provide examples of how those databases might be physically distributed in a deployment.

Database Type	Product	Optional component?
Configuration database	Team Foundation Server	No
Warehouse database	Team Foundation Server	No
Team project collection databases	Team Foundation Server	No
SharePoint Products databases	SharePoint Products	Yes
Reporting databases	SQL Server Reporting Services	Yes

Analysis databases	SQL Server Analysis Services	Yes
-----------------------	---------------------------------------	-----

Deployment topologies

Based on your deployment configuration, all the databases that require backing up might be on the same physical server, as in this example topology. This example does not include SharePoint Products or Reporting Services, so you do not have to back up any databases that are associated with reporting, analysis, or SharePoint Products.



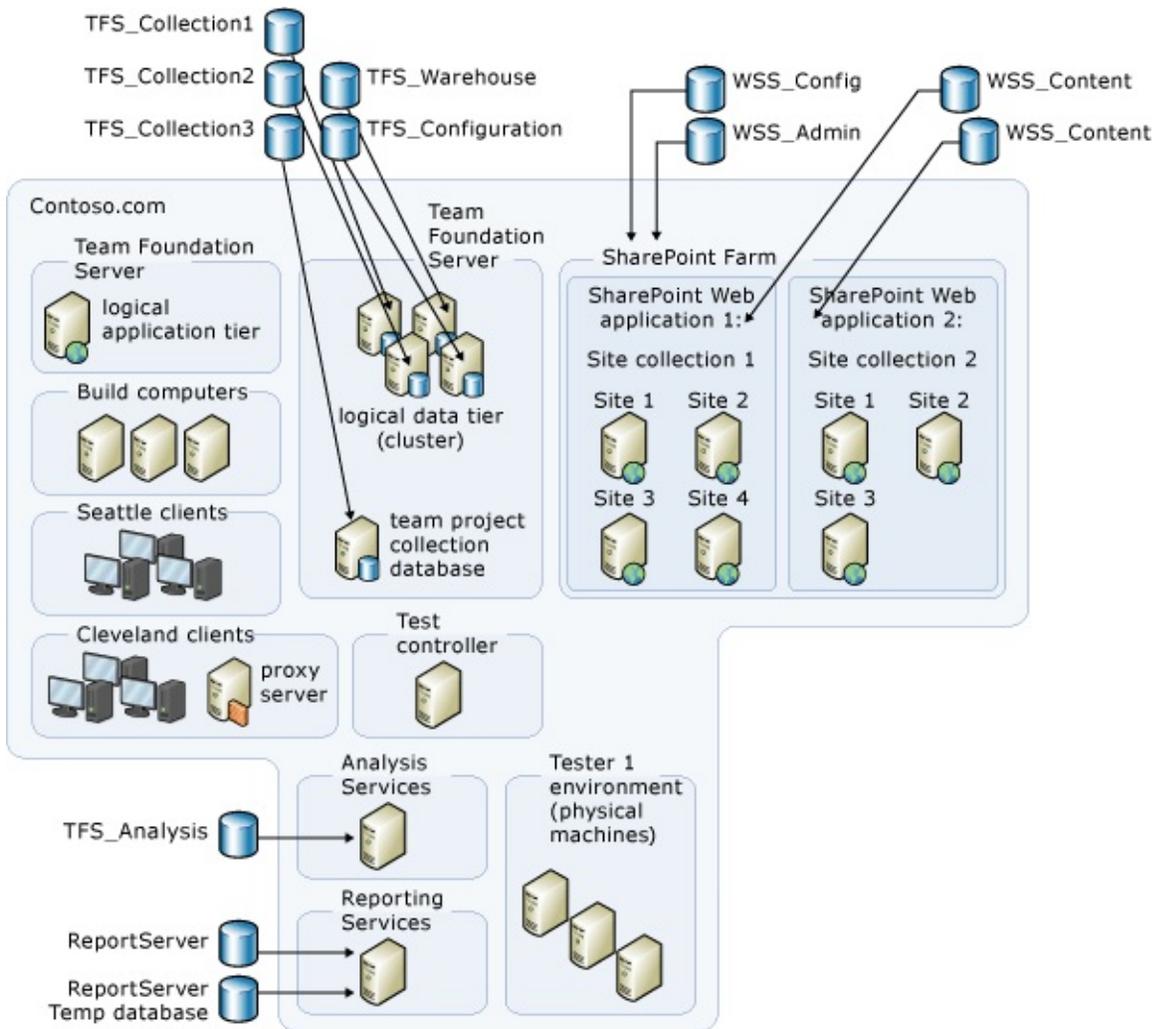
As an alternative, the databases might be distributed across many servers and server farms. In this example topology, you must back up the following databases, which are scaled across six servers or server farms:

- the configuration database
- the warehouse database
- the team project collection databases that are located on the SQL Server cluster
- the collection database that is located on the stand-alone server that is running SQL Server
- the SharePoint Products administrative databases and the site collection

databases for both SharePoint Web applications¹

- the databases that are located on the server that is running Reporting Services
- the database that is located on the server that is running Analysis Services

¹ If your SharePoint databases are scaled across multiple servers, you cannot use the Scheduled Backups feature to back them up. You will have to manually configure backups for those databases, and ensure that those backups are synchronized with the backups for TFS databases. See [Manually back up Team Foundation Server](#) for more details.



In both of these examples, you do not have to back up any of the clients that connect to the server. However, you might need to manually clear the caches for

Team Foundation Server on the client computers before they can reconnect to the restored deployment.

■ **Databases to back up**

The following list provides additional detail about what you must back up, depending on your deployment resources.

Caution

All the databases in the following list are SQL Server databases. Although you can use SQL Server Management Studio to back up individual databases at any time, you should avoid using such individual backups when possible. You might experience unexpected results if you restore from individual backups because the databases that TFS uses are all related. If you back up only one database, the data in that database will be out of synchronization with the data in the other databases.

- **Databases for Team Foundation Server** The logical data tier for Team Foundation Server includes several SQL Server databases, including the configuration database, the warehouse database, and a database for each team project collection in the deployment. These databases might all be on the same server, distributed across several instances in the same SQL Server deployment, or distributed across multiple servers. Regardless of their physical distribution, you must back up all of the databases to the same time stamp to help ensure against data loss. You can perform database backups manually or automatically by using maintenance plans that run at specific times or intervals.

Caution

The list of databases for Team Foundation Server is not static. A new database is created every time you create a collection. When you create a collection, make sure that you add the database for that collection to your maintenance plan.

- **Databases for SharePoint Products** If your deployment uses SharePoint

Products to host team project portals, you must back up several databases. These databases include the administration database for each SharePoint Web application that your deployment uses and the site collection databases that host team project portals. Ideally, your deployment has been configured to use a separate site collection for each team project collection in your deployment. Just as team project collections can be backed up and restored as a unit in Team Foundation Server, site collections can be backed up and restored in SharePoint Products. If one or more collections in your deployment are using sites or sub-sites instead of site collections as their root site, you might not be able to fully back up and restore the collections. For more information, see [Manage team project collections](#).

Note

You might assume that you must back up both databases and Web sites for the team project portal pages. However, SharePoint Products dynamically generates the web sites from the databases. So when you back up the databases, you also back up the sections of the team project that appear as web sites. If you have created custom site collections, site templates, or web parts in SharePoint Products but outside Team Foundation, you must back them up separately. For more information, see the following page on the Microsoft web site: [Backup \(SharePoint Foundation Foundation\)](#).

- **Databases for Reporting Services and Analysis Services** If your deployment uses SQL Server Reporting Services or SQL Server Analysis Services to generate reports for Team Foundation Server, you must back up the reporting and analysis databases. However, you must still regenerate certain databases after restoration, such as the warehouse.
- **Encryption key for the report server** The report server has an encryption key that you must back up. This key safeguards sensitive information that is stored in the database for the report server. You can manually back up this key by using either the Reporting Services Configuration tool or a command-line tool.

Advanced preparation for backups

When you deploy Team Foundation, you should keep a record of the accounts that you create and any computer names, passwords, and setup options that you specify. You should also keep a copy of all recovery materials, documents, and database and transaction log backups at a secure location. To safeguard against a disaster, such as a fire or an earthquake, you should keep duplicates of your server backups in a different location from the location of the servers. This strategy will help protect you against the loss of critical data. As a best practice, you should keep three copies of the backup media, and you should keep at least one copy offsite in a controlled environment.

Important

Perform a trial data restoration periodically to verify that your files are correctly backed up. A trial restoration can reveal hardware problems that do not appear with software verifications.

When you back up and restore a database, you must back up the data onto media with a network address (for example, tapes and disks that have been shared as network drives). Your backup plan should include provisions for managing media, such as the following tactics:

- A tracking and management plan for storing and recycling backup sets.
- A schedule for overwriting backup media.
- In a multi-server environment, a decision to use either centralized or distributed backups.
- A way of tracking the useful life of media.
- A procedure to minimize the effects of the loss of a backup set or backup media (for example, a tape).
- A decision to store backup sets onsite or offsite and an analysis of how this

decision might affect recovery time.

Because data for Team Foundation is stored in SQL Server databases, you do not have to back up the computers on which clients of Team Foundation are installed. If a media failure or disaster that involved those computers were to occur, you can reinstall the client software and reconnect to the server. By reinstalling the client software, your users will have a cleaner and more reliable alternative to restoring a client computer from a backup.

You can back up a server by using the Scheduled Backups features available in Update 2, or by manually creating maintenance plans in SQL Server to back up the databases that relate to your deployment of Team Foundation. The databases for Team Foundation Server work in relationship with one other, and if you create a manual plan, you should back them up and restore them at the same time. For more information about strategies for backing up databases, see the following page on the Microsoft Web site: [Introduction to Backup and Restore Strategies in SQL Server](#).

≡Types of backups

Understanding the types of backups available can help you determine the best options for backing up your deployment. For example, if you are working with a large deployment and want to protect against data loss while efficiently using limited storage resources, you can configure differential backups as well as full data backups. If you are using SQL Server AlwaysOn, you can take backups of your secondary database. You can also try using backup compression or splitting backups across multiple files. Here are brief descriptions of your backup options:

Full data backups (databases) A full database backup is necessary for the recoverability of your deployment. A full backup includes part of the transaction log so that you can recover the full backup. Full backups are self-contained in that they represent the entire database as it existed when you backed it up. For more information, see the following page on the Microsoft Web site: [Full Database Backups](#).

Differential data backups (databases) A differential database backup records only the data that has changed since the last full database backup, which is called the differential base. Differential database backups are smaller and faster than full database backups. This option saves backup time at the cost of increased

complexity. For large databases, differential backups can occur at shorter intervals than database backups, which reduces the work-loss exposure. For more information, see the following page on the Microsoft Web site: [Differential Database Backups](#).

You should also back up your transaction logs regularly. These backups are necessary for recovering data when you use the full database backup model. If you back up transaction logs, you can recover the database to the point of failure or to another specific point in time.

Transaction log backups The transaction log is a serial record of all modifications that have occurred in a database in addition to the transaction that performed each modification. The transaction log records the start of each transaction, the changes to the data, and, if necessary, enough information to undo the modifications made during that transaction. The log grows continuously as logged operations occur in the database.

By backing up transaction logs, you can recover the database to an earlier point in time. For example, you can restore the database to a point before unwanted data was entered or to a point of failure. Besides database backups, transaction log backups must be part of your recovery strategy. For more information, see the following page on the Microsoft Web site: [Working with Transaction Log Backups](#).

Transaction log backups generally use fewer resources than full backups. Therefore, you can create transaction log backups more frequently than full backups, which reduces your risk of losing data. However, sometimes a transaction log backup is larger than a full backup. For example, a database might have a high transaction rate, which causes the transaction log to grow quickly. In this situation, you should create transaction log backups more frequently. For more information, see the following page on the Microsoft Web site: [Troubleshooting a Full Transaction Log](#).

You can perform the following types of transaction log backups:

- A pure log backup contains only transaction log records for an interval, without any bulk changes.
- A bulk log backup contains log and data pages that were changed by bulk operations. Point-in-time recovery is not allowed.

- A tail-log backup is taken from a possibly damaged database to capture the log records that have not yet been backed up. A tail-log backup is taken after a failure to prevent work loss and can contain either pure log or bulk log data.

Because synchronization of data is critical for successful restoration of Team Foundation Server, you should use marked transactions as part of your backup strategy if you are configuring backups manually. For more information, see [Configure a backup schedule and plan](#) and [Manually back up Team Foundation Server](#).

■ Backups for services (application tier)

The only backup that you must perform for the logical application tier is for the encryption key for Reporting Services. If you use the Scheduled Backups feature to back up your deployment, this key will be backed up for you as part of the plan. You might assume that you must back up Web sites that are used as team project portals. If you integrated SharePoint Products as part of your deployment of Team Foundation Server, the portals will be backed up as part of backing up the databases for Team Foundation Server and SharePoint Products. However, if you specified a web site that was not created by using an integrated web application, you must back up and restore those sites manually. In addition, if you have any customizations to SharePoint Products or to services, you must also back those up or otherwise record them so that they can be reproduced on a new server.

Although you can back up an application tier more easily than a data tier, you must perform many steps when you restore an application tier. You must install another application tier for Team Foundation, redirect team project collections to use the new application tier, and redirect the portal sites for team projects.

Default database names

If you do not customize the names of your databases, you can use the following table to identify the databases used in your deployment of Team Foundation Server. As mentioned previously, not all deployments have all these databases. For example, if you did not configure Team Foundation Server with Reporting Services, you will not have the ReportServer or ReportServerTempDB databases. Similarly, you will not have the database for System Center Virtual Machine Manager (SCVMM), which is named VirtualManagerDB, if you did not configure Team Foundation Server to support Lab Management. In addition, the databases that Team Foundation Server uses might be distributed across more than one instance of SQL Server or across more than one server.

Note

By default, the prefix TFS_ is added to the names of any databases that are created automatically when you install Team Foundation Server or while it is operating.

Database	Description
TFS_Configuration	The configuration database for Team Foundation Server contains the catalog, server names, and configuration data for the deployment. The name of this database might include additional characters between TFS_ and Configuration, such as the user name of the person who installed Team Foundation Server. For example, the name of the database might be TFS_UserNameConfiguration.

TFS_Warehouse

The warehouse database contains the data for building the warehouse that Reporting Services uses. The name of this database might include additional characters between TFS_ and Warehouse, such as the user name of the person who installed Team Foundation Server. For example, the name of the database might be TFS_UserNameWarehouse.

TFS_CollectionName

The database for a team project collection contains all data for the team projects in that collection. This data includes source code, build configurations, and lab management configurations. The number of collection databases will equal the number of collections. For example, if you have three collections in your deployment, you must back up three collection databases. The name of each database might include additional characters between TFS_ and CollectionName, such as the user name of the person who created the collection. For example, the name of a collection database might be TFS_UserNameCollectionName.

TFS_Analysis

The database for SQL Server Analysis Services contains the data sources and cubes for your deployment of Team Foundation Server. The name of this database might include additional characters between TFS_ and Analysis, such as the user name of the person who installed Analysis Services. For example, the name of the database might be TFS_UserNameAnalysis.

Note

You can back up this database, but you must rebuild the warehouse from the restored TFS_Warehouse database.

The database for Reporting Services contains the reports and report settings for your deployment of Team Foundation Server.

Note

ReportServer

If Reporting Services is installed on a separate server from Team Foundation Server, this database might not be present on the data-tier server for Team Foundation. In that case, you must configure, back up, and restore it separately from Team Foundation Server.

However, you should synchronize the maintenance of the databases to avoid synchronization errors.

The temporary database for Reporting Services temporarily stores information when you run specific reports.

Note

ReportServerTempDB

If Reporting Services is installed on a separate server from Team Foundation Server, this database might not be present on the data-tier server for Team Foundation. In that case, you must configure, back up, and restore it separately from Team Foundation Server. However, you should synchronize the maintenance of the databases to avoid synchronization errors.

The configuration database for SharePoint Products contains a list of all sites, such as content databases, site templates, custom web parts, and other settings from SharePoint Central Administration.

Note

WSS_Config

You should not use marked transactions if you manually back up or restore the databases that SharePoint Products uses. However, to help avoid synchronization errors, you should attempt to synchronize your backup and restoration schedules for SharePoint Products and Team Foundation Server. For more information, see [Create a backup plan for SharePoint Foundation](#) .

The content database for SharePoint Products contains the actual content in team project portals.

 **Note**

WSS_Content

The name of this database will vary based on the version of SharePoint Products that is installed and whether the person who installed it customized the name. You should not use marked transactions if you manually back up or restore the databases that SharePoint Products uses. However, to help avoid synchronization errors, you should attempt to synchronize your backup and restoration schedules for SharePoint Products and Team Foundation Server. For more

information, see [Create a backup plan for SharePoint Foundation](#) .

The administration database for SharePoint Products contains the security information for users, roles, and databases.

Note

WSS_AdminContent

You should not use marked transactions if you manually back up or restore the databases that SharePoint Products uses. However, to help avoid synchronization errors, you should attempt to synchronize your backup and restoration schedules for SharePoint Products and Team Foundation Server. For more information, see [Create a backup plan for SharePoint Foundation](#) .

The administration database for SCVMM contains the information that you view in the SCVMM Administrator Console, such as virtual machines, virtual machine hosts, virtual machine library servers, and their properties.

Note

VirtualManagerDB

If SCVMM is installed on a separate server from Team Foundation Server, this database might not be present on the data-tier server for Team Foundation. In that case, you must configure, back up, and restore it separately from Team Foundation Server. However, you should use marked transactions and synchronize the maintenance of the databases to avoid synchronization errors.

See Also

Concepts

[Back up and restore TFS](#)

Other Resources

[Restore a deployment to new hardware](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Manually back up Team Foundation Server

[Send Feedback](#)

You can manually back up data for Visual Studio Team Foundation Server by using the tools that SQL Server provides. However, you might need to configure backups manually if your deployment has security restrictions that prevent use of that tool. To manually back up Team Foundation Server, you must not only back up all databases that the deployment uses, you must also synchronize the backups to the same point in time. You can manage this synchronization most effectively if you use marked transactions. If you routinely mark related transactions in every database that Team Foundation uses, you establish a series of common recovery points in those databases. If you regularly back up those databases, you reduce the risk of losing productivity or data because of equipment failure or other unexpected events.

Caution

You should not manually modify any of the TFS databases unless you're instructed to do so by Microsoft Support or you're following the procedures described in this document. Any other modifications can invalidate your service agreement.

If your deployment uses SQL Server Reporting Services, you must back up not only the databases but also the encryption key. For more information, see [How to: Back Up the Reporting Services Encryption Key](#).

The procedures in this topic explain how to create maintenance plans that perform either a full or an incremental backup of the databases and how to create tables and stored procedures for marked transactions. For maximum data protection, you should schedule full backups to run daily or weekly and

incremental backups to run hourly. You can also back up of the transaction logs. For more information, see the following page on the Microsoft website: [Creating Transaction Log Backups](#).

☑Note

Many procedures in this topic specify the use of SQL Server Management Studio. If you installed SQL Server Express Edition, you cannot use that tool unless you download SQL Server Management Studio Express. To download this tool, see the following page on the Microsoft website: [Microsoft SQL Server 2008 Management Studio Express](#).

In this topic:

- Required Permissions
- [Identify Databases](#)
- [Create Tables in Databases](#)
- [Create a Stored Procedure for Marking Tables](#)
- [Create a Stored Procedure for Marking All Tables At Once](#)
- [Create a Stored Procedure to Automatically Mark Tables](#)
- [Create a Scheduled Job to Run the Table-Marking Procedure](#)
- [Create a Maintenance Plan For Full Backups](#)
- [Create a Maintenance Plan For Differential Backups](#)
- [Create a Maintenance Plan For Transaction Backups](#)
- [Back Up the Reporting Services Encryption Key](#)
- [Create a Back Up Plan for SharePoint Foundation](#)
- [Back Up Additional Lab Management Components](#)

Required permissions

To perform this procedure, you must be a member of all the following groups:

- The Administrators security group on the server that is running the administration console for Team Foundation.
- The SQL Server System Administrator security group. Alternatively, your SQL Server Perform Back Up and Create Maintenance Plan permissions must be set to Allow on each instance of SQL Server that hosts the databases that you want to back up.
- The Farm Administrators group in SharePoint Foundation, or an account with the permissions required to back up the farm.

Identify databases

Before you begin, you should take the time to identify all the databases you will need to back up if you would ever have to fully restore your deployment. This includes databases for SharePoint Foundation and SQL Server Reporting Services. These might be on the same server, or you might have databases distributed across multiple servers. For a complete table and description of TFS databases, including the default names for the databases, see

[Understand backing up Team Foundation Server.](#)

To identify databases

1. Open SQL Server Management Studio, and connect to the database engine.
2. In SQL Server Management Studio, in Object Explorer, expand the name of the server and then expand Databases.
3. Review the list of databases and identify those used by your deployment.

For example, Fabrikam, Inc.'s TFS deployment is a single-server configuration, and it uses the following databases:

- the configuration database (Tfs_Configuration)
- the collection database (Tfs_DefaultCollection)
- the database for the data warehouse (Tfs_Warehouse)
- the reporting databases (ReportServer and ReportServerTempDB)
- the databases used by SharePoint Foundation (WSS_AdminContent, WSS_Config, WSS_Content, and WSS_Logging)

⚠ Important

Unlike the other databases in the deployment, the databases used by

SharePoint Foundation should not be manually backed up using the tools in SQL Server. Follow the separate procedure "[Create a Back Up Plan for SharePoint Foundation](#)" later in this topic for backing up these databases.

■ Create tables in databases

To make sure that all databases are restored to the same point, you can create a table in each database to mark transactions. You can use the Query function in SQL Server Management Studio to create an appropriate table in each database.

⚠ Important

Do not create tables in any databases that SharePoint Products uses.

To create tables to mark related transactions in databases that Team Foundation uses

1. Open SQL Server Management Studio, and connect to the database engine.
2. In SQL Server Management Studio, highlight the name of the server, open the submenu, and then choose New Query.

The Database Engine Query Editor window opens.

3. On the Query menu, choose SQLCMD Mode.

The Query Editor executes sqlcmd statements in the context of the Query Editor. If the Query menu does not appear, select anywhere in the new query in the Database Engine Query Editor window.

4. On the SQL Editor toolbar, open the Available Databases list, and then choose TFS_Configuration.

📌 Note

TFS_Configuration is the default name of the configuration database. This name is customizable and might vary.

5. In the query window, enter the following script to create a table in the

configuration database:

 **Copy Code**

```
Use Tfs_Configuration
Create Table Tbl_TransactionLogMark
(
  logmark int
)
GO
Insert into Tbl_TransactionLogMark (logmark) Values (1)
GO
```

6. Choose the F5 key to run the script.

If the script is well-formed, the message "(1 row(s) affected.)" appears in the Query Editor.

7. (Optional) Save the script.
8. Repeat steps 4–7 for every database in your deployment of TFS, except for those used by SharePoint Products. In the fictitious Fabrikam, Inc. deployment, you would repeat this process for all of the following databases:
 - Tfs_Warehouse
 - Tfs_DefaultCollection
 - ReportServer
 - ReportServerTempDB

■ Create a stored procedure for marking tables

After the tables have been created in each database that you want to back up, you must create a procedure for marking the tables.

To create a stored procedure to mark transactions in each database that Team Foundation Server uses

1. In SQL Server Management Studio, open a query window, and make sure that SQLCMD Mode is turned on.
2. On the SQL Editor toolbar, open the Available Databases list, and then choose TFS_Configuration.
3. In the query window, enter the following script to create a stored procedure to mark transactions in the configuration database:

Copy Code

```
Create PROCEDURE sp_SetTransactionLogMark
@name nvarchar (128)
AS
BEGIN TRANSACTION @name WITH MARK
UPDATE Tfs_Configuration.dbo.Tbl_TransactionLogMark SET logmark
COMMIT TRANSACTION
GO
```

4. Choose the F5 key to run the procedure.

If the procedure is well-formed, the message "Command(s) completed successfully." appears in the Query Editor.

5. (Optional) Save the procedure.
6. Repeat steps 2–5 for every TFS database. In the Fabrikam, Inc.

deployment, you would repeat this process for all of the following databases:

- Tfs_Warehouse
- Tfs_DefaultCollection
- ReportServer
- ReportServerTempDB

Tip

Make sure that you select the name of the database you want to create the stored procedure for from the Available Database list in Object Explorer before you create the procedure. Otherwise when you run the script the command will display an error that the stored procedure was already exists.

■ Create a stored procedure for marking all tables at once

To make sure that all databases are marked, you can create a procedure that will run all the procedures that you just created for marking the tables. Unlike the previous procedures, this procedure runs only in the configuration database.

To create a stored procedure that will run all stored procedures for marking tables

1. In SQL Server Management Studio, open a query window, and make sure that SQLCMD Mode is turned on.
2. On the SQL Editor toolbar, open the Available Databases list, and then choose TFS_Configuration.
3. In the query window, create a stored procedure that executes the stored procedures that you created in each database that TFS uses. Replace ServerName with the name of the server that is running SQL Server, and replace Tfs_CollectionName with the name of the database for each team project collection.

In the example deployment, the name of the server is FABRIKAMPRIME, and there is only one team project collection in the deployment, the default one created when she installed Team Foundation Server (DefaultCollection). With that in mind, you would create the following script:

Copy Code

```
CREATE PROCEDURE sp_SetTransactionLogMarkAll
@name nvarchar (128)
AS
BEGIN TRANSACTION
EXEC [FABRIKAMPRIME].Tfs_Configuration.dbo.sp_SetTransactionLogMark
EXEC [FABRIKAMPRIME].ReportServer.dbo.sp_SetTransactionLogMark (
EXEC [FABRIKAMPRIME].ReportServerTempDB.dbo.sp_SetTransactionLogMark
```

```
EXEC [FABRIKAMPRIME].Tfs_DefaultCollection.dbo.sp_SetTransaction
EXEC [FABRIKAMPRIME].Tfs_Warehouse.dbo.sp_SetTransactionLogMark
COMMIT TRANSACTION
GO
```

4. Choose the F5 key to run the procedure.

Note

If you have not restarted SQL Server Management Studio since you created the stored procedures for marking transactions, one or more red wavy lines might underscore the name of the server and the names of the databases. However, the procedure should still run.

If the procedure is well-formed, the message "Command(s) completed successfully." appears in the Query Editor.

5. (Optional) Save the procedure.

■ Create a stored procedure to automatically mark tables

When you have a procedure that will run all stored procedures for table marking, you must create a procedure that will mark all tables with the same transaction marker. You will use this marker to restore all databases to the same point.

To create a stored procedure to mark the tables in each database that Team Foundation Server uses

1. In SQL Server Management Studio, open a query window, and make sure that SQLCMD Mode is turned on.
2. On the SQL Editor toolbar, open the Available Databases list, and then choose TFS_Configuration.
3. In the query window, enter the following script to mark the tables with 'TFSMark':

Copy Code

```
EXEC sp_SetTransactionLogMarkAll 'TFSMark'  
GO
```

■ Note

TFSMark is an example of a mark. You can use any sequence of supported letters and numbers in your mark. If you have more than one marked table in the databases, record which mark you will use to restore the databases. For more information, see the following page on the Microsoft website:

[Using Marked Transactions.](#)

- Choose the F5 key to run the procedure.

If the procedure is well-formed, the message "(1 row(s) affected)" appears in the Query Editor. The WITH MARK option applies only to the first "BEGIN TRAN WITH MARK" statement for each table that has been marked.

- Save the procedure.

■ Create a scheduled job to run the table-marking procedure

Now that you have created and stored all the procedures that you need, you must schedule the table-marking procedure to run just before the scheduled backups of the databases. You should schedule this job to run approximately one minute before the maintenance plan for the databases runs.

To create a scheduled job for table marking in SQL Server Management Studio

1. In Object Explorer, expand SQL Server Agent, open the Jobs menu, and then choose New Job.

The New Job window opens.

2. In Name, specify a name for the job. For example, you might choose "MarkTableJob" for your job name.
3. (Optional) In Description, specify a description of the job.
4. In Select a page, choose Steps and then choose New.
5. The New Job Step window opens.
6. In Step Name, specify a name for the step.
7. In Database, choose the name of the configuration database. For example, if your deployment uses the default name for that database, TFS_Configuration, you would choose that database from the drop-down list.
8. Choose Open, browse to the procedure that you created for marking the tables, choose Open two times, and then choose OK.

■ **Note**

The procedure that you created for marking the tables runs the following step:

Copy Code

```
EXEC sp_SetTransactionLogMarkAll 'TFSSMark'
```

9. In Select a page, choose Schedules, and then choose New.

The New Job Schedule window opens.

10. In Name, specify a name for the schedule.
11. In Frequency, change the frequency to match the plan that you will create for backing up the databases. For example, you might want to run incremental backups daily at 2 A.M., and full backups on Sunday at 4 A.M. For marking the databases for the incremental backups, you would change the value of Occurs to Daily. When you create another job to mark the databases for the weekly full backup, you would keep the value of Occurs at Daily, and select the Sunday check box.
12. In Daily Frequency, change the occurrence so that the job is scheduled to run one minute before the backup for the databases, and then choose OK. In the example deployment, in the job for the incremental backups, you would specify 1:59 A.M.. In the job for the full backup, you would specify 3:59 A.M..
13. In New Job, choose OK to finish creating the scheduled job.

■ Create a maintenance plan for full backups

After you create a scheduled job for marking the databases, you can use the Maintenance Plan Wizard to schedule full backups of all of the databases that your deployment of TFS uses.

⚠ Important

If your deployment is using the Enterprise or Datacenter editions of SQL Server, but you think you might want to restore databases to a server running Standard edition, you must use a backup set that was made with SQL Server compression disabled. Unless you disable data compression, you will not be able to successfully restore Enterprise or Datacenter edition databases to a server running Standard edition. You should turn off compression before creating your maintenance plans. To turn off compression, follow the steps in the

[Microsoft Knowledge Base article](#).

To create a maintenance plan for full backups

1. In SQL Server Management Studio, expand the Management node, open the Maintenance Plans sub-menu, and then choose Maintenance Plan Wizard.
2. On the welcome page for the SQL Server Maintenance Plan Wizard, choose Next.

The Select Plan Properties page appears.

3. In the Name box, specify a name for the maintenance plan.

For example, you might create a plan for full backups named TfsFullDataBackup.

4. Choose Single schedule for the entire plan or no schedule, and then choose Change.
5. Under Frequency and Daily Frequency, specify options for your plan. For example, you might specify a weekly backup to occur on Sunday in Frequency, and specify 4 A.M. in Daily Frequency.

Under Duration, leave the default value, No end date. Choose OK, and then choose Next.

6. On the Select Maintenance Tasks page, select the Backup Database (Full), Execute SQL Server Agent Job, and Back up Database (Transaction Log) check boxes, and then choose Next.
7. On the Select Maintenance Task Order page, change the order so that the full backup runs first, then the Agent job, and then the transaction log backup, and then choose Next.

For more information about this dialog box, choose the F1 key. Also, search for Maintenance Plan Wizard on the following page of the Microsoft website: [SQL Server Books Online](#).

8. On the Define Back Up Database (Full) Task page, choose the down arrow, choose All Databases, and then choose OK.
9. Specify the backup options for saving the files to disk or tape, as appropriate for your deployment and resources, and then choose Next.
10. On the Define Execute SQL Server Agent Job Task page, select the check box for the scheduled job that you created for table marking, and then choose Next.
11. On the Define Back Up Database (Transaction Log) Task page, choose the down arrow, choose All Databases, and then choose OK.
12. Specify the backup options for saving the files to disk or tape as appropriate for your deployment and resources, and then choose Next.
13. On the Select Report Options page, specify report distribution options, and then choose Next two times.

14. On the Complete the Wizard page, choose Finish.

SQL Server creates the maintenance plan and backs up the databases that you specified based on the frequency that you specified.

■ Create a maintenance plan for differential backups

You can use the Maintenance Plan Wizard to schedule differential backups for all databases that your deployment of TFS uses.

⚠ Important

SQL Server Express does not include the Maintenance Plan Wizard. You must manually script the schedule for your differential backups. For more information, see the following topic on the Microsoft website:

[How to: Create a Differential Database Backup \(Transact-SQL\).](#)

To create a maintenance plan for differential backups

1. Log on to the server that is running the instance of SQL Server that contains the databases that you want to back up.
2. Open SQL Server Management Studio.
 1. In the Server type list, choose Database Engine.
 2. In the Server name and Authentication lists, choose the appropriate server and authentication scheme.
 3. If your instance of SQL Server requires it, in User name and Password, specify the credentials of an appropriate account.
 4. Choose Connect.
3. In SQL Server Management Studio, expand the Management node, open the sub-menu, choose Maintenance Plans, and then choose Maintenance Plan Wizard.
4. On the welcome page for the SQL Server Maintenance Plan Wizard, choose

Next.

5. On the Select Plan Properties page, in the Name box, specify a name for the maintenance plan.

For example, you could name a plan for differential backups
TfsDifferentialBackup.

6. Choose Single schedule for the entire plan or no schedule, and then choose Change.
7. Under Frequency and Daily Frequency, specify options for your backup plan.

Under Duration, leave the default value, No end date. Choose OK, and then choose Next.

8. On the Select Maintenance Tasks page, select the Back up Database (Differential) check box, and then choose Next.
9. On the Define Back Up Database (Differential) Task page, choose the down arrow, choose All Databases, and then choose OK.
10. Specify the backup options for saving the files to disk or tape as appropriate for your deployment and resources, and then choose Next.
11. On the Select Report Options page, specify report distribution options, and then choose Next two times.
12. On the Complete the Wizard page, choose Finish.

SQL Server creates the maintenance plan and backs up the databases that you specified based on the frequency that you specified.

■ Create a maintenance plan for transaction logs

You can use the Maintenance Plan Wizard to schedule transaction log backups for all databases that your deployment of TFS uses.

⚠ Important

SQL Server Express does not include the Maintenance Plan Wizard. You must manually script the schedule for transaction-log backups. For more information, see the following topic on the Microsoft website:

[How to: Create a Transaction Log Backup \(Transact-SQL\)](#).

To create a maintenance plan for transaction log backups

1. Log on to the server that is running the instance of SQL Server that contains the databases that you want to back up.
2. Open SQL Server Management Studio.
3. In the Server type list, choose Database Engine.
 1. In the Server name and Authentication lists, choose the appropriate server and authentication scheme.
 2. If your instance of SQL Server requires it, in User name and Password, specify the credentials of an appropriate account.
 3. Choose Connect.
4. In SQL Server Management Studio, expand the Management node, open the submenu, choose Maintenance Plans, and then choose Maintenance Plan Wizard.
5. On the welcome page for the SQL Server Maintenance Plan Wizard, choose

Next.

The Select Plan Properties page appears.

6. In the Name box, specify a name for the maintenance plan.

For example, you could name a plan to back up transaction logs
TfsTransactionLogBackup.

7. Choose Single schedule for the entire plan or no schedule, and then choose Change.

8. Under Frequency and Daily Frequency, specify options for your plan.

Under Duration, leave the default value, No end date.

9. Choose OK, and then choose Next.

10. On the Select Maintenance Tasks page, select the Execute SQL Server Agent Job and Back up Database (Transaction Log) check boxes, and then choose Next.

11. On the Select Maintenance Task Order page, change the order so that the Agent job runs before the transaction-log backup, and then choose Next.

For more information about this dialog box, choose the F1 key. Also, search for Maintenance Plan Wizard on the following page of the Microsoft website: [SQL Server Books Online](#).

12. On the Define Execute SQL Server Agent Job Task page, select the check box for the scheduled job that you created for table marking, and then choose Next.

13. On the Define Back Up Database (Transaction Log) Task page, choose the down arrow, choose All Databases, and then choose OK.

14. Specify the backup options for saving the files to disk or tape as appropriate for your deployment and resources, and then choose Next.

15. On the Select Report Options page, specify report distribution options, and

then choose Next two times.

16. On the Complete the Wizard page, choose Finish.

SQL Server creates the maintenance plan and backs up the transaction logs for the databases that you specified based on the frequency that you specified.

■ Back up the encryption key for reporting services

You must back up the encryption key for Reporting Services as part of backing up your system. Without this encryption key, you will not be able to restore the reporting data. For a single-server deployment of TFS, you can back up the encryption key for SQL Server Reporting Services by using the Reporting Services Configuration tool. You could also choose to use the **RSKEYMGMT** command-line tool, but the configuration tool is simpler. For more information about **RSKEYMGMT**, see the following page on the Microsoft website:

[RSKEYMGMT Utility](#).

To back up the encryption key by using the Reporting Services Configuration tool

1. On the server that is running Reporting Services, open Reporting Services Configuration Manager.

The Report Server Installation Instance Selection dialog box opens.

2. Specify the name of the data-tier server and the database instance, and then choose Connect.
3. In the navigation bar on the left side, choose Encryption Keys, and then choose Backup.

The Encryption Key Information dialog box opens.

4. In File Location, specify the location where you want to store a copy of this key.

You should consider storing this key on a separate computer from the one that is running Reporting Services.

5. In Password, specify a password for the file.

6. In Confirm Password, specify the password for the file again, and then choose OK.

■ Create a backup plan for SharePoint Foundation

Unlike Team Foundation Server, which uses the scheduling tools in SQL Server Management Studio, there is no built-in scheduling system for backups in SharePoint Foundation, and SharePoint specifically recommends against any scripting that marks or alters its databases. To schedule backups so that they occur at the same time as the backups for TFS, SharePoint Foundation guidance recommends that you create a backup script by using Windows PowerShell, and then use Windows Task Scheduler to run the backup script at the same time as your scheduled backups of TFS databases. This will help you keep your database backups in sync.

⚠ Important

Before proceeding with the procedures below, you should review the latest guidance for SharePoint Foundation. The procedures below are based on that guidance, but might have become out of date. Always follow the latest recommendations and guidance for the version of SharePoint Products you use when managing that aspect of your deployment. For more information, see the links included with each of the procedures in this section.

To create scripts to perform full and differential backups of the farm in SharePoint Foundation

1. Open a text editor, such as Notepad.
2. In the text editor, type the following, where *BackupFolder* is the UNC path to a network share where you will back up your data:

Copy Code

```
Backup-SPFarm -Directory BackupFolder -BackupMethod Full
```

Tip

There are a number of other parameters you could use when backing up the farm. For more information, see

[Back up a farm](#) and [Backup-SPFarm](#).

- Save the script as a .PS1 file. Consider giving the file an obvious name, such as "SharePointFarmFullBackupScript.PS1" or some meaningful equivalent.
- Open a new file, and create a second backup file, only this time specifying a differential backup:

Copy Code

```
Backup-SPFarm -Directory BackupFolder -BackupMethod Differential
```

- Save the script as a .PS1 file. Consider giving the file an obvious name, such as "SharePointFarmDiffBackupScript.PS1".

Important

By default, PowerShell scripts will not execute on your system unless you have changed PowerShell's execution policy to allow scripts to run. For more information, see [Running Windows PowerShell Scripts](#).

After you have created your scripts, you must schedule them to execute following the same schedule and frequency as the schedule you created for backing up Team Foundation Server databases. For example, if you scheduled differential backups to execute daily at 2 A.M., and full backups to occur on Sundays at 4 A.M., you will want to follow the exact same schedule for your farm backups.

To schedule your backups, you must use Windows Task Scheduler. In addition, you must configure the tasks to run using an account with sufficient permissions to read and write to the backup location, as well as permissions to execute backups in SharePoint Foundation. Generally speaking, the simplest way to do

this is to use a farm administrator account, but you can use any account as long as all of the following criteria are met:

- The account specified in Windows Task Scheduler is an administrative account.
- The account specified for the Central Administration application pool and the account you specify for running the task have read/write access to the backup location.
- The backup location is accessible from the server running SharePoint Foundation, SQL Server, and Team Foundation Server.

To schedule backups for the farm

1. Choose Start, choose Administrative Tools, and then choose Task Scheduler.
2. In the Actions pane, choose Create Task.
3. On the General tab, in Name, specify a name for this task, such as "Full Farm Backup." In Security options, specify the user account under which to run the task if it is not the account you are using. Then choose Run whether user is logged on or not, and select the Run with highest privileges check box.
4. On the Actions tab, choose New.

In the New Action window, in Action, choose Start a program. In Program/script, specify the full path and file name of the full farm backup .PS1 script you created, and then choose OK.

5. On the Triggers tab, choose New.

In the New Trigger window, in Settings, specify the schedule for performing the full backup of the farm. Make sure that this schedule exactly matches the schedule for full backups of the Team Foundation Server databases, including the recurrence schedule, and then choose OK.

6. Review all the information in the tabs, and then choose OK to create the

task for the full backup for the farm.

7. In the Actions pane, choose Create Task.
8. On the General tab, in Name, specify a name for this task, such as "Differential Farm Backup." In Security options, specify the user account under which to run the task if it is not the account you are using, choose Run whether user is logged on or not, and select the Run with highest privileges check box.

9. On the Actions tab, choose New.

In the New Action window, in Action, choose Start a program. In Program/script, specify the full path and file name of the differential farm backup .PS1 script you created, and then choose OK.

10. On the Triggers tab, choose New.

In the New Trigger window, in Settings, specify the schedule for performing the full backup of the farm. Make sure that this schedule exactly matches the schedule for full backups of the Team Foundation Server databases, including the recurrence schedule, and then choose OK.

11. Review all the information in the tabs, and then choose OK to create the task for the differential backup for the farm.
12. In Active Tasks, refresh the list and make sure that your new tasks are scheduled appropriately, and then close Task Scheduler. For more information about creating and scheduling tasks in Task Scheduler, see [Task Scheduler How To](#).

■ Back up additional lab management components

If you use Visual Studio Lab Management in your deployment of Team Foundation Server, you must also back up each machine and component that Lab Management uses. The hosts for the virtual machines and the SCVMM library servers are separate physical computers that are not backed up by default. You must specifically include them when you plan your backup and restoration strategies. The following table summarizes what you should back up whenever you back up Team Foundation Server.

Machine	Component
Server that is running System Center Virtual Machine Manager 2008 (SCVMM) R2	<ul style="list-style-type: none">• SQL Server database (user accounts, configuration data)• Virtual machines (VMs)
Physical host for the virtual machines	<ul style="list-style-type: none">• Templates• Host configuration data (virtual networks)• Virtual

SCVMM library server

machines

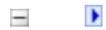
- Templates
- Virtual hard disks (VHDs)
- ISO images

The following table contains tasks and links to procedural or conceptual information about how to back up the additional machines for an installation of Lab Management. You must perform the tasks in the order shown, without skipping any tasks.

To back up the machines that are running any SCVMM components, you must be a member of the Backup Operators group on each machine.

Common Tasks	Detailed instructions
1. Back up the server that is running System Center Virtual Machine Manager 2008 R2.	
2. Back up the library servers for SCVMM.	•
3. Back up each physical host for the virtual machines.	

[Backing Up and Restoring the SCVMM Database](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Restore an application-tier server

[See Also](#) [Send Feedback](#)

The databases for Team Foundation store all data for your deployment of Team Foundation Server. Even if you back up the application-tier server, you will not back up any data for Team Foundation Server. However, if the hardware of an application-tier server fails, you can install another application-tier server and configure it to use the databases for your deployment. That server will then replace the offline server as the application-tier server for the deployment. If your application-tier server hosted SharePoint Products, you must also restore that software on the new hardware. For more information, see [Backup \(SharePoint Foundation\)](#), [Backup and Recovery \(SharePoint Server\)](#), or [Protecting and restoring a farm \(Office SharePoint Server 2007\)](#).

Note

After you restore an application tier to new hardware, make sure that all users, groups, and service accounts for your deployment are configured with the permissions that they require to perform necessary tasks. For example, administrators for Team Foundation must be members of the local Administrators group on the application-tier server so that they can open the administration console. For more information, see [Add users to team projects](#), [Set administrator permissions for team project collections](#), [Set administrator permissions for Team Foundation Server](#), and [Service accounts and dependencies in Team Foundation Server](#).

You can also add more than one application-tier server to a deployment of Team Foundation Server, but you must configure clients to connect to that server as a separate application tier. You cannot configure automatic load balancing between application-tier servers. For actual load balancing and transparency to clients, you must first install and configure a hardware or software device for network

load balancing (NLB).

To install and configure a server as the application-tier server

1. Stop the application pools and services that Team Foundation Server uses.

For more information, see [TFSServiceControl Command](#).

2. If you are using Network Service as the service account for Team Foundation (TFSService), on the application-tier server, open a Command Prompt window, and change directories to Drive:%Program Files%\Microsoft Team Foundation Server 12.0\Tools. At the command prompt, enter the following command:

```
TfsConfig Accounts /add /account:"NT Authority\Network Service"  
/accountType:ApplicationTier /SQLInstance:ServerName  
/DatabaseName:DatabaseName
```

Note

For more information, see [Accounts Command](#).

3. Install Team Foundation Server on the new server, and start the Application-Tier Only wizard.
4. If you are using Visual Studio Lab Management, install the System Center Virtual Machine Manager (SCVMM) Administrator Console on the application tier, and configure it to connect to the server that is running SCVMM.

For more information, see [Configure Lab Management for SCVMM environments](#).

5. If the computer name has changed, open the administration console for Team Foundation.
6. In the navigation bar, choose Application Tier, and then choose Change URLs.

The Change URLs window opens.

7. In Notification URL, specify the URL for the new application-tier server, and then choose OK.

Note

The name of the old application-tier server will still appear in the list of application-tier servers in the administration console for Team Foundation. If you select the Filter out machines that have not connected in more than 3 days check box, the old server will disappear from the list within three days.

See Also

Concepts

[Restore Lab Management components](#)

[Team Foundation Server architecture](#)

Other Resources

[Restore a deployment to new hardware](#)

[Open the Team Foundation Administration Console](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Restore a deployment to new hardware

[Send Feedback](#)

No matter how well you maintain your hardware, server failures happen. As an experienced administrator, you know to prepare against such an event. If the server that hosts your deployment of Team Foundation Server (TFS) fails or is destroyed in a disaster, you can use the backups you made of the databases and of the SharePoint Foundation 2010 farm to restore the deployment to a new server. While restoring an entire deployment isn't simple, with reliable data backups and the installation media for the programs you need to install, you can restore your deployment to full functionality.

In addition to the backup data, you need the installation media for TFS, SQL Server, and SharePoint Foundation, including access to all service packs and cumulative updates you applied to your original deployment. You must install and configure the software on the same operating system as the original server, and you should ensure that you restore the backups to the same versions, editions, and servicing levels of the software that you backed up.

Note

This tutorial contains fictitious users, servers, and companies as part of the examples that help illustrate the procedures. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, places, or events is intended or should be inferred.

Tutorial contents

Title	Description
Prepare the new hardware	Learn how to choose the right replacement hardware, review the prerequisites for a replacement server, and install the software you'll need in order to restore the deployment. As part of this process, you must choose a replacement server, and then install SQL Server, SharePoint Foundation, and the tools that are required to reconfigure the restored TFS databases.
Restore the databases	Learn how to restore the databases from backups, and to restore the SharePoint farm. As part of this process, you must restore the configuration, collection, and reporting databases, and restore the SharePoint farm from backup.
	Learn how to use the

[Install and configure Team Foundation Server](#)

Application-Tier Only wizard to install the web services and applications for TFS. As part of this process, you must install Team Foundation Server and the extensions for SharePoint Products, and configure URLs, service accounts, reporting, and analysis resources to complete the data restoration part of the recovery.

[Reconnect services and users](#)

Learn how to complete configuration of TFS after its databases have been restored. As part of this process, you will start team project collections, add groups of users to the new server, and clear the data caches to help prevent data errors. Learn how to clear the work item cache on the new server, and learn how users can clear their version control caches on their own computers so that they don't experience data caching problems when connecting to the new server.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Prepare the new hardware

[Send Feedback](#)

This topic, the first part of the Restore a Single-Server tutorial, teaches you how to install and configure the prerequisites, how to restore the databases and the SharePoint farm, and how to install but not configure Team Foundation Server (TFS) so that you can restore the databases on the new hardware.

In this topic

1. [Choosing hardware and naming the server](#)
2. [Installing SQL Server on the new server](#)
3. [Installing SharePoint Foundation on the new server](#)
4. [Install Team Foundation Server](#)

Required permissions

To perform the procedures in this topic, you must be a member of the Administrators security group on the server where you want to install the software.

▣ **Choosing hardware and naming the server**

Restoring a deployment that has unexpectedly gone offline due to a hardware failure is always stressful, and usually involves acquiring hardware as quickly as possible. You might be tempted to grab an unused machine and try to use it as the new server. However, for the long-term health of your deployment and to help ensure a successful restoration of your deployment, you should not use any server that has less RAM than the previous machine, and you should make sure that the new server is running the same operating system as the old server. You also must match the service pack level of the previous server as closely as possible.

In addition, this can be an opportune time to expand the hard drive space or processing power for your deployment, particularly if your previous server was not performing as quickly as you wanted, or if you were running low on hard drive space. Upgrading the numbers or speed of the processors, adding more RAM, or adding more hard drive space should not adversely affect your restoration, and you and your users can benefit from a faster, more powerful server.

▣ **Naming the server**

Your users will experience the least amount of confusion and inconvenience if you give the new server the exact same name as the old one. You will also find it much easier to restore the deployment if the new server has the same name as the old server. Unless you have compelling reasons not to reuse the name, such as underscores in the old server name that interfered with navigation in Team Web Access, use the same name for the new server as the old server.

▣ **The new server for the example deployment**

This example continues the tutorial that follows an IT Administrator at the fictitious company, Fabrikam, Inc., as she recovers from a hardware failure of the original deployment. When faced with the unexpected need for a new server,

she briefly considers using another machine that exactly matches the specifications of the previous server. However, she knows that the adoption of TFS is growing, and that several other departments are considering using her deployment. With that in mind, she negotiates funds to purchase a new server with a faster dual-core processor system, 16 GB of RAM, and two 500 GB hard drives. She gives the new server the same name as the old server, FabrikamPrime.

■ **Installing SQL Server on the new server**

After you acquire a server that meets the requirements mentioned above and give it the same name, you must install the same version and edition of SQL Server on that new hardware. You must also make sure to choose the same collation settings as your previous deployment, or you will not be able to restore the databases you backed up for the old deployment.

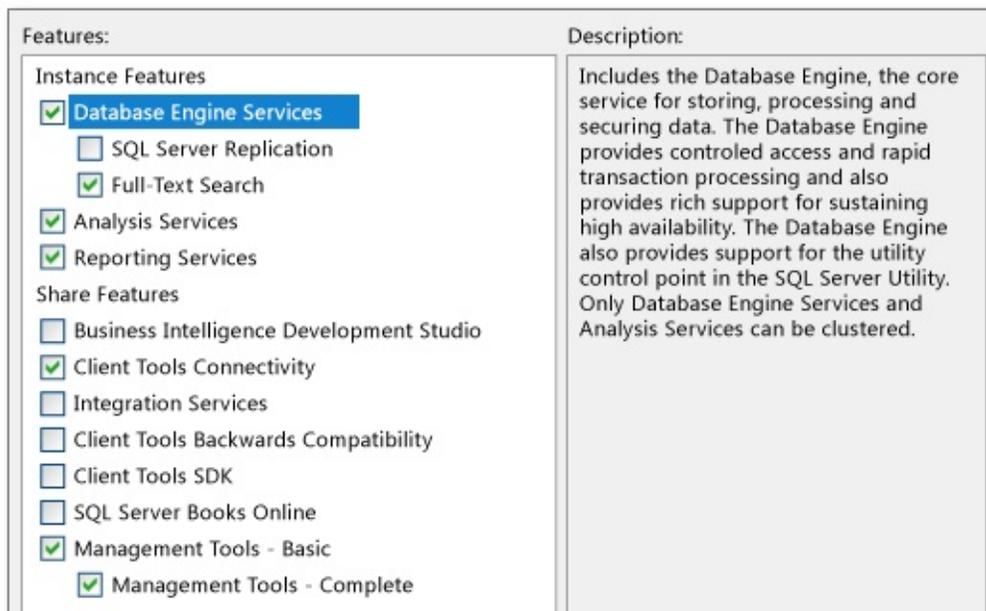
■ **Tip**

Most installations of SQL Server use the default collation settings. The default collation settings are determined by the Windows system locale on the server where you install SQL Server.

To install SQL Server to support Team Foundation Server

1. Launch the SQL Server Installation Center.
2. On the SQL Server Installation Center page, choose Installation, and then choose New installation or add features to an existing installation.
3. On the Setup Support Rules page, verify that all rules have passed, and then choose OK.
4. On the Product Key page, provide your product key, and then choose Next.
5. On the License Terms page, review the license agreement. If you accept the terms, select I accept the license terms. Optionally, you can select the check box to send usage data to Microsoft, and then choose Next.
6. On the Setup Support Files page, choose Install.
7. On the Setup Support Rules page, review the setup information. Correct any failure conditions, and then choose Next.

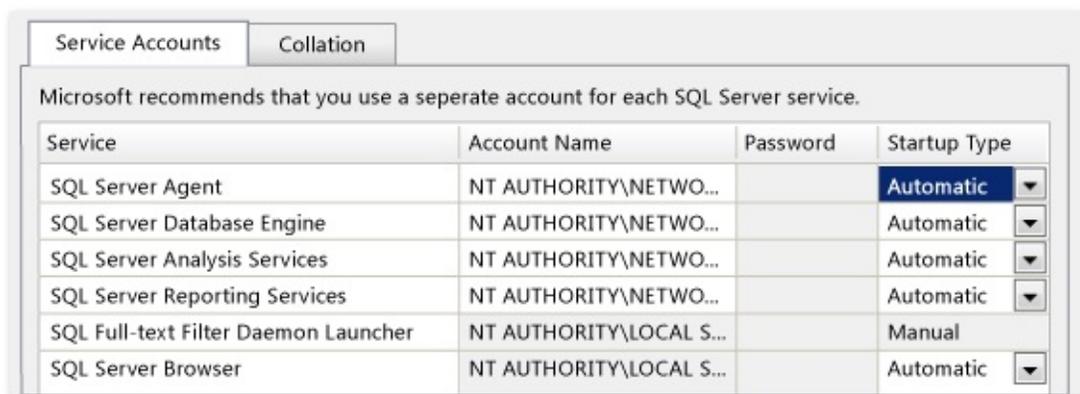
8. On the Setup Role page, choose SQL Server Feature Installation, and then choose Next.
9. On the Feature Selection page, select the following check boxes, and then choose Next:
 - Database Engine Services
 - Full-Text Search
 - Analysis Services, if reporting was part of the deployment you want to restore
 - Reporting Services, if reporting was part of the deployment you want to restore
 - Client Tools Connectivity
 - Management Tools - Basic
 - Management Tools - Complete



10. On the Installation Rules page, review any warnings and correct any failures, and then choose Next.

11. On the Instance Configuration page, choose Default instance, and then choose Next.
12. On the Disk Space Requirements page, review the information to make sure you have sufficient disk space, and then choose Next.
13. On the Server Configuration page, choose Use the same account for all SQL Server services. In the Use the same account for all SQL Server services window, choose or specify **NT AUTHORITY\NETWORK SERVICE**, and then choose OK.

In the Startup Type column, specify Automatic for all services that you can edit, and then choose Next.



14. On the Database Engine Services page, on the Account Provisioning tab, choose Windows authentication mode and then choose Add Current User to add your account as an administrator for this instance of SQL Server. Optionally add any other user accounts for users you want to act as database administrators, and then choose Next.
15. On the Analysis Services Configuration page, on the Account Provisioning tab, choose Add Current User to add your account as an administrator for the analysis services database. Optionally add any other user accounts for users you want to act as administrators, and then choose Next.
16. On the Reporting Services Configuration page, choose Install the native mode default configuration, and then choose Next.
17. On the Error Reporting page, choose whether to send information about

errors to Microsoft, and then choose Next.

18. On the Installation Rules page, review any failures or warnings, and then choose Next.
19. On the Ready to Install page, review the list of components to be installed, and if they match the list of features shown in the illustration below, then choose Install. If you need to make any changes, choose Back.



On the Installation Progress page, optionally monitor the installation progress of each component. When all components have finished installing, the Complete page appears. Review any messages, and then close the page.

20. After you finish the installation, visit Microsoft Update to apply the same service packs or updates that you applied to the installation on the old server. Do not apply any service packs or updates that were not applied to that server.

■ **Installing SharePoint Foundation on the new server**

Unlike a new installation of Team Foundation Server, you cannot use the installation wizard for TFS to install SharePoint Foundation for you. If you want to be able to restore the team project portals and other information used in the SharePoint Foundation portion of your deployment, you must first install SharePoint Foundation manually, and then restore the farm.

■ **Using SharePoint Tools to install SharePoint Foundation**

You can use the tools and features provided with SharePoint Foundation to install it on the new server. You should choose the same deployment configuration as your previous deployment. For example, if you installed SharePoint Foundation automatically during the installation process for Team Foundation Server in your original deployment, you should choose to install SharePoint Foundation on a single server, but using the installation of SQL Server you have already installed on the new server. You should also run the Microsoft SharePoint Products Preparation Tool prior to installing SharePoint Foundation.

To install SharePoint Foundation in a single-server farm configuration

1. Launch the installation media for SharePoint Foundation. On the SharePoint Foundation Start page, choose Install SharePoint Foundation.
2. On the Read the Microsoft Software License Terms page, review the terms, select the I accept the terms of this agreement check box, and then choose Continue.
3. On the Choose the installation you want page, choose Server farm.
4. On the Server Type tab, choose Complete.

If you want to install SharePoint Foundation in a custom location, choose the Data Location tab, and then either type the location or choose Browse to specify the location.

5. Choose Install Now.
6. When setup completes, choose Close. Do not configure the initial farm.

■ Using Windows PowerShell to install SharePoint Foundation

While there are a number of ways to install SharePoint Foundation, including the tools provided with SharePoint Foundation, restoring the farm requires Windows PowerShell. You can also use Windows PowerShell to install SharePoint Foundation. For convenience, consider using Windows PowerShell for both installing and restoring SharePoint Foundation on the new server. For more information about Windows PowerShell, see

[Getting Started: Windows Server Administration with Windows PowerShell.](#)

To install SharePoint Foundation using Windows PowerShell

1. Open a Windows PowerShell command prompt.
2. Input the following command, where Drive is the location of the installation media for SharePoint Foundation:

Copy Code

```
Install-SharePoint -SetupExePath "Drive:\SharePoint 2013\Setup\:
```

This installs SharePoint Foundation using a PID key in a farm deployment, but does not configure it or create any databases. Instead, you will restore the farm and its databases to this installation.

✔ Tip

As an alternative, you can choose to use a configuration XML file with the Install-SharePoint command to install SharePoint Foundation.

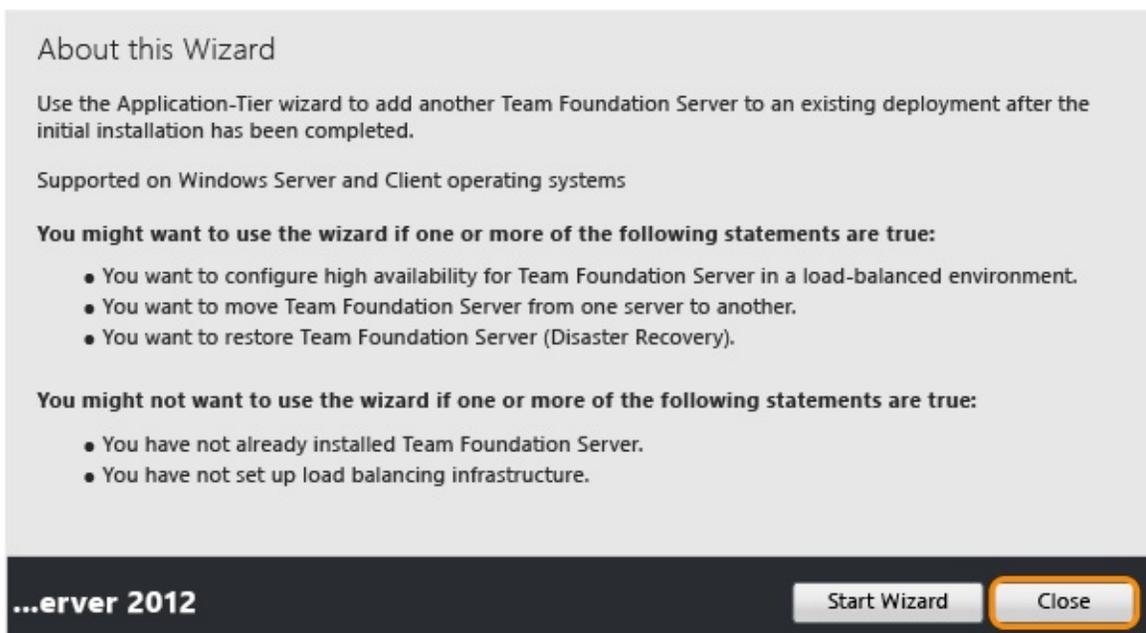
For more information, see [Install SharePoint Foundation by using Windows PowerShell](#).

■ Install Team Foundation Server

To restore the databases using the Scheduled Backups features, you must install, but not configure, the Team Foundation Server software.

To install Team Foundation Server binaries and tools

1. Launch the installation media for Team Foundation Server. On the Team Foundation Server Setup page, choose Install.
2. When the installation completes, the Team Foundation Server Configuration Center opens. Choose Cancel.



The administration console opens automatically in an unconfigured state. This is expected.

Home |

[Prepare the new hardware](#) | [Restore the databases](#) | [Install and configure Team Foundation Server](#) | [Reconnect services and users](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Restore the databases

[Send Feedback](#)

This topic, the second part of the Restore a Single-Server tutorial, teaches you how to use the backups you made of the original server to restore the databases for Team Foundation Server (TFS) and the SharePoint farm on the new server.

You can use the backups you made of the original deployment to restore the data from that deployment to the new server. When restoring the data, be sure to restore all databases and the SharePoint Farm to the same point in time. If you followed the guidance in [Configure a backup schedule and plan](#), you used the Scheduled Backups feature to create your backups. You will use these backups to restore your data.

If you manually configured your backups, you cannot use the Restore wizard in Scheduled Backups to restore those databases. You must manually restore them using the software you used to back them up.

In this topic

- [Restore Team Foundation Server databases](#)
- [Restore the SharePoint farm](#)

Required permissions

To perform these procedures, you must be a member of the following groups or have the following permissions:

- A member of the Administrators security group on the server.
- Either a member of the SQL Server System Administrator security group, or your SQL Server Perform Back Up and Create Maintenance Plan

permission must be set to Allow.

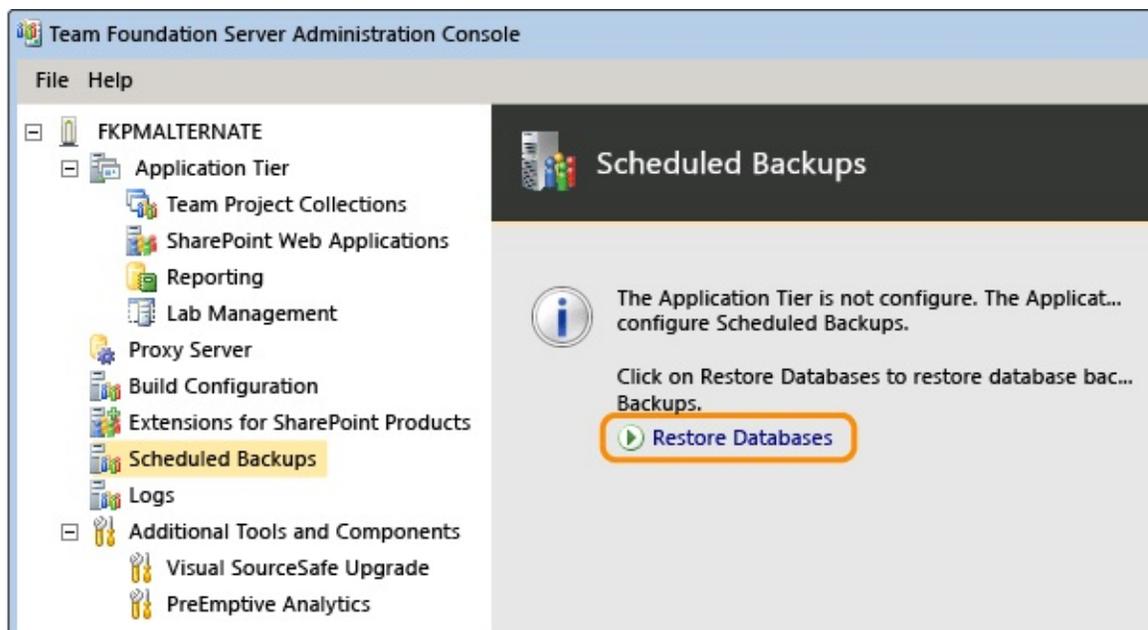
- A member of the sysadmin security group for the databases for Team Foundation.
- A member of the Farm Administrators group.

Restore Team Foundation Server databases

Installing and configuring software isn't sufficient to recover a deployment. You must restore the data before your users will be able to get back to work. There's a wizard to help you do this..

To restore databases

1. To start the Restore wizard, open the administration console for TFS and navigate to Scheduled Backups.



2. Specify the path to the backup set and choose the set you want to use for restoration.

- TFS_Warehouse
- TFS_DefaultCollection

This is the default name of the collection database. If you customized the name, make sure to use that name.

- TFS_Configuration
- ReportServer
- ReportServerTempDB
- WSS_Config
- WSS_AdminContent
- WSS_Content

The Restore wizard also restores the encryption key for SQL Server Reporting Services as part of its operation. You can choose to restore that key manually, but it should not be necessary.

Restore the SharePoint farm

Although the Restore wizard restored the SharePoint databases used by your deployment, it cannot restore the farm. You must use Windows PowerShell with the `Restore-SPFarm` command to restore the backup you made of the SharePoint Farm. In some cases, you can choose to use the Central Administration website to restore the farm instead of `Restore-SPFarm`, but the PowerShell command is the preferred method for restoring a farm.

To restore a farm, either you must be logged in with an account that is a member of the Farm Administrators group, or you must provide the credentials for an account that is a member of that group when prompted to do so.

To restore the farm for SharePoint Foundation using `Restore-SPFarm`

1. Open Windows PowerShell or SharePoint Management Shell.
2. At the Windows PowerShell command prompt, enter the following command, where *UNCPath* is the fully-qualified UNC path of the directory where the farm backup is located:

```
Restore-SPFarm -Directory UNCPath -RestoreMethod Overwrite
```

This command will restore the farm using the most recent backup available. If you want to use a different backup, you must specify which backup to restore by using the `-BackupID` parameter with the GUID of the specific backup you want to use.

3. At the Windows PowerShell command prompt, enter the following command, where *ServiceApplicationID* is the GUID of the restored farm:

```
Start-SPServiceInstance -Identity ServiceApplicationID
```

Tip

If you do not know the GUID, you can use the `Get-SPServiceInstance`

command to list the service instance GUIDs for all SharePoint applications on the server.

4. For more information about restoring a farm, see

[Restore a farm \(SharePoint Foundation\)](#) and [Restore-SPFarm](#).

To restore the farm using SharePoint Central Administration

1. Open SharePoint Central Administration, and on the Home page, in the Backup and Restore section, choose Restore from a backup.

The Restore Wizard opens.

2. On the Restore from Backup - Step 1 of 3 page, choose the farm backup job that you want to restore, and then choose Next.
3. On the Restore from Backup - Step 2 of 3 page, select the check box next to the farm option, and then choose Next.
4. On the Restore from Backup - Step 3 of 3 page, in the Restore Component section, make sure that Farm appears in the Restore the following component list. In the Restore Only Configuration Settings section, choose Restore content and configuration settings. In the Restore Options section, under Type of Restore, choose Same configuration. When a dialog box appears asking you to confirm your choices, choose OK, and then choose Start Restore.
5. Monitor the general status of the recovery as it appears in the Readiness section of the Backup and Restore Job Status page. The status automatically updates every 30 seconds. You can also choose to manually update the status by choosing Refresh.
6. When restoration is complete, return to the Home page in Central Administration. In Application Management, choose Manage services on server.
7. On the Services on Server page, find the service applications for the restored farm, and in the Actions column, choose Start for each of those

service applications.

8. For more information about restoring a farm, see [Restore a farm \(SharePoint Foundation\)](#).

[Home](#) | [Prepare the new hardware](#) | [Restore the databases](#) | [Install and configure Team Foundation Server](#) | [Reconnect services and users](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Install and configure Team Foundation Server

[Send Feedback](#)

This topic, the third part of the Restore a Single-Server tutorial, teaches you how to install Team Foundation Server (TFS) using the Application-Tier Only wizard. You will also learn how to install the Extensions for SharePoint Products on the new server. Additionally, you will learn how to configure SharePoint Foundation, reporting, and analysis services to work with the new server.

In this topic

1. [Install Team Foundation Server and the Extensions for SharePoint Products](#)
2. [Configure SharePoint Foundation](#)
3. [Configure reporting and Analysis Services](#)

Required permissions

You must be a member of the following groups or have the following permissions:

- A member of the Administrators security group on the server.
- A member of the Team Foundation Administrators group.
- Either a member of the SQL Server System Administrator security group, or your SQL Server Perform Back Up and Create Maintenance Plan permission must be set to Allow.
- A member of the sysadmin security group for the databases for Team Foundation.

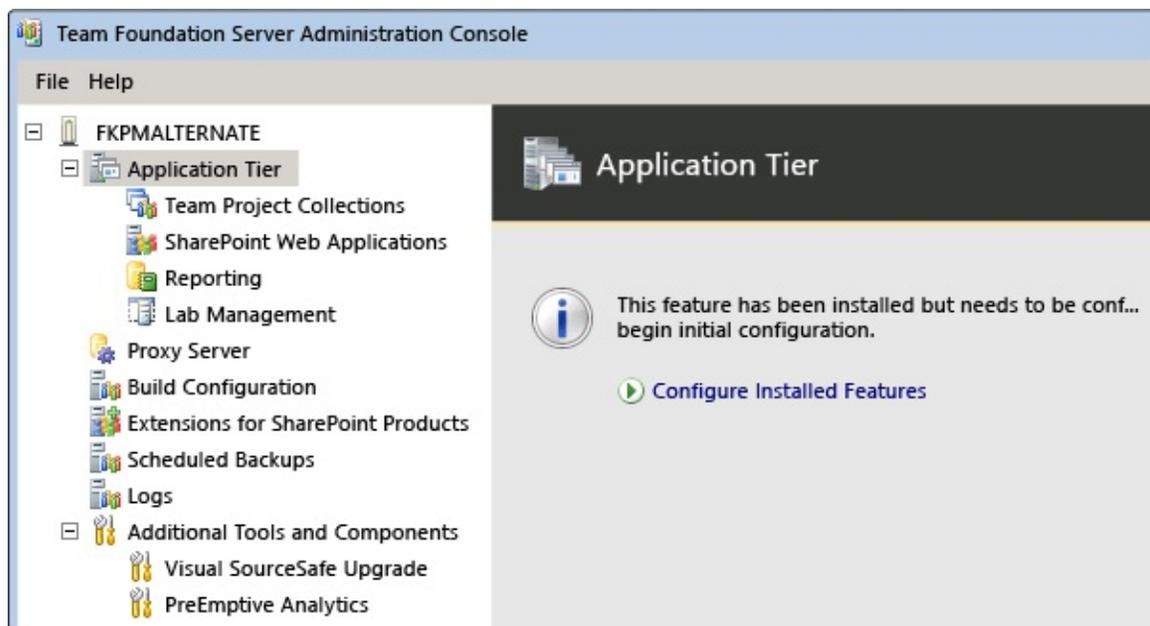
- A member of the Farm Administrators group.

Install Team Foundation Server and the Extensions for SharePoint Products

By restoring the databases that TFS uses, you have already restored the data tier of the original deployment. Now you must complete the restoration process by installing the application tier—the application and services that compose Team Foundation Server—by using the Application-Tier Only wizard. You must also install the extensions used by TFS and SharePoint Foundation by running the Extensions for SharePoint Products wizard, if your original deployment was configured with SharePoint Foundation.

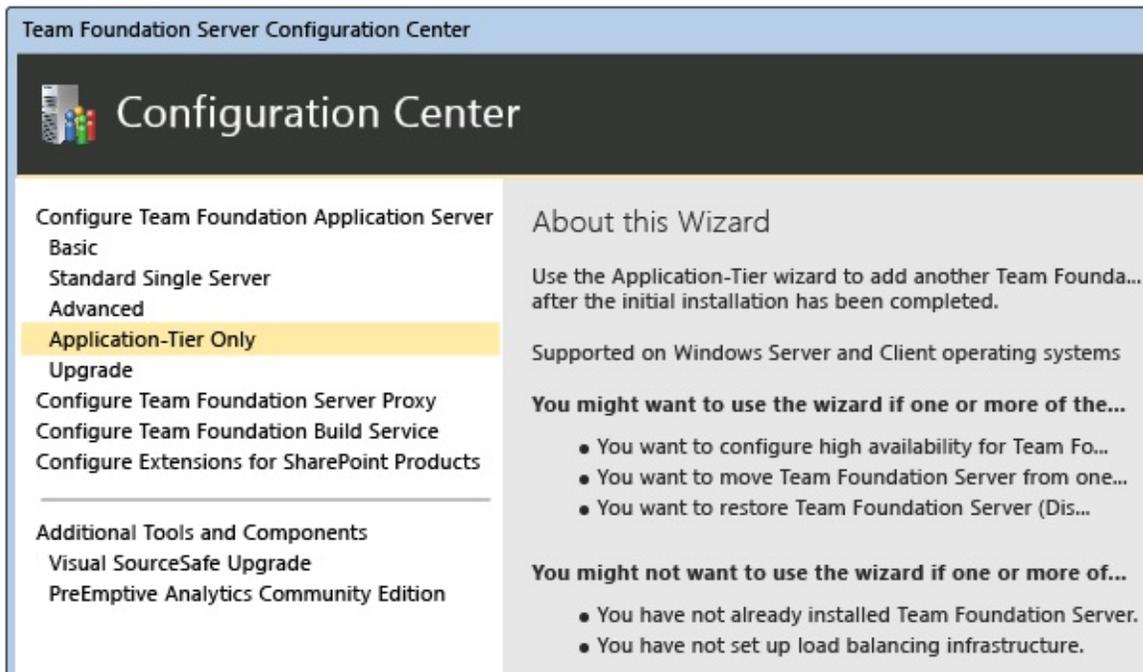
To install the application tier and the extensions

1. Open the administration console and start the configuration process.

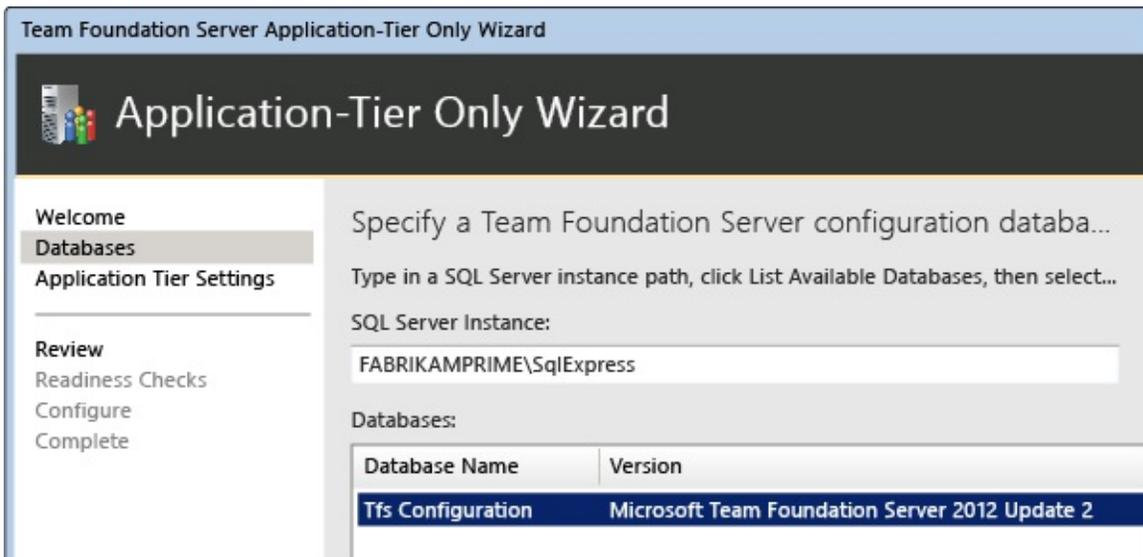


If you don't see the administration console for TFS, you haven't installed the TFS software on the server. To learn how, go to [Install Team Foundation Server](#).

2. Launch the application-tier only wizard.



3. Specify the name of the SQL Server where you just restored the databases and choose List Available Databases to populate the list. Choose the configuration database.



4. Provide the same service account information as used in the prior deployment, if possible.

...ion-Tier Only Wizard

...Only Wizard

Provide the settings for your new Application-Tier

Provide the service account to use for your new Application-Tier.

Server Url:

Service Account: 

Password: [Test](#)

Authentication Method

NTLM

Negotiate (Kerberos)

Negotiate (Kerberos) is the more secure configuration to use with Integrated Windows authentication, but may require additional configuration by the domain administrator. [More information.](#)

5. Complete the wizard.

Team Foundation Server Application Tier Only Wizard

Application-Tier Only Wizard

Welcome
Databases
Application Tier Settings

Review
Readiness Checks
Configure
Complete

Configuration Progress



-  Preparing Configuration
-  Configuration Database
-  Creating Web Sites
-  Configuring Services

 **Success**

Configuration completed successfully. Click Next to view deta...

6. In the Configuration Center, choose Configure Extensions for SharePoint Products and complete that wizard to install the extensions.

7. Open the administration console, and in the Application Tier pane, update or refresh the URLs to reflect the new URLs for the restored server. Even if

you have not changed the name or ports used for the restored server, you should reapply this information.

The screenshot displays the 'Application Tier' configuration page. The 'Application Tier Summary' section lists the following details:

Service Account:	NT AUTHORITY\NETWORK SERVICE
Web Site:	Team Foundation Server
Application Pool:	Microsoft Team Foundation Server Application Pool
Authentication:	NTLM
Notification URL:	http://fabrikamprime:8080/tfs
Server URL:	http://localhost:8080/tfs
Web Access URL:	http://fabrikamprime:8080/tfs
Machine Name:	FABRIKAMPRIME
Ports:	8080
Virtual Directory:	/tfs

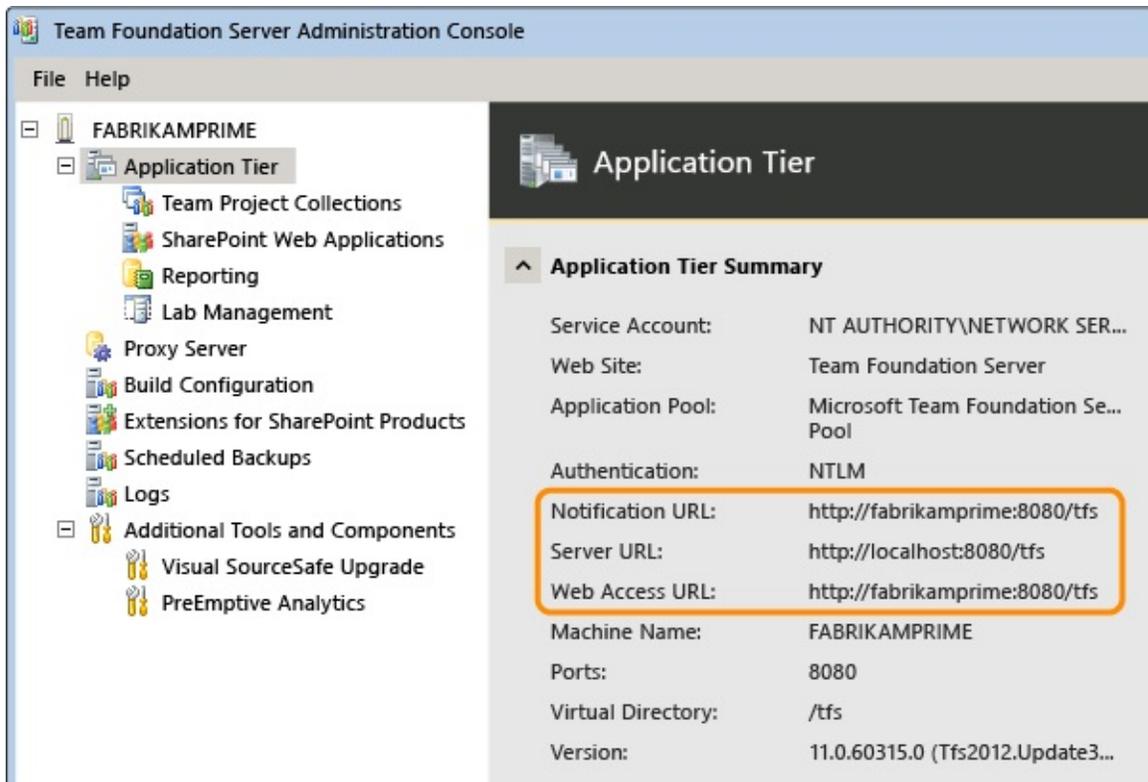
On the right side of the summary, there are several action links: Update Password, Change Account, Reapply Account, Authentication Settings, Group Membership, Administer Security, and Change URLs. The 'Change URLs' link is highlighted with an orange border.

The 'Change URLs' dialog box is open, showing the following configuration options:

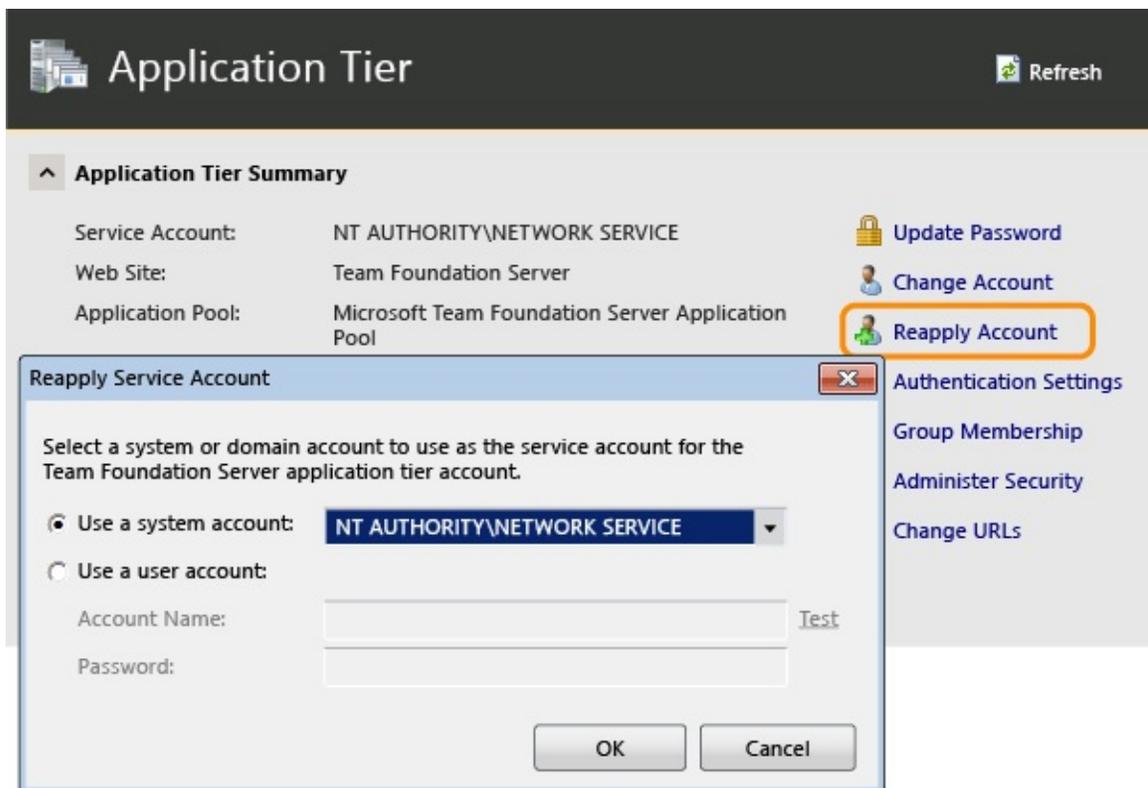
- Notification URL:** The primary URL that users use to connect to the system and is the URL used in the text of e-mail alerts. The current value is `http://FABRIKAMPRIME:8080/tfs`. A green checkmark and 'Test' button are visible next to the input field.
- Server URL:** Used for server-to-server communication. Under most configurations, the Server URL should be set to the localhost. More information about the Server URL can be found at <http://go.microsoft.com/fwlink/?LinkID=155633>.
- Use localhost:** Selected by default.
- Use:** The current value is `http://localhost:8080/tfs`. A 'Test' button is visible next to the input field.

At the bottom of the dialog box, there are 'OK' and 'Cancel' buttons.

8. After finishing the update, verify that the URLs appear correctly.



9. Reapply the service account for TFS.



■ **Configure SharePoint Products**

After you install the application tier and the extensions, you must reconfigure the settings for the SharePoint web applications that your deployment uses.

To configure the settings for SharePoint web applications

1. Open the administration console for Team Foundation.
2. In the navigation bar, choose SharePoint Web Applications.
3. In the SharePoint Web Applications list, highlight the web application used by the original deployment, and then choose Change.

Tip

By default, the web application should appear automatically in the list of web applications. If it does not, choose Add, and then add it manually.

4. The SharePoint Web Application Settings window opens. In Web Application URL and Central Administration URL, review the settings, and if necessary, specify or change the values to reflect the URLs on the new server.
5. When you are satisfied with the settings, choose OK.

For more information about how to configure SharePoint web applications as part of your deployment, see

[Add SharePoint products to your deployment.](#)

■ **Configure reporting and Analysis Services**

After you have configured SharePoint Foundation, you must redirect Team Foundation Server to the location of the report server, restart the warehouse, and manually rebuild the database for Analysis Services, if your deployment used reporting.

■ **Note**

You must complete this procedure even if you restored the TFS_Warehouse and TFS_Analysis databases, as the previous section describes.

To reconfigure reporting and Analysis Services

1. In the navigation bar for the administration console, choose Reporting.
2. On the Reporting page, choose Edit.
3. In the Take Offline dialog box, choose OK.

The Reporting dialog box opens.

4. Select the Use Reporting check box.
5. On the Warehouse tab, in Server, specify the name of the report server, and in Database, type the name of the warehouse database. Optionally choose Test Connection to verify that the database is valid.
6. On the Analysis Services tab, in the Server list, specify the name of the server that is running SQL Server Analysis Services. In Database, specify the name of the analysis services database, and in Account for accessing data sources, specify the username and password of the data sources account.

7. On the Reports tab, in the Server list, specify the name of the report server, and then choose Populate URLs. In Username and Password, specify the account name and password (if any) for the data sources account. In Default Path, specify the relative path for the location where reports are stored, and then choose OK.
8. In the administration console, choose Start Jobs to restart reporting.
9. Open a Command Prompt window, and then change directories to %ProgramFiles%\Microsoft Team Foundation Server 12.0\Tools.
10. Enter the following command to rebuild the database for Analysis Services:

TFSSConfig RebuildWarehouse /AnalysisServices
/ReportingDataSourcePassword:Password

Password is the password for the data sources account for Reporting Services.
11. Wait until the command is successfully completed.

Home |

[Prepare the new hardware](#) | [Restore the databases](#) | [Install and configure Team Foundation Server](#) | [Reconnect services and users](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Reconnect services and users

[Send Feedback](#)

This topic, the fourth part of the Restore a Single-Server tutorial, teaches you how to start team project collections and verify user groups on the new server. You will also learn how to clear the data caches on the new server, and how to help ensure that your users do not experience data caching problems when they connect to the new server.

In this topic

1. [Start team project collections and verify permissions](#)
2. [Refresh the version control cache on client computers and reconnect to Team Foundation Server](#)

Required permissions

To invoke the StampWorkitemCache web method, you must be a member of the Administrators security group on the server.

To use the **tf workspaces** command on the client computer, your **Read** permission must be set to **Allow**.

Start team project collections and verify permissions

After you have completed restoring the data and software on the new server, start the team project collections, and then make sure that all users, groups, and service accounts for your deployment are configured with the permissions that they require to perform tasks and function correctly.

You cannot automatically migrate all permissions. For example, administrators for Team Foundation must be members of the local Administrators group on the server so that they can open the administration console, and you must manually add them to that group.

To start a team project collection

1. Open the administration console, and in Team Project Collections, start the default collection.

The screenshot displays the 'Team Project Collections' administration console. At the top, there is a header with the title 'Team Project Collections' and buttons for 'Refresh' and 'Help'. Below the header is a table with two columns: 'Name' and 'State'. The table contains two rows: 'DefaultCollection' and 'Fabrikam Older Projects', both with a state of 'Offline'. To the right of the table are two buttons: 'Create Collection' and 'Attach Collection'. Below the table are three tabs: 'General', 'Status', and 'Team Projects'. The 'General' tab is selected, showing details for 'DefaultCollection'. The details include a URL: 'http://fabrikamprime:8080/tfs/DefaultCollection/' and a SQL Server Instance: 'FABRIKAMPRIME\SqIExpress'. To the right of the details are three buttons: 'Start Collection', 'Edit Settings', and 'Detach Collection'. The 'Start Collection' button is highlighted with an orange border.

2. Repeat this action for any other team project collections in the deployment.

To verify permissions

- Log on to the server and make sure that users, groups, and service accounts are configured with the permissions that they need to operate.

For more information, see

[Add users to team projects](#), [Set administrator permissions for team project collections](#), [Set administrator permissions for Team Foundation Server](#), and [Service accounts and dependencies in Team Foundation Server](#).

Refresh the work item cache

Note

This procedure is optional. You should perform it only if you receive errors with work item tracking.

To update the cache for tracking work items, you invoke the StampWorkitemCache web method. This method forces client computers to update the cache the next time that they connect to the application-tier server. This method also synchronizes the workspaces that are defined on the client computers.

To refresh the cache for tracking work items on client computers

1. On the new server, open Internet Explorer.
2. In the Address bar, enter the following address to connect to the ClientService web service:

<http://PublicURL:8080/VirtualDirectory/WorkItemTracking/v3.0/ClientS>

Note

Even if you are logged on with administrative credentials, you might need to start Internet Explorer as an administrator, and you might be prompted for your credentials.

3. Choose StampWorkitemCache, and then choose Invoke.

Note

The StampWorkitemCache method returns no data.

Refresh the version control cache on client computers and reconnect to Team Foundation Server

To prevent workspace errors from occurring during version control or build operations in Team Foundation, you have to update the data cache on client computers.

To refresh the version control cache, each user must run the **tf workspaces** command on any computer that uses version control and that connects to the restored server.

To refresh the version control cache on client computers

1. On the client computer, open a Command Prompt window with administrative permissions, and change directories to Drive:\Program Files (x86)\Microsoft Visual Studio 12.0\Common7\IDE.
2. At the command prompt, enter the following command, including the URL of the collection, which includes the server name and the port number of the new server:

tf workspaces

/collection:http://ServerName:Port/VirtualDirectoryName/CollectionName

For example, to refresh the version control cache for a project that is a member of the DefaultCollection collection, which is hosted in the FabrikamPrime deployment of Team Foundation Server, type the following string:

```
tf workspaces /collection:http://FabrikamPrime:8080/tfs/DefaultCollection
```

For more information, see Workspaces Command.

[Prepare the new hardware](#) | [Restore the databases](#) | [Install and configure Team Foundation Server](#) | [Reconnect services and users](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Restore data to the same location

[See Also](#) [Send Feedback](#)

You can restore data from a backup to the same server and instance of SQL Server for Team Foundation from which that data was backed up. For example, you might want to restore a corrupted set of databases to the last known good state.

Tip

If your original deployment used the Enterprise or Datacenter editions of SQL Server, and you want to restore databases to a server running Standard edition, you must use a backup set that was made with SQL Server compression disabled. Unless you disable data compression, you will not be able to successfully restore Enterprise or Datacenter edition databases to a server running Standard edition. To turn off compression, follow the steps in the [Microsoft Knowledge Base article](#).

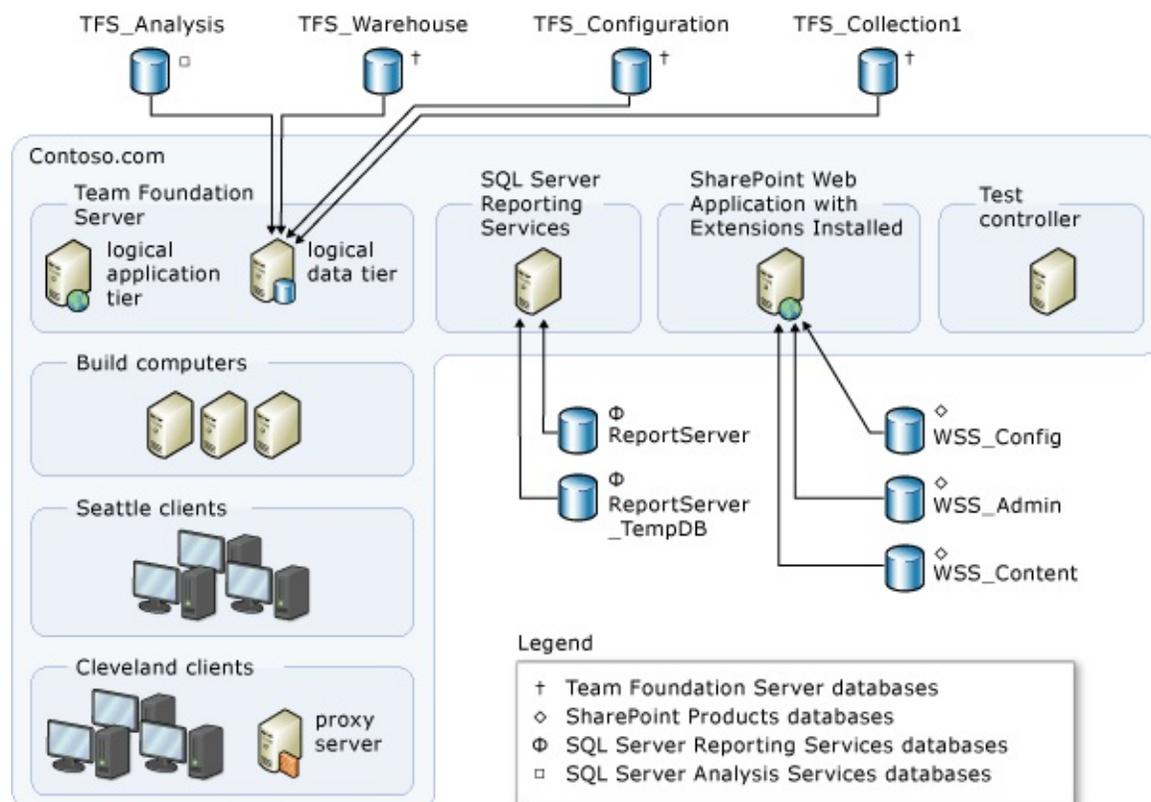
If you want to restore data to another server or another instance of SQL Server, see [Restore a deployment to new hardware](#).

Note

If you use SharePoint Products in your deployment, when you restore data, you do not have to restore the websites that are automatically generated based on the data for each team project. The data for the team project portals is contained in the databases that you restore.

The steps to restore data to the same server or servers vary based on how Team Foundation Server is installed and configured. For simplicity, the procedures in

this topic are structured for a moderately complex deployment of Team Foundation Server, as the following illustration shows:



If your topology does not exactly match this example, you might have to adjust the steps in this procedure to follow it successfully. For example, if you have a deployment where all components are installed on a single physical server, you would perform all procedures on that server. If databases for team project collections are deployed on more than one server, you must perform the steps to restore each collection database on the appropriate server. For more information about which components might be deployed on each server, see the following topics:

- [Understand backing up Team Foundation Server](#)
- [Team Foundation Server architecture](#)
- A Simple Topology for Team Foundation Server
- A Moderate Topology for Team Foundation Server

- A Complex Topology for Team Foundation Server

In this topic

1. [Required Permissions](#)
2. [Stop services that TFS uses](#)
3. [Rename Databases You Want To Restore](#)
4. [Restore Team Foundation Databases](#)
5. [Update All Service Accounts](#)
6. [Restore the Warehouse](#)
7. [Clear the Data Cache On Servers](#)
8. [Restart Services that Team Foundation Server Uses](#)
9. [Refresh the Data Cache on Client Computers](#)

Required Permissions

To perform this procedure, you must be a member of the following groups or have the following permissions:

- A member of the Administrators security group on the server or servers that are running the administration console for Team Foundation.
- Either a member of the SQL Server System Administrator security group or your SQL Server Perform Back Up and Create Maintenance Plan permission must be set to Allow on the instance of SQL Server that hosts the databases.
- A member of the **sysadmin** security group for the database instance for Team Foundation and for the Analysis Services instance of the warehouse database.
- An authorized user of the TFS_Warehouse database.

- A member of the TFSEXCROLE database role.
- If the deployment uses SharePoint Products, a member of the Farm Administrators group for the farm to which the SharePoint Products databases are being restored.

For more information, see the following page on the Microsoft website: [User Account Control](#).

■ Stop services that TFS uses

Stopping the services helps protect against data loss or corruption during the restoration process, particularly if you rename databases.

1. On the server that is running the application-tier services for Team Foundation, open a Command Prompt window, and change directories to Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools.
2. Type the following command:

 **Copy Code**

```
TFSServiceControl quiesce
```

For more information, see

[TFSServiceControl Command](#).

■ Rename Databases that You Want to Restore

Before you can use the Restore wizard to restore a database that Team Foundation Server, you must first take it offline, and then rename it.

To stop databases that Team Foundation Server uses

1. Open SQL Server Management Studio.

■ Note

For more information about how to restore databases, see the following page on the Microsoft website:

[Implementing Restore Scenarios for SQL Server Databases.](#)

The Connect to Server dialog box opens.

- In Server type, choose Database Engine.
- In Server name, choose or type the name of the data-tier server and database instance, and then choose Connect.

■ Note

If SQL Server is installed on a cluster, the server name is the name of the cluster and not the computer name.

SQL Server Management Studio opens.

- Expand the Databases node to show the list of databases that make up the data tier for Team Foundation.

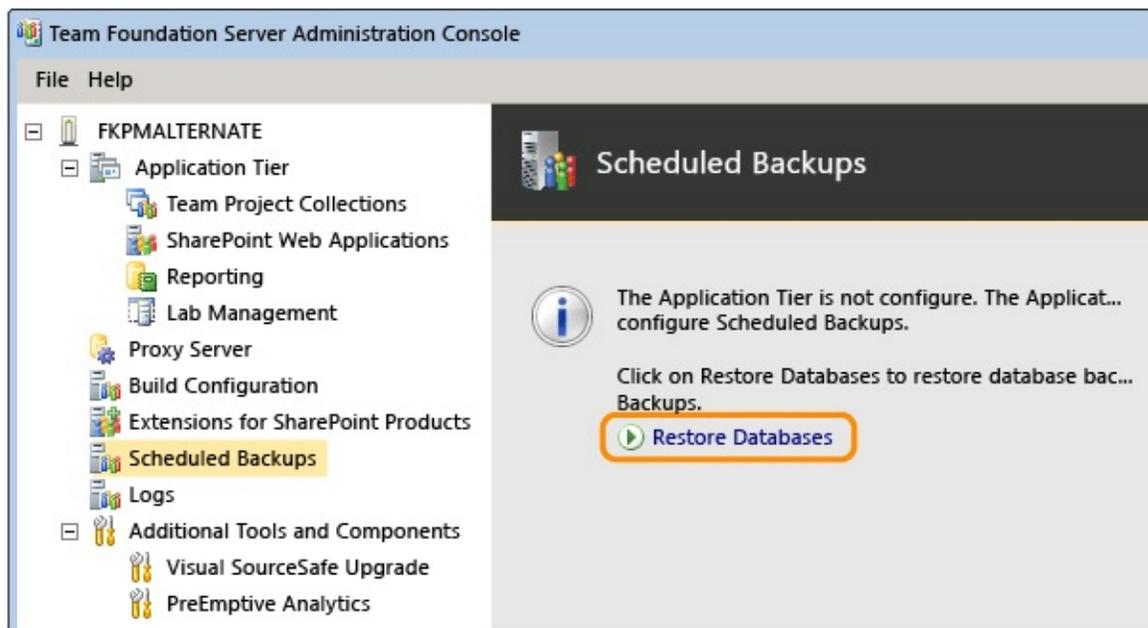
- Rename and then stop each database you want to restore, following [the guidance for your version of SQL Server](#). Give the database a name that indicates it is the old version of the database you will replace with the restored version. For example, you might rename TFS_DefaultCollection to TFS_DefaultCollection_Old.

Restore Team Foundation Databases

You can restore data for Team Foundation Server by using the Restore wizard in the administration console in TFS. The Restore wizard also restores the encryption key used for reporting.

To restore databases

1. Open the administration console for TFS and navigate to Scheduled Backups to start the Restore wizard.



2. Specify the path to the backup set and choose the set you want to use for restoration.

Update All Service Accounts

You must update the service account for Team Foundation Server (TFSService) and the data sources account (TFSReports). Even if these accounts have not changed, you must update the information to ensure that the identity and the format of the accounts are appropriate.

To update service accounts

1. On the server that is running SQL Server Reporting Services, open Computer Management, and start the following components if they are not already started:
 - ReportServer or ReportServer\$InstanceName (application pool)
 - SQL Server Reporting Services (TFSINSTANCE)
2. On the application-tier server, open a Command Prompt window, and change directories to Drive:\%programfiles%\ Microsoft Team Foundation Server 12.0\Tools.
3. At the command prompt, enter the following command to add the service account for Team Foundation, where DatabaseName is the name of the configuration database (by default, TFS_Configuration):

```
TfsConfig Accounts /add /AccountType:ApplicationTier  
/account:AccountName
```

For more information about how to use this command, see

[Accounts Command](#).

- Use the **Accounts** command to add the data sources account for the report server and the proxy account for Team Foundation Server Proxy, if your deployment uses these resources.

Rebuild the Warehouse

You can rebuild the data warehouse instead of restoring the TFS_Warehouse and TFS_Analysis databases. You will require a significant amount of time to rebuild the warehouse if your deployment contains a lot of data. However, that strategy helps ensure that all data is properly synchronized. When you rebuild the warehouse, Team Foundation Server creates an instance of it, which you must then process to populate it by using data from the operational stores.

Note

If you restored the TFS_Warehouse and TFS_Analysis databases in the previous section, you do not have to perform the following procedure.

To rebuild the warehouse

1. On the server that is running the application-tier services for Team Foundation, open a Command Prompt window, and change directories to Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools.
2. Enter the following command:

```
TFSSConfig rebuildwarehouse /all  
/ReportingDataSourcePassword:Password
```

where Password is the password for the data sources account for Reporting Services (TFSReports).

3. Wait until the command is successfully completed.

Note

If the command is not completed successfully, you should verify that you have all required permissions. For more information, see [Troubleshooting the Data Warehouse](#).

4. On the report server, open Internet Explorer and enter the following string in the Address bar:

`http://localhost:8080/VirtualDirectory/TeamFoundation/Administration/v3.0`

For VirtualDirectory, enter the virtual directory for Internet Information Services (IIS) that was specified when Team Foundation Server was installed. By default, this directory is named tfs.

The WarehouseControlWebService page opens.

 **Note**

The Microsoft Team Foundation Server Application Pool must be running for the Warehouse Control Web service to be available.

5. Choose GetProcessingStatus, and then choose Invoke.

 **Important**

The service should return a value of Idle for all jobs, which indicates that the cube is not being processed. If a different value is returned, repeat this step until Idle is returned for all jobs.

6. On the WarehouseControlWebService page, choose ProcessAnalysisDatabase, and then choose Invoke.

A browser window opens. The service returns True when it successfully starts to process the cube and False if it is not successful or if the cube is currently being processed.

7. To determine when the cube has been processed, return to the WarehouseControlWebService page, choose GetProcessingStatus, and then choose Invoke.

Processing has completed when the GetProcessingStatus service returns a value of Idle for all jobs.

8. On the application-tier server for Team Foundation, open **Computer Management**, and start the Visual Studio Team Foundation Background Job Service.

■ Clear the Data Cache on Servers

Each application-tier server in your deployment of Team Foundation uses a file cache so that users can quickly download files from the data-tier server. When you restore a deployment, you should clear this cache on each application-tier server. Otherwise, mismatched file IDs might cause problems when users download files from version control. If your deployment uses Team Foundation Server Proxy, you must also clear the data cache on each server that is configured as a proxy.

■ Note

By clearing the data caches, you can help prevent the download of incorrect versions of files in version control. You should routinely do this unless you are replacing all hardware in your deployment as part of your restoration. If you are replacing all hardware, you can skip this procedure.

To clear the data cache

1. On a server that is running the application-tier services for Team Foundation or that is configured with Team Foundation Server Proxy, open a Command Prompt window and change directories to `Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Application Tier\Web Services_tfs_data`.
2. Delete everything in the `_tfs_data` directory.
3. Repeat these steps for each application-tier server and each server that is running Team Foundation Server Proxy in your deployment.

Restart Services that Team Foundation Server Uses

After you restore the data, you must restart the services to return the server to an operational state.

To restart services that Team Foundation Server uses

1. On the server that is running the application-tier services for Team Foundation, open a Command Prompt window, and change directories to Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools.
2. Enter the following command:

```
TFSServiceControl unquiesce
```

For more information, see

[TFSServiceControl Command](#).

Refresh the Caches on Client Computers

To refresh the cache for tracking work items on client computers

1. On the new server, open Internet Explorer.
2. In the Address bar, enter the following address to connect to the ClientService web service:

`http://PublicURL/VirtualDirectory:8080/WorkItemTracking/v3.0/ClientS`

Note

Even if you are logged on with administrative credentials, you might need to start Internet Explorer as an administrator, and you might be prompted for your credentials.

3. Choose StampWorkitemCache, and then choose Invoke.

Note

The StampWorkitemCache method returns no data.

To refresh the version control cache on client computers

1. On the client computer, open a Command Prompt window with administrative permissions, and change directories to Drive:\Program Files (x86)\Microsoft Visual Studio 12.0\Common7\IDE.
2. At the command prompt, enter the following command, including the URL of the collection, which includes the server name and the port number of the new server:

tf workspaces

/collection:http://ServerName:Port/VirtualDirectoryName/CollectionName

In the example deployment, a developer needs to refresh the version control cache for a project that is a member of the DefaultCollection collection, which is hosted in the FabrikamPrime deployment of Team Foundation Server. He types the following string:

```
tf workspaces /collection:http://FabrikamPrime:8080/tfs/DefaultCollection
```

For more information, see [Workspaces Command](#).

See Also

Concepts

[Permission reference for Team Foundation Server](#)

[Team Foundation Server architecture](#)

[Service accounts and dependencies in Team Foundation Server](#)

[Restore the databases](#)

[Restore Lab Management components](#)

Other Resources

[Restore a deployment to new hardware](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Refresh the data caches on client computers

[See Also](#) [Send Feedback](#)

To prevent workspace errors from occurring during version control or build operations in Team Foundation, the data cache on client computers must be updated after certain maintenance operations. After you move, restore, rename, or fail over a data-tier or application-tier server or after you recover from a failure such as a hardware malfunction, you must refresh the cache for tracking work items and users must refresh the version control cache on client computers.

Required permissions

To invoke the StampWorkitemCache Web method, you must be a member of the Administrators security group on the application-tier server for Team Foundation. For more information, see [Permission reference for Team Foundation Server](#).

To use the **tf workspaces** command on the client computer, your **Read** permission must be set to **Allow**.

Refresh the Work Item Cache

Note

This procedure is optional. You should perform it only if errors occur with work item tracking.

To update the cache for tracking work items, you invoke the StampWorkitemCache Web method. This method forces client computers to update the cache the next time that they connect to the application-tier server. This method also synchronizes the workspaces that are defined on the client computers.

Note

When you invoke the StampWorkitemCache Web method, the performance of Visual Studio Team Foundation Server might temporarily degrade. The performance impact depends on how many Team Foundation users are connected when you invoke the method.

To refresh the cache for tracking work items on client computers

1. On the new server, open Internet Explorer.
2. In the Address bar, enter the following address to connect to the ClientService web service:

<http://PublicURL/VirtualDirectory:8080/WorkItemTracking/v3.0/ClientS>

Note

Even if you are logged on with administrative credentials, you might need to start Internet Explorer as an administrator, and you might be prompted for your credentials.

3. Choose StampWorkitemCache, and then choose Invoke.

Note

The StampWorkitemCache method returns no data.

Refresh the Version Control Cache

To refresh the version control cache, each user runs the **tf workspaces** command on any computer that must be updated. They must update any computer that uses version control and that connects to a team project collection whose databases were relocated.

To refresh the version control cache on client computers

1. On the client computer, open a Command Prompt window with administrative permissions, and change directories to Drive:\Program Files (x86)\Microsoft Visual Studio 12.0\Common7\IDE.
2. At the command prompt, enter the following command, including the URL of the collection, which includes the server name and the port number of the new server:

tf workspaces

/collection:http://ServerName:Port/VirtualDirectoryName/CollectionName

In the example deployment, a developer needs to refresh the version control cache for a project that is a member of the DefaultCollection collection, which is hosted in the FabrikamPrime deployment of Team Foundation Server. He types the following string:

```
tf workspaces /collection:http://FabrikamPrime:8080/tfs/DefaultCollection
```

For more information, see Workspaces Command.

See Also

Other Resources

Open the Team Foundation Administration Console
Workspaces Command
Managing Data



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Restore Lab Management components

[See Also](#) [Send Feedback](#)

By following the procedures in this topic, you can restore one or more components of Visual Studio Lab Management that have failed. You can decide which procedures to complete based on which components failed, but you should complete the procedures in the following order:

1. [Restore the library server for System Center Virtual Machine Manager \(SCVMM\)](#)
2. [Restore the server that is running SCVMM](#)
3. [Restore Team Foundation Server](#)
4. [Restore the Hyper-V hosts](#)

If you follow the recommended order, Team Foundation Server will recognize all virtual machines, templates, hosts and other objects in SCVMM. However, Team Foundation Server will not recognize any objects that were deleted from SCVMM after it was backed up but before Team Foundation Server was backed up. You can remove any such objects from Microsoft Test Manager, as described later in this topic.

Before you can complete these procedures, the following conditions must be true:

- The network names of the new machines must match the names of the machines that were replaced.
- All security accounts must have been restored.
- All software must have been recovered to the same state.

Required Permissions

To perform this procedure, you must be a member of the following groups:

- The Administrators security group on the server that is running the administration console for Team Foundation.
- The SQL Server System Administrator security group or your SQL Server Perform Back Up and Create Maintenance Plan permissions must be set to Allow on each instance of SQL Server that hosts the databases that you want to back up.
- The Administrator user role in SCVMM.

■ Restore the Library Server for SCVMM

When you restore the library server, make sure that you use the same names for the host and the library shares. For more information, see the following page on the Microsoft website:

[Backing Up and Restoring the SCVMM Database](#)

Restore the Server That Is Running SCVMM

To restore the server that is running SCVMM

1. Restore the server that is running SCVMM, and give it the same machine name as the old server.

For more information, see the following page on the Microsoft website:

[Backing Up and Restoring the SCVMM Database.](#)

- Make sure that the \$machinename of the application-tier server for Team Foundation is a member of the Administrators group on the server that is running SCVMM.
- Add or remove any host groups or library shares that were added or removed after the most recent backup. For more information, see the following page on the Microsoft website: [Backing Up and Restoring the SCVMM Database](#)

Restore Team Foundation Server

If Team Foundation Server has failed, you must restore both the application-tier and the data-tier servers that compose the deployment. These components might be hosted on the same server or on multiple servers.

To restore Team Foundation Server

1. Restore the server or servers that are running Team Foundation Server.

For more information, see

[Restore an application-tier server](#), [Restore data to the same location](#), [Restore Data to a Different Server or Instance](#), or [Restore a Single Server Deployment to New Hardware](#).

The machine name, accounts, and software must all be returned to the same state. The machine must be joined and connected to the domain.

If you restore Team Foundation Server to a new machine, you must update the notification URL for Lab Management to match the URL of the new machine. For information about how to set the Lab URL in the Team Foundation Administration Console, see [Configure Lab Management for SCVMM environments](#).

- Make sure that Team Foundation Server is running under a user account that is a member of the Administrators group on the server that is running SCVMM. If Team Foundation Server is running as Network Service, the TFS_MACHINE\$ account should be a member of the Administrators group on the server that is running SCVMM.
- Perform the following steps to make sure that Team Foundation Server is running under a user account that is a member of the Administrators group on all Hyper-V hosts.

1. Open the Team Foundation Administration Console.

2. On the Lab Management tab, click Configure Host Groups, and then click Verify.
 3. When you are prompted for a user name and password, type the user name and password of a member of the Administrators group on all the Hyper-V hosts. The service account for Team Foundation Server is automatically configured for you.
 4. Repeat steps b and c for each team project collection.
- In SCVMM, manually delete any virtual machines that were created after the point to which you restored Team Foundation Server.

These machines will not appear in either Microsoft Test Manager or Microsoft Environment Viewer. You can find them in SCVMM by looking for the team project name in the description field of the virtual machine.

- In Microsoft Test Manager, manually delete any virtual machines that were deleted after the point to which you restored Team Foundation Server.

These machines are no longer in SCVMM. In both Microsoft Test Manager and Microsoft Environment Viewer, the environment will have a status of Failed, and the virtual machines will have a status as Deleted. If a host was added to a SCVMM host group that was already associated with a team project collection, that host will be available for placement the next time that you deploy an environment.

- In Microsoft Test Manager, manually delete any hosts that were removed from SCVMM after the point to which you restored Team Foundation Server.

Because these hosts can no longer run virtual machines and environments, all environments on these hosts will appear as Failed in Microsoft Test Manager and Microsoft Environment Viewer.

- Manually reassociate any a host groups that were associated with a team project collection and team project after the point to which you restored Team Foundation Server.

The virtual machines on this host group will not be associated with the team project.

- At the levels of both team project collections and team projects, manually unassociate any host groups that were unassociated after the point to which you restored Team Foundation Server.

For more information, see [How to: Change the Host Groups for Your Team Project Collections](#).

- Reassociate any host groups that were associated with a team project after the point to which you restored Team Foundation Server.
- Reassociate any library shares that were associated with a team project after the point to which you restored Team Foundation Server, and reimport all virtual machines and templates that were imported into the team project.

Restore the Hyper-V Hosts

If a host that had virtual environments crashed, they will be in a Failed state.

To restore a physical host that was running virtual machines

1. Restore the host from your backup, and connect it to SCVMM.

For more information, see the following page on the Microsoft website:

[Backing Up and Restoring the SCVMM Database.](#)

If the virtual machines are running again on the host, no additional action is required. Lab Management will update the new status of all running virtual machines to the correct state. If you cannot restore the virtual machines and this host is new, the environments will appear as Failed and the virtual machines as Deleted in Microsoft Test Manager or in Microsoft Environment Viewer.

- Refresh Microsoft Test Manager, manually delete the environments that ran on this host, and then re-create them.

If you must determine the environments that ran on this host, SCVMM provides a list of which virtual machines ran on which host.

See Also

Concepts

[Manually back up Team Foundation Server](#)
[Back up and restore TFS](#)

Other Resources

[Move a team project collection](#)



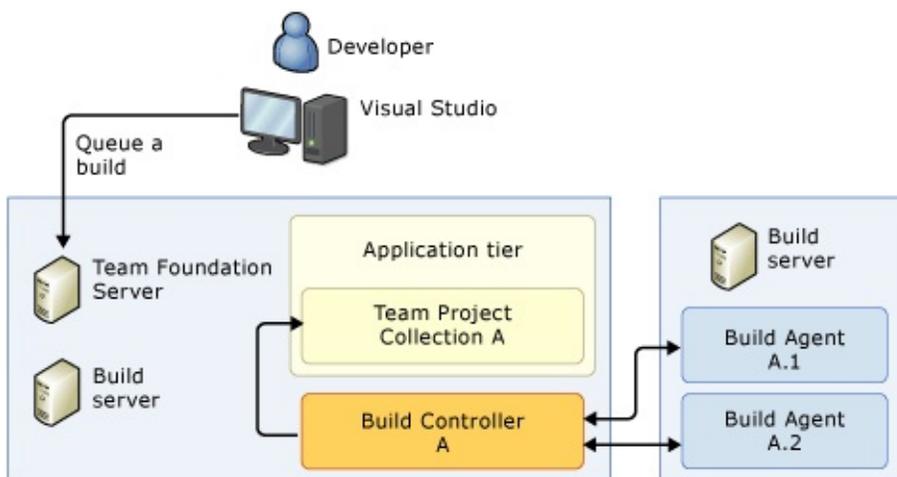
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Configure and manage your build system

[Send Feedback](#)

Do you want to automatically and consistently build, test, and deploy your app in a distributed environment? To get your team started using Team Foundation Build, you need one or more build servers in your build system.



Tip

If your team project collection is hosted on [Visual Studio Online](#) you might be able to use the Hosted Build Controller instead of deploying your own build system.

▣ What do you want to do?

Deploy and work with a build server

To use Team Foundation Build with an on-premises Team Foundation Server, you must deploy and dedicate at least one build server to your team project collection.

Deploy and work with a build controller

Use a build controller to perform lightweight tasks and distribute the processor-intensive work of your build process to its pool of build agents.

Deploy and work with build agents

Use build agents to do the processor-intensive work of your build, including provisioning the workspace, getting files from version control, compiling the code, and running tests.

Set up drop folders

Prepare and then designate one or more drop folders so that your build system can deliver binaries and log files to your team.

Scale out your build system

As your team and your code base grow, expand your build system incrementally with relative ease.

Manage your build system

Occasionally, you will need to monitor and manage your build system, confirm the health of your build server, or diagnose problems.

Use your build system to compile, test, and deploy your app

After your build system is in place, use it to automatically compile and test your applications, and perform other important functions.

Receive build notifications

Sign up for email alerts for yourself, and if you have sufficient permissions, for your team.

Start, stop, delete, or destroy builds from the command prompt

Use the command prompt directly or use a script. (We are not currently republishing this guidance. However, you can read the Visual Studio 2010 versions of the topics.)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Deploy and configure a build server

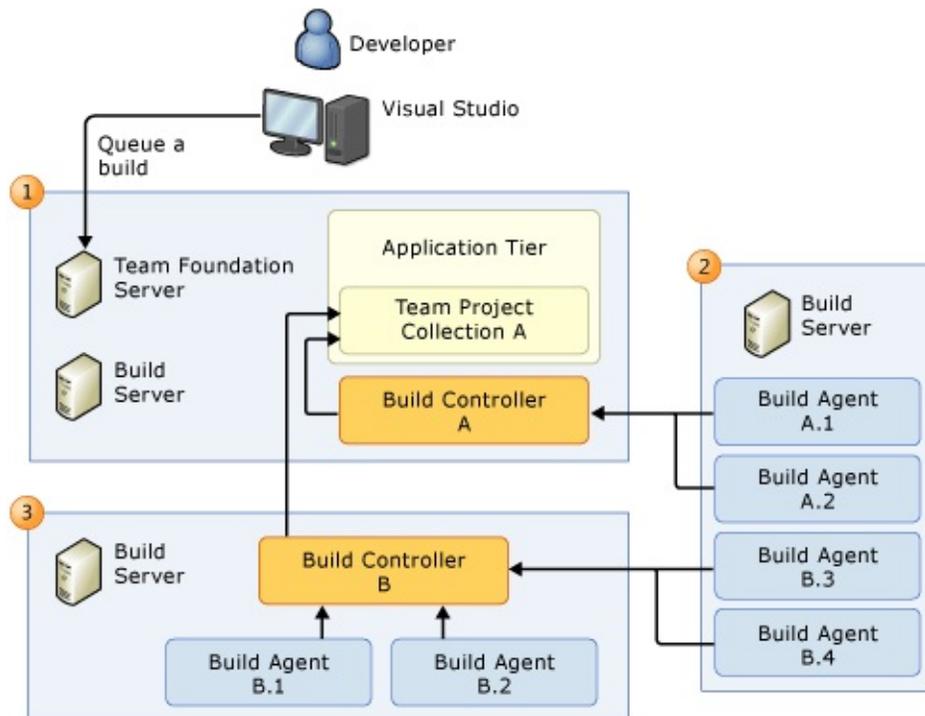
[Send Feedback](#)

To use Team Foundation Build (TFBuild) with an on-premises Team Foundation Server, you must deploy at least one build server.

 **Tip**

If your team project collection is hosted on [Visual Studio Online](#) and your team's needs can be met by a single standard build agent, you can use the Hosted Build Controller instead of deploying your own build agent.

Each build server serves a single team project collection. In fact, although you configure, modify, and manage a build server directly on the computer where Team Foundation Build Service is running, the configuration data is stored in the team project collection.



On a build server, you can run:

1. A single build controller
2. One or more build agents
3. A single build controller and one or more build agents

You can host a build server on the same computer as your Team Foundation Application-Tier Server, but, in most of these situations, this build server should not host any build agents. Build agents place heavy demands on the processor, which could significantly decrease the performance of your application tier. In addition, you might want to avoid running build server components on the application tier to avoid [increasing the attack surface](#). For more detailed examples of viable build system topologies, see [Scale out your build system](#).

Required Permissions

You must be a member of the Windows Administrators group on the build server and a member of the Project Collection Build Administrators group on your team project collection. See [Permission reference for Team Foundation Server](#).

What do you want to do?

- [Understand security risks](#)
- [Deploy a build server](#)
- [Begin configuring a build server](#)
- [Connect a build server to a team project collection](#)
- [Specify service accounts](#)
- [Run your build server in interactive mode](#)
- [Take next steps](#)

■ **Understand security risks**

Installing Team Foundation Build Service increases the attack surface of the computer. Because developers are treated as trusted entities in the build system, a malicious user could, for example, construct a build definition to run arbitrary code that is designed to take control of the server and steal data from Team Foundation Server. Customers are encouraged to follow security best practices as well as deploy defense in-depth measures to ensure that their build environment is secure. This includes developer workstations. For more information regarding security best practices, see the TechNet Article

[Security Guidance](#).

Deploy a build server

You deploy a build server by installing the Team Foundation Build Service. Before you begin this process, here are some tips:

- You can connect a TFBUILD 2010 or TFBUILD 2012 server to your on-premises Visual Studio Team Foundation Server 2013 application-tier server.
- You cannot run Visual Studio Team Foundation Server 2013 TFBUILD on the same computer as TFBUILD 2012 or TFBUILD 2010.
- If you install the build service while you are logged on as a member of the Project Collection Administrators, the installation automatically adds the build service account to the Project Collection Build Service Accounts group, so you don't need to do it manually.
- You can replace an existing build server by copying its configuration to the new build server. See [Set up Team Foundation Build Service](#).
- You can set up an ad-hoc build server on any client or server computer that has adequate processing and storage capacity. For example, an individual developer who has an extra computer could set it up as a build server.
- You can deploy a build server on a physical computer or a virtual machine.

For step-by-step instructions to deploy a build server, see [Set up Team Foundation Build Service](#).

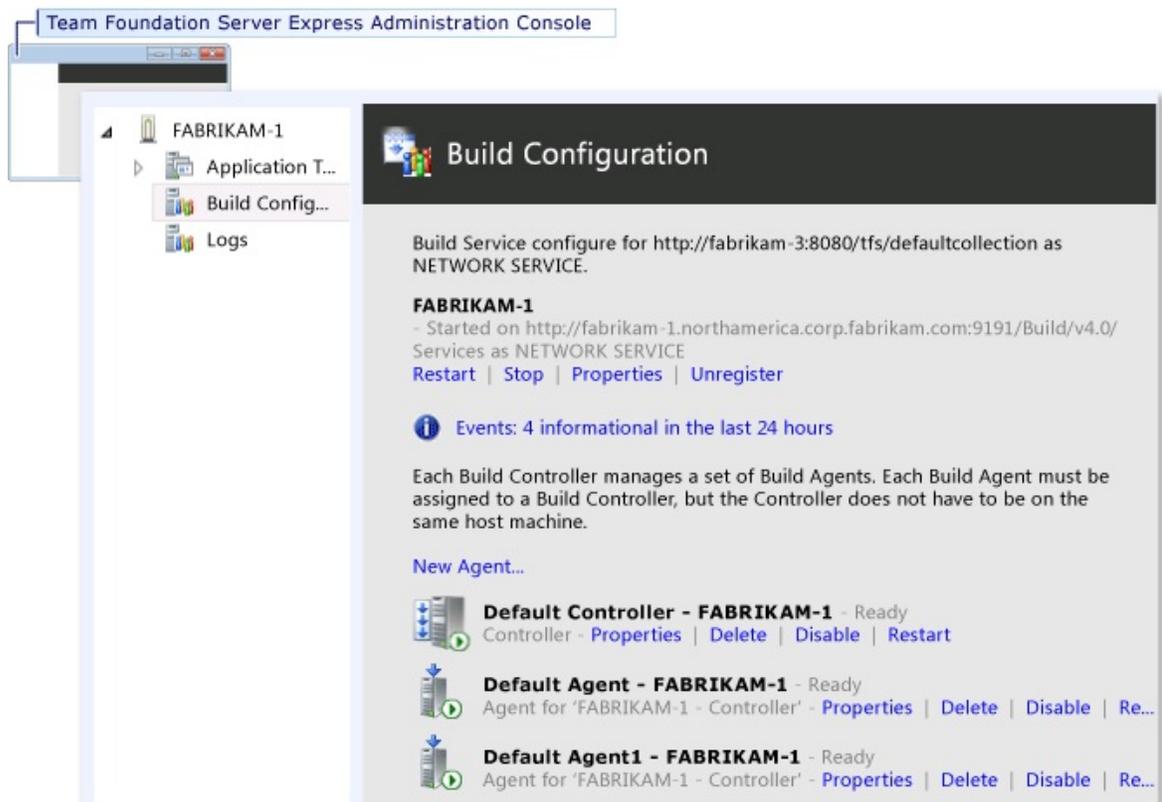
Begin configuring a build server

After you deploy your build server, you can configure it to meet your team's needs.

1. Log on to the build server that you want to configure.
2. From Windows Start, run Team Foundation Administration Console.

The Team Foundation Administration Console appears.

3. In the tree pane, expand the name of the server.
4. Choose the Build Configuration node.



Note

If the message  Configure Installed Features appears instead of a build

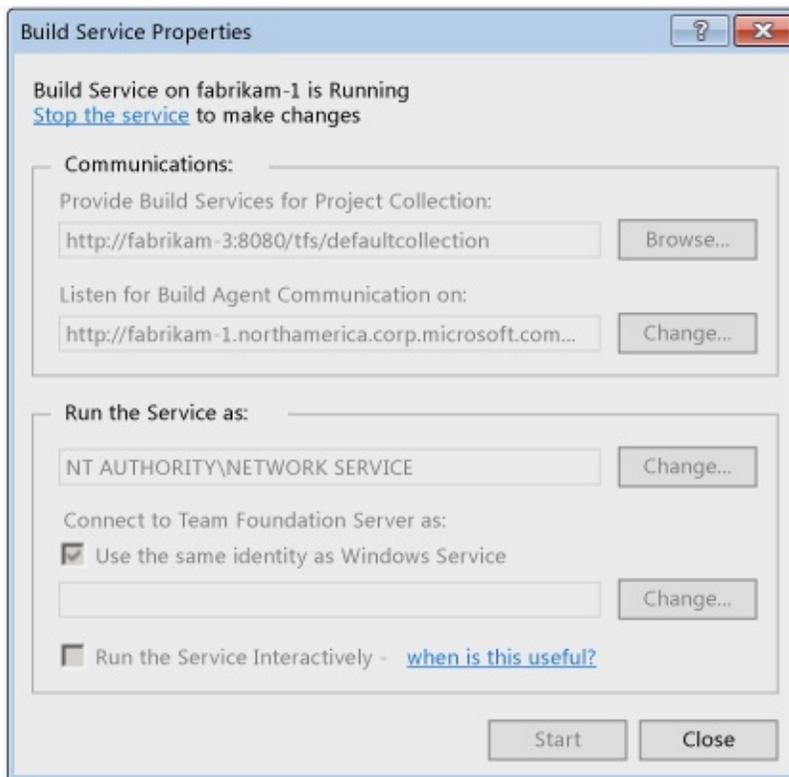
controller or build agents, as shown above, see

[Deploy a build server.](#)

- Choose Properties.



The Build Service Properties dialog box appears.



Before you can configure the build server, you must choose the Stop the service link. See the sections below for details about how to configure your build server.

■ **Connect a build server to a team project collection**

Under Communications, next to Provide Build Services for Project Collection, choose the Browse button to connect your build server to a team project collection on an on-premises Team Foundation Server or on

[Visual Studio Online](#).

You can strengthen security by using Hypertext Transfer Protocol Secure (HTTPS) with Secure Sockets Layer (SSL). See [Set up HTTPS with Secure Sockets Layer \(SSL\) for Team Foundation Server](#).

Specify service accounts

Under Run the Service as you can specify the accounts that enable the build server to provide its services.

Specify the build service account

Immediately under Run the Service as, you can specify the build service account.

NETWORK SERVICE account

For most purposes, the best setting is NT AUTHORITY\NETWORK SERVICE.



One advantage of this approach is that if someone changes the password of a user account (some network administrators require such a change on a regular basis), the build server does not go offline.

User account

Occasionally, you might be required to specify a user account, such as NORTHAMERICA\FABBUILD.



Examples of situations where you must specify a user account include:

- You want to run your build server in interactive mode,

[as explained below](#).

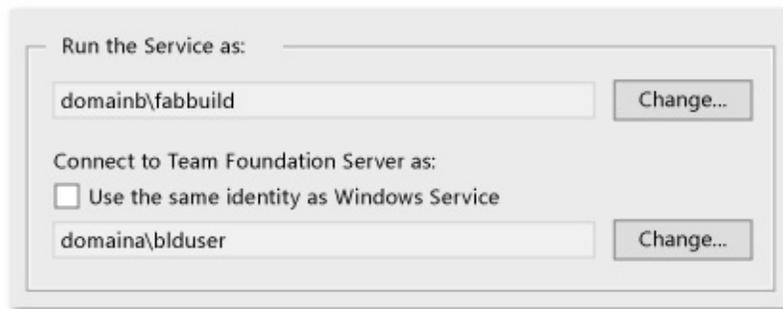
- Your Team foundation Server is inside your firewall, but the build server is outside your firewall.

Regardless of the account you specify, the build service account must belong to the Project Collection Build Service Accounts group.

■ Specify the account used to connect to your Team Foundation Server

You can usually leave the second text box empty. However, in the following cases, your build server can't connect to your Team Foundation Server using the build service account.

Domain trust differences: The domain of the Team Foundation Server does not trust the domain of the build server. For example, the build server is in domainb, and Team Foundation Server is in domaina, which does not trust domainb. You could specify the build service account in the first box, and an account from domaina in the second box:



Team project collection hosted on Visual Studio Online: When you connect your on-premises build server to

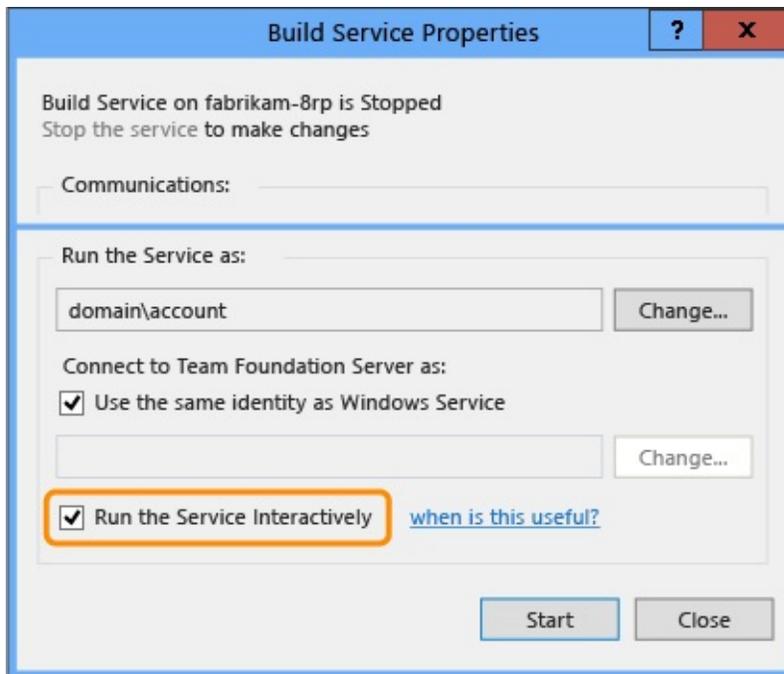
[Visual Studio Online](#), then the Use same identity as Windows Service check box is automatically cleared and the account you used to connect to Visual Studio Online (for example, a Windows Live account) is specified beneath it.

Run your build server in interactive mode

For most purposes, you should run your build server as a Windows service, which is the default setting. However, there are a few tasks (for example running coded UI tests or running tests on a Windows Store app) that a build agent can perform only on a build server that is running as an interactive process.

To run your build server in interactive mode

1. Identify the user account that will act as the build service account. The build service account must:
 - Be a member of the Windows Administrators group on the build server.
 - Be a member of the Build Service Accounts group on your team project collection. See [Grant a build server permission to serve a team project collection](#).
 - Have Change and Read privileges on the drop folder, if any, that you plan to specify in your build definition. See [Select a staging location and set up a drop folder](#).
2. On the Build Service Properties dialog box, choose Stop the service.
3. Under Run the Service as, choose Change, and then specify the credentials of the build service account.
4. Select Run the Service interactively.



5. Choose Start, and then choose OK.
6. Leave the build service account logged on to the build server.

Next Steps

Deploy and configure a build controller

Use a build controller to perform lightweight tasks and distribute the processor-intensive work of your build process to its pool of build agents. You can host one build controller on a build server.

Deploy and configure build agents

Use a build agent to do the processor-intensive work of your build, includes getting files from version control, provisioning the workspace, compiling the code, and running tests. You host can one or more build agents on a build server.

Set up drop folders

You can prepare and then designate one or more drop folders so that your build system can deliver binaries, test results, and log files to your team.

Scale out your Team Foundation Build system

As your team and your code base grow, you can expand your build system incrementally, with relative ease.

Manage your build system

After you deploy your build server, you can manage it from the Team Foundation Administration Console. You can manage the build controller and build agents from either Team Foundation Administration Console or from Visual Studio.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Grant a build server permission to serve a team project collection

[Send Feedback](#)

Tip

You can skip this topic if you deploy your build server while you are logged on as a member of the Project Collection Administrators group.

A build server can build and test code in a team project collection only if the build service account is a member of the Project Collection Build Service Accounts group of that collection.

Required permissions

You must be a member of the Project Collection Administrators group. See [Permission reference for Team Foundation Server](#).

■ Grant a build server permission to serve a team project collection

1. In Visual Studio, in Team Explorer:

1. If you are not already connected to a team project in the team project collection, then connect to the team project.
2. Choose  Home, and then choose  Settings.

2. Under Team Project Collection, choose Group Membership.

Your web browser displays the Security tab and the Groups sub-tab.

3. Under TFS Groups, choose Project Collection Build Service Accounts.

The members of the group appear in the pane on the right side. Verify that your build service account is already present. If not, continue to the next step.

4. Choose Add, and then choose Add windows user or group.

5. On the Add a windows User or Group dialog box, type one of the following values into the Identities box, depending on the type of build service account that your build server uses:

- User account: Type the account name into the box. For example: NORTHAMERICA\FABBUILD
- NETWORK SERVICE on a computer other than your Team Foundation Server: Type computer_name\$. For example, if your computer is named FABRIKAM-1, then you would specify: FABRIKAM-1\$
- NETWORK SERVICE on the same computer as your Team Foundation Server: Type NETWORK SERVICE.

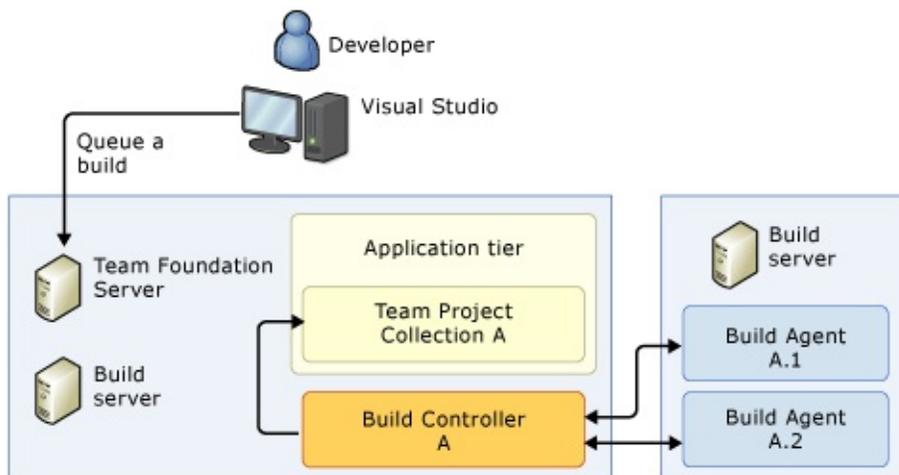
6. Choose Check Name to confirm that the name is valid, and if it is valid, choose Save Changes.
-

- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Deploy and configure a build controller

[Send Feedback](#)

To use Team Foundation Build, your team must have at least one build controller to perform lightweight tasks and distribute the processor-intensive work of your build process to its pool of build agents.

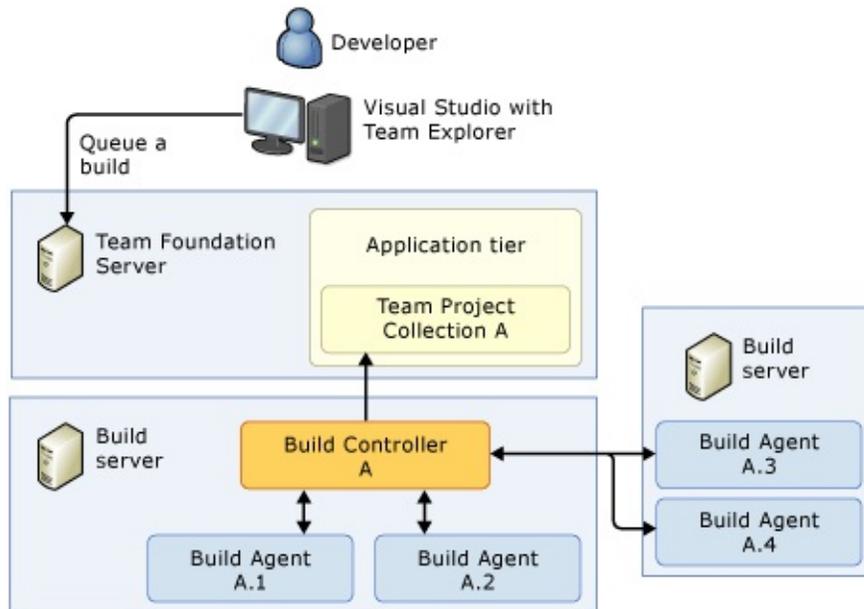


Each build controller is dedicated to a single team project collection. The build controller performs some lightweight tasks, such as determining the name of the build, creating the label in version control, logging notes, and reporting status from the build. The build controller uses the AgentScope activity to delegate processor-intensive work, such as compiling code or running tests, to the build agents within its pool. These build agents can be hosted either on the same build server, or on a different build server.

Because a build controller does not typically require significant processor time, in many cases you can host it on the same computer as your Team Foundation Server, as shown above, or on a low-powered physical or virtual machine.

However, a build controller can demand a significant amount of memory in

certain situations, so you should provide sufficient memory as needed to ensure it functions properly. Or, depending on your team's needs and resources, you might want to run the build controller on a different computer than your Team Foundation Server.



Required Permissions

You must be a member of the Windows Administrators group on the build server, and a member of the Project Collection Build Administrators group on your team project collection. See [Permission reference for Team Foundation Server](#).

What do you want to do?

- [Create or modify a build controller](#)
- [Enable your build processes to leverage supplemental binaries](#)
- [Specify the maximum number of concurrently running builds](#)
- [Remove a Build Controller](#)
- [Use a build controller with a team project collection hosted on Visual Studio Online](#)

■ Create or modify a build controller

To create or modify a build controller from the build server

1. Log on to the build server.
2. From Windows Start, run Team Foundation Administration Console.
3. In the tree pane of the Team Foundation Administration Console, expand the name of the server, and then choose the Build Configuration node.

📌 Note

If the  Configure Installed Features message appears, see Deploy a build server.

4. On the Build Configuration page:
 - If a controller is not listed, choose New Controller.
 - If a controller is already listed:



Choose Properties.

The Build Controller Properties dialog box appears.

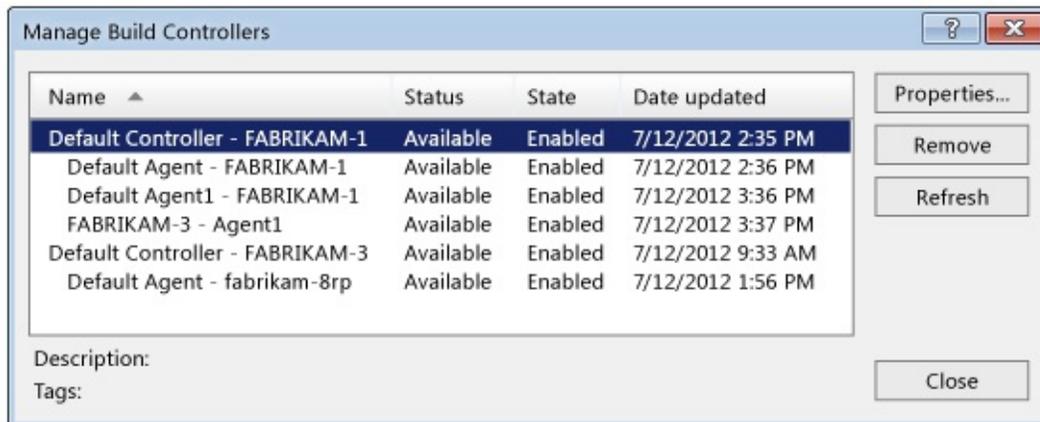
To modify a build controller from Visual Studio

1. In Visual Studio, in Team Explorer:
 1. If you are not already connected to a team project in the team project collection, then connect to the team project.

2. Choose  Home, and then choose  Builds.

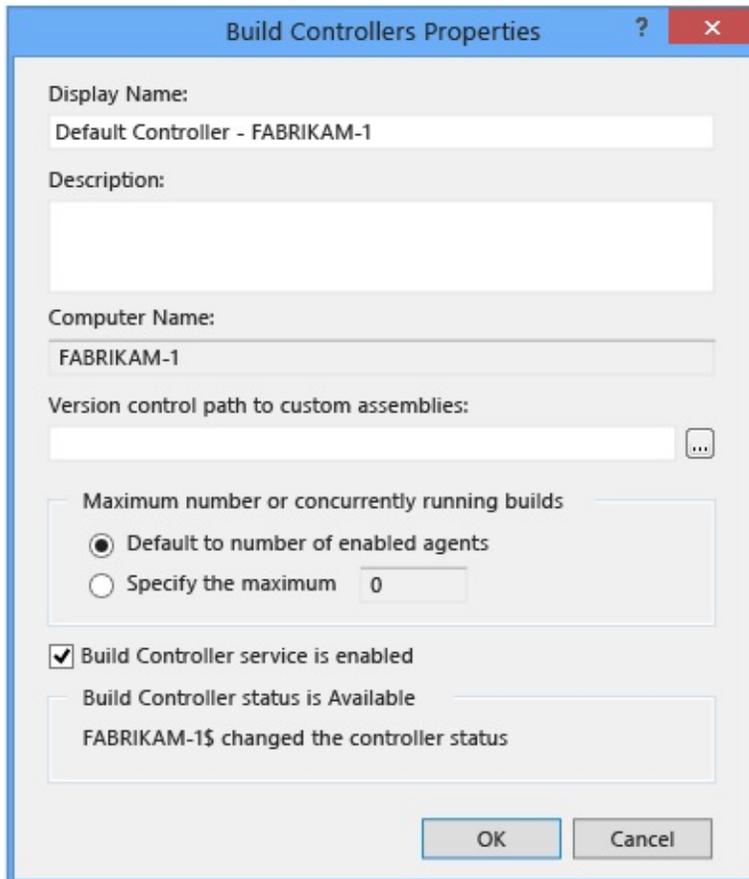
2. On the Builds page, choose Actions, and then Manage Build Controllers.

The Manage Build Controllers dialog box appears.



3. Select the build controller that you want to modify, and then choose Properties.

The Build Controller Properties dialog box appears.



In the Display Name and Description fields: Type a name and a description that can help team members easily identify the appropriate build controller when they define their build processes.

See the sections below for details about how to configure your build controller.

Enable your build processes to leverage supplemental binaries

You can enable your build process to leverage binaries that you have uploaded to your Team Foundation Server, for example:

- Assemblies that contain your custom workflow activities. See [Use and develop custom build process activities](#).
- Third-party unit test frameworks. See [Run tests in your build process](#).
- Custom MSBuild tasks

To enable your build processes to leverage these kinds of code, upload the binaries to the folder (or any of its descendant folders) that you specify in the [Version control path to custom assemblies](#) box. When you set or modify the value in this box, the build server automatically restarts to load the assemblies.

Tip

If your build results include a TF215097 error message, your build process might include a custom activity that is not in this folder or in any of its descendant folders.

■ Specify the maximum number of concurrently running builds

In most cases, you should leave this value set to Default to number of enabled agents. However, in some cases, you might want to control the depth of the build queue. For example:

- You observe that too many builds show a status of Running even though many of them are stalled as they wait for a build agent to be assigned to them. This situation can confuse the team. To avoid the problem, select Specify the maximum and then specify a value that is lower than the number of build agents that are pooled under the build controller.
- You design a custom build template to enable your builds to leverage more than one build agent at a time, from a pool of 12 build agents under the build controller. The builds that are processed by this build controller are based on a custom build template that concurrently delegates to three build agents. To make the build queue data more useful, you could configure the build controller with Maximum number of concurrently running builds set to Specify the maximum, with a value of 4.

Remove a Build Controller

1. In Visual Studio, open the Manage Build Controllers dialog box, as explained above in

[Modify a build controller from Visual Studio.](#)

- Select and then remove each build agent under the build controller, either by choosing Remove, or by choosing Properties and then using the Build Agent Properties dialog box to assign the build agent to another build controller.
- Select the build controller that you want to remove, and then choose Remove.

Note

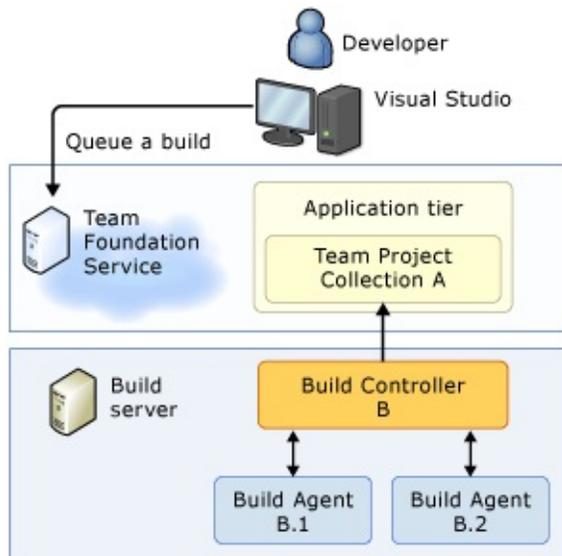
You can also use the Team Foundation Administration Console to remove the build controller while you are logged on to the build server. But if you do, you might still need to use Visual Studio, or log on to other build servers first, to remove any remote build agents pooled by the build controller.

Use a build controller with a team project collection hosted on Visual Studio Online

Use the Hosted Build Controller: If your team project collection is hosted on

[Visual Studio Online](#), you might be able to skip deploying your own build controller and use the Hosted Build Controller instead. To use the Hosted Build Controller, you simply select it when you define your build process. See [Use the Hosted Build Controller](#).

Use an on-premises build controller: If your build process requires resources outside of the [Hosted Build Controller capabilities](#), then you can connect your on-premises build controller to your hosted team project collection.



Next Steps

Deploy and configure build agents

Use a build agent to do the processor-intensive work of your build. This includes getting files from version control, provisioning the workspace, compiling the code, and running tests. You can host one or more build agents on a build server.

Set up drop folders

You can prepare and then designate one or more drop folders so that your build system can deliver binaries, test results, and log files to your team.

Scale out your Team Foundation Build system

As your team and your code base grow, you can expand your build system incrementally, with relative ease.

Manage your build system

After you deploy your build server, you can manage it from the Team Foundation Administration Console. You can manage the build controller and build agents from either Team Foundation Administration Console or from Visual Studio.

Define your build process

After your build system is in place, you are ready to define your build process, which contains your instructions about which code projects to compile, what actions trigger a build, what tests to run, and any other procedures required by your team.

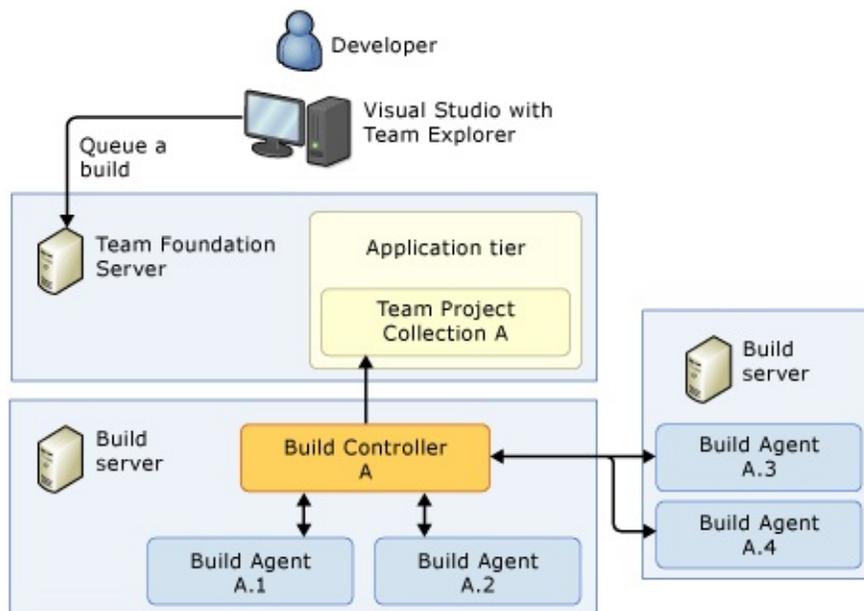
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Deploy and configure build agents

[Send Feedback](#)

To use Team Foundation Build, your team must have at least one build agent to perform the processor-intensive work of your build process.



Each build agent is dedicated to and controlled by a single build controller. Build agents can be hosted on the same build server that hosts their build controller, but this is not required, and in some cases your team's needs can most efficiently be met by a single build server to host a build controller that controls build agents on multiple build servers.

The build agent executes the steps of your build process that are contained in the AgentScope activity. Typically, these steps include getting files from version control, provisioning the workspace, compiling the code, running tests, and merging files back into version control.

Make sure the build server that hosts your build agents has sufficient storage and processing capabilities to match the size and complexity of the codebase and tests on the team project collection. Typically, you should host no more than one build agent per processor core on the build server. You can also enhance performance by dedicating a single physical hard drive to the [working directory](#) of each build agent.

Tip

If your team project collection is hosted on [Visual Studio Online](#), and your team's needs can be met by a single standard build agent, you can use the Hosted Build Controller instead of deploying your own build agent.

Required permissions

You must be a member of the Windows Administrators group on the build server and a member of the Project Collection Build Administrators group on your team project collection. See [Permission reference for Team Foundation Server](#).

What do you want to do?

- [Create or modify a build agent](#)
- [Install Visual Studio and other software to enable compilation and other capabilities](#)
- [Specify the working directory](#)
- [Enable your build agent to run tests](#)
- [Assign tags to represent build agent capabilities or purposes](#)
- Deploy a build agent that can compile and test a Windows Store app
- [Remove a build agent](#)

■ Create or modify a build agent

To create or modify a build agent from the build server

1. Log on to the build server that you want to configure.
2. From Windows Start, run Team Foundation Administration Console.

The Team Foundation Administration Console appears.

3. In the tree pane of the Team Foundation Administration Console, expand the name of the server, and then choose the Build Configuration node.

Information about the build server appears in the content pane.

If the message  Configure Installed Features appears, see Deploy a build server.

4. On the Build Configuration page:
 - To create a new build agent, choose New Agent.
 - To modify an existing build agent



choose Properties.

The Build Agent Properties dialog box appears.

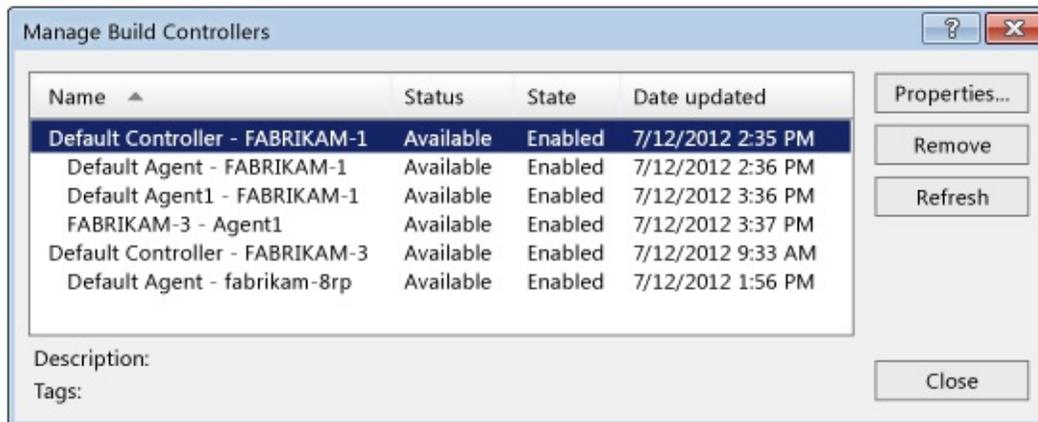
To modify a build agent from Visual Studio

1. In Visual Studio, in Team Explorer:
 1. If you are not already connected to a team project in the team project collection, then connect to the team project.

2. Choose  Home, and then choose  Builds.

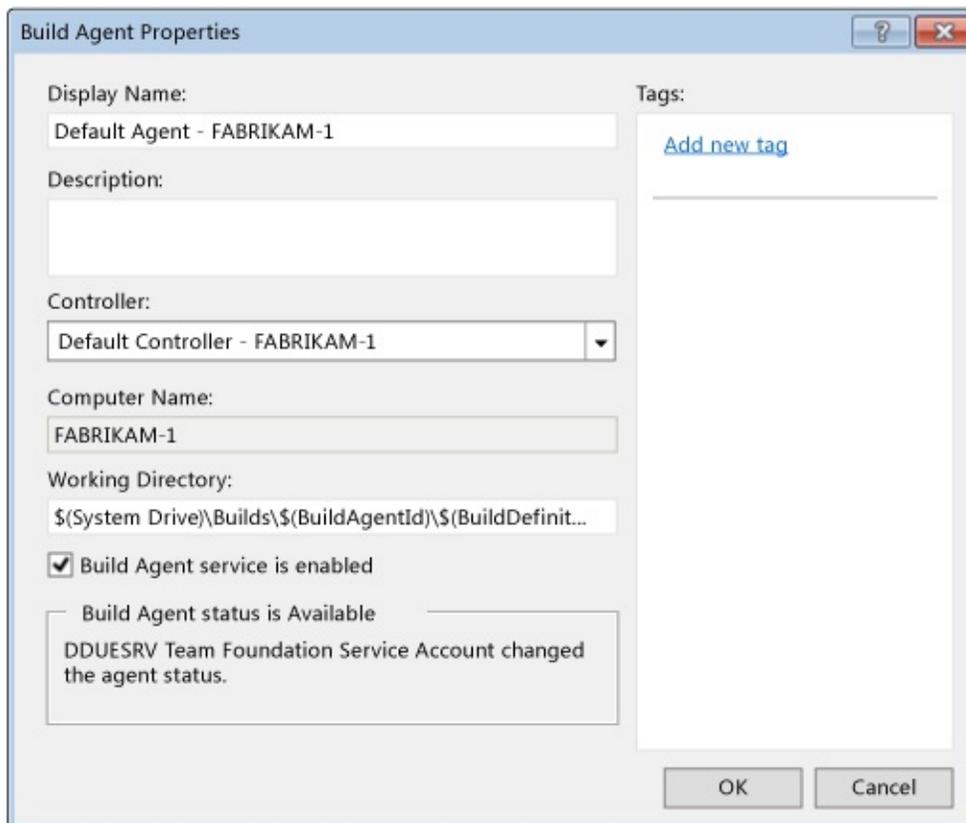
2. On the Builds page, choose Actions, and then Manage Build Controllers.

The Manage Build Controllers dialog box appears.



3. Select the build agent that you want to modify, and then choose Properties.

The Build Agent Properties dialog box appears.



Display Name, Description: Type a name and a description to help team members easily identify the build agent.

Controller: Select the build controller that you want to control this build agent. The build controller can run on the same build server as this build agent or on a different build server.

See the sections below for details about how to configure your build agent.

■ Install Visual Studio and other software to enable compilation and other capabilities

You must install on the build agent the version of Visual Studio that your team uses on its dev machines. See [Installing Visual Studio](#). You must also install any other software and components that are installed on your dev machines and that are required to build your app.

■ Specify the working directory

You can specify the working directory, which the build agent uses to read from or write to files. For example, source files are copied to subdirectories in this folder, and binaries are created and stored in other subdirectories in this folder.

📌 Tip

You can enhance performance by dedicating a single physical hard drive to the [working directory](#) of each build agent.

■ Use working directory tokens

Although you can specify a literal path for the Working Directory property (for example, c:\temp\build\), a simpler and more durable approach is to use tokens to specify the path. You can use two kinds of tokens:

Environment variables

Environment variables contain information about the environment for the system and the user who is logged on. The most typical variable you might use is **SYSTEMDRIVE**, but for some situations you might also use variables such as **USERNAME** or **HOME**.

📌 Tip

To list the environment variables on your build server, open a command prompt and type **set**.

Team Foundation Build variables

You can use the following variables in a build agent working directory:

- **\$(BuildAgentId)**: An automatically generated integer that uniquely identifies a build agent within a team project collection.

- **\$(BuildAgentName)**: The Display Name of the build agent.
- **\$(BuildDefinitionId)**: An automatically generated integer that uniquely identifies a build definition within a team project collection.
- **\$(BuildDefinitionPath)**: The team project name and the build definition name, separated by a backslash.

▣ **Working directory example**

For example, you have a build agent that is named BuildBot3. You have defined two builds that are called NightlyBuild and WeeklyBuild in a team project that is called CoolApp. In the Working Directory box, you specify the following value: \$(SystemDrive)\TeamBuilds\\$(BuildAgentName)\\$(BuildDefinitionPath). As a result, the BuildBot3 build agent creates and uses the following two working directories:

C:\TeamBuilds\BuildBot3\CoolApp\NightlyBuild

C:\ TeamBuilds\BuildBot3\CoolApp\WeeklyBuild

▣ **Make sure the path to the working directory is not too long**

The working directory that you specify must not cause the build agent to produce any physical paths that are longer than 259 characters. Otherwise, your builds could fail and log this error: **TF10128: The path PhysicalPath contains more than the allowed 259 characters. Type or select a shorter path.**

To solve this problem, specify a working directory that results in a shorter physical path. For example you could specify \$(HOMEDRIVE)\bld\\$(BuildAgentID)\\$(BuildDefinitionID), which would result in a working directory such as: **c:\bld\3\2**.

▣ **Subdirectories created in the working directory**

The build agent creates and works in the following subdirectories under this path.

|

Subdirectory Used to Store Files...

Sources

Read by the build agent, such as source files. You specify the files that it downloads in the Workspace settings for each build definition. See [Work with build workspaces](#).

Binaries

Compiled by the build agent, such as compiled application files.

TestResults

Produced by whatever tests the build agent ran.

Enable your build agent to run tests

You can define a build process that performs one or more automated test runs.

⚠️ Important

Many kinds of tests and test operations require that you install on your build agent the same version of Visual Studio that your team uses on its dev machines. See [Installing Visual Studio](#).

The build agent can run:

- Code Coverage
- Coded UI Tests (Requires a build server that is running in interactive mode. See [Run your build server in interactive mode and Verifying Code by Using UI Automation](#).)
- Database Test Data Generation
- Database Unit Tests
- Generic Tests
- Load Tests
- Unit Tests
- Ordered Tests
- Test Impact Analysis
- Web Tests

Assign tags to represent build agent capabilities or purposes

As the scale of your build system grows, you might need to define specialized build agents. Whenever a build agent has special capabilities or is intended for a specific kind of usage, you should assign one or more tags to that agent. This way, when a team member creates a build definition that requires a certain kind of build agent, they can specify the tag in their build definition.

You can assign tags from the

[Build Agent properties dialog box](#), described above. You can then apply the tags to your build definitions.

The following table offers examples of tag names and the build agent capabilities that they could represent.

You could apply the following tag	To identify a build agent that can...
...	
x86	Compile 32-bit applications
x64	Compile 64-bit applications
bvt	Run your BVT tests, which are run by your nightly BVT build.

WindowsStore	Compile and test a Windows Store app.
IIS	Compile an ASP.NET Web application and then stage and host it on the computer on which the build agent is running.
interactive	Perform tasks that require an agent on a build server that is running in interactive mode.

You can apply more than one tag to a build agent. For example, you could create a build agent with the tags `x86` and `Release` to indicate an agent that is set up to compile the `Release` configuration of a 32-bit application.

If you run multiple build agents on the same build server, they probably will all have the same capabilities. Therefore, you probably will want to apply the same tags to all build agents on that build server.

Remove a build agent

1. In Visual Studio, open the Manage Build Controllers dialog box, as explained previously in

[Create or modify a build agent.](#)

- Select the build agent that you want to remove, and then choose Remove.

Tip

You can also use the Team Foundation Administration Console to remove the build agent while you are logged on to the build server.

Next steps

Scale out your Team Foundation Build system

As your team and your code base grow, you can expand your build system incrementally, with relative ease.

Manage your build system

Occasionally, you will need to monitor and manage your build system.

Use your build system to compile, test, and deploy your app

After your build system is in place, your team is ready to create a simple build process (for example, a continuous integration build) and benefit from automated building and testing of your app.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Select a staging location and set up a drop folder

[Send Feedback](#)

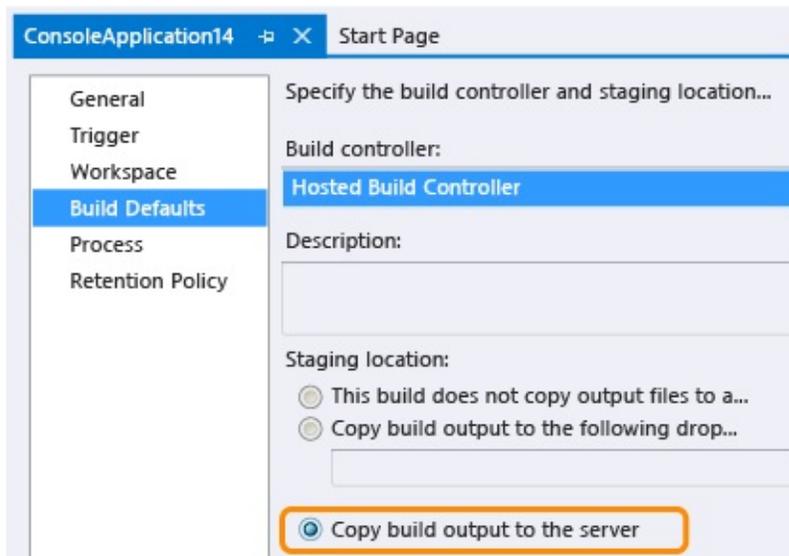
When you define your build process, you can specify a staging location so that you can deliver binaries to your team.

What do you want to do?

- [Drop outputs into TFS](#)
- [Set up a drop folder on a file share for your on-premises build controllers](#)

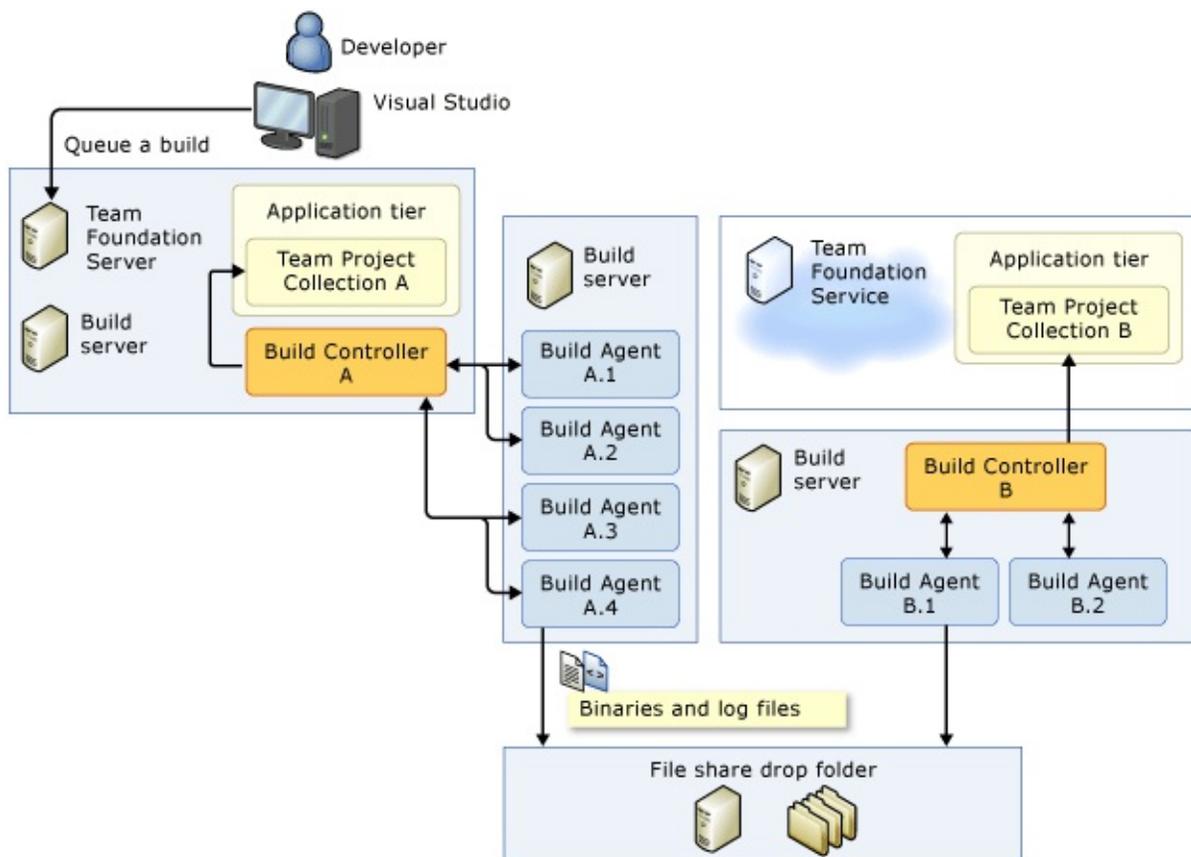
Drop outputs into TFS

You might find it most convenient to drop your build outputs into your Team Foundation Server.



Set up a drop folder on a file share for your on-premises build controllers

You can specify a path to a file share when you define or queue a build definition that is run by an on-premises build controller. The on-premises build controller can be dedicated to a team project collection on either an on-premises Team Foundation Server (TFS) or on Visual Studio Online.



Tip

If you see a TF270016 error in your build results log, it's possible that the build server does not have the required permissions to access the drop folder. Follow the instructions below to resolve this problem.

Requirements

You must be a member of the Administrators group on the computer where the drop folder resides.

You can use a file share drop folder on the following operating systems:

- Windows Server 2012 R2
- Windows Server 2012
- Windows Server 2008
- Windows Server 2003
- Windows 8.1
- Windows 8
- Windows 7
- Windows Vista

To set up a drop folder on a file share

1. Log on to the computer where the drop folder will reside.
2. Open Windows Explorer (File Explorer in Windows 8), and browse to the folder that contains (or will contain) the drop folder.
3. If necessary, create the drop folder.

Tip

Try to keep the path to the drop folder as short as possible (for example: c:\drops). The drop folder must not cause the build agent to produce any physical paths that contain more than 259 characters. Otherwise, your builds will fail and log the **TF205022** error message.

4. Open the context menu for the folder, choose Properties, and then choose the Sharing tab.
5. On Windows Server 2012 R2, Windows Server 2012, Windows Server 2008, Windows 8.1, Windows 8, Windows 7, and Windows Vista only: Choose Advanced Sharing.

The Advanced Sharing dialog box appears.

6. Select Share this folder, and then choose Permissions.

The Permissions for FolderName dialog box appears.

7. Choose Add.

The Select Users, Computers, or Groups dialog box appears.

8. Perform one of the following steps, depending on what type of account you've specified as the build service account on the build server that hosts your build controller:
 - NETWORK SERVICE running on the computer that contains the drop folder: In the Enter the object names to select box, enter NETWORK SERVICE.
 - NETWORK SERVICE running on a different computer from the one that contains the drop folder:
 1. Choose Object Types.

The Object Types dialog box appears.
 2. Select the Computers check box, and then choose OK.
 3. Verify that the location is correct.
 4. In the Enter the object names to select box, enter BuildMachine\$ where BuildMachine is the name of the build machine on which the build agent is running.
 - A domain account: Verify that the location is correct. In the Enter the

object names to select box, enter the name of the account.

9. Choose OK.
10. Repeat the previous two steps for the build service account on the build server that hosts your build agents.
11. In the Permissions for FolderName dialog box, choose the account that you just added to the Group or user names list.
12. Select the Change and Read check boxes, and then choose OK.

Next Steps

Once you have identified and set up the drop folder, you can specify the path to this folder when you define your build and when you queue a build.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Scale out your build system

[Send Feedback](#)

To use Team Foundation Build for automated building and testing of your app, you must first set up a build server, add a build controller and a few build agents, and finally designate a drop folder. If you have a small start-up team working on a new project, you can probably deploy all these build system components on a single computer in a few minutes. As your team and your code base grow, you can expand your build system incrementally, with relative ease.

Tip

If your team project collection is hosted on Visual Studio Online, you might be able to skip all these steps, and use the [Hosted Build Controller](#) instead, as explained below.

Here are some examples that demonstrate how you can start small and simple, and then later scale out the build system as your requirements become more demanding.

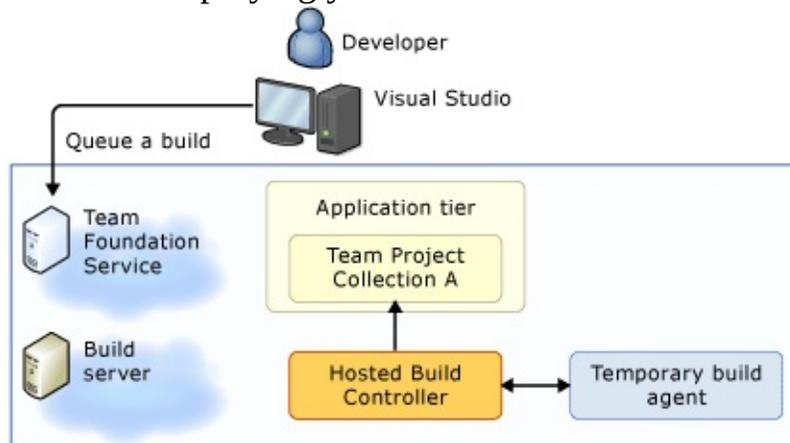
- [Visual Studio Online with Hosted Build Controller](#)
- [Visual Studio Online with on-premises build servers](#)
- [Build system for trial usage or a very small team](#)
- [Build system for a small team](#)
- [Multiple-build-server systems](#)
- [Build system to support multiple team project collections](#)

- [Next Steps](#)

Visual Studio Online with Hosted Build Controller

If your team project collection is hosted on

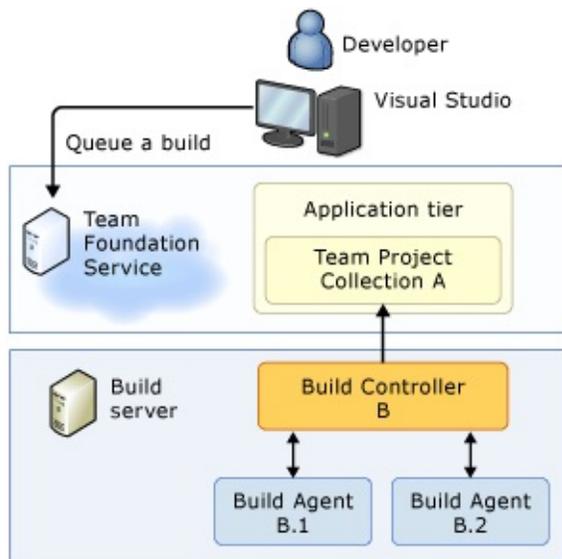
[Visual Studio Online](#) you might be able to use the Hosted Build Controller instead of deploying your own build servers.



See [Use the Hosted Build Controller in a team project collection hosted on Visual Studio Online](#).

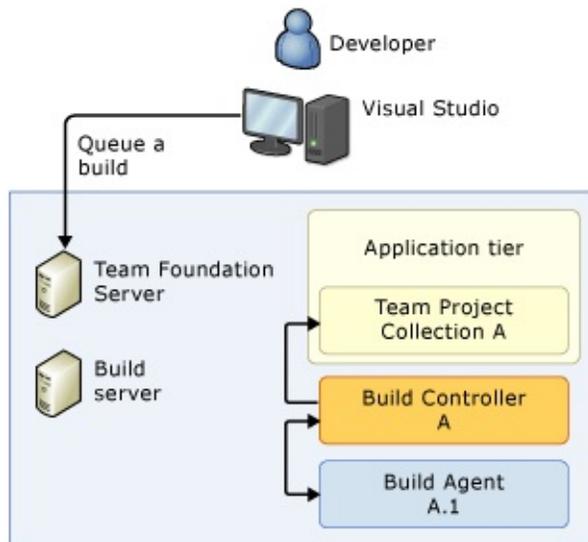
Visual Studio Online with on-premises build servers

If your team project collection is hosted on Visual Studio Online and your team needs larger scale or customized build agents, you can connect your on-premises build servers to Visual Studio Online.



Build system for trial usage or a very small team

If you are using Team Foundation Server on a trial basis or work on a very small team, the following topology might work for you.

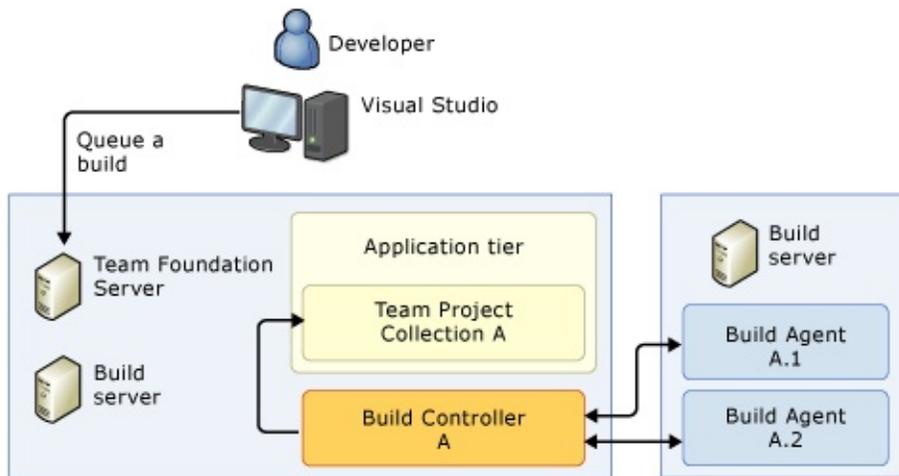


This topology might work for a team that runs builds infrequently and only during off-hours, such as a team that runs only a single nightly build. However, for many teams, it is insufficient because:

- The build agent places heavy demands on the processor, which could significantly decrease the performance of the application tier.
- The build controller can exert pressure on the system's memory, especially if the controller is managing many active build agents at the same time.
- Installing Team Foundation Build Service increases the attack surface of the computer. See [Build Server: Understand security risks](#).

Build system for a small team

If you work on a small team with an on-premises Team Foundation Server, consider this topology:

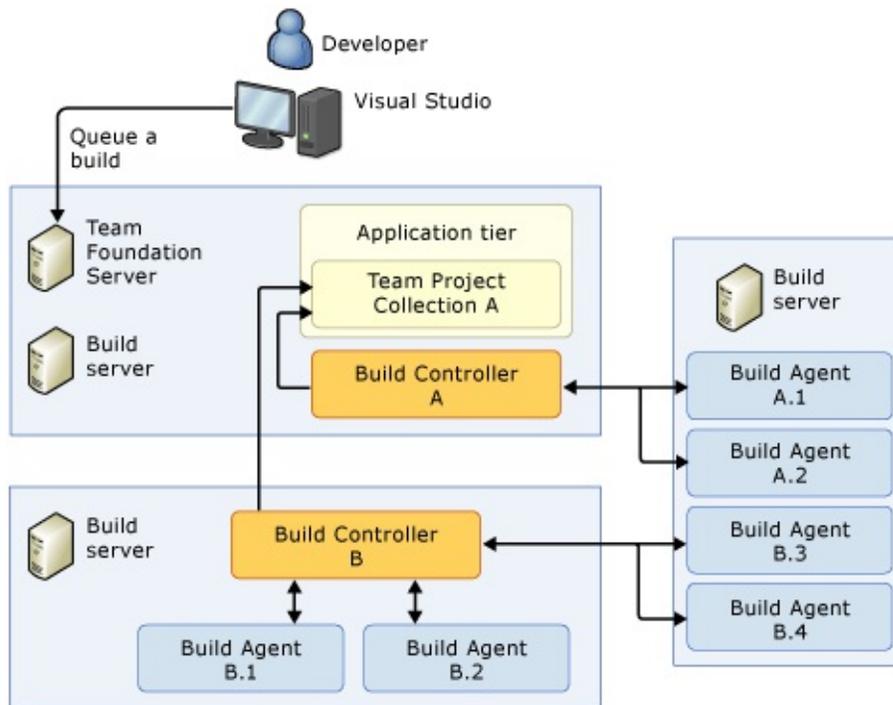


Because build agents perform the processor-intensive work on a separate machine, they do not affect the performance of the application tier when builds are run.

You could also run the build controller on the dedicated build server. However, the topology in the illustration has the advantage of making build system changes less disruptive, such as when you must repair or replace the build server.

Multiple-build-server systems

As the size of your team and your code base increases, you can incrementally add resources to meet your requirements. For example, you could add an additional controller and build agents.



The presence of Build Controller A on the same machine as the application tier is generally not a problem from a processor standpoint. However, you might move the build controller to another server because of the memory pressure and attack surface issues mentioned previously.

By using multiple build servers, you can dedicate each server to a different purpose, as described in the following examples:

- A build server on a high-performance computer dedicated to build agents that process continuous integration or gated check-in builds. The team needs these kinds of builds—especially gated check-in builds—to run quickly so that their work is not held up waiting for a build.
- A build server dedicated to nightly scheduled BVT builds that require a lot

of time to run processes such as large test runs and code analysis.

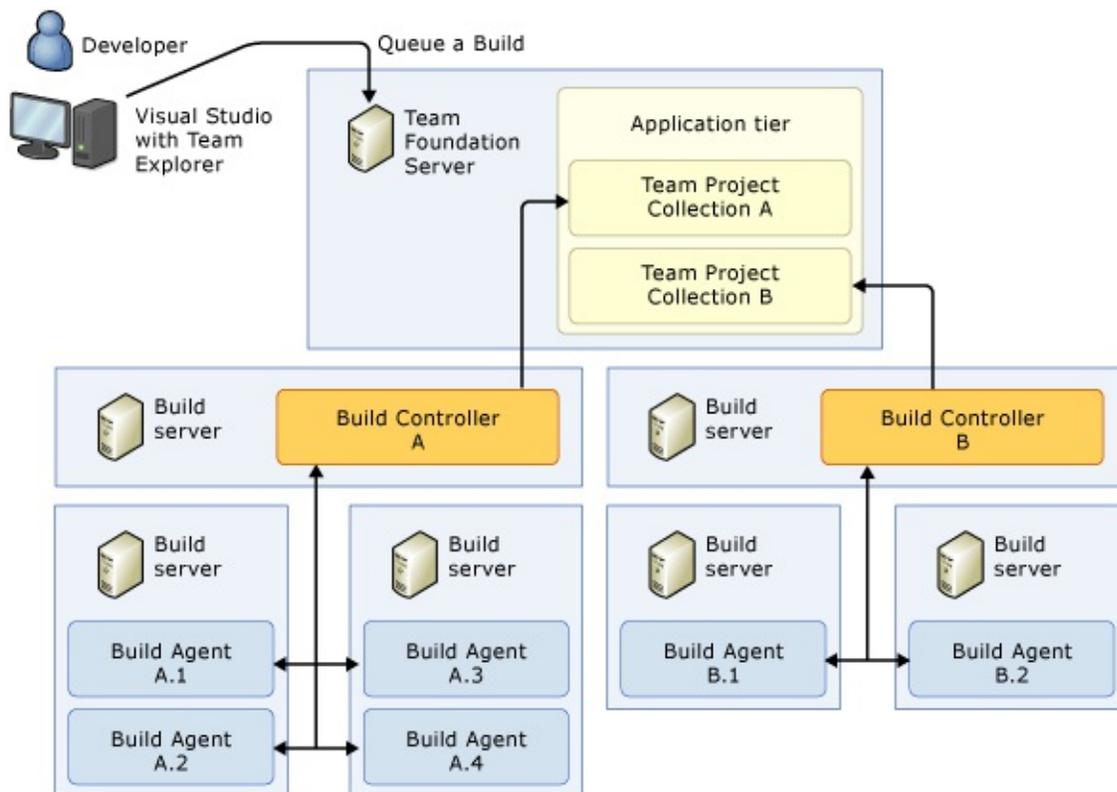
- A build server prepared and dedicated to specialized tasks such as building and testing a Windows Store app.

 **Tip**

In scenarios such as these you can apply tags to specialized build agents and then constrain your build definitions to use only build agents with the correct set of tags. See [Assign tags to represent build agent capabilities or purposes](#), [Specify Which Build Agents Process Your Build](#) for a simple default build process, and [Run activities on the build agent](#) for an advanced custom build process.

Build system to support multiple team project collections

The following build system topology example could support an enterprise-level software effort.



Each team project collection must have its own build controller, as shown in above. Notice how this topology isolates the build servers. Team members who work on Team Project Collection A can use only the build agents that Build Controller A controls. This constraint could be useful in situations where you need to tightly control access to more sensitive intellectual property.

Next Steps

Deploy and work with a build server

To use Team Foundation Build with an on-premises Team Foundation Server, you must deploy at least one build server. You can also connect one or more on-premises build servers to Visual Studio Online.

Tip

As you scale out your system, you can replace an existing build server when you deploy a new build server. For example, you might want to host the same configuration and set of build controllers and build agents on a new, more powerful computer. See [Set up Team Foundation Build Service](#).

Deploy and configure a build controller

Use a build controller to pool one or more build agents. You can host one build controller on a build server.

Deploy and configure build agents

Use a build agent to do the processor-intensive work of your build, includes getting files from version control, provisioning the workspace, compiling the code, and running tests.

Set up drop folders

You can prepare and then designate one or more drop folders so that your build system can deliver binaries, test results, and log files to your team.

Manage your build system

After you deploy your build server, you can manage it from the Team Foundation Administration Console. You can manage the build controller and build agents from either Team Foundation Administration Console or from Visual Studio.

Use Team Foundation Build

With your build system in place, your team is ready to create a simple build process (for example, a continuous integration build) and benefit from automated building and testing of your app.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Manage your build system

[Send Feedback](#)

You've configured your build servers, including your build controllers and build agents. You've defined build processes to meet your team's needs. Your team is now running and managing your automated builds to compile and test your app.

Occasionally, you will run into situations in which you need to monitor and manage your build system, such as when you are diagnosing problems or scaling out your build system.

Required permissions

You must be a member of the Windows Administrators group on the build server, and a member of the Project Collection Build Administrators group on your team project collection. See [Permission reference for Team Foundation Server](#).

What do you want to do?

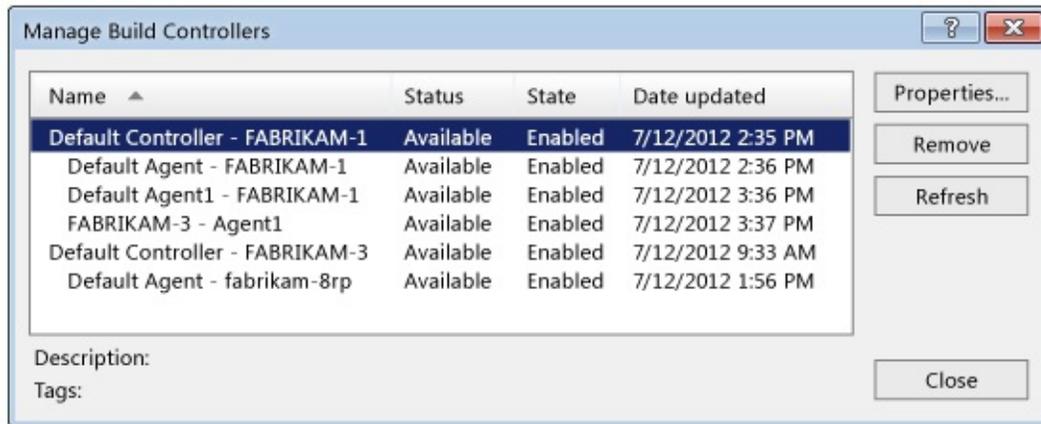
- [Check the status, manage, and modify your controllers and agents from Visual Studio](#)
- [Check the status, manage, and modify your server, controller, and agents from the administration console](#)
- [Start, stop, or restart a build server](#)
- [Restart a build controller or build agent](#)
- [Enable or disable a controller or agent](#)
- Specify the maximum number of concurrently running builds for a

controller

- [Register or unregister a build server](#)
- [Monitor the health of your build server](#)
- [Use event logs to diagnose problems](#)
- [Take next steps](#)

Check the status, manage, and modify your controllers and agents from Visual Studio

Sometimes you want to check the status, and manage your build controllers and build agents, for example, when a queued build process is not starting or is failing to make progress. You can do all this directly from Visual Studio.



1. In Visual Studio, in Team Explorer:
 1. If you are not already connected to a team project in the team project collection, then connect to the team project.
 2. Choose  Home, and then choose  Builds.
2. On the Builds page, choose Actions, and then Manage Build Controllers.

Next Steps

Check the status of your controllers and agents

- If the State is Available and Status is Enabled, then the component is probably functioning correctly.

- If the State is Disabled, you can select the component, choose Properties, and then re-enable the component.
- If the Status is Offline, then the component is disconnected. Log on to the build server and make sure it is registered and can connect to your Team Foundation Server.

View, modify, enable, or disable a controller or agent

Select it and choose Properties, and then see

[Deploy and configure a build controller](#) or [Deploy and configure build agents](#).

Remove a controller or agent

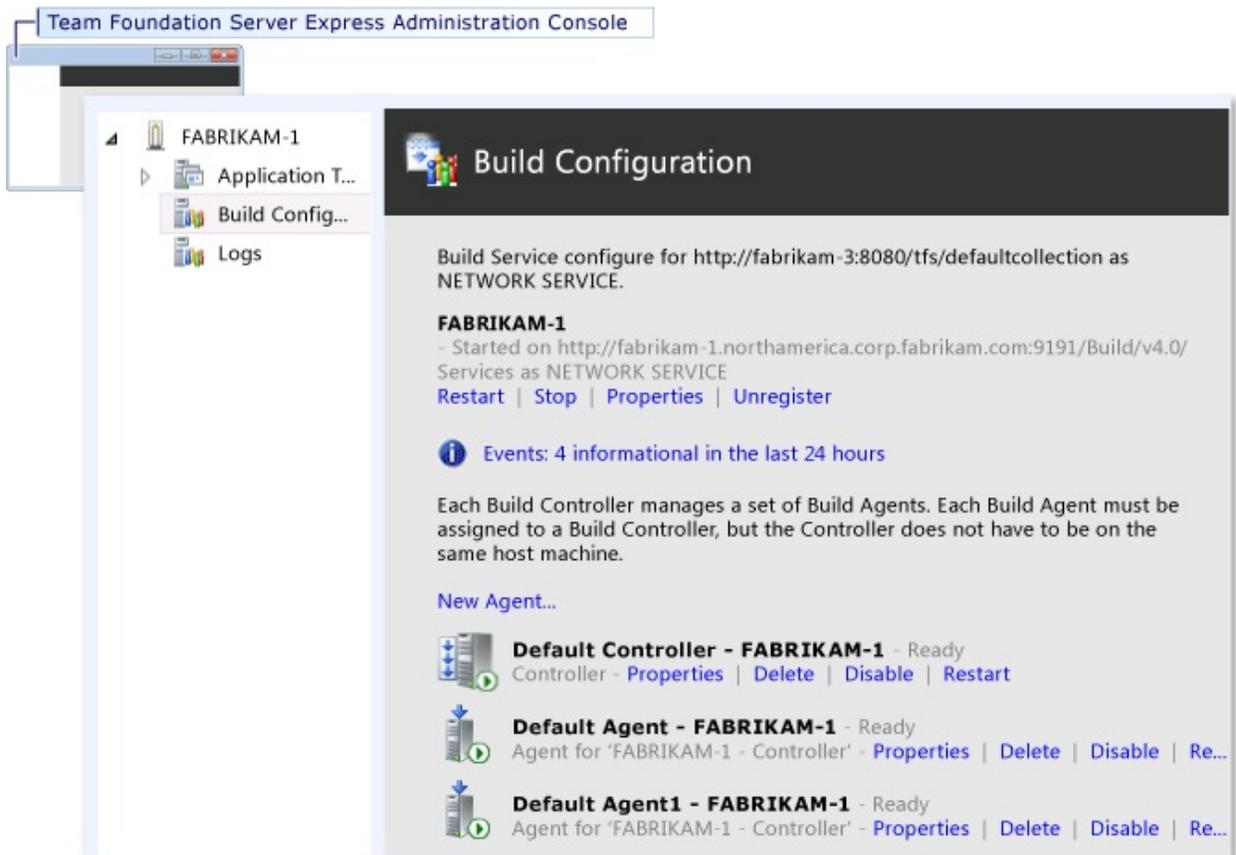
See Remove a build controller or Remove a build agent.

Manage or resolve problems with a build server, controller or agent

See the sections below for more information.

Check the status, manage, and modify your server, controller, and agents from the administration console

When you want to check or manage your build system components, you sometimes need the additional capabilities that are available when you log on to the build server.



1. Log on to the build server that you want to configure.
2. From Windows Start, run Team Foundation Administration Console.

The Team Foundation Administration Console appears.

3. In the tree pane, expand the name of the server.
4. Choose the Build Configuration node.

Note

If the message  Configure Installed Features appears instead of a view such as the one above, see [Deploy a build server](#).

Next Steps

Check the status of your controllers and agents

- If any of the build controllers or build agents are processing a build, you can see messages about this ongoing work in real time from the

[Team Foundation Administration Console](#).



For more information about monitoring a running build process, see [Run, monitor, and manage builds and Diagnose problems in your build](#).

- If a  controller or  agent is red, then the component is disconnected. Make sure your build server can connect to your Team Foundation Server.

Work with your build server

Build Service configure for http://fabrikam-3:8080/tfs/defaultcollection as NETWORK SERVICE.

FABRIKAM-1

- Started on

http://fabrikam-1.northamerica.corp.fabrikam.com:9191/Build/v4.0/
Services as NETWORK SERVICE

[Restart](#) | [Stop](#) | [Properties](#) | [Unregister](#)

 Events: 6 informational in the last 24 hours

- Choose Restart, Stop, or Start to [start, stop, or restart the build server](#).
- Choose Properties to deploy and configure a build server.
- Choose Register or Unregister to [register or unregister a build server](#).
- Choose Events... to [use event logs to diagnose problems](#).
- [Monitor the health of your build server](#).

Work with your build controller

 **Default Controller - FABRIKAM-1** - Ready
Controller - [Properties](#) | [Delete](#) | [Disable](#) | [Restart](#)

- Choose Properties and see [Deploy and configure a build controller](#).
- Choose Restart to [restart the controller](#).
- Choose Enable or Disable to [enable or disable the controller](#).

Work with your build agent

 **Default Agent - FABRIKAM-1** - Ready
Agent for 'Default Controller - FABRIKAM-1' - [Properties](#) | [Delete](#) | [Disable](#) | [Restart](#)

- Choose Properties and see [Deploy and configure build agents](#).
- Choose Restart to [restart the agent](#).
- Choose Enable or Disable to [enable or disable the agent](#).

■ Start, stop, or restart a build server

It is a good idea to stop the build server whenever you are performing maintenance on it. From the

[Team Foundation Administration Console](#), choose Stop to take all build controllers and build agents hosted on the build server offline. Choose Start when you are ready to put the controllers and agents back in service.

If the build server fails to respond (for example, if a build process fails to start or make timely progress), choosing Restart might help. When you restart the build server, the Team Foundation Build Service process is closed and re-launched into a state that is nearly as fresh as if you rebooted the computer. If builds are currently being processed by the build controller or any build agents on this build server, those builds are cancelled.

Restart a build controller or build agent

When restarting the build server is too disruptive or unnecessary, you can choose the Restart link of the build controller or the build agent from the

[Team Foundation Administration Console](#). When you restart a build controller, it reloads assemblies, and all builds it is processing are aborted. When you restart a build agent, it discards the app domain, reloads assemblies, and the build that it is processing is aborted.

Enable or disable a controller or agent

You can disable a  controller or  agent to prevent builds from being assigned to it. If you disable a controller or agent that is processing a build, the build completed before the component is disabled.

This capability can help you tune or temporarily modify the way your build system works. For example, you suspect that a build server is running too many build agents because builds are being processed too slowly. You can disable some build agents and assess how this affects performance. You can then later re-enable one or more build agents as necessary.

To enable or disable a build agent or build controller:

- From the

[Visual Studio Manage Build Controllers dialog box](#), select the build controller or build agent that you want to disable or enable, and then choose Properties. Either the Build Controller Properties or Build Agent Properties dialog box appears. Clear or select either the Build Controller service is enabled or the Build Agent service is enabled check box.

- From the [Team Foundation Administration Console](#), locate the build controller or build agent that you want to disable or enable, and then chose its Enable or Disable link.

Register or unregister a build server

From the

[Team Foundation Administration Console](#), choose Unregister if you want to take the build server out of your build system. When you choose Unregister, you are asked whether you want to delete any build controllers and build agents that are running on the build server:

- Choose No if you want to temporarily decommission the build server. The controllers and agents on this build server are not deleted from your team project collection and appear with a Status of Offline in the [Visual Studio Manage Build Controllers dialog box](#). You can later choose Register and restore the build controller and build agents.
- Choose Yes if you want to delete all configuration settings for this build from the team project collection. All information about any build controller and any build agents is deleted.

If the message Build Service is not registered appears, you can choose Register to re-commission the build server and enable it to host build controllers and build agents.

Tip

You can also replace an existing build server when you deploy a new build server. For example, you might want to host the same configuration and set of build controllers and build agents on a new, more powerful computer. See [Configure Team Foundation Build Service using the Team Foundation Server configuration tool](#).

Monitor the health of your build server

While logged on to the build server, you can confirm Team Foundation Build Service is running, get information about the resources it is consuming, and confirm the general health of the build server.

1. Run Windows Task Manager (Task Manager on Windows 8).
2. On Windows 8, if the More details link appears, choose it.
3. Choose the Process tab.
4. On versions of Windows other than Windows 8, make sure Show processes from all users is selected.
5. On what version of Windows is your build server running?
 - Windows 8: Locate the Visual Studio Team Foundation Build Service Host process. It should be located in the Background processes section, or if your build server is running in interactive mode, in the Apps section. Observe the CPU, memory, disk, and network resources that the process is consuming.
 - Another Windows version: Locate the TFSBuildServiceHost.exe process. Observe the CPU and memory resources that the process is consuming.
6. Use the other tabs in Task Manager to confirm the general health of the build server. For example, you can choose the Performance tab to confirm the computer has sufficient processor and memory resources. You can then choose Resource Monitor (on Windows 8, Open Resource Monitor).

▣ Use event logs to diagnose problems

Use the Event Viewer to get information that can help you monitor how your build server is operating and to diagnose problems with your build server or your build process.

▣ View recent events

Begin by checking the most recent events. From the

[Team Foundation Administration Console](#), choose Events... to open Event Viewer to display the most recent informational, warning, and error messages.

▣ View the operational log

To get more detailed information over a longer period of time, in the Event Viewer, in the tree pane expand Applications and Services Logs, Microsoft, Team Foundation Server, Build-Services, and then choose Operational.

▣ View the analytic log

In most cases you don't need to read the internal messages exposed by the analytic log. However, this data may be useful when you are working with customer support to resolve an issue.

To enable the analytic log

1. In Event Viewer, choose View, Show Analytic and Debug Logs.
2. In the tree pane, expand Applications and Services Logs, Microsoft, Team Foundation Server, Build-Services, and then choose Analytic.
3. While Analytic is still selected, open its shortcut menu and then choose Enable.

☑ Tip

By default when this log reaches a data limit, the Event Viewer stops gathering data. To modify the limit or change this behavior, select the Analytic node, open its shortcut menu, and choose Properties.

☐Customize and view the data

To save the data in a permanent file that is easy to scan and view, in the tree pane, select one of the nodes described above, open its shortcut menu, and then choose Save All Events.... Use the Save As dialog box to save the data in the format that meets your needs.

Next Steps

Deploy and work with a build server

To use Team Foundation Build with an on-premises Team Foundation Server, you must deploy at least one build server. The server can be deployed on a physical computer or a virtual machine.

Deploy and configure a build controller

Use a build controller to perform lightweight tasks and distribute the processor-intensive work of your build process to its pool of build agents. You can host one build controller on a build server.

Deploy and configure build agents

Use a build agent to do the processor-intensive work of your build, includes getting files from version control, provisioning the workspace, compiling the code, and running tests. You host can one or more build agents on a build server.

Set up drop folders

You can prepare and then designate one or more drop folders so that your build system can deliver binaries, test results, and log files to your team.

Scale out your Team Foundation Build system

As your team and your code base grow, you can expand your build system incrementally with relative ease.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Configure and administer Lab Management

[See Also](#) [Send Feedback](#)

Visual Studio Lab Management lets you create and manage lab environments for a team project on Team Foundation Server. You can use lab environments to develop, test, or run your application. Each machine in a lab environment represents a role required for the application that you intend to develop, test, or run. For example, you might be developing a multi-tiered application that requires three roles: a desktop client, a web server, and a database server. There are two types of lab environments: standard environments and SCVMM environments. A standard environment can contain physical machines and virtual machines using any virtualization platform. An SCVMM environment can only contain virtual machines that are managed by SCVMM on the Hyper-V virtualization platform. For more information about lab environments, see [Using a Lab Environment for Your Application Lifecycle](#).

Tasks

Use the following topics to help you configure and administer Lab Management:

Tasks	Associated Topics
Setting up Lab Management for the first time: To use Lab Management to create standard environments, you need to set up a test controller. To use Lab Management to create SCVMM environments, you must first configure Hyper-V and SCVMM.	<ul style="list-style-type: none">• Configure Lab Management for SCVMM environments
Changing your Lab Management configuration: You might want to add more host groups, change the library share that is used, modify the integration service account, or add build controllers or	<ul style="list-style-type: none">• Change Existing Lab Management Configurations

test controllers after you have configured Lab Management for the first time.

Assigning users permissions to use Lab Management resources: The default Team Foundation Server security groups automatically give members certain permissions to use Lab Management resources. You might want to modify those permissions or assign permissions to custom groups or users.

- [Managing User Permissions for Lab Management](#)

Backing up and restoring your lab environments: You can back up and restore your lab environments.

- [Back up and restore TFS](#)
- [Restore Lab Management components](#)

▣ Forums

[Visual Studio Lab Management](#)

Blogs

[Visual Studio Lab Management team blog](#)

See Also

Concepts

Using a Lab Environment for Your Application Lifecycle

[Administer Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Configure Lab Management for SCVMM environments

[See Also](#) [Send Feedback](#)

To create and manage System Center Virtual Machine Manager (SCVMM) environments, you must first configure the required components for Lab Management. These components are:

- Hyper-V hosts
- SCVMM
- Team Foundation Server
- A test controller

For more information about Lab Management, see [Using a Lab Environment for Your Application Lifecycle](#).

If you are upgrading from earlier versions of Lab Management or SCVMM, see [upgrading](#).

 **Note**

Any computers that you use as Library Servers (including the SCVMM machine that has a default library server) and as Hyper-V hosts should be on a gigabit network that is connected to a common network switch. The faster network speed will help improve performance when you deploy virtual machines to hosts and when you save virtual machines to the library from your host.

Prerequisites

The Team Foundation Server administrator must set up and configure Team Foundation Server before configuring Lab Management. For more information about how to set up and configure Team Foundation Server, see the Team Foundation Server Installation Guide. After the administrator has configured Team Foundation Server and created a default team project collection, make sure that you have the following information:

- TfsMachine: The name of the machine that is running Team Foundation Server.
- domain: The Active Directory domain that is used by all machines.
- tfssvc: The account under which Team Foundation Server is running.

Note

You must use a domain user account and password for the TFSservice account to set up an SCVMM environment for lab management.

- domain\tfsadmin: The account for the Team Foundation Server administrator.

System requirements

The following table summarizes the minimum and recommended hardware and software requirements for the machines used with Lab Management. To make the requirements easier to understand, each role is broken out as if it were installed on a separate machine. For a simpler setup, you can run the HyperVHost machine, the VmmMachine and the LibraryMachine all on the same computer.

Role/System	Processor	Supported operating system	RAM (min/recommended)	Hard disk (min/recommended)
		Windows Server 2012 R2 operating system with Hyper-V 64-bit, Standard, Enterprise and Datacenter Editions		
		VMM Server 2012 R2 operating system with Hyper-V 64-bit, Standard, Enterprise and Datacenter		

		Editions		
		Hyper-V Host and Library Server 2012 R2. The Hyper-V Host and Library servers can be running on Windows Server 2012 with VMM server running on 2012 R2.		
HyperVHost	64-bit with hardware virtualization enabled	Windows Server 2008 R2 operating system with Hyper-V 64-bit, Standard, Enterprise, and Datacenter Editions	4GB / 8GB	100 GB/ 200GB
		Windows Server 2008 R2- Server Core installation, Standard, Enterprise, and		

Datacenter
Editions

Windows
Server 2008
operating
system with
Hyper-V 64-
bit, Standard,
Enterprise,
and
Datacenter
Editions with
Service
Pack 2

Hyper-V
Server 2008
R2 operating
system

Windows
Server 2012
R2 operating
system with
Hyper-V 64-
bit, Standard,
Enterprise
and
Datacenter
Editions

VMM Server
2012 R2
operating
system with
Hyper-V 64-
bit, Standard,

VmmMachine	64-bit	Enterprise and Datacenter Editions	2GB / 4GB	200GB
		Hyper-V Host and Library Server 2012 R2. The Hyper-V Host and Library servers can be running on Windows Server 2012 with VMM server running on 2012 R2.		
		Windows Server 2008 R2 operating system 64-bit Standard, Enterprise, and Datacenter Editions		
		Windows Server 2008 operating system Standard, Enterprise,		

and
Datacenter
Editions with
Service
Pack 2

Windows
Server 2012
R2 operating
system with
Hyper-V 64-
bit, Standard,
Enterprise
and
Datacenter
Editions

VMM Server
2012 R2
operating
system with
Hyper-V 64-
bit, Standard,
Enterprise
and
Datacenter
Editions

LibraryMachine

64-bit

Hyper-V
Host and
Library
Server 2012
R2. The
Hyper-V
Host and
Library
servers can
be running

2GB / 4GB

200GB /
500GB

on Windows Server 2012 with VMM server running on 2012 R2.

Windows Server 2008 R2 operating system 64-bit Standard, Enterprise, and Datacenter Editions

All operating systems that are supported by Team Foundation Server are supported by Visual Studio Lab Management, except Windows 8 clients and servers.

All operating systems that are supported by Team

TfsMachine

32-bit x86

64-bit x64

2 GB / 4GB

120GB / 230GB

ControllerMachine

32-bit x86

64-bit x64

Foundation
Server are
supported by
Visual Studio
Lab
Management,
except any
operating
systems that
are not joined
to a domain.

2GB / 4GB

120GB /
230GB

Configuration procedures

The following table details the procedures that you must follow to complete the configuration for Hyper-V, SCVMM, and Lab Management, and the permissions that are required. For a simple setup, the HyperVHost machine, the VmmMachine and the LibraryMachine can be the same machine.

Note

All the machines that are used for Lab Management must be joined either to the same domain or domains that have two-way trust between them.

Step Number	Procedure	Details	Machine Name In Procedure	Permissions Required	Step Complete
1	Set Up And Configure Hyper-V Hosts	Hyper-V lets you manage virtual machines and their resources.	HyperVHost	Administrator for the HyperVHost	
2	Install and configure SCVMM	SCVMM helps you manage your virtual machines and templates, and where and how	VmmMachine Note You can use a separate machine as the LibraryMachine to use for the library share to store the	Administrator for VmmMachine	

you store virtual machines.
them.

3

[Configure Team Foundation Server to enable Lab Management](#)

You configure TFS to give you access to the virtual machines and templates that you have created.

TfsMachine

Team Foundation Server Administrator

4

[Verify the configuration using Microsoft Test Manager](#)

Verify that you can connect to a team project and access the Lab Center by using Microsoft Test Manager.

DesktopClient

Any user account that has been added to the computer

[Install test](#)

If you want to build and deploy your application by using Team

TfsMachine

Note

It is recommended Team Project

5

[controller](#)

Foundation that you use a
Build and separate machine
run tests, as the
you will ControllerMachine
need build to install the test
controllers controller.
and test
controllers.

Collection
Administrator

■ Set up and configure Hyper-V hosts

To set up Hyper-V, you can do one of the following:

- Enable the Hyper-V role on your computer that has Windows Server 2008 R2 or Windows Server 2012 installed. If you do not have the Hyper-V role enabled on the HyperVHost computer, you must first enable this by using Server Manager on each of the Hyper-V hosts so that you can create and manage virtual machines using the steps in the following procedure. For more information about Hyper V, see the following

[Microsoft Web site](#).

- Install Hyper-V Server 2008 R2 on your HyperVHost. For information about Hyper-V server and how to install it, see the following [Microsoft Web site](#).

To enable the Hyper-V role on each Hyper-V host with Windows Server

1. On the Hyper-V host, log on as an administrator.
2. Choose Start, point to Administrative Tools, and then choose Server Manager.
3. In the left pane of Server Manager, right-choose Roles and point to Add Roles.
4. Follow the Add Roles Wizard. You can just choose Next in all except the following pages of the wizard:
 1. On the Select Server Roles page, select Hyper-V and then choose Next.
 2. On the Create Virtual Networks page, you must select an active network adapter and then choose Next.

■ **Note**

To find which network adapters are active, use the **ipconfig** command to view which network adapters have IP addresses. In the wizard, select the network adapter that should be used for creating a virtual network. You must select an adapter that is connected to the same network as the other machines in the lab. If there are several adapters, select the adapter to which the virtual machines that were created by Lab Management should be connected.

3. On the Confirm Installation Selections page, choose Install.
4. On the Installation Results page, choose Close.
5. On the Do you want to restart now dialog box, choose Yes.

The machine will restart and continue with Resume Configuration Wizard.

6. On the Installation Results page, you will see a message that states **Installation succeeded**. Choose Close.
5. Install the update described on this [Microsoft Web site](#). This update supports the network isolation capability of SCVMM environments.

■ **Improve the reliability of WinRM**

Windows Remote Management (WinRM) is used by SCVMM to communicate with Hyper-V hosts. You must make the following changes to the default configuration settings for WinRM. The changes to these settings help improve the reliability of WinRM because Lab Management performs a larger number of operations on Hyper-V hosts than are performed in a typical production scenario for Hyper-V. These changes to the WinRM settings must also be performed on any computer that is used for a library share.

To change Windows Remote Management configuration settings

1. Choose Start, open the shortcut menu for the Command Prompt, and then choose Run as administrator.

☑Note

You must be a domain user who has been added to the administrators group for this machine.

2. At the command prompt, run the following command:

```
winrm set winrm/config @{MaxTimeoutms = "1800000"}
```

3. Restart the winrm service and the SCVMM agent on the host using the following commands:

```
net stop winrm
```

```
net start winrm
```

```
net start scvmmagent
```

4. If the host OS is Windows Server 2008:

1. Run the following command:

```
winrm set winrm/config/Service  
@{MaxConcurrentOperations="200"}
```

2. Restart winrm and the SCVMM agent on the host using the following commands:

```
net stop winrm
```

```
net start winrm
```

```
net start scvmmagent
```

☐Verify the Hyper-V setup

To verify that Hyper-V is installed correctly and is working

- Create a virtual machine using Hyper-V and check that you can start the

virtual machine.

If you have enabled the Hyper-V role for your HyperVHost, create a virtual machine using the information from the following [Microsoft Web site](#).

Note

You can change the settings for the virtual machine based on your needs. You can update the hard disk size and the memory. For more information about these settings, see the following [Microsoft Web site](#).

If you are using Hyper-V Server, create a virtual machine using the information in the Hyper-V Server Getting Started Guide from the following [Microsoft Web site](#).

Note

If you have multiple hard disk drives on your Hyper-V machine, you must configure the paths for the virtual machines to be the larger disk, or you may not have sufficient disk space to create your virtual machines.

For more information about Hyper-V, see [this overview of Hyper-V](#).

Note

When you create or start a virtual machine, issues may occur due to antivirus software. For more information, see the following [Microsoft Web site](#).

■ **Install and configure SCVMM**

To access the virtual machines that you create with Hyper-V from Lab Management, you must install and configure SCVMM. SCVMM is a tool for managing your Hyper-V host machines from a central console. Lab Management communicates with SCVMM to be able to use the virtual machines and templates to create environments. Two versions of System Center Virtual Machine Manager are supported for Lab Management: SCVMM 2012 or SCVMM 2008 R2.

📌 **Note**

If you are using the stand-alone Hyper-V Server 2008 R2 operating system, you must install SCVMM on a separate VmmMachine. You cannot use the HyperVHost computer. For more information about the unique characteristics of Hyper-V Server 2008 R2, see the following

[Microsoft Web site](#)

■ **Install and configure SCVMM 2012**

For information about prerequisites, see: [System Requirements: VMM Management Server](#)

Install SCVMM server and its administrator console on the VMMMachine

1. On VmmMachine, log on as a domain user who is also a local administrator on the machine.
2. Run setup.exe.
3. In System Center 2012 setup, choose Install.

📌 **Tip**

If you do not have Microsoft .NET Framework 3.5 SP1 installed, VMM Setup prompts you to install it. Choose OK.

4. In Select features to add, choose VMM management server and then choose Next. VMM Console is automatically installed with the server.
5. Type your name, organization and product key and choose Next.
6. Choose I have read, understood, and agreed with the terms of the license agreement and choose Next.
7. Choose whether to participate in the customer experience improvement program and choose Next.
8. To accept the recommended installation location, choose Next.
9. Specify the SQL Server instance to use, as well as any particulars about it, and choose Next.
10. Specify the local system account or a domain account as the identity the VMM service will use and choose Next.
11. Choose Next to accept the default port assignments or to enter different values.
12. Choose Create a new library share and choose Next.
13. On the Installation summary screen, choose Install, and then choose Close after the wizard finishes.

After you install VMM manager, you must add your Hyper-V host machines to a host group, and optionally add more library shares.

To configure SCVMM

1. Open Virtual Machine Manager Console.
 1. Choose Start, All programs, Microsoft System Center 2012, Virtual Machine Manager, and then Virtual Machine Manager Console.

2. On the Connect to Server dialog box, type localhost:<port number>or <the name of the SCVMM Server>:<port number>, and then choose Connect.
2. To add the HyperVHost machines to a host group in SCVMM, choose Fabric, and then choose Add resources, and then choose Hyper-V Hosts and Clusters. Choose Next in all except the following pages of the wizard:
 1. On Specify the credentials to use for discovery, choose Manually enter the credentials, type the credentials of an administrator for the Hyper-V host, and choose Next.
 2. On the Specify the search scope for virtual machine host candidates wizard page, choose Specify Windows Server computers by names, in the Computer names box, type the computer name of the host that you are adding to SCVMM (for example, HyperVHost), and then choose Next.
 3. On the Select the computer that you want to add as hosts page, select the Hyper-V host computers and choose Next.
 4. On the Summary wizard page, choose Finish.

Note

It is recommended that you configure a separate host group for each team project collection. For example, if you plan to configure Lab Management in two project collections in Team Foundation Server, you can create two host groups in System Center Virtual Machine Manager.

3. (Optional) By default, SCVMM configures the machine on which it is installed to act as a library server. It also sets up a default library share on the machine. You can add more library servers or shares to improve the performance.
 1. To add a new library server to SCVMM, choose Library, and then choose the Add library server button from the ribbon and follow the instructions in the wizard.

2. To add a new library share to SCVMM, first create a Windows file share on a library server. Then choose Library, in the navigation pane, select the library server where you created the Windows file share, and choose Library Server from the menu. In the ribbon, choose Add library shares and follow the instruction in the wizard.

Note

We recommend that you configure a separate library share for each team project collection.

4. To ensure that virtual machines start quickly and that the host service and any network agents do not timeout, set the minimum amount of memory that the HyperVHost machine will always have in reserve. The default memory reserves on the host machine are often too low to accommodate the longer startup time required for virtual machines.
 1. In the SCVMM Administrator Console, choose Fabric.
 2. In the navigation pane, expand Servers, All Hosts, choose the Hyper-V host machine, and then choose the Properties button from the ribbon.
 3. In the properties dialog box, choose Reserves and then select the Override host reserves settings from the parent host group check box.
 4. In the Memory (MB) edit box, type or select 1024, and then choose OK.
 5. Repeat steps b. through d. for each host.

Install and configure SCVMM 2008 R2

To install SCVMM server and SCVMM Administrator Console on the VmmMachine

1. On VmmMachine, log on as a domain user who is also an administrator on the machine.

2. To install System Center Virtual Machine Manager, run setup.exe.
3. To install the SCVMM Server, under Setup choose VMM Server. Choose I accept the terms of this agreement. Then choose Next in all except the following pages of the wizard:

1. On the SQL Server Settings wizard page, choose Install SQL Server 2005 Express Edition SP3, and then choose Next.

Note

If you do have an existing instance of SQL that you want to use, then select Use a supported version of SQL Server.

2. On the Library Share Settings wizard page, select Create a new library share.

Note

You can accept the default names and locations for the library share or enter different values.

3. On the Installation Settings wizard page, you can either accept the default port numbers and VMM service account or enter different values.
 4. On the Summary of Settings wizard page, choose Install.
 5. On the Installation wizard page, choose Close.
4. To install the Virtual Machine Manager Administrator Console, under Setup choose VMM Administrator Console. Choose I accept the terms of this agreement. Then choose Next in all except the following pages of the wizard:
 1. On the Summary of Settings wizard page, choose Install.
 2. On the Installation wizard page, choose Close.

5. On the System Center Virtual Machine Manager 2008 startup page, choose Exit.

After you install VMM manager, you must add your Hyper-V host machine to a host group, and optionally add more library shares.

Note

A host group lets you group the Hyper-V host machines together in a meaningful way based on the purpose of the host machine. For example, a set of host machines might belong to a particular department in your company.

To configure SCVMM

1. Open Virtual Machine Manager Administrator Console.
 1. Choose Start, All programs, Microsoft System Center, Virtual Machine Manager 2008 R2, and then Virtual Machine Manager Administrator Console.
 2. On the Connect to Server dialog box, enter localhost:<port number>or <the name of the SCVMM Server>:<port number>, and then choose Connect.

Note

If you receive an error message that states that you cannot connect because the service is not running, run services.msc and verify that Virtual Machine Manager Service is running.

2. To add the HyperVHost machines to a host group in SCVMM, in the Actions pane choose Add host. Choose Next in all except the following pages of the wizard:
 1. On Select Host Location wizard page of the Add Hosts wizard, enter the credentials of an administrator for the Hyper-V host.

2. On the Select Host Servers wizard page, enter the computer name of the host that you are adding to SCVMM-for example, HyperVHost-and then choose Add.
3. On the Summary wizard page, choose Add Hosts.

Note

We recommend that you configure a separate host group for each team project collection. For example, if you plan to configure Lab Management in two project collections in Team Foundation Server, you can create two host groups in System Center Virtual Machine Manager.

3. (Optional) By default, SCVMM configures the machine on which it is installed to act as a library server. It also sets up a default library share on the machine. You can add more library servers or shares to improve the performance.
 1. To add a new library server to SCVMM, choose Add library server in the Actions pane on the right side and follow the instructions in the wizard.
 2. To add a new library share to SCVMM, first create a Windows file share on a library server. Then select the library server on the Library tab of SCVMM Administrator Console and choose Add library shares under the Actions pane on the right side.

Note

We recommend that you configure a separate library share for each team project collection.

3. On each SCVMM library server, install the update described on this [Microsoft website](#). This update supports the network isolation capability of SCVMM environments.

4. To ensure that virtual machines start quickly and that the host service and any network agents do not timeout, set the minimum amount of memory that the HyperVHost machine will always have in reserve. The default memory reserves on the host machine are often too low to accommodate the longer startup time required for virtual machines.
 1. In the SCVMM Administrator Console, choose Go, Hosts.
 2. In the center All Hosts pane, open the shortcut menu for the name of the host machine, and then choose Properties.
 3. In the Host Group Properties for host machine dialog, choose the Reserves tab.
 4. In the Memory edit box, enter or select 1024, and then choose OK. For more information about the minimum required memory for Hyper-V hosts, see this [Microsoft website](#).
 5. Repeat steps b through d for each host.

⚠Important

If you have two SCVMM servers that are both configured to be used by Lab Management with a specific Team Foundation Server, then each SCVMM server must be configured to use a unique MAC range. To configure the MAC range, from the Virtual Machine Manager Administration Console, in the lower-left navigation window choose Administration, choose Networking and then open Global Static MAC Address Range. For more information about how to configure the MAC range, see the following [Microsoft Web site](#).

To verify that SCVMM is installed correctly and is working

1. In the Virtual Machine Manager Administrator Console, in the Actions pane on the right side, choose New virtual machine. Choose Next in all except the following pages of the wizard.
 1. On the Select Source wizard page, select Create the new virtual machine with a blank virtual hard disk.

2. On the Virtual Machine Identity wizard page, in Virtual machine name, enter blank.
3. On the Select Destination wizard page, select Place the virtual machine on a host.
4. On the Select Host wizard page, view the ratings, and then choose Next.

You should see all the HyperVHost machines that you are using for Lab Management. All of them should have positive star ratings. If not, you might have an issue that must be corrected. To determine why the HyperVHost machines are not usable by SCVMM, examine the Rating Explanation.

5. On the Summary wizard page, choose Create.

A new virtual machine called blank should be created without any errors and should be visible in the Virtual Machines tab of SCVMM. There might be a warning that SCVMM could not locate the boot volume. This is expected and can be ignored.

2. In the Virtual Machine Manager Administrator Console, choose Virtual Machines, open the shortcut menu for the virtual machine displayed in the list, and then choose Start. If the virtual machine does not start, review the information located on the following sites to identify the cause of the problem:
 - [New Installation of SCVMM 2008](#)
 - [System Center Virtual Machine Manager TechCenter](#)
3. After the Status of the VM is Running, open the shortcut menu for the VM, and then choose Stop.

This confirms that your Hyper-V hosts are configured correctly.

4. To store the virtual machine in the library, open the shortcut menu for the VM, and then choose Store in library.

1. On the Select Library Server wizard page, choose one of the library shares.
2. On the Select Path wizard page, browse to the path for your library share that you plan to use for your team project collection on Team Foundation Server.
3. When you are finished, confirm that a virtual machine called blank appears in the Library pane in the SCVMM Administrator Console.

This confirms that the library share is configured correctly.

You can use the following procedure to improve the performance of virtual machine transfers from the SCVMM library share to a host and also when storing a virtual machine to the SCVMM library share. This performance improvement removes encryption that has security implications if the images are transmitted unencrypted over the network.

To improve performance of VM transfers

1. Enable unencrypted file transfers for library servers.
 1. Choose the Library tab in the SCVMM Administrator console. Open the shortcut menu for each library server in the Library Servers tree, and then choose Properties.
 2. Select Allow unencrypted file transfers.
 3. Choose OK to close the dialog box.
2. Enable unencrypted file transfers for host groups.
 1. Choose the Hosts tab in the SCVMM Administrator console. Open the shortcut menu for each host group in the Hosts tree, and then choose Properties.
 2. Select Allow unencrypted file transfers.
 3. Choose OK to close the dialog box.

■ **Configure Lab Management for Team Foundation Server**

To enable Team Foundation Server to communicate with the SCVMM server, you must also install the SCVMM Administrator Console on your TfsMachine if this is a different computer from the computer that you are using as your VmmMachine.

■ **Note**

If your deployment of Team Foundation Server has more than one application-tier machine, install the console on every application tier in the deployment.

To install SCVMM 2012 Console on <TFSMachine>

1. Log on to TfsMachine as domain\tfsadmin.
2. To install Virtual Machine Manager Administrator Console, run setup.exe for Virtual Machine Manager.
3. In VMM setup, choose Install.
4. You can just choose Next in all except the following pages of the wizard:
 1. On the Select features to install page, choose VMM console and then choose Next.
 2. On the license page, you must choose I agree with the terms of this notice and then choose Next.
 3. On the Installation summary page, choose Install.
 4. On the Setup completed successfully page, choose Close.

To install the SCVMM 2008 R2 Administrator Console on

<TfsMachine>

1. Logon to TfsMachine as domain\tfsadmin.
2. To install Virtual Machine Manager Administrator Console, run setup.exe for Virtual Machine Manager.
3. Under Setup on the System Center Virtual Machine Manager 2008 R2 startup wizard page, choose VMM Administrator Console.
4. You can just choose Next in all except the following pages of the wizard:
 1. On the Summary of Settings wizard page, choose Install.
 2. On the Installation wizard page, choose Close.

■ **Configure Lab Management for Team Foundation Server**

You must configure Lab Management in Team Foundation Server. This enables access to the virtual machines and templates that you create from Microsoft Test Manager. You can then use these virtual machines or templates to create SCVMM environments and store them in a library share. This requires you to select the SCVMM server to use with Team Foundation Server.

To configure Lab Management for Team Foundation Server

1. On TfsMachine, choose Start, All programs, Microsoft Visual Studio Team Foundation Server, and then choose Team Foundation Server Administration Console.

☑ **Note**

You must log on to TfsMachine as domain\tfsadmin.

2. In the Administration Console, in the Application Tier node, choose Lab Management.
3. Choose Configure.

The Lab Management Settings dialog box appears.

4. On the Virtual Machine Manager tab, in VMM Server Name, type the name of VmmMachine, the server that is running Virtual Machine Manager and that you will use to manage the virtual machines.
5. Choose Test to determine whether Team Foundation Server can communicate with the SCVMM server.
 1. If Team Foundation Server service account does not have the appropriate permissions on the specified SCVMM, a dialog box prompts you for a user name and password.
 2. Specify the credentials of an existing member of SCVMM Administrators Role. By using this user name and password, the Team Foundation Server service account will be added to the SCVMM Administrator Role.

If Team Foundation Server cannot contact the SCVMM server, a red x and an error message is displayed. For more information about how to troubleshoot installation issues, see [Troubleshooting Lab Management](#).

6. You can use network isolation in Visual Studio Lab Management to enable multiple copies of a lab environment to run at the same time without causing network conflicts, such as conflicts in computer names and Domain Name System (DNS) registration. For more information about network isolation, see [Creating and using a network isolated environment](#). On the Network Isolation tab, follow these steps:
 1. In IP Block, type the range of Internet Protocol (IP) addresses to be assigned to the virtual machines in an environment when an isolated network is created.

You must type the IP Block using Classless Inter-Domain Routing (CIDR) notation. The CIDR notation is constructed from the IP address, the forward slash (/), and the prefix size. For example, 192.168.0.0/24.

The addresses that you specify are used only for internal routing among virtual machines and are not exposed beyond the boundaries of

an environment. Therefore, you can specify any address range that is not used within your public network. In most cases, you can use the default range of 192.168.23.0/24.

2. In DNS Suffix, type the suffix of the domain name to be assigned to the environment when an isolated network is created.

Team Foundation Server uses the suffix that you entered when it registers a unique external name with DNS for each virtual machine in a network-isolated environment. The DNS alias record makes it possible for machines and other objects outside the isolated network to communicate with machines inside the isolated network. Because Team Foundation Server goes into the DNS zone to register the alias record, the service account under which Team Foundation runs must have permissions to add or delete alias records in the specified DNS zone.

If your Team Foundation Server deployment has more than one application tier and each application tier runs under a different service account, then each application-tier service account must have permission to edit the DNS alias records created by the other application tiers.

Choose Test to determine whether Team Foundation Server can use the suffix.

If the suffix is valid, a green check mark appears. Choose OK.

If the suffix is not valid, a red x and an error message appear. You must fix the error before you can continue.

7. If your deployment of Team Foundation Server has more than one application-tier machine, repeat the following steps on each of the other application-tier machines:

1. In the Team Foundation Administration Console, choose the Application Tier node, choose Lab Management, and then choose Reconfigure.

The Lab Management Settings dialog box appears.

2. Choose the Virtual Machine Manager tab, and then choose Test to determine whether the application tier can communicate with SCVMM.

If the Team Foundation Server service account does not have the correct permissions on the specified SCVMM, a dialog box is displayed prompting you for a user name and password. Type the credentials of an existing member of SCVMM Administrators role. When you type this user name and password and choose OK, the Team Foundation Server service account of this application tier will be added to the SCVMM Administrator role.

■ **Configure Lab Management for Each Team Project Collection**

You must configure the host group and library share from SCVMM that you want to use with each team project collection. For example, you might decide to allocate one host group to each team project collection to evenly divide the virtual machine resources.

To use the workflow capability from a Team Foundation Build build definition to deploy your application, or run tests by using a test controller on your SCVMM environments, you must add a domain user account that will be used by test agents and build agents.

📌 **Note**

You must have a team project collection already created for this step of the configuration.

To configure Lab Management for each team project collection

1. To select the library share to use to store virtual machines, templates, and SCVMM environments, choose Team Project Collections under Application Tier.

⚠️ **Important**

You must configure a library share for each team project collection that you want to configure for Lab Management.

2. In the right-side pane, choose the appropriate team project collection from the list of project collections.
3. Choose the Lab Management tab, and then choose Configure Library

Shares.

The Lab Management Settings dialog box appears.

4. On the Library Shares tab, choose Add and Verify.

The Select Library Shares dialog box appears.

5. In the Select Library Shares dialog box, choose one or more SCVMM library shares that this team project collection will use, and then choose Add.

The verify process now confirms that Team Foundation Server can connect to the library share. If the verification fails, then a log is generated. You can view the log to identify the problems and then fix them. When they are fixed, you can choose Verify to confirm that Team Foundation Server can connect to the library share.

6. To add the library share to each team project in the team project collection, select the Auto Provision check box for the library share.

Note

If you select to auto-provision the library share, Team Foundation Server automatically adds the library share to all the team projects in this team project collection. For team projects that have not yet been created, the library share is added when the New Team Project wizard creates the project. For team projects that have already been created in this team project collection, the library share is added when these settings are saved. If you clear Auto Provision the library share will no longer be added to new team projects that you add to your team project collection.

7. To select the host group to use for a team project collection, choose Host Groups.

Important

You must select the host group for each team project collection that you

want to configure for Lab Management.

8. On the Host Groups tab, choose Add and Verify.
9. In the Select Host Groups dialog box, choose one or more SCVMM host groups that this team project collection will use, and then choose Add.
10. To add the host group to each team project in the team project collection, choose the Auto Provision check box for the host group.

 Note

If you select to auto-provision the host group, Team Foundation Server automatically adds the host group to all the team projects in this team project collection. For team projects that have not yet been created, the host group is added when the New Team Project wizard creates the project. For team projects that have already been created in this team project collection, the host group is added when these settings are saved. If you clear Auto Provision, the host group will no longer be added to new team projects that you add to your team project collection.

■ Install a Test Controller

You can use Lab Management to deploy an application to your SCVMM environment and also to run tests on that environment. This requires a test controller. For more information about how to run tests using test settings and environments, see [Setting Up Test Machines to Run Tests or Collect Data](#).

To install the test controller

1. Install the test controller on ControllerMachine or on a different machine. For information about how to install and configure a test controller, see [Installing and Configuring Test Agents and Test Controllers](#).

■ Note

You must register the test controller with the team project collection that you are using for Lab Management.

2. Specify the user account to use for communication between the test agents and build agents and between the test controllers and Team Foundation Server.

⚠ Caution

The service account that you specify must be a domain account. For security reasons, this account must have limited privileges. The account that you use must not have any administrative permission or be the account that is used by any trusted service. This includes the build controller and test controller.

We recommend that you create two accounts to use for this service account. For more information about how to switch between these two accounts and how existing environments are updated with the service account, see

[How to: Configure the Lab Service Account.](#)

You must configure the user account for each team project collection that you want to configure for Lab Management.

▣ **See Also**

Concepts

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

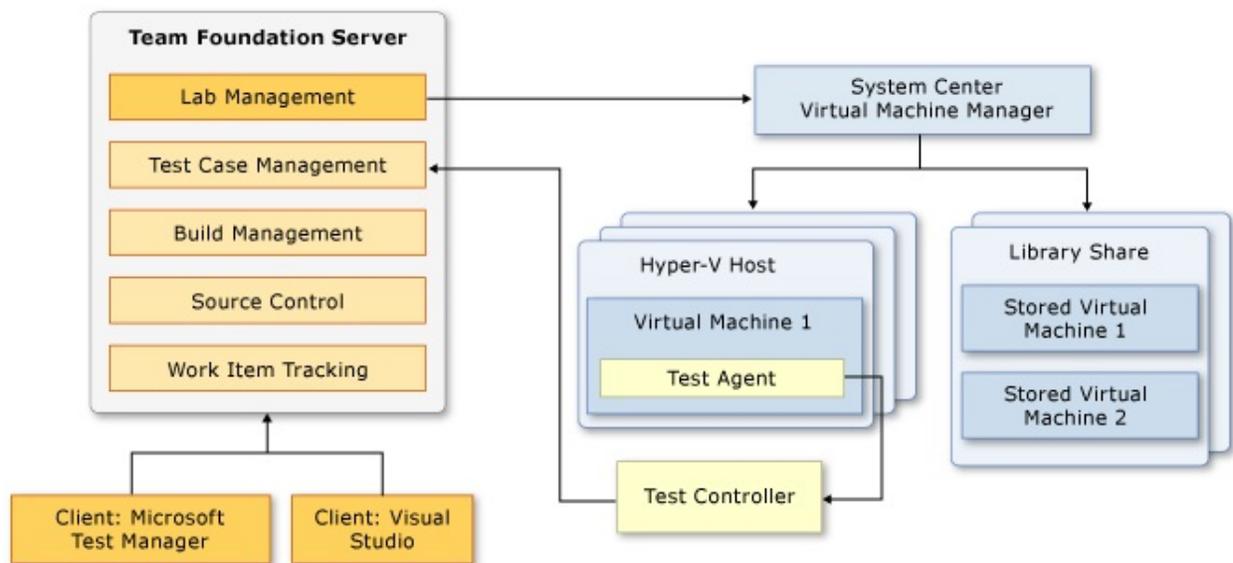
Create and store virtual machines and templates ready for Lab Management

[See Also](#) [Send Feedback](#)

If you are a system administrator, you might have to regularly create new golden virtual machines to satisfy the needs of various team projects. Because you cannot create a virtual machine (VM) from scratch using Lab Management, you must start from an existing virtual machine created in Hyper-V or System Center Virtual Machine Manager (SCVMM) and import it into a team project. After you create the virtual machines in SCVMM, you can:

- Leave the virtual machine on the host and compose an environment around it.
- Convert the virtual machine into a template and store the template in the team project library.

The following diagram illustrates how the various machines and agents interact with one another.



This topic contains instructions for creating virtual machines with all the necessary agents by using SCVMM, for converting those virtual machines into templates, and for storing the templates in the SCVMM library.

■ Prepare a Virtual Machine with a Test Agent

To create a virtual machine

1. Log in as an Administrator to the server that is running Virtual Machine Manager.
2. Open the Virtual Machine Manager Administrator Console.
3. Create a virtual machine in SCVMM by using one of the following three methods:
 1. [How to Create a Virtual Machine from an Existing Virtual Hard Disk.](#)
 2. [How to Create a Virtual Machine with a Blank Virtual Hard Disk.](#)
 3. [Creating Virtual Machines from a Template.](#)
4. Connect to the virtual machine by using the VM console in the SCVMM Administrator Console.
 1. In SCVMM, click the Virtual Machines tab, right-click the virtual machine, and then click Connect to virtual machine.
 2. Make sure that you have installed the appropriate operating system and application pre-requisites onto this virtual machine.
5. Enable remote desktop connections to this virtual machine.

This enables guest-based connections to the virtual machine. For more information about the types of connections, see [How to: Connect to machines in a Lab Environment](#).

1. On the virtual machine, click Start, right-click Computer, and then click Properties.

2. In the System dialog box, click Remote settings.
3. On the Remote tab of the System Properties dialog box, click the appropriate Allow connections option, and then click Select Users.
4. In the Remote Desktop Users dialog box, confirm the appropriate user names appear or click Add to add other user accounts, and then click OK.

You can use Lab Management to deploy an application to your environment and run tests on that environment. This is called workflow integration. For workflow integration to deploy an application, you create a build definition that includes the workflow to deploy your application to each virtual machine in your environment. You can then run tests on the role that you selected in your test settings. For more information about environments and roles, see [Setting Up Test Machines to Run Tests or Collect Data](#).

Next you will install a test agent on each virtual machine that you created.

To install a test agent on a virtual machine

- Install a test agent on your virtual machine. You can find the test agent on the same DVD from which you installed Team Foundation Server.

You do not have to register your test agent with a test controller when you install your test agent. This will be done automatically when the virtual machine is deployed by Lab Management.

If you will compose a virtual environment using this virtual machine, you are finished. If you will store the virtual machine in the library, continue to the next section.

Caution

Lab Management has very strict requirements for how a network should be configured when you create an isolated environment. Do not do any of the following:

- Set virtual LAN IDs for network adapters. The VLAN ID will be cleared

when the virtual machine is deployed. For more information about how to use VLAN IDs, see [Configuring Virtual Networks in VMM](#).

- Create multiple network adapters on a virtual machine that connect to specific networks. Lab Management overrides this configuration and creates two adapters. One adapter connects to the lab network and the other adapter handles communication among virtual machines.
- Configure the MAC address on a network adapter. Lab Management clears the MAC address at the time the network-isolated environment is created.

■ Store the Virtual Machine into the SCVMM library

When you store a virtual machine in the SCVMM library, you must decide whether to store it as a virtual machine or a template. The decision to store the virtual machine as-is or as a template will depend on several factors:

- Whether applications that are installed on the virtual machine can continue to work if the unique identifying information about the virtual machine is removed.
- Whether virtual machines will be used only as part of a network-isolated environment.

For example, if you will make multiple copies of this virtual machine for use in the same environment, that is, you will use the machine only once in an environment, then compose a virtual environment from the virtual machine without storing it to the library. If you will use the virtual machine in more than one environment on the same network, you must either convert it into a template and store it in library, or configure each environment to use network isolation. For more information about the factors to consider, see [Guidance for Creating and Managing SCVMM Environments](#).

To store a virtual machine as-is in the library

- Store the virtual machine so that it can be imported into your team project, right-click the VM and point to Store in library.
 1. In SCVMM Administrator Console, right-click the virtual machine and then click Shutdown.
 2. Select the SCVMM library share that you added to your team project collection and then click Next.
 3. Select the path for the share location for this virtual machine and then click Next.

4. Click Store.

To first convert a VM into a template and then store it in the library

1. Prepare the virtual machine to be converted into a template by clearing the administrator password and local password policy.
 1. If the virtual machine is joined to a domain, remove it from the domain and join it to a workgroup. Restart the machine when you are prompted to do this.
 2. Open gpedit.msc, and then click Local Computer Policy, Computer Configuration, Windows Settings, Security Settings, Account Policies, and then Password Policy.

Set the following values:

Enforce password history: 0

Minimum password length: 0

Password must meet complexity requirements: Disabled

3. Close gpedit.msc
 4. Change the password of local Administrator account to be empty. You can do this by logging in as .\Administrator and then using CTRL+ALT+DEL to change the password.
2. Make sure that any applications already installed on the virtual machine will continue to work properly after the name of the virtual machine is changed.

⚠Caution

If your application depends on a specific computer name or on joining a specific domain, your application might not work when a new virtual machine is deployed from the template.

3. Make sure that there is no media (DVD) loaded into the virtual machine. For more information about how to remove media, see [How to Add a DVD or CD Drive to a Virtual Machine](#).
4. Shutdown the virtual machine. In SCVMM Administrator Console, select the virtual machine, right-click it, and then click Shutdown.
5. Select the virtual machine and create a clone of it. This is recommended because the original virtual machine is destroyed when it is converted into a template.
6. Select the clone, and remove all the checkpoints on the virtual machine. This is required as SCVMM cannot convert a virtual machine with checkpoints into a template.
 1. Right-click the virtual machine, and then click Manage checkpoints.
 2. Click each checkpoint, and then click Remove.
7. If you remove the checkpoints from a virtual machine, Hyper-V merges the disks associated with the checkpoints. This could take several minutes depending on the size of the checkpoints. You can verify the progress of this merging using Hyper-V manager on the host on which the virtual machine is deployed. Wait for the merging to be complete before moving to the next step.
8. In SCVMM Administrator Console, right-click the virtual machine, and then click New template.
 1. Accept the defaults on all the screens. When prompted for a library share location, select the SCVMM library share.
 2. Click Create to start the SCVMM job. If the SCVMM job is not completed successfully, review the error message and take the appropriate actions.

See Also

Concepts

[Configure Lab Management for SCVMM environments](#)

Creating Lab Environments

Creating and using a network isolated environment



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Prepare a domain controller virtual machine using SCVMM

[See Also](#) [Send Feedback](#)

A virtual environment that uses network isolation requires that the virtual machines in the environment must be either in a workgroup or joined to a private domain that is served by a domain controller (DC) virtual machine within the environment. Among the steps to set up an environment that uses network isolation is to prepare a domain controller virtual machine. For more information about the other steps, see [Creating and using a network isolated environment](#).

Required Permissions

To perform these procedures, you must be a member of the following groups:

- Either the Administrator user role or the Self-Service User role in System Center Virtual Machine Manager (SCVMM).

For more information about SCVMM roles, see [Managing User Roles](#).

To prepare a domain controller virtual machine

1. Use SCVMM Administrator Console to create a new virtual machine with the Windows Server 2008 R2 operating system.
 1. We recommend that you use a new installation of the operating system instead of reusing or customizing an existing image.
 2. Make sure that you allocate at least 1GB of memory your virtual machine.
2. Install Active Directory and DNS server roles in the virtual machine. For more information about how to install Active Directory on Windows Server

2008 R2, see [AD DS Installation and Removal Step-by-Step Guide](#).

1. Lab Management does not support use of separate virtual machines in an environment for Active Directory and DNS.
 2. Make sure that the domain controller is the only DC in the forest and that it is the root of the forest.
 3. Do not use a domain name that is already used in your corporate or lab network. For example, if one of the domains in your network is called test.contoso.com, do not use the same name for your private domain.
3. Install the test agent on the domain controller virtual machine. For more information, see [Installing and Configuring Test Agents and Test Controllers](#).
 4. Verify that the Active Directory and DNS are installed correctly.
 5. If the virtual machine was ever connected to a network, make sure that the only DNS entries are for the virtual machine. You only have to perform these steps one time: after you install Active Directory and DNS and before you store the virtual machine into library. Perform these steps and then immediately shut down the virtual machine. If you restart the virtual machine after you perform these steps, or if you leave the virtual machine running for a long time after you perform these steps, you must repeat them.
 1. Open the DNS tool by clicking Start, Administrative Tools, and then DNS.
 2. In the DNS tool, navigate to Forward Lookup Zones. Select each zone that is listed under Forward Lookup Zones. Then delete all the DNS entries under that zone that have an IP address listed in the Data column.
 3. In the DNS tool, navigate to Reverse Lookup Zones. If any reverse lookup zone is configured, delete all the DNS entries under that zone.
 4. Close the DNS tool.

6. Shut down the virtual machine, and store it in the SCVMM library.
 1. Do not turn off the Active Directory VM. You have to shut it down correctly.
 2. Do not generalize the Active Directory VM, either by running Sysprep or by storing the virtual machine as a template in SCVMM.

See Also

Concepts

Creating and using a network isolated environment

[Configure Lab Management for SCVMM environments](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Change Existing Lab Management Configurations

[See Also](#) [Send Feedback](#)

After you have configured Lab Management, you might want to change the resources that you are using for a team project collection. You might want to add more resources as you create more virtual machines, templates, and environments, or change the integration service account, or add a build controller or test controller.

Tasks

Use the following topics to help you change existing Lab Management configurations:

Tasks	Associated Topics
Change the host groups for your team project collection: You can add or remove host groups from your team project collection by using the Administration Console for Team Foundation, or you can use the command-line utility TFSLabConfig.	<ul style="list-style-type: none">• How to: Change the Host Groups for Your Team Project Collections
Change the library shares for your team project collection: You might want to add another library share to provide more storage for your virtual environments, templates, or virtual	<ul style="list-style-type: none">• How to: Change the Library Share for Your Team Project Collections

machines.

Change the integration service account that provides communication between the agents and controllers: We recommend that you change this user account regularly for security reasons.

- [Setting Up Test Controllers in Lab Environments](#)

Change Active Directory domain to which the SCVMM server belongs.

- [How to: Join the SCVMM Server to another Domain](#)

See Also

Concepts

[Configure Lab Management for SCVMM environments](#)

[Configure and administer Lab Management](#)

[Administer Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

How to: Change the Host Groups for Your Team Project Collections

[See Also](#) [Send Feedback](#)

You can use the host groups in System Center Virtual Machine Manager as a convenient way to monitor and manage virtual machine hosts. Hosts are the computers in the host group that you use to deploy virtual environments in Visual Studio Lab Management. You create and manage host groups by using SCVMM. In Lab Management, you can specify which host groups to use for a team project collection.

If you want to add a host to an existing host group, you must use SCVMM to accomplish this task. For more information, see this [Microsoft Web site](#). If you do add a host, you must also verify the following:

- The logical network of the host group is the same as the other hosts so that the virtual machines can communicate with each other.
- The host has been added to the correct host group.

After you have configured Lab Management for the first time, you might later want to change the host groups that are used by your team project collection. For example, you need more hosts because the volume of use has increased. You can choose to add another host group that you can use to your existing team project collection to increase your capacity. You can add multiple host groups to any team project collection.

Note

To configure Lab Management for the first time, see [Configure Lab Management for SCVMM environments](#).

You can change the host groups for a team project collection using the Administration Console for Team Foundation. Or, you can use the command-line to accomplish the same task. For more information about the command-line utility TFSLabConfig, see [Configure Lab Management with TFSLabConfig](#).

Note

If you want to assign a host group that you have created using SCVMM to a specific team project that you have added to your team project collection, you must use the command-line utility to accomplish this task. You cannot use the administration console to assign host groups to specific team projects. For more information, see [TFSLabConfig CreateTeamProjectHostGroup Command](#).

Use the following procedure to change the host groups for your team project collection by using the Administration Console for Team Foundation. For each host in a host group that you add, you must also run the steps in the procedure to improve the reliability of WinRM. To expand your virtual lab, see [Expanding or Changing Your Virtual Lab](#).

To add or remove host groups for your team project collection

1. In the Team Foundation Administration Console, under Application Tier, click Team Project Collections.
2. In the right-side pane, click the appropriate team project collection from the list of project collections.
3. Click the Lab Management tab, and then click Configure Host Groups.

The Edit Project Collection Level Lab Management Settings dialog box is displayed.

4. To add a host group, on the Host Groups tab, click Add and Verify.
 1. In the Select Host Groups dialog box, click one or more SCVMM host groups that this team project collection will use, and then click Add.
 2. To add the host group to each team project in the team project

collection, select Auto Provision for the host group.

Note

If you select to auto provision the host group, Team Foundation Server automatically adds the host group to all the team projects in this team project collection. For team projects that have not yet been created, the host group is added when the New Team Project wizard creates the project. For team projects that have already been created in this team project collection, the host group is added when these settings are saved. If you clear Auto Provision the host group will no longer be added to new team projects that you add to your team project collection.

5. To delete a host group from your team project collection, on the Host Groups tab, select the host group from the list, and click Delete.

Important

Before you can delete a host group from a team project collection, you must first remove all Lab Management environments that are in this host group using Microsoft Test Manager. Then you must remove the association of this host group from each team project in that project collection using the command line utility TFSLabConfig. For more information about how to do this, see [TFSLabConfig DeleteTeamProjectHostGroup Command](#).

6. Click OK.

■ Improve the Reliability of WinRM

You must make following changes to the default configuration settings for Windows Remote Management for each host in a host group that you have added. The changes to these settings help improve the reliability of WinRM because Lab Management performs a larger number of operations on Hyper-V hosts than are performed in a typical production scenario for Hyper-V.

To change Windows Remote Management configuration settings

1. Click Start, right-click Command Prompt, and then click Run as administrator.

■ Note

You must be a domain user that has been added to the administrators group for this machine.

2. At the command prompt, run the following command:

```
winrm set winrm/config @{MaxTimeoutms = "1800000"}
```

3. If the host operating system is Windows Server 2008 SP2, also run the following command:

```
winrm set winrm/config/Service @{MaxConcurrentOperations="200"}  
-r:http://<HyperVHost>
```

(Replace HyperVHost with the computer name of the Hyper-V host.)

■ Note

To create a WinRM listener on http://<HyperVHost>, you might also have to run the following command:

```
winrm quickconfig
```

4. Restart winrm and the SCVMM agent on the hosts after you make these changes using the following commands:

```
net stop winrm
```

```
net start winrm
```

```
net start vmmagent
```

See Also

Concepts

[Manage team project collections](#)

Other Resources

[Change Existing Lab Management Configurations](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

How to: Change the Library Share for Your Team Project Collections

[See Also](#) [Send Feedback](#)

You can use the library shares in the System Center Virtual Machine Manager (SCVMM) library as a place to create and store the virtual machines, templates, and environments in your team projects. You use SCVMM to create and manage the library shares and to specify on which computer a library share is located.

After you have configured Lab Management for the first time, you might later want to change the library shares that are used by your team project collection.

Note

To configure Lab Management for the first time, see [Configure Lab Management for SCVMM environments](#).

Using SCVMM, you can create multiple library shares on a single library server. The library shares enable you to allocate the disk space on your library server. If you do this you can allocate disk space for each team project collection. You can add multiple library shares to each team project collection. For example, you can add an extra library share from a different library server to a specific team project collection to provide more disk space for the environments, virtual machines, and templates for the teams that use that team project collection. If you are maximizing the network bandwidth to a specific library server for your team project collection, you may want to add a library share from another library server. Or, you might want to add a library share that has been added to a library server in a different physical location.

Note

For improved performance, it is better to locate library shares and host groups

in the same physical location. For example, if you have a distributed team with half your team located in one country and the other half in another country, then it is more efficient to have library shares and host groups that are physically located in each of those countries and have the team members use the library shares that are located together in their country. You can change the library shares by using the Administration Console for Team Foundation. Or, you can use the command-line to accomplish the same task. For more information about the command-line utility TFSLabConfig, see [Configure Lab Management with TFSLabConfig](#).

Use the following procedure to change the library shares for your team project collection using the Administration Console for Team Foundation. For each library share that you add that is located on a different library server, you must also run the steps in the procedure to improve the reliability of WinRM. To expand your virtual lab, see [Expanding or Changing Your Virtual Lab](#).

To add or remove library shares for a team project collection

1. In the Team Foundation Administration Console, under Application Tier, click Team Project Collections.
2. In the right-side pane, click the appropriate team project collection from the list of project collections.
3. Click the Lab Management tab, and then click Configure Library Shares.

Note

You must create a library share using SCVMM before you can add this to be used by your team project collection.

4. To add a library share, on the Library Shares tab, click Add and Verify.

The Select Library Shares dialog box is displayed.

1. Click one or more SCVMM library shares that this team project collection will use, and then click Add.

2. To add the library share to each team project in the team project collection, select Auto Provision for the library share.

 **Note**

If you select to auto provision the library share, Team Foundation Server automatically adds the library share to all the team projects in this team project collection. For team projects that have not yet been created, the library share is added when the New Team Project wizard creates the project. For team projects that have already been created in this team project collection, the library share is added when these settings are saved. If you clear Auto Provision the library share will no longer be added to new team projects that you add to your team project collection.

5. To delete a library share, on the Library Shares tab, select the library share from the list, and then click Delete.

 **Important**

Before you can delete a library share from a team project collection, you must first remove all Lab Management environments, virtual machines and templates that are stored on this library share using Microsoft Test Manager. Then you must remove the association of this library share from each team project in that project collection by using the command-line utility TFSLabConfig. For more information about how to do this, see [TFSLabConfig DeleteTeamProjectLibraryShare Command](#).

6. To confirm that Visual Studio Team Foundation Server can connect to the library share, click Verify.
7. Click OK.

■ Improve the Reliability of WinRM

You must make following changes to the default configuration settings for Windows Remote Management for each library server that has library shares that you use for Lab Management. The changes to these settings help improve the reliability of WinRM because Lab Management performs a larger number of operations on Hyper-V hosts than are performed in a typical production scenario for Hyper-V.

To change Windows Remote Management configuration settings

1. Click Start, right-click Command Prompt, and then click Run as administrator.

■ Note

You must be a domain user who has been added to the administrators group for this machine.

2. At the command prompt, run the following command:

```
winrm set winrm/config @{MaxTimeoutms = "1800000"}
```

3. If the host operating system is Windows Server 2008 SP2, also run the following command:

```
winrm set winrm/config/Service @{MaxConcurrentOperations="200"}  
-r:http://<LibraryServerMachine>
```

(Replace LibraryServerMachine with the computer name of the library server.)

■ Note

To create a WinRM listener on `http://<LibraryServerMachine>`, you might also have to run the following command:

winrm quickconfig

4. Restart winrm and the SCVMM agent on the hosts after you make these changes using the following commands:

net stop winrm

net start winrm

net start vmmagent

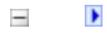
See Also

Concepts

[Manage team project collections](#)

Other Resources

[Change Existing Lab Management Configurations](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

How to: Configure the Lab Service Account

[Send Feedback](#)

This topic has been superseded by [Setting Up Test Controllers in Lab Environments](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Expanding or Changing Your Virtual Lab

[See Also](#) [Send Feedback](#)

As you use and become familiar with a virtual lab, you might want to either expand or change your lab in one or more ways:

- [Move Team Foundation Server from One Server to Another](#)

You might have to change or replace the server that is running the Team Foundation Server application-tier.

- [Move a Team Project Collection from One Team Foundation Server to Another](#)

You might want to reuse the machines and other resources in your initial team project and move those resources.

- [Add a New Library Server and Retire an Old Library Server](#)

You might find you need more space to store virtual machines, templates, and environments. You can add more library shares or replace existing library shares with ones on larger SCVMM library servers.

- [Add a New Hyper-V Host and Retire an Old Hyper-V Host](#)

You might find you need more space to run virtual machines and environments. You can add more physical hosts to existing host groups to transparently add more capacity.

- [Move System Center Virtual Machine Manager from One Server to Another](#)

You might have to change or replace the server that is running SCVMM.

- [Move a Test Controller](#)

You might have to move or replace the machine that is running the test controller.

Each of these changes requires some modification to the configuration of Lab Management resources.

Move Team Foundation Server from One Server to Another

If you move Team Foundation Server from one server to another, or otherwise change the name of the server that is running Team Foundation Server, you must update the URL that the Lab Management components use to communicate with Team Foundation Server. For more information about how to move Team Foundation Server, see

[Administer Team Foundation Server](#).

After you have finished moving Team Foundation Server, update the Team Foundation Server URL in the following locations:

- On the machine that is running the Team Foundation Server application-tier, the URL appears in the Team Foundation Server Administration Console. In the Administration Console, go to the Lab Management node, click Reconfigure Lab Management, and then click the Advanced tab. In the Lab URL text box, type the new server name.
- On machines where a build agent is installed, the URL is stored in the Hyper-V-managed registry hive. If the server that is running Team Foundation Server is renamed, click Repair Workflow Capability for this environment. The setting will be updated automatically in all the virtual machines of this environment. Because .lvr files store the URL of Team Foundation Server internally, you must modify the .lvr file manually to update the XML.

Note

You do not have to revert snapshots, because the URL is updated in the virtual machines by Team Foundation Server after it reverts to a snapshot.

Move a Team Project Collection from One Team Foundation Server to Another

If you started to explore Lab Management by using composed environments or by creating a small team project to pilot test Team Foundation Server, you might want to migrate the lab assets to a different Team Foundation Server instance later. Because Team Foundation Server does not support merging one team project with another, you must move your lab resources manually. For more information about how to migrate source code, work items, and build definitions, see

[Administer Team Foundation Server.](#)

Note

You can move virtual machines only within the same network location because they are already configured to use that network location. If you move a team project collection between instances of Team Foundation Server that are located in different domains, you must move the project collection, un-configure Lab Management for that project collection, and then configure it from fresh using the settings of the new Team Foundation Server instance and domain.

Because you are moving between domains, you cannot recover everything.

To move from one server that is running Team Foundation Server to another server that is running Team Foundation Server

1. Move the team project collection to a new application tier. For the steps see [Move a team project collection.](#)
2. Un-configure Lab Management for the team project collection after attaching the collection to the destination Team Foundation Server. For the steps to un-configure, see [TFSSConfig Lab /Delete Command.](#)

3. Reconfigure Lab Management for the collection in the destination Team Foundation Server. For the steps to configure Lab Management, see [Configure Lab Management for SCVMM environments](#).

■ Add a New Library Server and Retire an Old Library Server

If you must expand the storage for System Center Virtual Machine Manager (SCVMM), you can add new library shares. You can also replace existing shares with shares on servers that have faster processors, more hard disks, or both. The process to add or remove library shares to the SCVMM used by Lab Management is as follows:

1. Prepare to move to a new library share.
2. Move the templates.
3. Move the imported virtual machines that are currently not used in virtual environments.
4. Move the stored environments.
5. Remove the old library share.

■ Note

This scenario will not update existing .lvr files that may be part of active bugs. If there are stored environments that are required for reproducing bugs, those should remain in the source library until the bugs are resolved.

To prepare for the move

1. Make sure that the new library server meets the system requirements for a SCVMM library server. For more information about the system requirements, see

[System Requirements: VMM Library Server.](#)

- On the new server, add a new library share:

1. In Windows Explorer, create a new folder that uses the name of the new library share.
 2. Share the new folder with the SCVMM server by adding VMM_MACHINE\$ as a contributor to the list of users this folder is shared with.
 3. In the Virtual Machine Manager Administrators Console, associate this library share.
- Add the new library share to the relevant team project collections and team projects.
 1. In the TFS Administration Console, click TFS Server. Click Application Tier, and then click Team Project Collections.
 2. On the Lab Management tab, click Configure library shares.
 3. Add the share that you create in SCVMM.
 - Back up the original library share. For the steps to back up the library share, see Backing Up and Restoring the VMM Database.
 - Do one of the following:

Take the project collections offline. It is recommended, although not required, to take the relevant project collections offline during the move.

-or-

Contact your system administrator and ask them to notify all users not to deploy environments or import virtual machines or templates during the time of the move.

To move templates

1. Identify the files that are related to the template that you want to move. Move those files from the source to the destination. In the case of differencing-disk VHDs, make sure that you move all the versions of the .vhd. To find the path of the .vhd that this template points to, right-click the

template and then click Properties. Click Hardware Configuration. Move this .vhd to the new library share.

2. In the SCVMM Administrators Console, click Library. In the Action pane, click Refresh. Refresh both the old library share and the new library share.
3. In Microsoft Test Manager, click Library. Point to the new Library Share, and re-import the templates. Repeat for each team project that is enabled for Lab Management. Make sure that the OS profile and the hardware profile that you define to the new instance of the template are the same as the one that you used for the old template.
4. If you have environments in the library that are based on templates, you should update the environment definitions manually. In Microsoft Test Manager, click Library. Click Environments and then click each environment that used the templates. Remove the old template and add the new template.
5. Delete the old templates from the library. In Microsoft Test Manager, click the Lab Center. Click the Library tab, and then click VMs and Templates.

You can distinguish between the old and the new templates by looking at the library share column. This does not delete the underlying template.

6. Confirm the move was successful. If you did not take the team project collection offline, deploy one the templates to a host. It should complete without errors.

If there are virtual machines in a library that are currently not used in an environment, you must deploy those virtual machines to a host and then save those virtual machines to the new library.

To move imported virtual machines

1. Create a new virtual environment
2. Add all the virtual machines that must be moved. Do not select any capabilities.
3. Deploy the environment, but do not start it.

Note

If you try to start the environment, and there are running virtual machines that are using the same name of the virtual machines you are now moving, name conflicts in the domain might occur.

4. In Microsoft Test Manager, click the first virtual machine in the environment. Click Store to Library and then select the target library you want to store to. Repeat for all the virtual machines in the environment.
5. After all the virtual machines are stored in the library, delete the environment
6. Delete the virtual machines in the source library.

Note

After the move, you will not be able to import the virtual machines into other team projects.

To move stored environments

1. In Microsoft Test Manager, deploy the environment.

You do not have to start the environment.

2. On the Environment tab, click Store to library on the destination library share.
3. Confirm the move was successful:
 1. Deploy the environment.
 2. Take a snapshot.
 3. Revert to an earlier snapshot.
 4. Revert to the latest snapshot.

4. If you can successfully revert, delete the environment from the source library share.

After the move is complete, place the team project collection back online.

If you have bugs that included snapshots for environments that are in the library, we recommend that you keep the environments in this library until those bugs are resolved. There is no easy way to find which environments has .lvr files pointing to them. Therefore, a practical solution will be to keep environments that were created in the last 60 days.

To remove the share

1. Delete virtual machines and templates from the original library share: In Microsoft Test Manager, click Lab Center. Click Library, Virtual Machines and Templates and then delete the virtual machines in the library share that are still shown at located in the original library share.

You can easily group the virtual machines by library share, by dragging the Library Share column header and dropping it above the header row.

2. Deleting library share association from team projects:

1. Open a command prompt.

At the command prompt, use the **TFSLabConfig** command line tool for each project that is associated with the library share that you want to delete.

2. Run the command **TFSLabConfig DeleteTeamProjectLibraryShare**, to specify the project collection and project name, in addition to the library share name.

For more information about the **TFSLabConfig DeleteTeamProjectLibraryShare** command, see [TFSLabConfig DeleteTeamProjectLibraryShare Command](#).

3. Repeat this step for each team project.

3. In the Team Foundation Server Administration Console, delete the original

library share from the team project collections that it was associated with.

■ Add a New Hyper-V Host and Retire an Old Hyper-V Host

To add a Hyper-V host

1. In the SCVMM Administrators Console, add the new Hyper-V server to the host group. For the steps to add a server to the host group, see

[Adding Hosts.](#)

- In the Team Foundation Server Administration Console, click TFS Server. Click Lab Management and then click Host Groups. Click Verify. If you are prompted for a username and password, provide a user who is a member of the Administrators security group on the new host.
- Prevent placement of new virtual machines on the host that you are migrating from.
 1. In the SCVMM Administrators Console, click Hosts. Click the name of the host that you are migrating from, and then click Properties.
 2. In the Host Properties window, click the Status tab. Clear the check box This host is available for placement, and then click OK.
- In the Host Properties window, click the Status tab. Clear the check box labeled This host is available for placement and then click OK.
- Migrate environments that are not network-isolated:
 1. In Microsoft Test Manager, click Lab Center. Click the Lab tab and then click Environments. Identify the environments that reside in the host group in which the source and destination hosts are.
 2. Open each environment and identify which of them have the Network Isolation capability off. The rest of this step applies only to those environments that are not network-isolated.

3. In SCVMM Administrators Console, find the virtual machines that are part of the environments that are not network-isolated.

You can do this by typing the environment name in the search box of the Virtual Machines pane.

4. If any of the virtual machines that were found in the previous step reside on the host that is being discontinued, move those virtual machines to the new host.

- Migrate environments that are network-isolated:

1. In Microsoft Test Manager, click Lab Center. Next, click the Lab tab and then click Environments. Identify the environments that reside in the host group where the source and destination hosts are.

2. Open each environment and identify which of them have the Network isolation capability on.

3. In SCVMM Administrators Console, find the virtual machines that are part of network-isolated environments.

You can do this by typing the environment name in the search box of the Virtual Machines pane.

4. If the virtual machines of any network-isolated environment reside on the host being discontinued, store the network-isolated environment in the team project library.

All virtual machines within a network-isolated environment are located on the same physical host.

5. Deploy the environment from the library back to the host group.

- In Microsoft Test Manager, pause and then resume each environment. This verifies that the environment was successfully migrated.

- In the SCVMM Administrators Console, click Hosts, click the name of the old host, and then click Remove host.

Move System Center Virtual Machine Manager from One Server to Another

There are at least two situations in which you might move SCVMM from one server to another. First, you began to use Lab Management on a pilot basis on temporary machines and now want to deploy the functionality to production servers. Because you are starting over in a production mode, it is not important to retain the environments, templates, and other lab assets you had been using during pilot testing. Second, you have to change the server that is running SCVMM because of capacity or availability issues. Because you want to continue operating as before, you have to preserve the same environments and related assets.

To move SCVMM without retaining lab assets

1. Delete the lab objects from each team project collection.
 1. On the Team Foundation Server application tier, open a command prompt and type the following command **Tfsconfig.exe lab /delete /collectionName:myCollection**. You must run **TFSCONFIG** from a server that is running the Team Foundation application tier. By default, **TFSCONFIG** is located in Drive:\Program Files\Microsoft Team Foundation Server 2010\Tools. For more information about the **TfsConfig Lab /Delete** command, see

[TFSCONFIG Lab /Delete Command](#).

- Repeat this command for each team project collection on the server.
- Enter the new name in Team Foundation Server Administration Console.
 1. In Team Foundation Server Administration Console, click the Lab Management node.

2. On the Lab Management page, click Reconfigure Lab Management.
3. On the Virtual Machine Manager tab, type the fully-qualified domain name of the new server that is running SCVMM. Then click Test to confirm that the new server is available.
 1. If the Team Foundation Server service account does not have the appropriate permissions on the specified SCVMM, a dialog box prompts you for a user name and a password.
 2. Type the name and the password of a user who is an existing member of the SCVMM Administrator role. The Team Foundation Server service account will be added to the SCVMM Administrator role.
4. Click OK.

To move SCVMM while retaining lab assets

1. Back up the database of the original SCVMM:
 1. In the SCVMM Administrators Console, click Administration and then click General. In the Actions pane, click Back up Virtual Machine Manager.
 2. In the Virtual Machine Manager Backup dialog box, type the path of a destination folder for the backup file. The folder must not be a root directory and must be accessed by SQL Server.

Note

You can follow the status of the backup in Jobs view.

2. Copy the database backup files from step 1b to a location on the new SCVMM machine.
3. Install SCVMM on the new server. Either point the wizard to the current SCVMM database or create a new database. For the steps to restore the backup, see [Backing Up and Restoring the VMM Database](#).

4. Update the SCVMM server name in Team Foundation Server.

1. Open a command prompt and run the command: **Tfsconfig.exe lab /settings /scvmmservername:my_new_scvmmservername /force**. You must run **TFSConfig** from a server that is running the Team Foundation application tier. By default, **TFSConfig** is located in Drive:\Program Files\Microsoft Team Foundation Server 2010\Tools. You must use the **/force** switch because the library server and host groups are already bound to a team project collection; changing the SCVMM is not allowed unless you use the **/force** option. For more information about the **TFSConfig Lab Settings** command, see [TFSConfig Lab /Settings Commands](#).
2. In the Team Foundation Server Administration Console, click the Lab Management node. Click Reconfigure Lab Management and then click Test near VMM Server name. This confirms that Team Foundation Server can connect to the new SCVMM and has the correct permissions.
 1. If the Team Foundation Server service account does not have the appropriate permissions on the specified SCVMM, a dialog box prompts you for a user name and a password.
 2. Type the name and the password of a user who is an existing member of the SCVMM Administrator role. The Team Foundation Server service account will be added to the SCVMM Administrator role.
3. In the Team Foundation Server Administration Console, do the following for each team project collection.
 1. Click the Lab Management tab and then click Configure Host Groups. Select all host groups, and then click Verify. This confirms that Team Foundation Server and the new SCVMM can connect to the hosts and have the correct permissions.
 2. Click , Lab Management, and then click Configure Library shares. Select all library shares and then click Verify. This confirms that Team Foundation Server and the new SCVMM can connect to the library shares and have the correct permissions.

5. Confirm the move was successful:

1. In Microsoft Test Manager, click Lab Center and then Environments. Confirm that your existing environments appear in the list and that they are in the correct state.
2. Click Library and then VMs and Templates. Confirm that all the existing virtual machines and templates appear in the list and that they do not display new errors.

Move a Test Controller

If you must move or reinstall a test controller that is associated with environments, possibly for a hardware upgrade or because the current server is becoming too old, you must back up the database and reinstall on the new machine. If you are moving a test controller used for load tests, you back up and restore the load test database additionally. You will also have to go to each agent and reregister it with the new controller.

To move or reinstall a test controller

1. Locate the QTControllerConfig.XML file and make a backup copy.

This file is usually located in your Visual Studio installation in the Common7\IDE folder.

2. Rebuild the machine with the same name.
3. Reinstall the test controller on the new machine.

Do not configure the Test Controller yet.

4. Replace the default QTControllerConfig.XML file on the machine with the file that you backed up in step 1.
5. Configure the Test Controller and connect it to the Project Collection it was connected before.
6. Restart the machine.
7. If you change the name of a machine where a test controller is installed and the testing capability is configured for an environment, you must update the name of the test controller in the environments using that test controller. You will use Microsoft Test Manager to modify the environment:

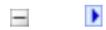
1. Stop the environment.

2. Edit the environment, and update the Test Controller setting in the Capabilities page. Select the Test Controller using its new name.
3. Restart the environment.

See Also

Tasks

[How to: Change the Library Share for Your Team Project Collections](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

How to: Join the SCVMM Server to another Domain

[Send Feedback](#)

Occasionally, it might be necessary to change your Team Foundation Server from one active directory domain to another, or from a workgroup to a domain. For more information, see [Move Team Foundation Server from one environment to another](#). As part of this task, you should move SCVMM and Lab Management to the new domain.

In order to move an SCVMM server to a domain, or change its domain, you must re-install SCVMM. Therefore, you have to save the virtual machines and templates, and then re-define the environments in the new installation. This topic describes this procedure in more detail.

Uninstall SCVMM

1. Remove the hosts and library servers from the SCVMM server.

1. Open the Virtual Machine Manager Administrator Console:

Choose Start, Microsoft System Center, Virtual Machine Manager, Virtual Machine Manager Administrator Console.

2. Delete the library servers.

3. Delete the host groups.

2. Delete all lab artifacts from Microsoft Test Manager Library Center:

In the Lab tab, delete the environments. In the Library tab, delete the virtual machines and the environments.

Because the hosts and library servers are no longer associated with

SCVMM, removing these items from Microsoft Test Manager does not delete the actual virtual machines and VHDs.

3. Unconfigure SCVMM from Team Foundation Server.
 1. Log on to the Team Foundation Server as tfsadmin.
 2. Open Team Foundation Administrator Console.
 3. In the Application Tier node, choose Lab Management, then choose Reconfigure Lab Management.
 4. Delete the links between the library shares and the team project collection.
4. Uninstall SCVMM from the computer on which it was installed.

Join the computers to domain

1. Change the domain of the hosts and library servers.
2. Change the domain of the SCVMM server computer.

Re-install SCVMM and rebuild artifacts

1. Re-install SCVMM on its computer.
2. Add the hosts and library servers back to SCVMM.

For more information, see the SCVMM section in [Configure Lab Management for SCVMM environments](#).

3. If any of the virtual machines has to be joined to the new domain, do so.

Network isolated environments have their own domain or workgroup. Therefore, their constituent machines do not have to be altered.

4. Configure the new SCVMM to couple it to your team project collection.

Set the new lab service account in Team Foundation Server.

For more information, see the SCVMM section in [Configure Lab Management for SCVMM environments](#).

5. Using Lab Management, re-import the virtual machines and templates into the Library.

For more information, see [How to: Import a Virtual Machine or Template from SCVMM](#).

6. Compose virtual machines into environments in Microsoft Test Manager.

For more information, see [How to: Store an SCVMM Environment](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Upgrading SCVMM 2008 R2 to SCVMM 2012

[See Also](#) [Send Feedback](#)

Lab Management for Visual Studio Team Foundation Server 2012 supports SCVMM 2008 R2 and SCVMM 2012. If you are upgrading Team Foundation Server 2010 to Visual Studio Team Foundation Server 2012, and plan to upgrade SCVMM 2008 R2 to SCVMM 2012, we recommend that you upgrade to SCVMM 2012 after you complete your upgrade to Visual Studio Team Foundation Server 2012. This topic describes how to upgrade SCVMM 2008 R2 to SCVMM 2012 when you are using Lab Management on Visual Studio Team Foundation Server 2012.

Important

When you upgrade SCVMM, certain steps will cause some downtime for your Team Foundation Server. Those steps are indicated below.

▣ Upgrading to SCVMM 2012



1. Use the SCVMM 2012 installer to upgrade SCVMM 2008 R2 Server to SCVMM 2012 Server.
2. Upgrade the SCVMM agents on your hosts and library shares.
3. Use the SCVMM Administration Console to verify that all of your SCVMM components are working.
4. Install SCVMM 2012 Administration Console on all machines of the application tier of your Team Foundation Server.

⚠Caution

This step will disrupt the services on your Team Foundation Server.

5. Use the **iisreset** command to restart the Team Foundation Server web service. Then restart the Team Foundation Server job agent.

⚠Caution

When you use the **upgradeSCVMM** command, Team Foundation Server will create a new template object on your SCVMM server for every team project that uses that template. This ensures that your templates are upgraded to be compatible with SCVMM 2012 without losing any data. However, when the new templates are created, the team project name is appended to the template name. If the new template name is greater than 64 characters, this will cause an SCVMM failure. To resolve this error, you must give those templates a shorter name.

6. Open an elevated command prompt on your Team Foundation Server and enter the following command: **C:\Program Files\Visual Studio Team**

Foundation Server 2011\bin\tools> tfsconfig lab /upgradeSCVMM /collectionName:*. This command will upgrade the data and templates in each project collection database so it is compatible with SCVMM 2012. If you encounter any errors or warnings when you run this command, see the next section to resolve those errors. If you do not encounter any errors or warnings, your upgrade is complete and you can begin using SCVMM environments with Lab Management.

■ **Resolving errors and warnings when using the upgradeSCVMM command**

After you use the **upgradeSCVMM** command, you must resolve any errors or warnings you receive then rerun the command before you can start using Lab Management. The **upgradeSCVMM** command generates a log file that contains information about any errors and warnings that you encounter. The location of the log file is displayed when you run the **upgradeSCVMM** command.

SCVMM failure: If you receive an error that is related to an SCVMM failure, use your SCVMM job history to get additional information about the error. After you resolve the error in SCVMM, rerun the **upgradeSCVMM** command.

See Also

Other Resources

[Upgrading Lab Management Components from Visual Studio 2010](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Managing User Permissions for Lab Management

[See Also](#) [Send Feedback](#)

You can control the level of access that various members of your team have to Lab Management resources by adding each member to security groups for each team project. By default, Team Foundation creates several groups for each project, and each group has its own set of permissions and rights for that project. If the default groups do not provide the appropriate permissions, you can create custom groups that have a specific combination of permissions. For more information about adding users to the default groups, see [Adding and Removing Users To and From Groups](#). For more information about creating custom groups, see [Custom Groups](#).

Default Groups and Permissions

When you create a team project collection, Team Foundation automatically creates the following default collection-level groups:

- Project Collection Administrators
- Project Collection Valid Users
- Project Collection Service Accounts
- Project Collection Build Service Accounts
- Project Collection Proxy Service Accounts
- Project Collection Test Service Accounts

Of these groups, the Project Collection Administrators and the Project Collection Build Service Accounts are given explicit permissions to access or control Lab Management resources. For more information about all the default collection-level groups, see [Default Groups in Team Foundation Server](#).

Similarly, when you create a team project, Team Foundation automatically creates the following default project-level groups:

- Project Administrators
- Contributors
- Readers
- Builders

Of these groups, the Project Administrators, Contributors, and Readers are given explicit permissions to access or control Lab Management resources. For more information about the permissions for all the default collection-level and project-level groups, see [Permission reference for Team Foundation Server](#).

The following table lists the specific Lab Management permissions that are assigned to users when you add the user to a default group. For many team projects, assigning users to either the Project Administrators group or the Contributors group is sufficient for the individual people to do their jobs.

Name of Lab Management permission	Name of the permission at the command line	Users who have this permission can:	Project Collection Administrators	Project Collection Build Service Accounts
View Lab Resources	Read	<p>View information for the various Lab Management resources, which include collection host groups, project host groups, and environment. X</p> <p>To view information about a specific lab resource, you must have the View Lab Resources permission for that resource.</p> <p>Import a</p>	X	X

Import
Virtual
Machine

Create

virtual machine from a VMM library share. This permission differs from Write because users can create an object in Lab Management but not write anything to the Virtual Machine Manager host group or library share. X

Write
Environment
and Virtual
Machines

Write

Users who have this permission for a project host group can create environments. Users who have this permission for a project library share can store environments and templates. X X

Edit Environment and Virtual Machines	Edit	Users who have this permission can edit environments and templates. The permission is checked for the object that is being edited.	X	X
Start	Start	Start an environment.	X	X
Stop	Stop	Stop an environment.	X	X
Pause Environment	Pause	Pause an environment.	X	X
Manage Snapshots	ManageSnapshots	Users who have this permission can perform all snapshot management tasks for an environment, which include taking a snapshot, reverting to a	X	X

Delete
Environment
and Virtual
Machine Delete

snapshot,
renaming a
snapshot,
deleting a
snapshot, and
reading a
snapshot.

Delete
environments
and
templates.
The
permission is X
checked for
the object that
is being
deleted.

Edit the
locations of
Lab
Management
resources,
which include
collection
host groups,
collection
library shares,
project host
groups, and
project library
shares. To
change a
specific
location, you
must have the

Manage Lab Location	ManageLocation	<p>Manage Lab Locations permission for that location. This permission for collection-level locations (collection host groups and collection library shares) also lets you create project-level locations (project host group and project library share).</p>	X
Delete Lab Locations	DeleteLocation	<p>Delete the locations for Lab Management resources, which include collection host groups, collection library shares, project host groups, and project library shares. To delete a</p>	X

location, you must have the Delete Lab Location permission for that location.

Users who have this permission can change the permissions of all the child Lab Management objects. For example, if a user has Manage Child X Permission for a team project host group, the user can change permissions for all the environments under that team project host group.

Modify the permissions for a Lab

Manage
Child
Permissions

ManageChildPermissions

Manage
Permissions

ManagePermissions

Management
object. This
permission is X
checked for
the object
whose
permissions
are being
modified.

■ Custom Groups and Permissions

If the permissions granted by the default Team Foundation security groups are too inclusive or exclusive, you can create new security groups that have different combinations of permissions. For example, your team might have some users who function as test leads and other users who function as just testers. The policies in your organization require that only test leads are authorized to create virtual machines and templates. Because adding a tester to the default Contributors group would automatically give the tester permission to create new virtual machines and templates, you might want to create a custom group named "Tester" that has just the permissions indicated in the following table. Similarly, your organization has some users who function as a team project administrator and other users who function as test lab administrators. The policies in your organization require that only team project administrators can manage user permissions. Because adding a lab administrator to the default Project Administrators group would automatically give the test lab administrator the ability to manage permissions, you may want to create a custom group named "Lab Administrators." The following table illustrates the specific permissions that might be given to the "Lab Administrators" group and three other custom groups that differ from the permissions in the default groups. For the steps to create custom groups, see Custom Groups in Team Foundation Server.

Name of Lab Management permission	Custom group: Lab Administrator	Custom group: Test Lead	Custom group: Tester	Custom group: Developer
View Lab Resources	X	X	X	X
Create VM Template using VMM	X	X		

Virtual
Machines

Write

Environment X
and Template

X

Edit

Environment X
and
Templates

X

X

X

Start

Environment X

X

X

X

Stop

Environment X

X

X

X

Pause

Environment X

X

X

X

Manage

Environment X
Snapshots

X

X

X

Delete

Environment X
and
Templates

X

Manage Lab

X

Location

Delete Lab
Location X

Manage
Child
Permissions

Manage
Permissions

After you create the custom group, you must add users to those groups. Use the command line utility **TFSLabConfig** to manage the Lab Management permissions for groups or users. For the syntax and other considerations, see [Configure Lab Management with TFSLabConfig](#).

To view what groups or users have existing permissions on the various Lab Management objects, see the [TFSLabConfig Permissions Command](#).

See Also

Concepts

[Configure Lab Management for SCVMM environments](#)

Other Resources

[Change Existing Lab Management Configurations](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Planning Your Lab

[See Also](#) [Send Feedback](#)

As you prepare for widespread use of Lab Management in your test lab, several questions are likely to come to mind:

- How many physical servers do I need?
- What kinds of servers should I buy?
- How much storage capacity do I need?
- Can I use a SAN for storage?
- Can I setup everything on one powerful machine?
- How do I set up an isolated lab?

This topic provides general guidelines for estimating the numbers and types of physical servers, virtual machines (VMs), and controllers you will need for using Visual Studio Lab Management. In general, the number of servers is not as important as the capacity of each server. For example, a server that uses a dual-core or quad-core processor will be able to support more VMs than a server that uses a single-core processor. Similarly, a server that has 32 GB of RAM can host more VMs at the same time than a server that has only 8GB of RAM.

▣ Planning for Capacity

The following sections contain guidelines to help you provide sufficient capacity for your virtual testing lab. The guidelines are expressed as things to do (Do's) and not do (Don'ts) when you acquire or configure hardware or when you install and configure the required software.

▣ Planning for Team Foundation Server

Don't

- Set up everything on a single machine. Only if you will use a single machine just for demonstration or proof of concept purposes should you set up all the components on a single machine.
- Use the NetworkService account as the service account for Team Foundation Server if your instance of Team Foundation Server uses more than one server to run the logical application tier, and you will be using the Lab Management feature of Visual Studio Using the NetworkService account requires more manual work later to maintain the physical host machines for virtual environments. This extra work is necessary because the NetworkService account for each new application-tier machine has to be added to the local Administrator group on each physical host machine. For example, if you run a virtual lab with 20 physical hosts and add or replace an application-tier machine, you would then have to update each of the 20 host machines with the name of the new application-tier machine and assign permissions. Instead of using the NetworkService account, use a standard domain user account and password for the TFSservice account. By doing this, the domain user account is added once at the initial configuration of the physical host and each subsequent application-tier machine uses the same account. For more information about the limitations of the NetworkService account, see [NetworkService Account](#).

▣ Planning for System Center Virtual Machine Manager

Don't

- Install SCVMM on a virtual machine. Installing SCVMM on a virtual machine will make it harder to administer the physical host that virtual machine is running on, and it will slow the performance of the library if you set the library up on the same virtual machine.
- Use clustering with the SCVMM 2008 R2 library servers. Lab Management only supports clustering in SCVMM environments when using SCVMM 2012, not SCVMM 2008 R2.
- Set virtual LAN IDs for network adapters. Lab Management does not support setting and using a virtual LAN ID in System Center Virtual Machine Manager. If you manually set the VLAN ID on a network adapter for a virtual machine and then store the virtual machine in the SCVMM library, the VLAN ID will be cleared when the virtual machine is deployed. For more information about how to use VLAN IDs, see [Configuring Virtual Networks in VMM](#).

Do

- Provide the SCVMM machine enough resources. If you expect to have fewer than 50 VMs in your lab, the machine running SCVMM should have at least:
 - A 64-bit processor
 - 4 GB of memory
 - A 300 GB hard disk drive
 - Windows Server 2008 R2 operating system

If you expect to have more than 50 VMs, increase these resources. If you plan to install SCVMM along with some other software on the same machine, give SCVMM server the amount of resources that were described earlier in this topic. However, be sure to determine the amount after you deduct the resource consumption of the other software. For instance, if you want to install SCVMM on the machine that is running Team Foundation Server, add the requirements to those of Team Foundation Server, and then ensure that the machine has enough capacity.

- Provide the server that runs the library at least 200GB of free space on the hard disk drive. In the default installation, make sure that the drive used by the library share has more than 200GB free space.
- Create the default library share on D: and not C:. By default, SCVMM creates the library share on the same machine it is on and creates the library in the C: drive. Changing the default library share to D: makes it easier to upgrade the machine later.
- Use a hard disk drive with sufficient speed for the library. If you plan to use the library lightly, a hard disk with sufficient speed will be sufficient. If you plan to use the library moderately, use a RAID 5 disk configuration with 6 to 12 disks for better throughput. If you plan to use the library heavily, use multiple library servers. You can use direct-attached storage or SAN. When using SAN, create a LUN to be used solely for library machine.
- Run Team Foundation Server under a regular domain user account instead of the network service account. This is required if you put Team Foundation Server and SCVMM on the same machine.
- If SCVMM is installed on a Hyper-V host, store the Hyper-V hosted virtual machines on a different hard disk drive than the SCVMM library. For example, use C: from one disk for the library, and D: from another disk for Hyper-V virtual machines. SCVMM server, in this case, will be running in the primary OS in Hyper-V. This ensures that when the primary OS is loaded, all guest OS (VMs deployed in Hyper-V) will be impacted. To reduce this impact, configure the host reserves for that machine by adding the Hyper-V host reserves (described below) to the SCVMM machine requirements mentioned earlier. Host reserves can be configured using the SCVMM Administrator Console.
- Provide line-of-sight network routing between SCVMM and Team Foundation Server, hosts, and other library servers.
- Update the SCVMM machine with all the latest Windows updates and ensure these updates get applied automatically. If this is not feasible, you should plan to keep track of Windows and SCVMM updates, and apply them manually as they become available.

▣ **Planning for the Hyper-V hosts**

Don't

- Install any additional software such as Team Foundation Server on the physical host machine. If you have sufficiently powerful hosts (exceeding the aggregate needs of the hypervisor and virtual machines), then you can have SCVMM or library server co-located on the host, as long as you also account for the resource constraints of those servers. For example, if you want to install SCVMM on a Hyper-V host machine, then add the host requirements, virtual machine requirements, and SCVMM requirements, and then ensure that the machine has enough capacity.
- Use clustering with Hyper-V host servers. Lab Management supports clustering in SCVMM environments.
- Schedule tens of VMs deployments simultaneously. Limit the number of concurrent environment deployments to hosts.
- Use physical hosts that are in different geographic locations than the library servers. If you must use hosts that are in a different geographic location than the SCVMM library servers, the network speed between SCVMM and hosts should be at least 100 Mbps and not subject to high latencies.
- Create multiple network adapters on a virtual machine that connects to a specific network. Lab Management overrides this configuration and creates two adapters. One adapter connects to the lab network and the other adapter handles internal communication between virtual machines.
- Configure the MAC address on a network adapter used in a network-isolated environment. Lab Management clears the MAC address at the time the network-isolated environment is created.

Do

- Provide the host machines with enough resources and configure them correctly. The number of Hyper-V hosts and the capacity of each host depends on the number of VMs that you host in your lab. If you decide to setup a relatively small lab, install the Hyper-V role on machines with the following configuration:

- Two, dual-core, 64-bit processors that are Hyper-V capable
- 16 GB memory
- 300 GB hard disk space
- Windows Server 2008 R2 operating system
- The latest updates of the Windows operating system.

If you have relatively larger number of virtual machines, and you decide to set up a few, powerful hosts, configure each host as follows:

- two, quad-core, 64-bit processors that are Hyper-V capable
 - 64 GB memory
 - 1 TB hard disk space
 - Windows Server 2008 R2 operating system
 - The latest updates of the Windows operating system.
- Reserve enough RAM memory on the host. Out of the host capacity requirements listed above, you must set aside the following resources for the smooth functioning of the hypervisor. For a 16GB host, set aside 20% for the CPU and 2 GB memory. For a 64 GB host, set aside 30% for the CPU and 4 GB memory. These host reserves must be configured in the host properties pane of SCVMM Administrator Console. Only the resources remaining on the host after deducting the host reserves can be used for virtual machines.
 - Provide enough storage for virtual machines. You should use a different disk partition for virtual machine storage than the primary partition of Hyper-V server. For example, use D: for virtual machine storage and C: for the primary partition for the hypervisor. After you decide on the virtual machine storage location, configure that location in Hyper-V Manager or using SCVMM Administrator Console. In Hyper-V Manager, change the Virtual Hard Disks folder and the Virtual Machines folder. In the SCVMM Administrator Console, change the Placement Path under the host

properties.

- Provide the hosts with fast hard disk drives and configure the drives correctly. Disk types for hosts: A disk with good speed is necessary. RAID 5 configured disks are highly recommended. The storage for hosts can come from Direct-attached storage or from SAN. However, if you decide to have your host's disk come from a SAN drive for space and reliability needs, you will have to have separate LUNs mapped to each host. Even if the LUNs are managed by same controller, given that Visual Studio Lab Management does not leverage any of SAN functionalities, the underlying BITS copy during a virtual machine deployment will happen all the way from library to host via your LAN network.
- For SCVMM to be installed on a Hyper-V host, it is highly recommended that the hard disk drive used for storing Hyper-V hosted virtual machines is different from the disk used for library. SCVMM server, in this case, will be running in the primary OS in Hyper-V. So, when the primary OS is loaded, all Guest OS (VMs deployed in Hyper-V) will have performance impact. To reduce this impact, configure the host reserves for that machine by adding the Hyper-V machine's host reserves to the SCVMM machine requirements mentioned earlier. Host reserves can be configured using SCVMM administration console.
- For a Hyper-V host to be used as a library server as well, you must have multiple disks in the machine. You should use separate hard disks on the host for the virtual machines and for the library storage.
- Provide the Hyper-V host with line-of-sight networking to Team Foundation Server, SCVMM, and other library servers.
- If the Hyper-V hosts are in different geographic locations, have a local library server for each location as well.
- Update the hosts regularly. Hyper-V hosts should be on a network from where operating system updates can be automatically applied. If this is not feasible, you should plan on keeping track of Windows and SCVMM updates, and apply them manually when they become available.

▣ Planning for Controllers and Agents

Don't

- Install a test controller inside an environment. Only the build, test, and lab agents should be installed on the virtual machines inside an environment.

Do

- Use more than one build controller when building and deploying an application for testing. The first controller is used by the build process and is not heavily utilized. The second controller is used to deploy the build to virtual machines and run tests; therefore, it can be heavily used if there are a large number of virtual machines in your lab. The second controller is also used to take snapshots of the environment.
- Use test controllers in the same domain as Team Foundation Server. If esprtfs and a test controller are in a workgroup or untrusted domain, you must create a local user account with same user name and password on both machines, add this user on Team Foundation Server to the "[Project Collection]\Project Collection Test Service Accounts" security group, and then register the test controller with team project collection using this local account.

Planning for Topology

Do

- Use a gigabit network to connect the server where SCVMM is installed to the library servers and to the Hyper-V hosts.
- Establish a full, two-way trust relationship among the domains where Team Foundation Server, the test controller, the build controller, SCVMM, and the physical host of the virtual machines are running.

There are several topologies you can use when setting up Lab Management for testing your application. The simplest topology for using Lab Management requires only two servers: install all Team Foundation Server components on the same server and install all SCVMM 2008 components on an additional server. Alternatively, you might have complex networking topology requirements that restrict the networks in which Team Foundation Server, SCVMM, Hyper-V hosts, and virtual machines running the application-under-test can be located. In another alternative, you might want to configure network load balancing for your Team Foundation Server. The following list suggests several possible dimensions for your topology and the variations within each dimension.

Networking

- DNS
- Firewall
- Threat Management Gateway

Domain

- One-way trust
- Two-way trust
- No-trust

Team Foundation Server logical application tier

- Single server
- Multiple servers without network load balancing
- Multiple servers with network load balancing

Team Foundation Server logical data tier

- Single server
- Multiple servers without clustering
- Multiple servers with clustering

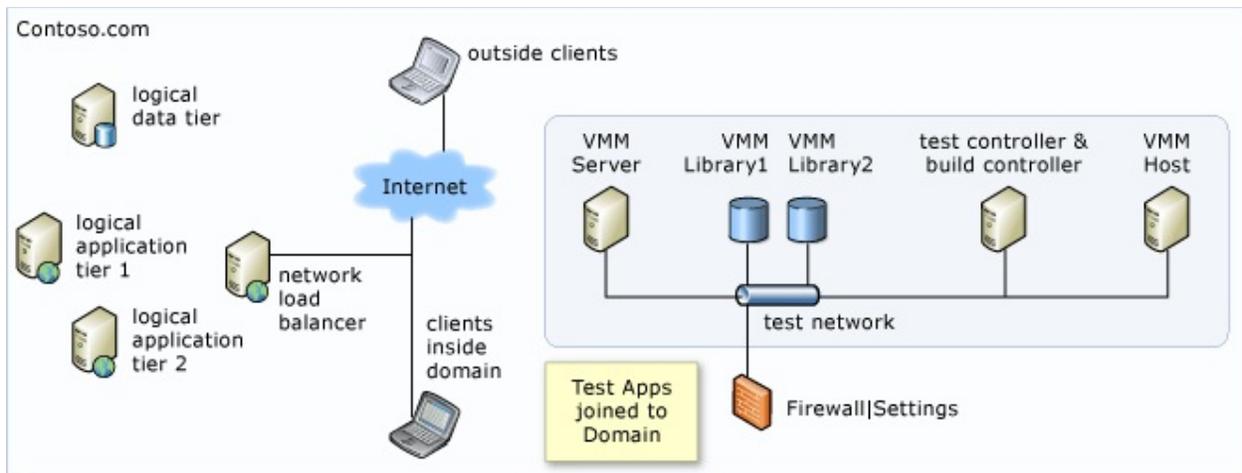
Tests

- Inside the environment
- Outside the environment

The following four sample topologies are examples of how you can set up combinations of the above dimensions according to your testing needs.

▣ **Sample Topology 1**

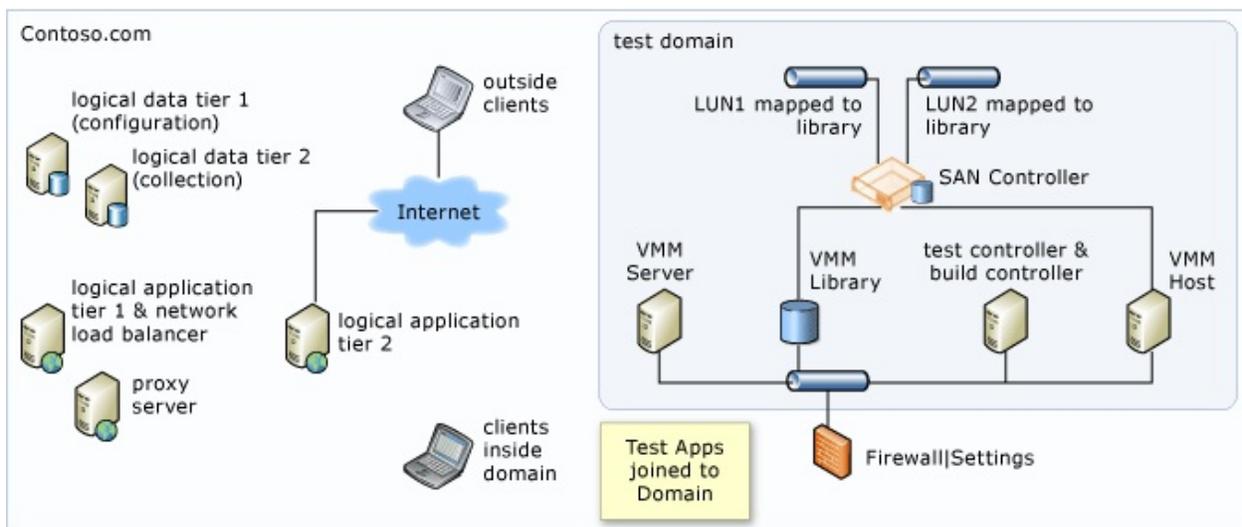
The Team Foundation Server logical application tier is run on several servers and those servers are controlled by a network load balancer. There is also a separate test network with firewall settings to control the test traffic into and out of the domain network. The following diagram illustrates topology 1.



For instructions to set up this topology, see [Setting up various topologies to test with Visual Studio Lab Management - Part 1](#).

Sample Topology 2

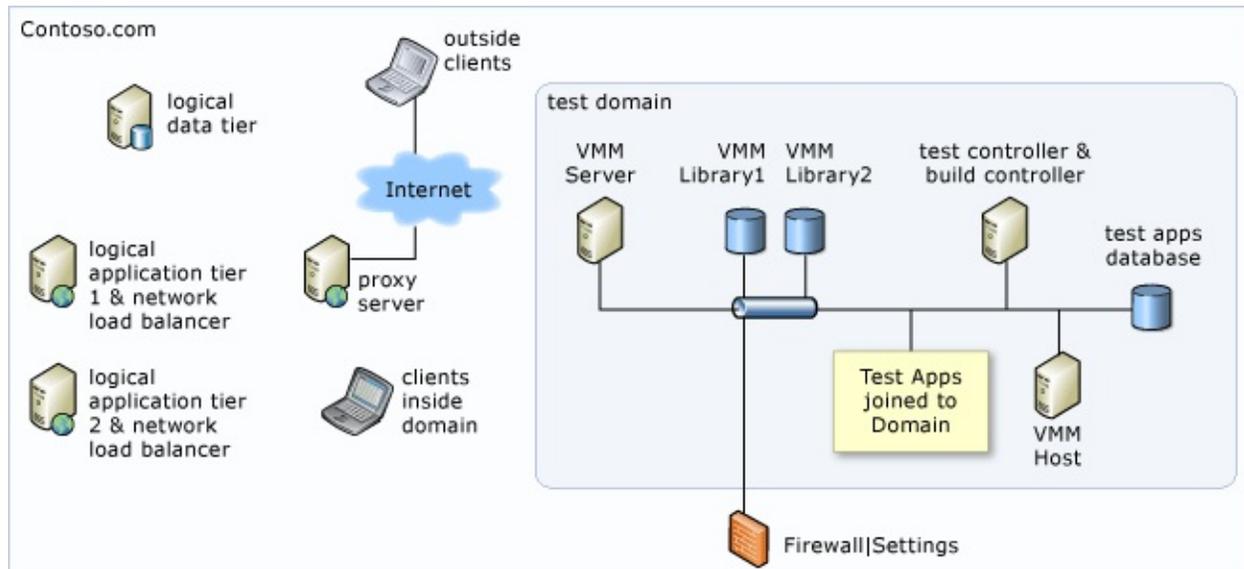
The Team Foundation Server logical application tier and data tiers are run on several servers, but those servers are not controlled by a load balancer. There is also a separate test network with a SAN-based library and host. The following diagram illustrates topology 2.



For instructions to set up this topology, see [Setting up various topologies to test with Visual Studio Lab Management - Part 2](#).

Sample Topology 3

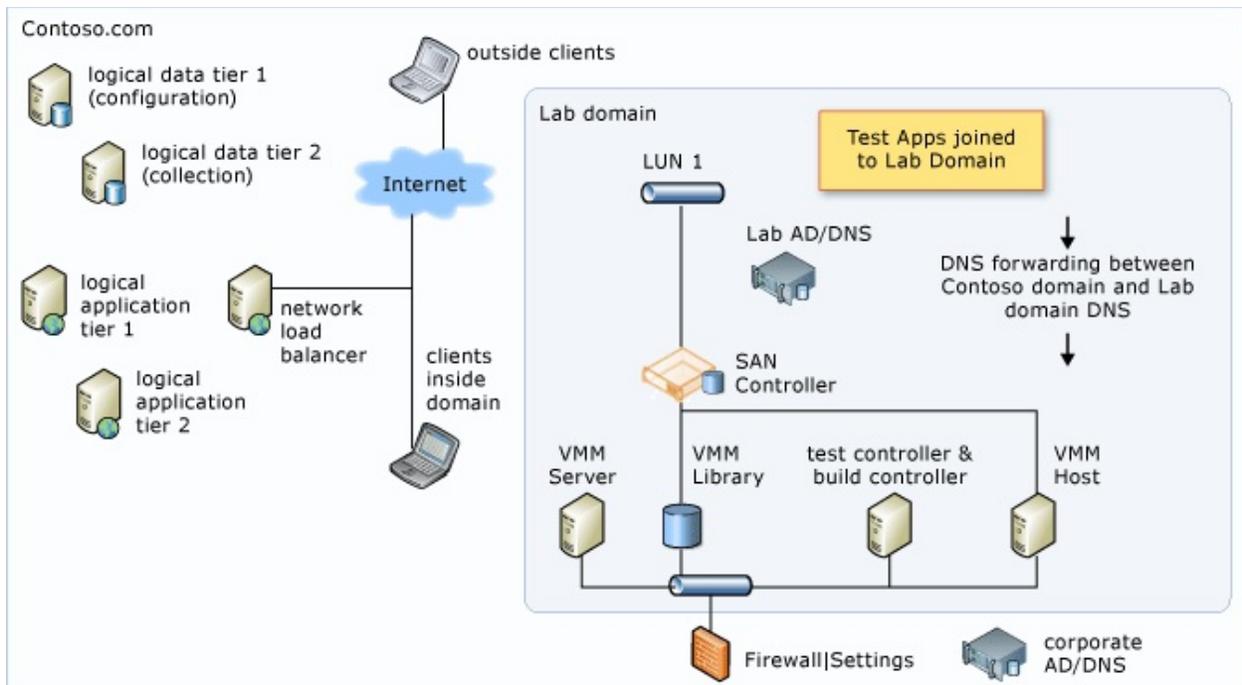
The Team Foundation Server logical application tier is run on several servers and those servers are controlled by a network load balancer. There is also a separate test network. The applications being tested make calls to a database outside the virtual environment. The following diagram illustrates topology 3.



For instructions to set up this topology, see [Setting up various topologies to test with Visual Studio Lab Management - Part 3](#).

■ Sample Topology 4

The Team Foundation Server logical application tier and data tiers are run on several servers and those servers are controlled by a network load balancer. The test network and environments are in a separate domain. The following diagram illustrates topology 4.



For instructions to set up this topology, see [Setting up various topologies to test with Visual Studio Lab Management - Part 4](#).

See Also

Concepts

[Team Foundation Server architecture](#)

[Configure and administer Lab Management](#)

Other Resources

Examples of Simple Topology

Examples of Moderate Topology

Examples of Complex Topology



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Upgrade from Visual SourceSafe

[Send Feedback](#)

Upgrading your code projects, files, version history, labels, and user information from [Visual SourceSafe](#) to Team Foundation Server (TFS) or [Visual Studio Online](#) version control has many benefits for your team. TFS version control is a modern version control system that is fully integrated with the suite of ALM tools in Visual Studio 2012 and Team Foundation Server.

To upgrade data from your Visual SourceSafe database

1. [Understand how VSS upgrade tools convert your data](#) Team Foundation and Visual SourceSafe have significant functional differences. As a result, the VSS upgrade tools must modify certain kinds of data as it is upgraded.
2. [Prepare to upgrade from Visual SourceSafe](#) Before you start the upgrade process, it's critical that you plan ahead and prepare your data.
3. Upgrade your data.
 - [Upgrade from Visual SourceSafe using the wizard](#) In most cases, you should use the VSS Upgrade Wizard. Advantages include:
 - A fast and easy-to-use step-by-step graphical interface.
 - Support for upgrading into a team project on either an on-premises Team Foundation Server or on [Visual Studio Online](#).
 - [Upgrade from Visual SourceSafe using the VSSUpgrade command-prompt tool](#) You can use the VssUpgrade command-prompt tool to upgrade to an on-premises Team Foundation Server:

- Using a script to upgrade your data.
 - While taking advantage of VssUpgrade features that are not available when using the wizard.
4. [Take next steps after upgrading from Visual SourceSafe](#) After the upgrade tool has finished processing your data, you should verify that the process led to the outcome you expected. You might need to take additional steps to complete the upgrade.
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Understand how VSS upgrade tools convert your data

[Send Feedback](#)

The Visual SourceSafe upgrade tools are a convenient method for a one-time, one-way upgrade of your codebase from Visual SourceSafe to Team Foundation Server (TFS). You should use the tools to upgrade your data only when you are ready to stop accessing it in Visual SourceSafe and begin maintaining it in TFS.

TFS and Visual SourceSafe have significant functional differences. As a result, the Visual SourceSafe upgrade tools modify certain kinds of data during the upgrade.

In this topic

- [How changesets are created](#)
- [How shared and pinned items are upgraded](#)
- [How historical data is ported](#)
 - [How data about the user name and the time stamp is ported](#)
 - [How specific types of events are converted](#)
- [How version control bindings are ported](#)

How changesets are created

TFS version control groups changes to multiple files into a single unit when a user checks in a set of changes. This single unit is known as a changeset.

Visual SourceSafe does not have a feature that is equivalent to changesets. However, during the conversion process, each set of changes is grouped into a changeset as long as the following conditions are true:

- The changes do not conflict with each other. For example, no two actions affect the same file or folder.
- The changes occurred within no more than a few minutes of each other.
- The changes were checked in by the same user.
- The changes have the same check-in comment.

How shared and pinned items are upgraded

In Visual SourceSafe, you can share a file across multiple folders. Changes that you make in a shared file are replicated across the folders in which the file is shared. TFS version control does not have an equivalent feature. During the upgrade, shared files in your Visual SourceSafe project are upgraded by creating an additional independent copy of the item on your server for TFS version control.

TFS version control also does not have a feature that is equivalent to the Pin feature in Visual SourceSafe. During the upgrade, pinned items in the Visual SourceSafe project are converted to labeled items on your server for TFS version control. For more information, see the next section.

How historical data is ported

Each event in the history of an item in your Visual SourceSafe database is transferred onto your Team Foundation Server as a changeset. After the upgrade is completed, you can view this data in the History window. For more information, see [Get the history of an item](#).

Some changes to the data occur during upgrade.

How data about the user name and the time stamp is ported

As each entry in the history of an item in your Visual SourceSafe database is upgraded to a changeset on your Team Foundation Server, the following changes occur:

- The time stamp on the changeset is set to the date and time when the item was upgraded.
- The original time stamp is stored in the Comment field of the changeset.
- The user name is stored in either the User field or the Comment field of the changeset, depending on the result of the user mapping process.

Note

If you want to control how user names are assigned to historical data, you must use VssUpgrade. See [Specify how user names are upgraded](#).

How specific types of events are converted

Events such as edit, rename, and delete are upgraded from your Visual SourceSafe database into changesets on your Team Foundation Server in a straightforward manner. However, the upgrade tools convert some events in ways that you might not expect, as the following table shows.

|

Visual SourceSafe Event	How it is upgraded into TFS version control
--------------------------------	--

Add file or folder	<p>This changeset is the first event in the history of each upgraded file and folder. Unlike in Visual SourceSafe, no event is logged for the parent of each child item that it contains.</p>
--------------------	---

	<p>Sharing is a precondition of branching in Visual SourceSafe, but TFS version control does not support sharing. Therefore, the upgrade of a branched file creates a copy of the file in the destination folder.</p>
--	---

Branching	<p>The shared files in your Visual SourceSafe database are upgraded to TFS version control by copying the version of the file that existed when it was shared and putting the copy in the destination folder. Thereafter, each changeset is replicated in both copies of the file until the branch event occurs.</p>
-----------	--

Label

Implicitly labeled items: In Visual SourceSafe, when you label a folder, the items it contains are implicitly labeled. The upgrade process explicitly labels all the upgraded items in your team project.

Explicitly-labeled files: In Visual SourceSafe, when you explicitly label an individual file, a new version of the file is created. The upgrade process ports the label to the applicable version in TFS version control, but it does not create a new version.

Note

If your Visual SourceSafe database contains many labels that are applied to many files, the upgrade process may be prolonged. If you want to exclude this data, you must use VssUpgrade and configure it to ignore labels. See `<Label migrate="false" />`.

The Move Folder event creates a new version of the folder in TFS version

control.

Move
folder

If you are using VssUpgrade, it will not upgrade the complete history of items in moved folders unless both the source and destination folders are upgraded at the same time. See Review and resolve issues found by the analyze feature.

 **Note**

If the move folder event is combined with a restore event, the history data may not be upgraded correctly.

Restore

No history data that occurs before a restore event is upgraded.

TFS version control does not support pinning. The upgrade tools upgrade a pinned file by creating two labels.

The **PINNED_LATEST** label is applied to the pinned versions of the pinned files and to the most recent version of the unpinned files. The

PIN and
UNPIN

PINNED label is applied to only the pinned versions of the pinned files. After upgrade, the **PINNED_LATEST** label will retrieve the same files as a Get Latest in Visual SourceSafe. However, one exception is that the **PINNED_LATEST** label might return different files if events other than checking in occurred after a file was pinned, such as renaming or deleting the file.

Sharing

TFS version control does not support sharing. Shared files in your Visual SourceSafe database are upgraded to TFS version control by copying the version of the file that existed when it was shared and putting the copy in the destination folder. Thereafter, each changeset is replicated in both copies of the file.

During an upgrade of undelete events of a file or folder, the tools replay the event to create a new version of the file and folder in TFS version

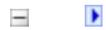
Undelete File or Folder	control. The upgrade tools create a changeset that includes the file or folder name, the date and time when it was undeleted, and the user name.
-------------------------------	---

How version control bindings are ported

The upgrade tools port version control bindings for each solution.

Next Step

[Prepare to upgrade from Visual SourceSafe](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Prepare to upgrade from Visual SourceSafe

[Send Feedback](#)

Before you begin the upgrade process from Visual SourceSafe (VSS) to Team Foundation Server (TFS), plan ahead and prepare your computers and databases to improve the outcome and avoid serious problems.

Note

If your Visual SourceSafe database is a version earlier than Visual SourceSafe 6.0, you must first upgrade it to Visual SourceSafe 2005. Then you can upgrade your data into TFS. This process is explained below.

Prepare to upgrade

1. [Make sure you have the required permissions](#)
2. [Schedule the upgrade with your team](#)
3. [Provide a database for upgrade tool to use](#)
4. [Prepare your instance of Team Foundation Server](#)
5. [Copy and prepare your Visual SourceSafe database](#)
6. [\(Optional\) truncate the history of items](#)
7. [Prepare the upgrade computer](#)

Make sure you have the required permissions

To upgrade from VSS to TFS, you must have the following permissions:

- In the VSS database that contains the data that you want to upgrade, you must know the password of the Admin account.
- On the temporary database that the upgrade tool uses, you must be a SQL Administrator or have the CREATE ANY DATABASE permission.
- On your Team Foundation server, you must be a member of the Team Foundation Administrators security group. See

[Permission reference for Team Foundation Server.](#)

- On the upgrade computer-the computer that contains the copy of the VSS database that you are upgrading-you must be a member of the Administrators group.

▣ **Schedule the upgrade with your team**

Try to schedule the upgrade when your team does not require access to the VSS database that you are upgrading. If you have lots of data, a large team, or if you have worked on the projects for a long time, you should allow time to prepare and upgrade your data.

Important

Inform your team members when the upgrade process will occur, and advise them to check in all files before the process begins.

■ Provide a database for the upgrade tool to use

The upgrade tools require a database-either

[SQL Server Express](#) or [SQL Server](#)-to use for temporary storage. After the upgrade process is complete, the database is never used again.

📌 Tip

By default, when you install Team Foundation Server Express, SQL Server Express is installed, and you are automatically given the required CREATE ANY DATABASE permission.

Prepare your instance of Team Foundation Server

Prepare the upgrade computer by following these steps:

1. Make sure that the data tier for TFS has enough storage space available. Typically, you will need about two times the data size of the projects in the VSS database that you are upgrading, but the exact amount of storage space depends on the following factors:
 - The size of the VSS database you are upgrading.
 - The number of actions to be upgraded.
2. The upgrade tools require that the destination team projects already exist on your Team Foundation server before the upgrade process starts. If you do not yet have the team project collection or the team project into which you want to port your upgraded VSS data, you must create them. See

[Create a team project.](#)

We recommend that you use a new team project that you have not yet begun using. You can upgrade your data into a team project that you are already using. However, if the path to any of your VSS items overlaps with an existing version control path, the upgrade process will fail.

■ Copy and prepare your Visual SourceSafe database

Copy and prepare your VSS database by following these steps:

1. Check in files. Ideally, all files in your VSS database should be checked in. If this is not possible, as many files as possible should be checked in before you upgrade.
2. Remove access to the Visual SourceSafe projects. You should be the only person who has access to the Visual SourceSafe projects that you are upgrading.
3. Copy the database. Follow the instructions on this page of the Microsoft Web site:

[How To Back Up a Visual SourceSafe Database.](#)

- Upgrade the copy of your database. If your Visual SourceSafe database is a version earlier than Visual SourceSafe 6.0, upgrade it to Visual SourceSafe 2005 by using the Visual SourceSafe [DDUPD Utility](#).
- Scan for and fix data integrity issues in the copy of your database.

⚠ Important

You must use the Visual SourceSafe ANALYZE utility to locate and fix data integrity issues in the database. For more information about how to use this tool, see the following pages on the Microsoft Web site: [ANALYZE Utility](#) and [How to Detect and Fix Database Corruption Errors in Visual SourceSafe](#).

☐ (Optional) Truncate the history of items

If you do not need all the history data, you can save time when you upgrade from Visual SourceSafe and preserve only the history after a specific date. This is called truncating the history. To do this, use the Archive feature in VSS.

☑ Tip

If you want to exclude all historical data and you plan to use the VSS Upgrade Wizard, you can skip this section.

⚠ Caution

Archiving permanently removes the version history from the VSS database. Therefore, make sure that you perform this procedure on a copy of the VSS database instead of the database that is in service.

You can specify the time stamp before which you want to truncate the history by using any of the following values:

- Label
- Version of a folder
- Date

For more information about how to archive in Visual SourceSafe, see [Visual SourceSafe Archive Databases](#).

☑ Note

The Visual SourceSafe Archive feature has a limitation of 2 gigabytes (GB) on the size of the archive file. If an error occurs while you are archiving, try to archive smaller projects separately.

Prepare the upgrade computer

Log on and prepare the computer on which you will perform the upgrade:

1. Make sure the computer is running one of the following operating systems:
 - Windows 8
 - Windows Server 2012
 - Windows 7
 - Windows Server 2008 R2
2. Install the

[Visual SourceSafe upgrade tools](#).

- Make sure that the computer has sufficient free disk space to complete the upgrade process. To estimate how much disk space is required, total the following items:
 - 5 GB for the upgrade tools to create temporary files and to generate log files.
 - Two times the size of the projects in the Visual SourceSafe database that you will upgrade.
- Make sure that you have followed the steps in [Copy and Prepare Your Visual SourceSafe Database](#) earlier in this topic.
- Copy the VSS database to a folder on the upgrade computer.

Note

If you use file sharing to enable the upgrade computer to access the data in the VSS database instead of copying the database, you must provide Read and Modify access to the account that you use to log on to the upgrade

computer. This approach is not recommended because it may prolong the upgrade process.

⚠Caution

Regardless of how you set up your upgrade computer to access your VSS database, make sure that you run the upgrade process on a copy of the database and not the database that is in service. This approach helps protect your data.

Next step

Choose one of the following processes:

- [Upgrade from Visual SourceSafe using the wizard](#) In most cases, you should use the VSS Upgrade Wizard. Advantages include:
 - A fast and easy-to-use step-by-step graphical interface.
 - Support for upgrading into a team project on either an on-premises Team Foundation Server or on [Visual Studio Online](#).
 - [Upgrade from Visual SourceSafe using the VSSUpgrade command-prompt tool](#) You can use the VssUpgrade command-prompt tool to upgrade to an on-premises Team Foundation server:
 - Using a script to upgrade your data.
 - While taking advantage of VssUpgrade features that are not available when using the wizard.
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Upgrade from Visual SourceSafe using the wizard

[Send Feedback](#)

After you understand how the upgrade tools work and have completed preparations, you can proceed to upgrade your Visual SourceSafe data. In most cases, you should use the VSS Upgrade Wizard. Advantages include:

- A brief and simple step-by-step procedure to initiate the upgrade.
- Support for upgrading into a team project on either an on-premises Team Foundation Server or on [Visual Studio Online](#).

Upgrade your Visual SourceSafe data using the wizard

⚠ Important

You must run Analyze on your Visual SourceSafe repository before you use the wizard. See Copy and prepare your Visual SourceSafe database.

1. From Windows Start, run VSS Upgrade Wizard.
2. On the Visual SourceSafe Repository page...

Provide and check the Visual SourceSafe information required for upgrade to Team Foundation Server.

Type in a Visual SourceSafe repository folder path and administrator password, click List Available Projects, and select one or more projects from the list for upgrade.

Visual SourceSafe Repository:

Use the **Browse...** button to find a Visual SourceSafe repository folder that has to include the srcsafe.ini file.

Visual SourceSafe Admin password:

The field is optional. Leave it empty if the Visual SourceSafe administrator account does not require a password.

Specify the repository, and if necessary, the Admin password.

3. To display the projects in your Visual SourceSafe repository, choose the List Available Projects link.

Visual SourceSafe Projects: [List Available Projects](#)

Select all

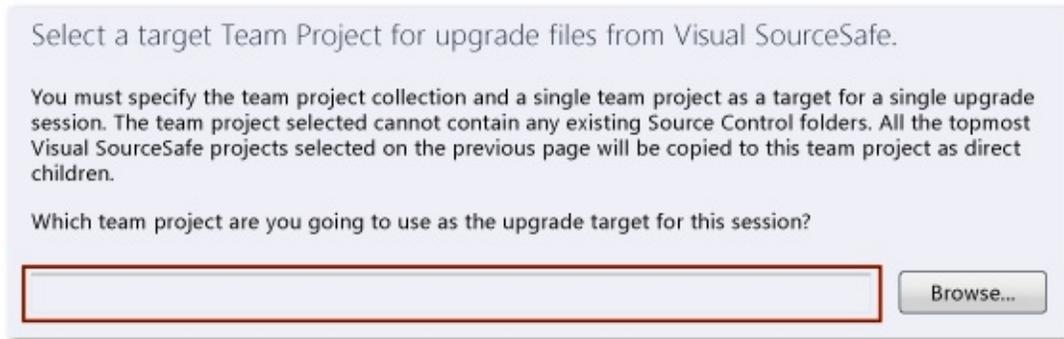
Project Name
<input checked="" type="checkbox"/> \$/Converter

Select the projects that contain the data you want to upgrade.

4. Select the check box at the bottom of the page to confirm you have run Analyze. See Copy and prepare your Visual SourceSafe database.

Choose Next.

5. On the Team Project page...



Select a target Team Project for upgrade files from Visual SourceSafe.

You must specify the team project collection and a single team project as a target for a single upgrade session. The team project selected cannot contain any existing Source Control folders. All the topmost Visual SourceSafe projects selected on the previous page will be copied to this team project as direct children.

Which team project are you going to use as the upgrade target for this session?

Choose Browse and then use the Select a Team project for Migration dialog box to specify the team project into which you want to port the upgraded data.

We recommend that you use a new team project that you have not yet begun using. You can also upgrade your data into a team project that you are already using. However, if the path to any of your Visual Source Safe items overlaps with an existing version control path, the upgrade process will fail.

Choose Next.

6. On the options page...



Select upgrade options and a SQL server instance.

Upgrade Options

Full history
This option will upgrade full history of changes.

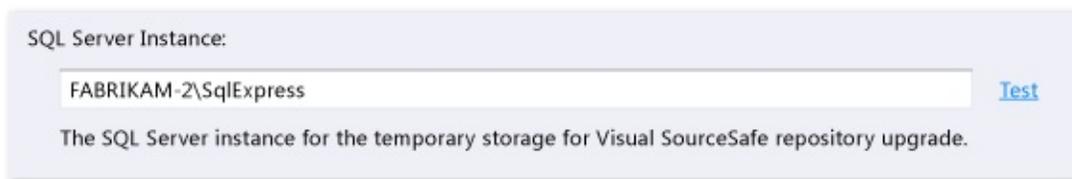
Tip
This option will upgrade only latest version of files.

Select whether you want to upgrade the Full history or Tip to omit historical data.

 **Tip**

If you want to upgrade a specific subset of your historical data, you can truncate it before you upgrade. Choose Cancel, and then see Truncate the history of items.

7. On the options page...



Specify the name of the SQL Server instance you want the wizard to use for temporary storage. See Provide a database for upgrade tool to use.

Choose Next.

8. Review the configuration settings and then choose Next.
 9. After the readiness checks all pass, choose Upgrade.
 10. The wizard analyzes your Visual SourceSafe data, and then upgrades and ports it to your Team Foundation Server. After the process is complete, choose Next.
 11. View and verify the results, and resolve issues as necessary.
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Upgrade from Visual SourceSafe using the VSSUpgrade command-prompt tool
[Send Feedback](#)

In most cases, you should use the VSS Upgrade Wizard instead of the VssUpgrade command-prompt tool to upgrade your Visual SourceSafe data. However, there are [a few situations](#) when you might want to use the VssUpgrade command-prompt tool to upgrade to an on-premises Team Foundation server (TFS).

To upgrade your Visual SourceSafe data using VSSUpgrade

1. Understand how the tool converts your data TFS and Visual SourceSafe have significant functional differences. As a result, the Visual SourceSafe upgrade tools modify certain kinds of data during the upgrade.
2. Prepare to upgrade Before you start the upgrade process, it's critical that you plan ahead and prepare your data.

Important

You must run the Visual SourceSafe ANALYZE utility on your Visual SourceSafe repository before you proceed. See Copy and prepare your Visual SourceSafe database.

3. [Learn about unique VssUpgrade features](#). If you don't need any of them, then you should use the VSS Upgrade Wizard instead.
4. [Analyze your data](#) Before you upgrade your data from Visual SourceSafe to Team Foundation version control, you must use the Analyze command to determine whether any issues in your data will affect the outcome of the upgrade. This process also generates a user mapping file that is required to

upgrade your data.

5. [Upgrade your data](#) To upgrade your data, you must specify how user names are upgraded, create a Migrate settings file, and then run the Migrate command.
6. View and verify the results, and resolve issues as necessary

▣ VssUpgrade features

Use VssUpgrade if you want to upgrade and port your data to an on-premises team project and:

- Perform the upgrade as part of an automated approach, such as a script.

-

[Migrate multiple Visual SourceSafe projects into a single team project](#), using a folder structure that is specified by you.

- [Specify how user data is converted](#), for example, from Kim in Visual SourceSafe to EUROPE\KimT in TFS.
- [Ignore labels](#) in the Visual SourceSafe data.
- [Resume the process after it has been interrupted](#) while upgrading a large amount of data-for example, because of a network error.

If you don't need any of these features, or if you need upgrade and port your data into a team project on [Visual Studio Online](#), then you should use the VSS Upgrade Wizard instead of VssUpgrade.

■ Analyze your data

Before you upgrade your data from Visual SourceSafe to TFS version control, you must first use the Analyze command to determine whether any issues in your data will affect the outcome of the upgrade. This command also generates a user mapping file that the Migrate command uses to upgrade your data.

■ Create an Analyze settings file

Before you run the Analyze command, you must create an Analyze settings file. In this file, you specify the path of the Visual SourceSafe database that you will upgrade and the folders that you want to upgrade.

The following XML is an example of an Analyze settings file.

Copy Code

```
<?xml version="1.0" encoding="utf-8"?>
<SourceControlConverter>
<ConverterSpecificSetting>
  <Source name="VSS">
    <VSSDatabase name="c:\ourvss"></VSSDatabase>
    <UserMap name="c:\ourvss\migrate\Usermap.xml"></UserMap>
    <SQL Server="SQLInstanceName"></SQL>
  </Source>
  <ProjectMap>
    <Project Source="$/Core"></Project>
    <Project Source="$/ProjectA"></Project>
    <Project Source="$/ProjectB"></Project>
  </ProjectMap>
</ConverterSpecificSetting>
<Settings>
  <Output file="c:\ourvss\migrate\logs\ContosoVSSAnalyze.xml"></Output>
</Settings>
</SourceControlConverter>
```

You can copy the previous example, paste it into your own Analyze settings file, and then modify it. The following information can help you adapt the example to meet your needs.

▣ <?xml encoding> attribute

The <?xml encoding> attribute must match the encoding that is used in your Analyze settings file. For example, if the file is saved as Unicode, the <?xml encoding> tag is as follows:

```
<?xml version="1.0" encoding="unicode">
```

▣ <VSSDatabase name> attribute

In the <VSSDatabase name> attribute, specify the path of the folder that contains the **srcsafe.ini** file for the copy of the Visual SourceSafe database that you are upgrading. For example:

Copy Code

```
<Source name="VSS">  
  ...  
  <VSSDatabase name="c:\ourvss"></VSSDatabase>  
  ...  
</Source>
```

The path must not contain the string **srcsafe.ini**. For example, the following <VSSDatabase name> attribute is incorrect and will cause the **VssUpgrade** command to fail:

Copy Code

```
<Source name="VSS">  
  ...  
  <VSSDatabase name="c:\ourvss\srcsafe.ini"></VSSDatabase>  
  ...  
</Source>
```

▣ <UserMap name> attribute

The Analyze command gathers and compiles data about your Visual SourceSafe users and stores it in an XML file. Optionally, you can specify the path and name of the file where you want this data stored in the <UserMap name> attribute. If

you do not specify this attribute, the Analyze command creates a file that is called **UserMap.xml** and puts it in the current directory.

☒ <ProjectMap> section

In the <ProjectMap> section, specify the path of each Visual SourceSafe project that you want to upgrade in the Source attribute of a <Project> item.

To upgrade all the data in your Visual SourceSafe database, make the <ProjectMap> section match the following example:

Copy Code

```
<ProjectMap>
  <Project Source="$/"></Project>
</ProjectMap>
```

Instead of upgrading your complete Visual SourceSafe database at the same time, you can upgrade selected projects at different times.

Tip

Use this option to avoid blocking your team during upgrade if you have lots of data to upgrade.

The paths in the Source attributes must not overlap. For example, the following <ProjectMap> section is not valid:

Copy Code

```
<ProjectMap>
  <Project Source="$/ProjectA"></Project>
  <Project Source="$/ProjectA/Controller"></Project>
</ProjectMap>
```

☒ <Output file> attribute

In the <Settings> section, in the <Output file> attribute, you can specify the path and name of the file where you want the analysis report to be written. If you do not want to specify this option, you can omit the <Output> tag. In this case, the converter writes the report to a file that is named **VSSAnalysisReport.xml** and puts it in the current directory.

▣<SQL> element

You must specify a database for VssUpgrade to use to store temporary data by adding a <SQL> element to the <Source> section of your Analyze settings file. This element uses the following syntax: <SQL Server="SQL_Server_name"></SQL>.

For example, if you are running VssUpgrade on a computer that is named **FABRIKAM-2**, to use a local SQL Server Express instance on the same computer:

Copy Code

```
<Source name="VSS">  
    ...  
    <SQL Server="fabrikam-2\sqlexpress"></SQL>  
    ...  
</Source>
```

To direct the converter to use SQL Server, for example a server that is named **FabrikamSQLServer**

Copy Code

```
<Source name="VSS">  
    ...  
    <SQL Server="fabrikamsqlserver"></SQL>  
    ...  
</Source>
```

▣Run the Analyze command

1. From Windows Start, run Command Prompt as administrator.

2. In the Administrator: Command Prompt window, enter:

```
VssUpgrade Analyze settings.xml
```

Replace settings.xml with the path and name of the Analyze settings file that you created.

3. When you are prompted, enter the administrator password for your Visual SourceSafe database. If your database has no password, press Enter.

VssUpgrade displays the ongoing status as the Analyze command proceeds. When the process is completed, the system summarizes the results. For example:

Copy Code

```
Analyze complete.  
Analyzed 859 files and 941 folders.  
Warnings 0 and Errors 0  
Pre-migration report file: C:\VSS\migrate\VSSAnalysisReport.xml
```

If there were any warnings or errors, you can find details about them in **ConverterErrors.txt** and **VSSUpgrade.log**.

The Analyze command generates:

- A report that you can use to get details about possible problems and changes that the Migrate command may cause.
- A user mapping file (**Usermap.xml**), which you can use to specify how user data will be converted during the upgrade process.

These files are explained in more detail below.

Review and resolve issues found by the Analyze command

The analysis report provides information about issues in your Visual SourceSafe database that may cause problems during the upgrade process. Try to resolve as many of these issues as possible to minimize problems with the upgrade process, as described in the next section.

Some Files are Checked Out

The report lists files that are currently checked out. The upgrade process does not preserve checkout information. Try to ensure that as many files as possible are checked in before upgrading.

Some Items have Data Integrity Issues

The report lists items whose data integrity has been compromised. The Visual SourceSafe ANALYZE utility may be able to fix these kinds of issues. Items that cannot be repaired will not be upgraded. For more information, see the following pages on the Microsoft Web site:

[ANALYZE Utility](#) and [How to Detect and Fix Database Corruption Errors in Visual SourceSafe](#).

Some Folders in Mapped Projects Contain History that is Not Included in the <ProjectMap> section

If a folder is moved from one project to another in a Visual SourceSafe database, the history of that folder is contained in both the original and current projects. To upgrade such a folder with all of its history, you must upgrade both the original and current projects.

For example, you are upgrading the Visual SourceSafe project Project2. This project contains the folder \$/Project2/FeatureA, which was moved from Project1 at some point in its history.

If your <ProjectMap> section contains...	For example...	Then...
Both projects.	 Copy Code <pre data-bbox="451 1835 1219 1900"><ProjectMap> <Project Source="\$/Project1"></Project></pre>	The folder is upgraded

```
<Project Source="$/Project2"></Project>
</ProjectMap>
```

with its full history.

The project that originally contained the folder but not the project that currently contains it.

 **Copy Code**

```
<ProjectMap>
  <Project Source="$/Project1"></Project>
</ProjectMap>
```

The folder is not upgraded.

The folder is upgraded with its history starting from when it was moved to the current project. The history that occurred before the folder was moved to the current project is not

The project that currently contains the folder but not the project that originally contained it.

 **Copy Code**

```
<ProjectMap>
  <Project Source="$/Project2"></Project>
</ProjectMap>
```

upgraded.

For more information about the <ProjectMap> section of the settings file, see [<ProjectMap> Section](#) earlier in this topic.

Some label names are not supported by Team Foundation version control

The report lists label names that will change when they are upgraded because they contain characters that TFS version control does not support. See [Naming restrictions in Team Foundation](#).

Upgrade your data

After you have run the Analyze command, you are almost ready to upgrade your data. Before you run the Migrate command, you must create a settings file. Optionally, you can specify how user names are upgraded.

Specify How User Names are Upgraded

You can control how user information is upgraded from Visual Source Safe to TFS version control. Specifically, you can specify what user name the Migrate command should associate with each changeset in the history of each item in TFS version control. You do this by editing the user mapping file that was created when you ran the Analyze command, as explained earlier in this topic.

The user mapping file is optional. If you omit the <UserMap name> attribute from your Analyze settings file, each changeset is built in the following manner:

- The User field is set to the name of the account under which VssUpgrade is running.
- The name of the user who performed the action in your Visual SourceSafe database is stored in the Comment field.

Example of a user mapping file

When you run the Analyze command, it compiles data about your Visual SourceSafe users and stores it in an XML file. This file lists every Visual SourceSafe user who has ever performed a version control operation in the Visual SourceSafe projects that you are upgrading.

The following example shows a user mapping file that was created by the Analyze command.

Copy Code

```
<?xml version="1.0" encoding="utf-8"?>  
<UserMappings>
```

```
<UserMap From="Admin" To=""></UserMap>
<UserMap From="Guest" To=""></UserMap>
<UserMap From="Kim" To=""></UserMap>
<UserMap From="Satomi" To=""></UserMap>
<UserMap From="Mark" To=""></UserMap>
</UserMappings>
```

You can specify the To attribute of none, some, or all of the UserMap items in the user mapping file. For example, you could modify the previous example in the following manner:

Copy Code

```
<?xml version="1.0" encoding="utf-8"?>
<UserMappings>
  <UserMap From="Admin" To="NORTHAMERICA\KenM"></UserMap>
  <UserMap From="Guest" To="Test1"></UserMap>
  <UserMap From="Kim" To="EUROPE\KimT"></UserMap>
  <UserMap From="Satomi" To="ASIA\SatomiH"></UserMap>
  <UserMap From="Mark" To=""></UserMap>
</UserMappings>
```

Notice that in the previous example, Guest is mapped to Test1, and no domain is specified. In these cases, VssUpgrade presumes that the account belongs to the default domain.

If you do not specify a <UserMap To> attribute, each changeset is built in the following manner:

- The User field is set to the name of the account under which VssUpgrade ran.
- The name of the user who performed the action in your Visual SourceSafe database is stored in the Comment field.
- If you specify a <UserMap To> attribute and the value is a valid user on your Team Foundation Server, the User field is set to the name of that account. If the value is not a valid user on your Team Foundation Server, VssUpgrade will display an error and end the upgrade process.

■ Create a Migrate settings file

You use the Migrate settings file to specify what Visual SourceSafe data you want to upgrade and to control several aspects of how you want to upgrade it. The easiest way to create this file is to copy the file that you created in

[Create an Analyze settings file](#) earlier in this topic. You then add more data to the file to make it usable by the Migrate command.

The following example shows a Migrate settings file.



```
<?xml version="1.0" encoding="utf-8"?>
<SourceControlConverter>
<ConverterSpecificSetting>
  <Source name="VSS">
    <VSSDatabase name="c:\ourvss"></VSSDatabase>
    <UserMap name="c:\ourvss\migrate\Usermap.xml"></UserMap>
    <SQL Server="SQLInstanceName"></SQL>
  </Source>
  <ProjectMap>
    <Project Source="$/Core" Destination="$/CoreTeamProject"><
    <Project Source="$/ProjectA" Destination="$/ClientTeamProje
    <Project Source="$/ProjectB" Destination="$/ClientTeamProje
  </ProjectMap>
</ConverterSpecificSetting>
<Settings>
  <TeamFoundationServer name="My_Server" port="8080" protocol="ht
  <Output file="c:\ourvss\migrate\logs\ContosoVSSMigrate.xml"></O
</Settings>
</SourceControlConverter>
```

The following information can help you modify the Migrate settings file to specify how the Migrate command will upgrade your data.

■<ProjectMap>

For each <Project> element in the <ProjectMap> section of your Migrate settings file, provide a Destination attribute to specify the path of the location on your Team Foundation Server where you want to upgrade the contents of the project in your Visual SourceSafe database (specified in the Source attribute).

For example, you want to upgrade the contents of ProjectA in your Visual

SourceSafe database into ProjectA at the root of a team project that is called Client.

Copy Code

```
<ProjectMap>  
  <Project Source="$/ProjectA" Destination="$/ClientTeamProject/Pr  
</ProjectMap>
```

For the value in the Destination attribute to be valid, the following conditions must be true:

- The team project in the Destination attribute (in the previous example, the team project is ClientTeamProject) must already be located in the team project collection before you start the upgrade process.
- The path in the Destination attribute of a <Project> element must not overlap the path in the Destination attribute of any other <Project> elements. For example, the following <ProjectMap> section is not valid:

Copy Code

```
<ProjectMap>  
  <Project Source="$/ProjectA" Destination="$/ClientTeamProject  
  <Project Source="$/ProjectB" Destination="$/ClientTeamProject  
</ProjectMap>
```

Note

If the folder in the Destination attribute contains any items, and if the path to any of your Visual Source Safe items overlaps with the version control path to any of these items, the upgrade process will fail.

<TeamFoundationServer> tag

In the <Settings> section, add a <TeamFoundationServer> tag, and specify the name, port, protocol, and path to the team project collection on your Team Foundation Server by using the following format:

```
<TeamFoundationServer name="ServerName" port="PortNumber" protocol='
```

☐ **<Label migrate="false" /> tag**

If your Visual SourceSafe database contains many labels that are applied to many files, the upgrade process may be prolonged. If your team does not need this data, you can configure VssUpgrade to ignore labels by adding the `<Label migrate="false" />` tag to the `<Settings>` section.

☐ **<Output file> attribute**

In the `<Settings>` section, in the `<Output file>` attribute, you can specify the path and file where you want the upgrade report written. If you do not include the attribute, the converter writes the report to a file that is named **VSSMigrationReport.xml** and puts it in the current directory.

☐ **Run the Migrate command**

1. From Windows Start, run Command Prompt as administrator.
2. In the Administrator: Command Prompt window, enter:

```
VssUpgrade Migrate settings.xml
```

Replace `settings.xml` with the path and name of the Migrate settings file that you created.

The Migrate command displays each project that you are upgrading from your Visual SourceSafe database and each folder into which the data will be upgraded on your server for Team Foundation version control.

3. When you are prompted, enter the administrator password for your Visual SourceSafe database. If your database has no password, press Enter.
4. The system summarizes how it will run the upgrade process and asks you to confirm that you want to proceed. For example:

Copy Code

This will start migration with following inputs:

SourceSafe Folders -> Team Foundation Server Folders

\$/ -> \$/Sequence2

Full history migration of VSS Database: c:\vss

Team Foundation Server: http://fabrikam-4:8080/tfs/DefaultCollection

Migration Settings File: migrate_settings.xml

Migration Report: C:\VSS\migrate\VSSMigrationReport.xml

Depending on the VSS database size, migration may take few hours.

Please verify all inputs are correct and confirm.

Start migration (Y/N)?

Press **Y** to proceed.

VssUpgrade displays the ongoing status as the Migrate command proceeds.

When the process is completed, the system summarizes the results. For example:

Copy

```
Verification of pinned versions: 0 file(s) added, 0 file(s) edited &
Verification of latest tip versions: No change was required (No fix
```

```
Started: Creating label "Pinned_Latest".
```

```
Finished: Creating label "Pinned_Latest".
```

```
Migration complete.
```

```
Migrated 6397 Actions
```

```
Warnings 7 and Errors 0
```

```
Post migration report file: C:\VSS\migrate\VSSMigrationReport.xml
```

If there were any warnings or errors, you can find details about them in

ConverterErrors.txt and **VSSUpgrade.log**.

The Migrate command generates a report that you can use to get details about possible problems and changes that the Migrate command may have caused. See

[Take next steps after upgrading from Visual SourceSafe.](#)

▣ Resume the process by using incremental upgrade

If the upgrade process is interrupted for some reason, you can resume the process as an incremental upgrade from the point at which the process ended. An incremental upgrade can be useful if the upgrade process failed because of an error or network problems. During incremental upgrade, the converter will upgrade only the data that was not upgraded in previous sessions.

To start an incremental upgrade, follow the steps in

[Run the Migrate command](#). When the Migrate command asks whether you want to perform an incremental migration, press **Y**.

Limitations of an incremental upgrade

An incremental upgrade will not succeed unless you comply with the following restrictions:

- In your Visual SourceSafe database, you must not have performed destroy, purge, archive, or restore activities.
- You must not change the <ProjectMap> section of your Migrate settings file.
- On your Team Foundation Server, you must not modify any folders (or any content in the folders) that are specified in the <ProjectMap> section of your Migrate settings file.

Next step

View and verify the results, and resolve issues as necessary



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Take next steps after upgrading from Visual SourceSafe

[Send Feedback](#)

After the upgrade is finished, you should make sure the results meet your expectations and that the data is ready for your team to use:

- [Check the outcomes from the upgrade process](#)
- [Resolve problems](#)
 - [Resolve failure caused by storage limit for SQL Server Express](#)
 - [Convert files in the MS-DOS-Compatible short name \(8.3\) format \(TF227014\)](#)

▣ Check the outcomes from the upgrade process

Review the report and, if necessary, the logs that were generated by the upgrade tool. Check the data on your Team Foundation server to make sure that the data from your Visual SourceSafe database was upgraded in the way that you expected.

▣ View the Migration Report

TFS and Visual SourceSafe have significant functional differences. As a result, the Visual SourceSafe upgrade tools modify certain kinds of data during the upgrade. And in some cases, errors or problems could occur. The migration report can advise you about these issues.

To open the migration report:

- If you used the VSS Upgrade Wizard: On the Complete page, choose the Upgrade Report link.
- If you used the VssUpgrade tool: Open the **VSSAnalysisReport.xml** report in the current directory, or whatever file name and location you specified using the <Output file> attribute.

Migration Report

Summary

Status	Migration completed. (0 Errors 7 Warnings) Click here for post migration steps
Type of Migration	Full
Source Control	Visual SourceSafe
VSS Location	C:\VSS\srcsafe.ini

VSS Folder	Team Foundation Folder
\$/Coverter	\$/Sequence/Coverter

Warnings

Label Name Change Warnings
TF227033: Label names were changed as a result of migration because they contain characters not supported by Team Foundation version control, or they contain more than 64 characters. Team...

VSS Label	Team Foundation Label
Labeling Version No. : 0	Labeling Version No. # 0
Labeling Version No. : 1	Labeling Version No. # 1
Labeling Version No. : 2	Labeling Version No. # 2

Information

User Input

View logs

If warnings or errors were reported, you might want to view the logs. You might also want to view the logs to get details on how your data was modified during the upgrade process. To view the logs:

- If you used the VSS Upgrade Wizard: On the Complete page, choose the Complete Log link.
- If you used the VssUpgrade tool: Open **ConverterErrors.txt** and **VSSUpgrade.log**.

Examine the data in your team project

Use source control explorer to examine the data on your Team Foundation server to make sure that the data from your Visual SourceSafe database was upgraded in the way that you expected.

Resolve problems

After you have examined the outcome of the upgrade process, you might have to troubleshoot problems.

Resolve failure caused by storage limit for SQL Server Express

The upgrade tools use a database to store temporary metadata. This metadata typically requires a small percentage of the total size of the data that you upgrade. In the unlikely event that your upgrade fails because the 4-GB limit of SQL Server Express is reached, you can use SQL Server instead of SQL Server Express. See

[Upgrade from Visual SourceSafe using the wizard](#) or VssUpgrade <SQL> Element.

Convert files in the MS-DOS-Compatible short name (8.3) format (TF227014)

TFS does not allow file names that are in the MS-DOS-compatible short name (8.3) format (for example, abcdef~1.txt). When you analyze or attempt to upgrade files that have such a name, a TF227014 error appears.

To work around this issue, you can temporarily apply a setting to your Team Foundation Server that will cause it to allow files that have such names. To do this, you must set **Allow8Dot3Paths** to **True** in the configuration database for TFS.

Important

To avoid issues with client machines that support MS-DOS-compatible short names, after you complete the upgrade process, it is strongly recommended that you set **Allow8Dot3Paths** to **False** as described in the following procedure.

To perform the following procedure, Windows PowerShell must be enabled on

the application-tier server for TFS. See

[Scripting with Windows PowerShell](#).

Required Permissions

You must be a member of the Administrators group on the application-tier server for TFS. For more information, see [Permission reference for Team Foundation Server](#).

To upgrade a Visual SourceSafe database that contains files that are named in the MS-DOS-compatible short name format

1. Log on to the application-tier server for Team Foundation.
2. Create a Windows PowerShell script that is called **Allow8Dot3Paths**.
 1. Copy the text in [Allow8Dot3Paths PowerShell Script](#) later in this topic, and paste the text into the script.
 2. Change ServerPath to match the path in the URL that you use to connect to Team Foundation Server. By default, the server path is "tfs".
 3. Change CollectionName to match the name of the team project collection into which you are upgrading your data (for example, DefaultCollection).

The end result, for example, would be the following line in the script:

Copy Code

```
$collectionBaseUrl = "http://localhost:8080/tfs/DefaultColl
```

3. Run the **Allow8Dot3Paths** script.
4. Recycle the application pool for your Team Foundation Server.
 1. From Windows Start run Computer Management.

2. In the navigation pane, expand Services and Applications.
3. Choose Internet Information Services (IIS) Manager, expand the local computer, and double-click Application Pools.
4. Open the shortcut menu of the application pool, and then choose Recycle.
5. Run the Migrate command.
6. Modify the **Allow8Dot3Paths** Windows PowerShell script that you created earlier, replacing "true" with "false".
7. Run the modified **Allow8Dot3Paths** script.
8. Recycle the application pool for your Team Foundation Server (as explained above).
9. In Visual Studio, if you are not already connected, then connect to the team project into which you upgraded the data.
10. In Source Control Explorer, rename any files that have names in the MS-DOS-compatible short name (8.3) format.

☰ **Allow8Dot3Paths PowerShell Script**



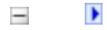
```
# Load client OM assembly.
[Reflection.Assembly]::Load("Microsoft.TeamFoundation.Client, Version=12.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35")

$collectionBaseUrl = "http://localhost:8080/ServerPath/CollectionName"

$tfcs = [Microsoft.TeamFoundation.Client.TeamFoundationServerFactory]::GetServer($collectionBaseUrl)
$collectionHive = $tfcs.GetService([Microsoft.TeamFoundation.Framework.Common.CollectionHive])

# Set some version control settings in the collection hive.
$collectionHive.SetValue("/Service/VersionControl/Settings/Allow8Dot3Paths", "false")

# Display all version control settings as a table.
$collectionHive.ReadEntries("/Service/VersionControl/Settings/...")
```



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Technical Reference for Team Foundation

[Send Feedback](#)

This technical reference provides you with reference material useful in understanding, managing, configuring, and customizing your deployment of Team Foundation Server (TFS).

In this guide

Title	Description
Architecture reference	Understand the underlying architecture of TFS, including naming conventions, service, database, and client architecture, and compatibility with different versions of clients.
Permission reference for Team Foundation Server	Describes TFS permissions, their default assignments to each of built-in TFS group, and the tools you can use to set permissions. There are three categories of built-in groups, four permissions levels, and five permission states.
Command-line tools for TFS Team Foundation Build Commands	Command-line tools you can use to manage the various aspects of TFS.

[Work item field and database schema reference](#)

A description of each field used to track work items, and more.

[XML element reference](#)

Modify the XML to customize your team project to support specific processes and practices that your team uses.

Extending Visual Studio Application Lifecycle Management
Customize some aspects of Visual Studio to extend existing features or to add new capabilities if you have special requirements.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Compatibility between Team Foundation clients and Team Foundation Server
[Send Feedback](#)

Whether you have upgraded Visual Studio or Team Foundation Server (TFS), you'll want to make sure that you can connect from your version of Visual Studio or Team Foundation client to the version of TFS that's running in your deployment. In general, follow this guidance:

Connecting to TFS:

- [Visual Studio Online](#) always runs the latest version of TFS.
- You can connect to the latest version of TFS from earlier versions of Team Explorer although you might need to install additional software. The current version of Team Explorer installs with each version of Visual Studio.
- To connect to Visual Studio Team Foundation Server 2013 from Visual Studio 2008 or Team Explorer for Visual Studio 2005 Team System requires installation of [Microsoft Source Code Control Interface \(MSSCCI\) Provider 2013](#). This configuration supports users in accessing Team Foundation version control from these earlier client versions.
- When you connect to a more recent version of TFS than that of the client that you run, you'll only be able to access those features supported by your client. For example, if you connect Visual Studio 2010 to TFS 2013, you can perform the same functions as if you connected to TFS 2010. You can't access any new features that Visual Studio 2010 doesn't support.
- To create a team project or manage process templates on an on-premises TFS, you must connect using the same version level of Visual Studio or Team Explorer. That is, to create a team project on TFS 2013, you must

connect from Team Explorer 2013.

- For information about compatibility and deploying a build server, see [Deploy and configure a build server](#).

You can work around most limitations by installing Team Explorer 2013 in a side-by-side configuration with an earlier version of Visual Studio or by using Team Web Access (TWA). Use Team Explorer 2013 to support TFS administrative and project administrative tasks.

Team Explorer is available as a [free download](#).

Clarification of terminology:

- Earlier versions refer to software that was made available with versions of TFS prior to Visual Studio Team Foundation Server 2013, the current version.
- Forward compatibility refers to how earlier versions of a client support the current version of Team Foundation Server, sometimes with the addition of a software update. Backward compatibility refers to how the current version of a client supports earlier versions of Team Foundation Server. A General Distribution Release (GDR) software update is available to support forward compatibility of Team Explorer.

Team Web Access and version compatibility

TWA is automatically installed and configured when you install or upgrade Team Foundation Server. Version compatibility is not an issue because all instances of Team Foundation Server are accessed by TWA services that are hosted on that server. To connect to TWA, use any of these browsers:

-

[Internet Explorer](#), versions 9, 10, and 11

- [Mozilla Firefox](#), latest version
- [Google Chrome](#), latest version
- [Safari](#) (supported on Macs only), versions 5, 6, and 7

To learn about the new features available through TWA with the upgrade to TFS 2013, go [here](#).

Visual Studio Online client support

To connect to Visual Studio Online, use any of these clients.

Visual Studio version	Team Explorer version	Required software
Visual Studio 2013	Team Explorer 2013	RTM release or latest update.
Visual Studio 2012	Team Explorer 2012	Requires latest Visual Studio 2012 update. Supports Git with Visual Studio Tools for Git extension .
Visual Studio 2010	Team Explorer 2010	Requires SP1 and Compat GDR .
Visual Studio 2008	Team Explorer for Visual Studio Team System 2008	Requires SP1 and Compat GDR . Version control is available using MSSCCI 2013 in addition to the Compat GDR .

For an overview of new features, see [Features Timeline](#).

TFS 2013 client support

To connect to a server that's running the current version of TFS, use any of these clients.

Visual Studio version	Team Explorer version	Required software
Visual Studio 2013	Team Explorer 2013	RTM release or latest update.
Visual Studio 2012	Team Explorer 2012	Requires latest Visual Studio 2012 update. Supports Git with Visual Studio Tools for Git extension .
Visual Studio 2010	Team Explorer 2010	Requires SP1 and Compat GDR .
Visual Studio 2008	Team Explorer for Visual Studio Team System 2008	Version control is available using MSSCCI 2013 .

Visual
Studio
2005

Version control
available using
[MSSCCI 2013](#). XP
users need to use
[MSSCCI 2010](#).

For an overview of new features, see [What's new for Application Lifecycle Management in Visual Studio 2013](#).

TFS 2012 client support

You can connect to a server that's running TFS 2012 from any of these clients:

Visual Studio version	Team Explorer version	Required software
Visual Studio 2013	Team Explorer 2013	RTM release or latest update
Visual Studio 2012	Team Explorer 2012	RTM release and latest update . Supports Git with Visual Studio Tools for Git extension .
Visual Studio 2010	Team Explorer 2010	Requires SP1 and Compat GDR .
Visual Studio 2008	Team Explorer for Visual Studio Team System 2008	Requires SP1 and Compat GDR .

Visual
Studio
2005

Limited access
available using
[MSSCCI 2012](#).
XP users need to
use [MSSCCI
2010](#).

Here are the client-side tasks that are available if you connect to TFS 2012 or TFS 2013 from a Visual Studio 2012 client:

Feature area	Availability
My Work, Pending Changes, and Request Code Review	These features are only available from Team Explorer 2012 and Team Explorer 2013.
Code analysis check-in policies	There are several differences in the way check-in policies are authored and evaluated when you use different versions of Team Explorer. See Version Compatibility for Code Analysis Check-In Policies .
	You cannot use local workspaces with earlier versions of Visual Studio.

You must use server workspaces. You can only share server workspaces across multiple clients. See [Decide between using a local or a server workspace](#).

Version control local workspaces and name format

Version Control now uses a display name, such as Jamal Hartnet, when it outputs user information instead of a user ID, such as fabrikamfiber\jamel. You will need to update any dependencies on the command line output format to parse names according to the new format.

Testing

Visual Studio 2012 introduced several changes that impact on compatibility of Microsoft Test Manager, unit tests, coded UI tests, Web performance tests, and load tests. See [What's new in testing the application in Visual Studio 2013](#).

Project

You must use Team

creation and process template management Explorer 2012 to create a team project or upload and download process templates when connecting to TFS 2012.

Manage groups and security To manage groups, group membership, and security permissions, you must use Team Web Access.

To complete the following tasks, you must connect to TFS 2012 or more recent version:

- Finds parent items based on filters specified for the child items, also known as bottom-up tree queries.

Work item tracking

- Find work items by filtering on text fields that are indexed for full-text search, which correspond to: Description, History, Steps to Reproduce, and Title that use Contains Words or Does Not Contain Words operators. See Query fields, operators, values,

and variables.

- Link work items to models and storyboards using the Model and Storyboard link type.
- Set project alerts.

Extensibility

The extensibility model for Team Explorer has been completely redesigned. Custom controls based on the earlier versions will need to be rewritten. You can view work item forms that contain new custom controls, but they display as missing. The location on the form where the control is missing displays as red, with the path of the missing control displayed. To view all new work item form controls, use the current version of Team Explorer in a side-by-side configuration, or use Team Web Access.

TFS 2010 client support

To connect to a server that's running TFS 2010, use any of these clients:

Visual Studio version	Team Explorer version	Required software
-----------------------	-----------------------	-------------------

Visual Studio 2013	Team Explorer 2013	RTM release or latest update
--------------------	--------------------	------------------------------

Visual Studio 2012	Team Explorer 2012	Requires latest Visual Studio 2012 update. Supports Git with Visual Studio Tools for Git extension .
--------------------	--------------------	--

Visual Studio 2010	Team Explorer 2010	Requires SP1 and Compat GDR .
--------------------	--------------------	---

Visual Studio 2008	Team Explorer for Visual Studio Team System 2008	Version control officially supported with MSSCCI 2013. Version control unofficially supported with SP1 and Compat GDR .
--------------------	--	---

Visual
Studio
2005

Requires [Visual
Studio Team System
2005 Service Pack 1
Forward
Compatibility
Update for Team
Foundation Server
2010 \(Installer\)](#).

Version control
available using
MSSCCI 2013. XP
users need to use
MSSCCI 2010.

Team Explorer Everywhere (TEE)

The following table lists version compatibility for TEE. TEE no longer stores credentials for each TFS connection.

Team Explorer Everywhere 2013 also supports connection to

[Visual Studio Online](#).

Clients and tools	Eclipse versions supported	Team Foundation Server 2013	Team Foundation Server 2012	Team Foundation Server 2010	Team System 2008 Team Foundation Server	Visual Studio 2005 System Team Foundation Se
Team Explorer Everywhere 2013	3.5-4.3	Yes	Yes	Yes	No	No
Team Explorer Everywhere 2012	3.5-4.3	Yes	Yes	Yes	Yes	Yes
Team Explorer Everywhere 2010 SP1	3.2-3.6	Yes	Yes	Yes	Yes	Yes
Team						

Explorer Everywhere 2010	3.0-3.5	Yes	Yes	Yes	Yes	Yes
--------------------------	---------	-----	-----	-----	-----	-----

■ **Team Explorer and side-by-side compatibility of Team Foundation clients**

You can run Team Explorer side-by-side with older versions of Team Explorer and Visual Studio. To use these Team Foundation clients, you must install the current version of Team Explorer side-by-side with earlier versions of Team Explorer.

- Microsoft Test Manager
- Microsoft Project
- Microsoft Excel
- PowerPoint Storyboarding

For example, you can start the Office Integration Add-Ins for Excel and Project from the 2005 version, but they will have 2013 functionality when they are side by side. Similarly, you can publish test cases only when you run the two versions of Team Explorer side by side. You must install a side-by-side update to use the earlier versions of Team Explorer with the current versions of Office Add-Ins for Team Foundation Server.

For more information about Office integration, see [Choose the Team Foundation client to support your tasks](#).

Microsoft Source Code Control Interface (MSSCCI) Provider

The MSSCCI power tool lets you can integrate Team Foundation version control with products that do not support integration with Team Explorer. Download the 2013 version of this tool here:

-

[Microsoft Visual Studio Team Foundation Server 2013 MSSCCI Provider 32-bit](#)

- [Microsoft Visual Studio Team Foundation Server 2013 MSSCCI Provider 64-bit](#)

XP users need to use [MSSCCI 2010](#).

MSSCCI Provider 2013 supports Visual Studio Team Foundation Server 2013 and is compatible with these software versions:

- Visual Studio 2008
- Visual Studio 2005
- Visual Studio .NET 2003
- Visual C++ 6 SP6
- Visual Basic 6 SP6
- SQL Server Management Studio

Q: How do I download, install, or upgrade to Visual Studio 2013 or Team Foundation Server 2013?

A: Downloads are available [here](#). To install or upgrade to TFS 2013, see Team Foundation Server install guide or Upgrade Team Foundation Server.

Q: How do I connect to TFS 2013?

A: Go here to learn how to connect to TFS 2013 from TWA, Team Explorer, or TFS Proxy server.

Q: How can I determine the TFS version I'm connecting to?

A: From a web browser and for an on-premises TFS, enter the following address: http://ServerName:8080/tfs/_home/About.

For Visual Studio Online, enter the following address:
http://AccountName.visualstudio.com/_home/About.

Q: What about compatibility between TFS and supporting software?

A: If you are upgrading from TFS 2005 or TFS 2008, you'll want to understand the supported operating systems, SQL Server versions, and other special considerations. Review TFS 2013 Compatibility and Dependencies to understand which versions of the following products and functions that TFS 2013 supports.

- .NET Framework
- Hyper-V
- Office (client products)
- Project Server
- SharePoint Products and Technologies
- SQL Server
- System Center Virtual Machine Manager
- Web browsers

Q: Where can I learn about the features that Visual Studio 2013 supports?

A: Go here for a [capabilities comparison](#) of Visual Studio products.

For an overview of new features, see [What's new for Application Lifecycle Management in Visual Studio 2013](#).

Q: What features have been deprecated in TFS 2013?

A: Not many features have been deprecated from TFS 2012. Go [here](#) for a summary of a few changes that were introduced.

To learn about changes made to TFS process templates, see [Changes made to team projects and default process templates during upgrade of Team Foundation Server](#).

Q: What features require Advanced access in TWA?

A: Agile portfolio management, team rooms, charting query results, request feedback, and test case management tools require users to have Advanced access to TWA. To change a person's access level, see [Change access level](#).

Q: Besides Visual Studio and TWA, what other clients connect to TFS 2013?

A: Go here to learn about all supported Team Foundation clients and the tasks you can perform with each client.

Q: Will my custom work item controls work if I upgrade to TFS 2013?

A: Custom controls for TWA that you developed for TFS 2012 are compatible with TFS 2013. You must rebuild any TWA custom controls for work item forms that were built for TFS 2010 or earlier versions. You must rebuild these controls by using the assembly references in the current version of Team Web Access. To learn about writing new controls, see the following Microsoft websites: [Work Item Custom Control Development in TF Web Access 2012 - Development](#) and [Work Item Custom Control Development in TF Web Access 2012 - Deployment](#).

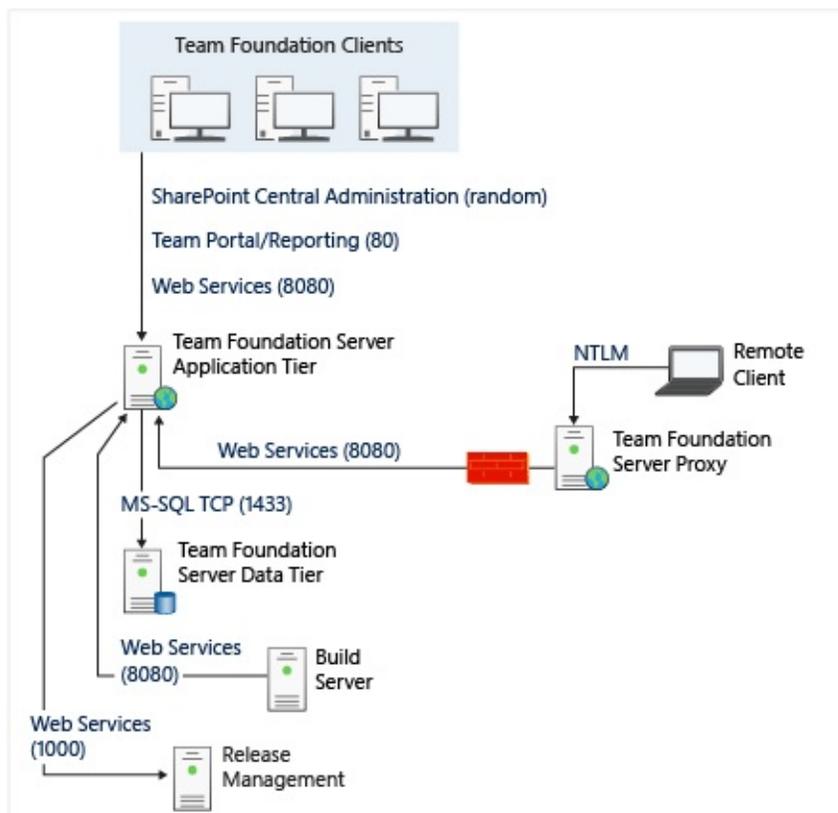


- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Architecture reference

[Send Feedback](#)

Administering Team Foundation Server (TFS) is much easier if you understand its architecture, naming conventions, services, and dependencies.





Title	Description
Naming restrictions in Team Foundation	A reference of naming restrictions in TFS.
Team Foundation Server architecture	Learn about the architecture of TFS, including web services, topologies, ports and protocols, and default settings.
Naming conventions for work item tracking objects	All work item tracking objects have reference names used internally by TFS.
Compatibility between Team Foundation clients and Team Foundation Server	Understand the limitations or restrictions that occur in earlier versions of Team Explorer that connect to Team Foundation Server 2013. You can configure Team

[Configure Team Foundation Build Service to Use Team Foundation Server Proxy](#) Foundation Build Service to retrieve files from Team Foundation Server Proxy by modifying a registry entry on the server that is running Team Foundation Build Service.

[Team Foundation Background Job Agent](#) Learn about the general scheduling mechanism for Web services and jobs for Team Foundation.

[Service accounts and dependencies in Team Foundation Server](#) Service accounts must function correctly for TFS to operate. Learn about how they work in various kinds of deployments, including interactions with other programs.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Team Foundation Server architecture

[See Also](#) [Send Feedback](#)

To best plan and manage your deployment, you should first understand the underlying architecture of Team Foundation Server (TFS). Understanding the architecture can help you maintain the overall health of the deployment and help ensure the overall availability of the servers and services your development teams require.

You can deploy TFS in several ways: on one server; on many servers; or in one domain or workgroup or across domains. Alternatively, you might choose to use Visual Studio Online, where all the server elements of your deployment are hosted for you by Microsoft. Understanding the architecture can help you decide what topology is most likely to meet your business needs. Regardless of your choice of topology, if you understand the architecture underlying TFS, you can better manage the physical and logical requirements. This topic provides a simple overview of the various architectures, with links to more information about example deployments. It also provides technical information about the services, databases, configuration information, and network ports and protocols of local deployments.

To understand the architecture of TFS and how it affects your deployment, you should consider the following:

- The logical application, data, and client tiers of Team Foundation, and whether you want to use one or more servers for the application and data tiers, or whether you want the application and data tiers hosted in the cloud for you by using Visual Studio Online
- The location of the physical or virtual servers that host those tiers
- Team Foundation Build and the number and location of build computers

that will run in your environment, including how many you might need to support your development practices

- The potential need for Team Foundation ServerProxy

In addition, you must consider the interactions between these entities. For example, if you choose to use the hosted TFS service, you must ensure that your clients can access the service on port 443. If you choose to deploy TFS locally, you must know what Web services, databases, and object models TFS uses. Also, you must know which network ports and protocols TFS uses by default and which network ports you can customize. Finally, you must understand what permissions you must set in Team Foundation Server and the components and programs on which your deployment depends.

Besides its own services, Team Foundation Server depends on other services in order to function. For information about these services, see [Team Foundation Server concepts](#) and [Components of the TFS data warehouse](#). For more information about the requirements and dependencies for installation, see Team Foundation Server install guide.

Caution

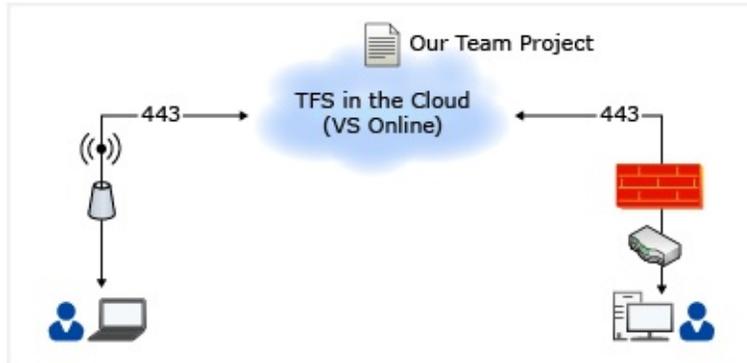
You should not manually modify any of the TFS databases unless you're instructed to do so by Microsoft Support or you're following the procedures described for manually backing up the databases. Any other modifications can invalidate your service agreement.

In this topic

- [Visual Studio Online](#)
- [The Object Model](#)
- [Web Services and Databases for Local Deployments](#)
 - [Collection-Level Services](#)
 - [Server-Level Services](#)

- [Data Tier](#)
- [Client Tier](#)
- [Configuration Information](#)
- [Groups and Permissions](#)
- [Network Ports and Protocols](#)
 - [Default Network Settings](#)
 - [Customizable Network Settings](#)

Visual Studio Online



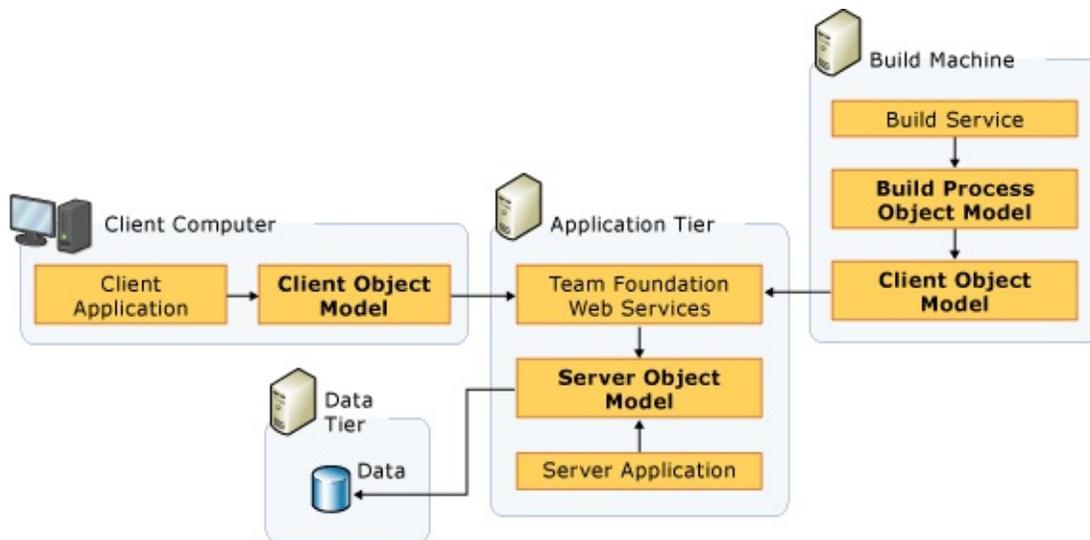
Microsoft offers the option of using Visual Studio Online, where all of the server-side aspects of the deployment are hosted for you. Your source code, work items, build configurations, and team features are all hosted in the cloud. From an architectural point of view, this greatly simplifies your deployment, as the only aspects of the architecture you need to consider are the client components and their Internet access.

When using the service, you use a web browser to connect to the service using your Microsoft account. You can create team projects, add members to your team, and work as you would with a locally installed deployment, without the overhead of administering the servers. Your application tier, data tier, and build servers are hosted for you in the cloud, using the Microsoft Cloud platform and SQL Server Azure.

For more information, see [Examples of Hosted Topology](#).

The object model

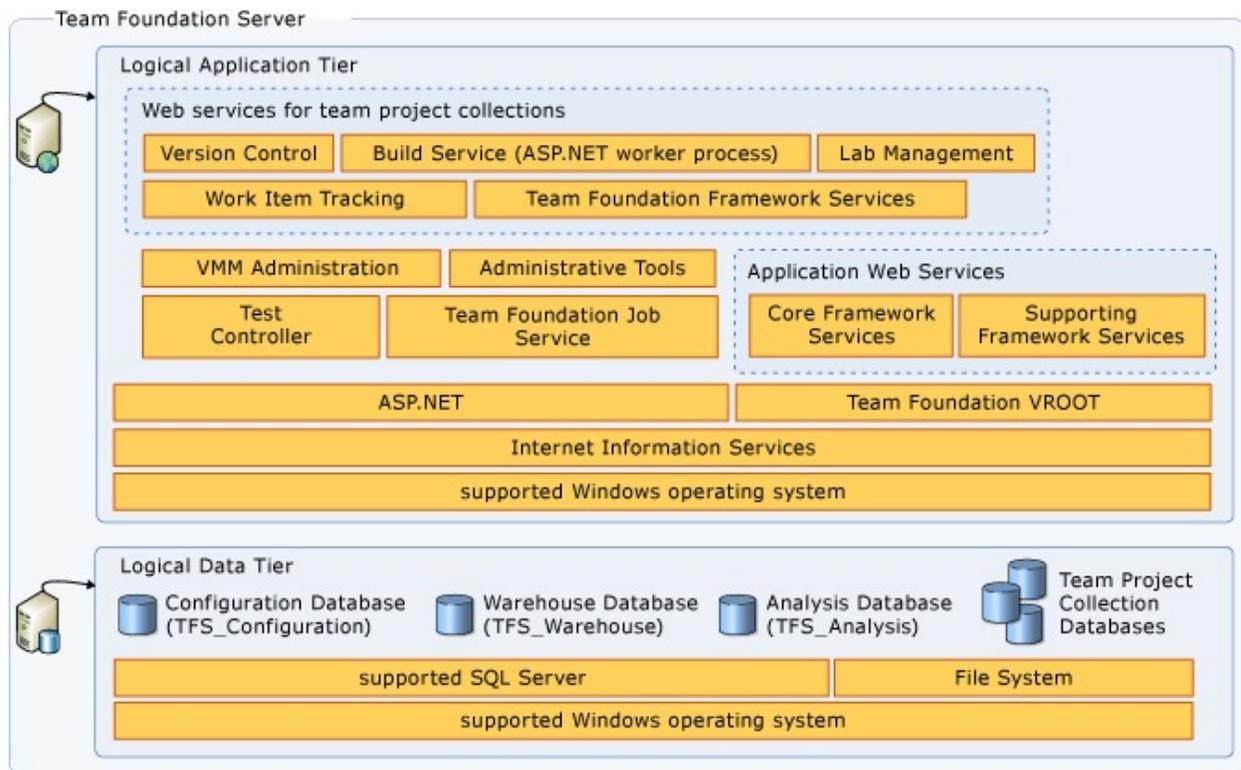
With either the hosted or the locally-deployed architecture, you can extend the features and functionality of Team Foundation by writing an application that is based on its server or client object model. In all deployment types, you can write applications that extend client capabilities. However, if you want to extend server capabilities, your application must run on the application-tier server. To extend the client capabilities, you must run the application on the same computer as Team Explorer.

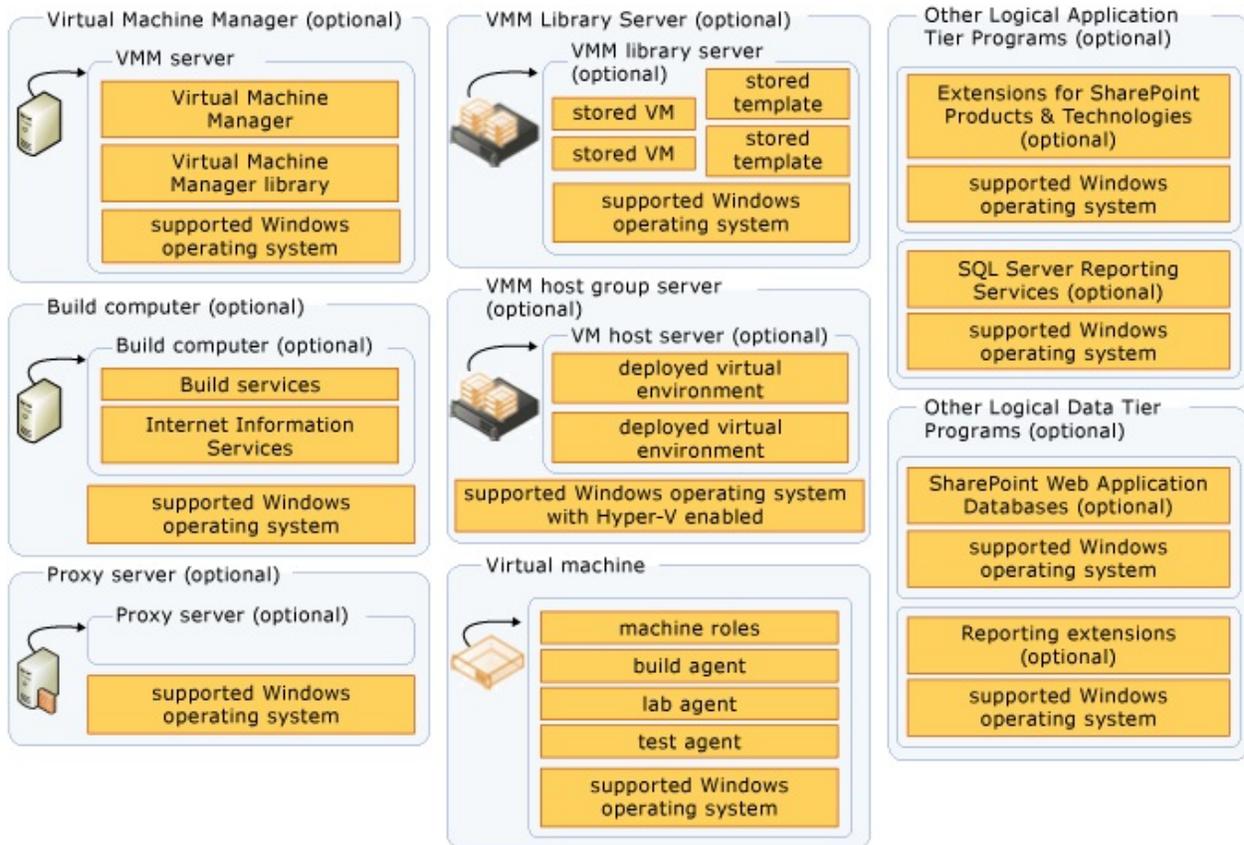


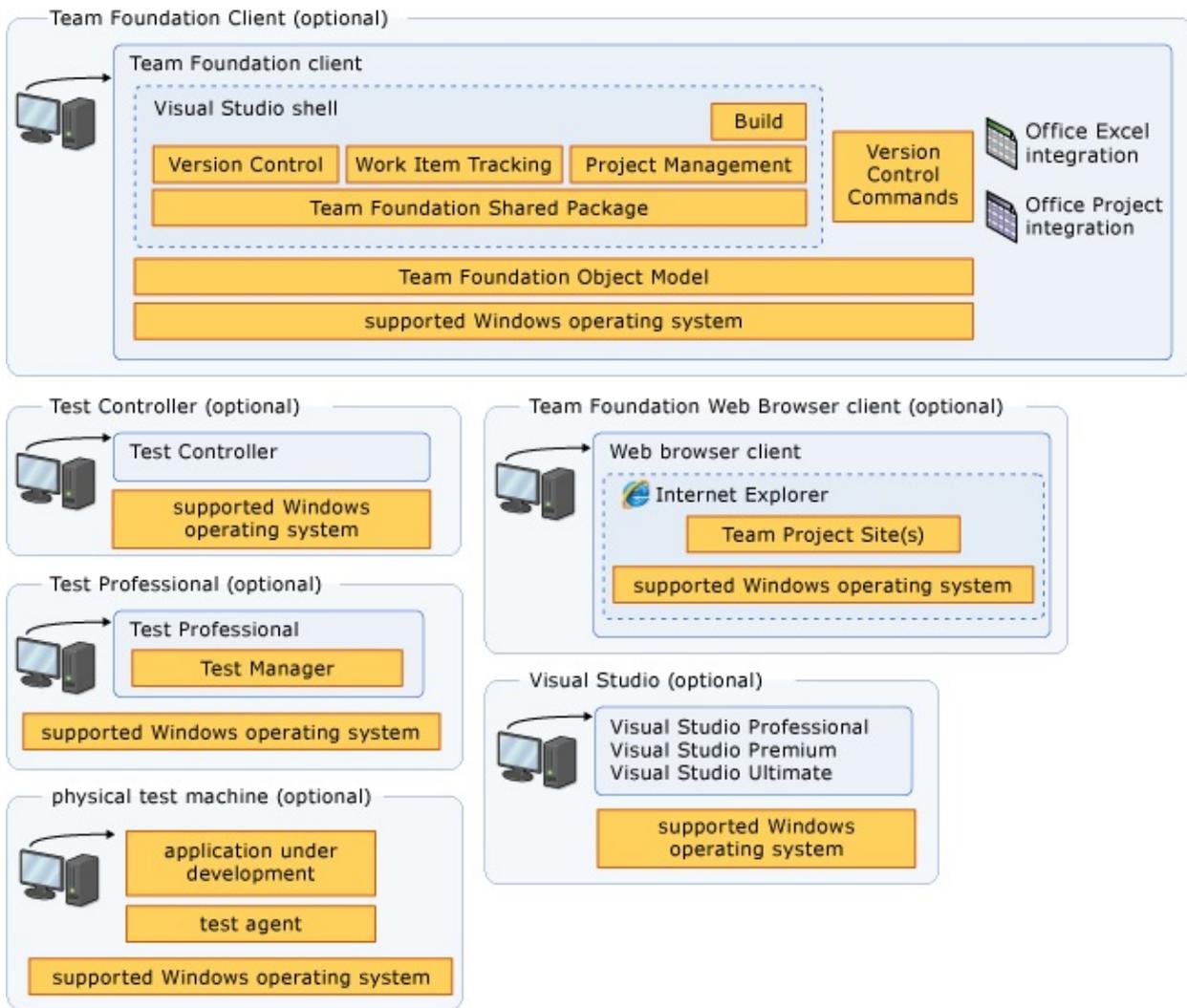
For more information, see [Extending Team Foundation](#).

Web services and databases for local deployments

Team Foundation Server includes a set of Web services and databases that you install and configure separately on the server or servers that host the logical application, data, and client tiers for Team Foundation. Some features, such as the task board, and backlog team-based features, are entirely web-based and accessed solely through Team Web Access, a client-side web based service. Others, such as the version control features, can be accessed through either Team Web Access or through a client application. The following illustrations provide a high-level view of web services, applications, and databases for local deployments of TFS.







Collection-level services

Collection-level services provide the functionality for operations at the level of the team project collection. You can create applications that extend Team Foundation Server by using some of these services. For more information about creating applications for TFS, see [Extending Team Foundation](#).

Tip

Some services appear in more than one level. For example, the Registry service functions at the collection level and the server level, and appears in both lists.

- Team Foundation Framework Services

- Registry service
- Registration service (for compatibility with earlier versions of Team Foundation Server)
- Property Service
- Event Service
- Security service
- Location service
- Identity Management service
- Version Control Web service
- Work Item Tracking Web service
- Team Foundation Build Web service
- Lab Management Web service
- VMM Administration Web service
- Test Agent Controller Web service

■ **Server-level services**

Server-level services (also known as application-level services) provide the functionality for operations for Team Foundation Server as a software application. You can create applications that extend Team Foundation Server by using some of these services. For more information, see [Extending Team Foundation](#).

- Team Foundation Framework Services
 - Registry service
 - Event service

- Team Project Collection service
- Property service
- Security service
- Location service
- Identity Management service
- Administration Service
- Collection Management Service
- Catalog Service

■Data tier

The data tier includes data, stored procedures, and other associated logic. When you use Visual Studio Online, the data tier is hosted for you using SQL Server Azure. In a local deployment of TFS, the logical data tier consists of the following operational stores within SQL Server. These stores might be located on one physical server or distributed across many servers. You can create applications that extend Team Foundation Server by using some of these operational stores. For more information, see [Extending Team Foundation](#).

- Configuration database (TFS_Configuration)
- Application warehouse (TFS_Warehouse)
- Analysis Services database (TFS_Analysis)
- Databases for team project collections (TFS_CollectionName)

The following table provides a list of the databases that Team Foundation Server uses in local deployments. Unless otherwise noted, you can move all databases in this list from the original server and instance where they are installed and restore them to another server or instance.

|

Database Name and Description	Server
TFS_Configuration This database stores the catalog of resources and the configuration information for Team Foundation Server. This database contains the operational stores for Team Foundation Server.	Instance of SQL Server that is used when Team Foundation Server is installed and configured.
TFS_Warehouse This database stores the data for reports.	Instance of SQL Server that is used when Team Foundation Server is installed and configured.
TFS_Analysis This multi-dimensional database stores the aggregated data from team project collections.	Instance of SQL Server that is used when Team Foundation Server is installed and configured.
Databases for team project	Instance of SQL

collections for each team project collection, containing data from all team projects in that collection.	One database Server that is compatible with Team Foundation Server.
--	---

≡ **Client tier**

The client tier communicates with the application tier through the server object model, and uses the same Web services that are listed for that tier. This is true whether you deploy TFS locally, or if you use Visual Studio Online. Besides that model, the client tier consists of Visual Studio Industry Partners (VSIP) components, Microsoft Office integration, command-line interfaces, and a framework for check-in policies.

▣ **Configuration information**

The hosted service depends on the client services, deployed locally, and an Internet connection to the application and data tiers hosted in the cloud. A local deployment of Team Foundation Server depends on SQL Server, Internet Information Services (IIS), and the Windows operating system. Contingent on your chosen topology, Team Foundation Server might also depend on SQL Server Reporting Services or SharePoint Products. Therefore, configuration information for Team Foundation Server can be stored in any of the following locations:

- IIS data stores.
- Configuration files for Team Foundation Server.
- Data sources for Reporting Services (for example, TFSREPORTS data).
- Configuration database for Team Foundation Server. The Team Foundation Server registry is part of the configuration database.
- Windows Registry.

For examples of different local deployment topologies and where these resources are stored, see [Examples of Simple Topology](#), [Examples of Moderate Topology](#), and [Examples of Complex Topology](#). As you maintain a local deployment of Team Foundation Server, you must take these configuration sources into account. To change the configuration in any way, you might need to modify information that is stored in multiple locations. You might also need to change configuration information for the data and client tiers. Team Foundation Server includes an administration console and several command-line utilities to help you make these changes. For more information, see

[Configure and manage TFS resources](#).

▣ **Synchronization of group identities between Active Directory and Team Foundation Server**

In local deployments where Team Foundation Server is running in an Active Directory domain, group and identity information is synchronized when any of the following events occur:

- The application-tier server for Team Foundation starts.
- An Active Directory group is added to a group in Team Foundation Server.
- The period of time that is specified in the scheduled job elapses. The default is one hour, and all groups in Team Foundation Server update every 24 hours.

Identity Management Services (IMS) synchronizes with Active Directory, and changed identities propagate from the server to the clients. By default, all groups update within 24 hours, but you can customize this to better suit the needs of your deployment. For more information, see [Trusts and Forests Considerations for Team Foundation Server](#). For local deployments that do not use Active Directory, see [Managing Team Foundation Server in a Workgroup](#).

Groups and permissions

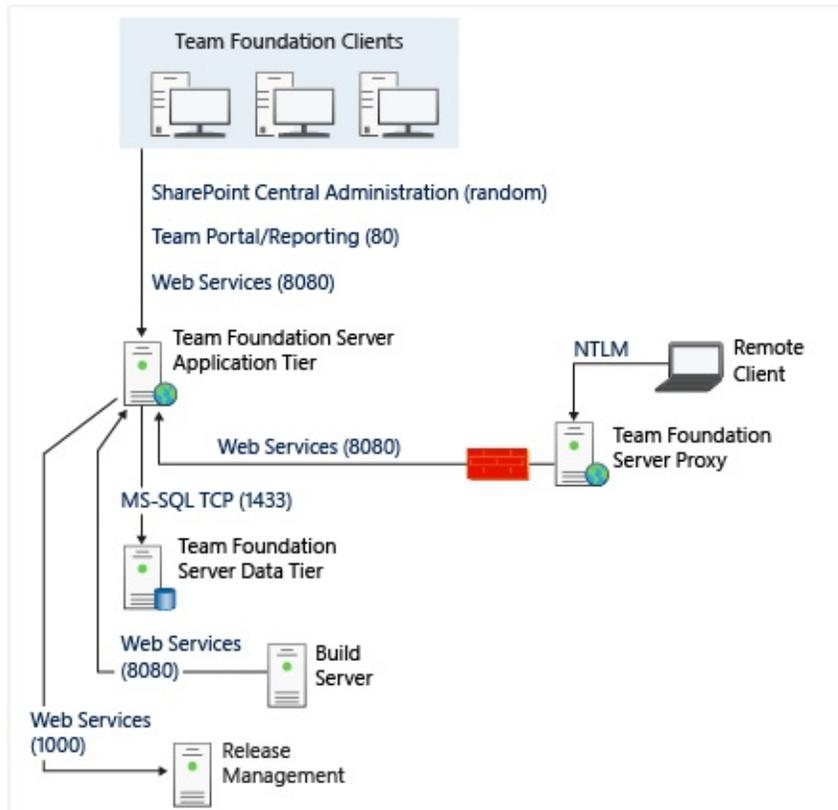
In a local deployment, Team Foundation Server has its own set of default groups and permissions that you can set at the project, collection, or server level. You can create custom groups and customize permissions at group and individual levels. However, users or groups that you add to Team Foundation Server are not automatically added to two components on which local deployments of Team Foundation Server can depend: SharePoint Products and Reporting Services. If your deployment uses these programs, you must add users and groups to them and grant the appropriate permissions before those users or groups will function correctly across all operations in Team Foundation Server. For more information, see

[Manage users or groups in TFS.](#)

For hosted deployments, access is controlled through a combination of Microsoft accounts and team membership. For more information, log onto the service with your Microsoft account and [choose Learn](#).

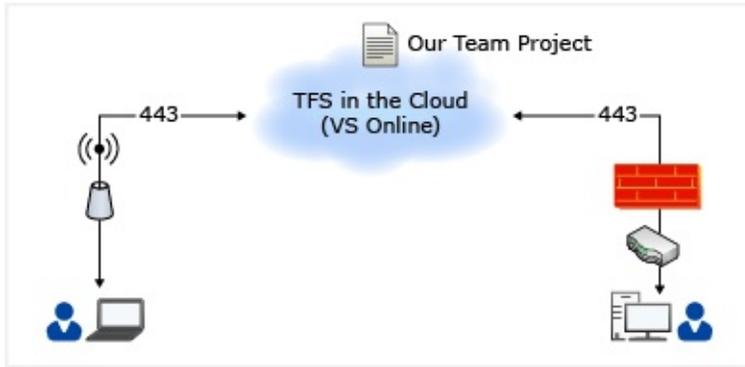
Network ports and protocols

By default, a local deployment of TFS is configured to use specific network ports and protocols. The following illustration shows network traffic for Team Foundation Server in a simple deployment.

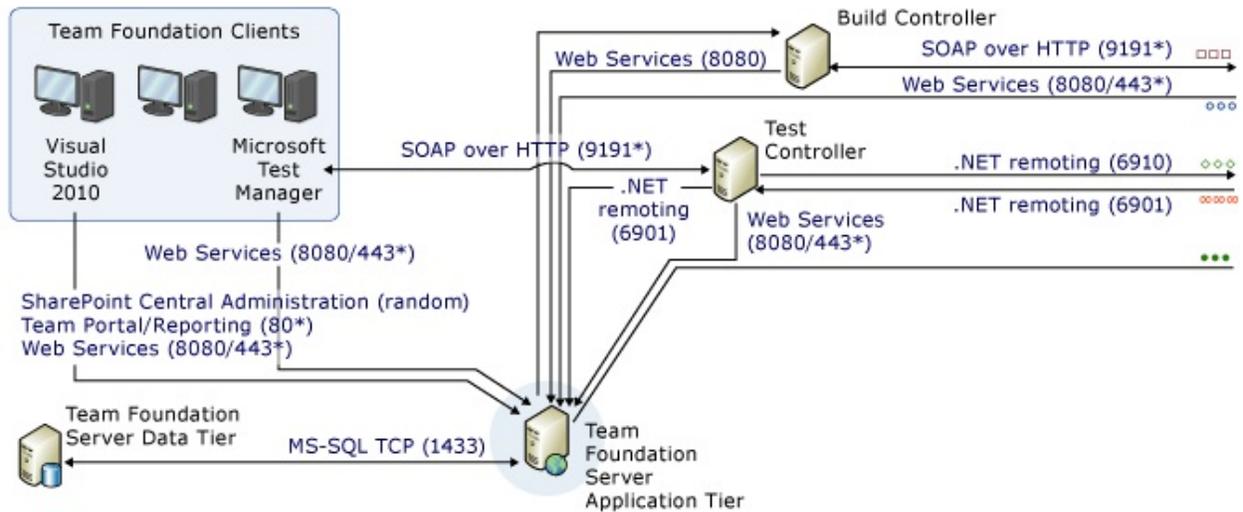


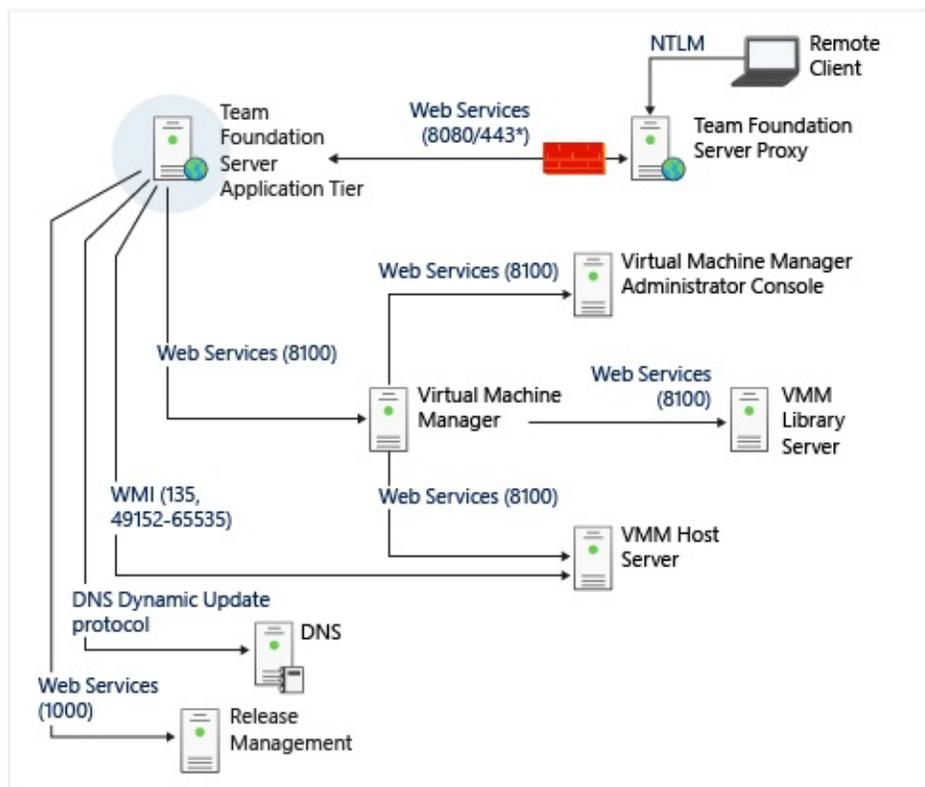
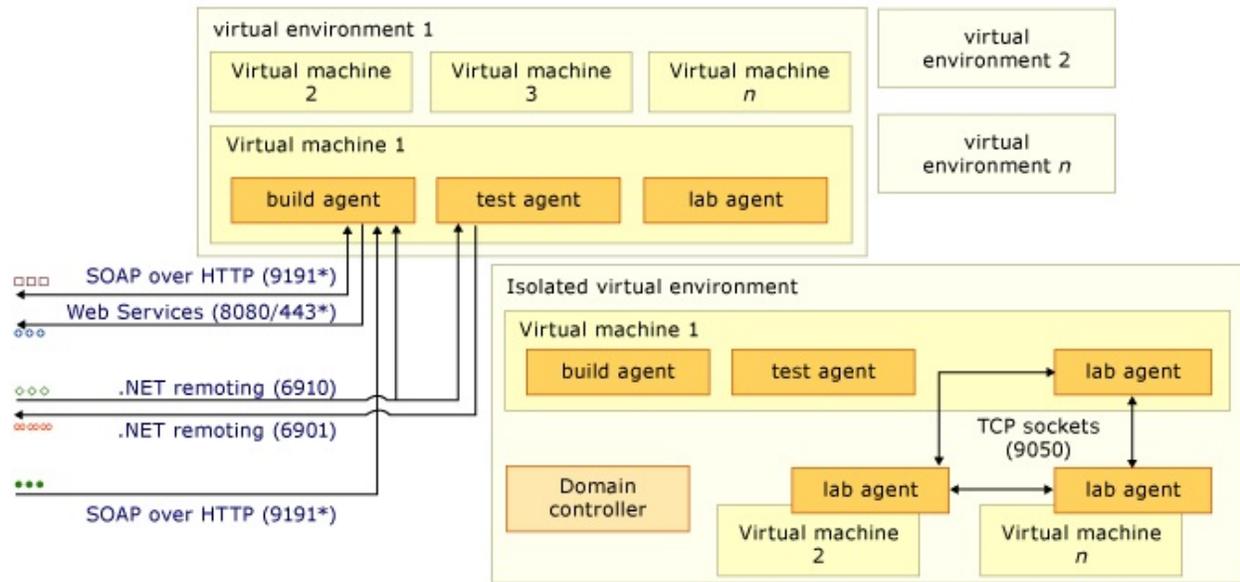
Hosted TFS

Similarly, the hosted service for TFS is configured to use specific network ports and protocols. The following illustration shows network traffic in a hosted deployment.



The following illustration shows network traffic in a more complex deployment that includes the components for Visual Studio Lab Management.





Virtual machines use port 80 to communicate with any test controller about the download of a lab management agent. Check that this port is enabled if you are having any communication issues.

Default network settings

By default, communication between the computers in a deployment of Team Foundation uses the protocols and ports shown in the following table. If an asterisk (*) follows the port number, you can customize that port.

Tier and service	Protocol	Port
Application tier - Web Services	HTTP/HTTPS	8080/443*
Application tier - SharePoint Products Administration	HTTP	17012* if SharePoint Products was installed with Team Foundation Server; otherwise, randomly generated
Application tier - SharePoint Products and Reporting Services	Windows Management Instrumentation (WMI) service (required during installation to specify and verify the URLs for reporting)	80* Dynamic port

		services)	
Data tier	MS-SQL	TCP	1433*
			default (2382 or 2383)*
Data tier (SQL Server Analysis Services)	MS-AS		The default port varies depending on the version of SQL Server you installed and the type of instance. Use SQL Server Configuration Manager to determine the ports used by your deployment.
Team Foundation Server Proxy - client to proxy	HTTP		8081*
Team Foundation Server Proxy - proxy to application tier	HTTP/HTTPS		8080/443*

Client tier - Reporting Services	HTTP	80*
Client tier - Web services	HTTP/HTTPS	8080/443*
Build controller to application tier	HTTP/HTTPS	8080/443
Build agent to application tier	HTTP/HTTPS	8080/443
Release Management Server	HTTP or HTTPS	1000*
Release Management Client	HTTP or HTTPS	1000*
Release Management Agent	HTTP or HTTPS	1000*
Test controller to application tier	HTTP/HTTPS	8080/443*

Application tier to test controller	.NET remoting	6901*
Application tier to Domain Name System (DNS)	DNS Dynamic Update	53
Application tier - Virtual Machine Manager	HTTP	8100
Test controller to test agent	.NET remoting	6910*
Test agent to test controller	.NET remoting	6901*
Build controller to build agent	SOAP over HTTP	9191
Lab agent to lab agent in an isolated environment	TCP sockets	9050
Build agent to build controller	SOAP over HTTP	9191

Virtual Machine Manager Administrator Console - Virtual Machine Manager	HTTP	8100
Virtual Machine Manager-Virtual Machine Manager hosts	Windows Remote Management (WinRM) to perform actions	80 to perform actions
	Background Intelligent Transfer Service (BITS) to transfer data	443 to transfer data
Virtual Machine Manager-Virtual Machine Manager library server	WinRM to perform actions	80 to perform actions
	BITS to transfer data	443 to transfer data
Application tier - Virtual Machine	Distributed Component Object Model/Windows Management	135 Dynamically assigned in

Manager hosts	Interface (DCOM/WMI) communication to transfer data	the range 49152 to 65535
Client tier - Virtual Machine Manager hosts	Host-based connection to the virtual machine.	2179 to perform host-based connections.
Hosted services	HTTPS	443

■ Customizable network settings

As the previous table shows, you can change communication between the application, data, and client tiers in local deployments by modifying Team Foundation Server to use custom ports. The following table describes example changes in ports from HTTP to HTTPS.

📌 Note

To configure Team Foundation Server to use HTTPS and Secure Sockets Layer, you must not only enable ports for HTTPS network traffic but also perform many other tasks. For more information, see

[Set up HTTPS with Secure Sockets Layer \(SSL\) for Team Foundation Server.](#)

Service	Protocol	Port
Web Services with SSL	HTTPS	Configured by the administrator

SharePoint Central Administration	HTTPS	Configured by the administrator
SharePoint Products	HTTPS	443
Reporting Services	HTTPS	443
Client Web Services	HTTPS	Configured by the administrator
Release Management	HTTPS	Configured by the administrator

See Also

Concepts

[Team Foundation Server concepts](#)

[Permission reference for Team Foundation Server](#)

[Components of the TFS data warehouse](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Naming restrictions in Team Foundation

[See Also](#) [Send Feedback](#)

Many components in Visual Studio Team Foundation Server (TFS) have certain naming restrictions. These restrictions help guarantee a consistent user experience and provide compatibility with other programs. These restrictions might include length, special characters, uniqueness, or other attributes.

This topic contains the following subsections:

Server, collection,
account, group,
and machine name
restrictions:

- [Common considerations](#)
- [TFS account names](#)
- [TFS group account names](#)
- [Computer names](#)
- [Team project collection names](#)

Project and work
item tracking name
restrictions:

- [Team project names](#)
- [Team names](#)
- [Work items](#)
- [Work item customizations](#)
- [Process templates](#)

Build and code
name
restrictions:

- [Team Foundation version control](#)
- [Team Foundation Build](#)

Common considerations

The length restrictions in this topic are measured by the number of Unicode characters permitted. For more information about Unicode, see "About Unicode and Character Sets" (

<http://go.microsoft.com/fwlink/?LinkId=76837>). Surrogate characters are composed of two Unicode characters and these will count as two characters against the length restriction.

As with other operating system files, ASCII control characters (ASCII 1-31) and surrogate combinations are also not allowed. For general information about the operating system restrictions applied to file names, see [Naming Files, Paths, and Namespaces](#).

Restrictions for Team Foundation Server account names

User accounts identify users in TFS. These accounts might be Windows user accounts, Active Directory accounts, or other account types. If you want examples of how to add a user to groups used in TFS, SharePoint Products, and SQL Server Reporting Services, see

[Add users to team projects.](#)

When a user is added to TFS, the user name must meet certain Team Foundation Server restrictions. The following table describes these restrictions.

Restriction type	Restriction
Account name length	<ul style="list-style-type: none">• Must not contain more than 256 Unicode characters
Uniqueness	<ul style="list-style-type: none">• Must not match any other name of an account in TFS
Reserved group names	<ul style="list-style-type: none">• Must not be named \$NAMESPACE at either the project or the server level• Must not include the following printable

characters: "/\ [] : | <
> + = ; ? *

Special
character
restrictions

- Must not include nonprintable characters in the ASCII value range of 1-31
- Must not end in a period (.)
- Must not include commas (,)
- Must not include the following Unicode categories:
LineSeparator,
ParagraphSeparator,
Control, Format,
OtherNotAssigned

 **Note**

You add existing user accounts to Team Foundation Server. You cannot create a user account in TFS.

Restrictions for Team Foundation Server group account names

Team Foundation group accounts enable you to apply certain rights or permissions to a group of users. This Team Foundation group can consist of Windows user accounts, Windows group accounts, Active Directory group accounts, Team Foundation group accounts, or any mixture of these types. For more information about managing users with groups, see [Users, groups, and permissions in Team Foundation Server deployments and](#)

[Manage users or groups in TFS](#). If you want examples of how to add a new member to groups used in TFS, SharePoint Products, and SQL Server Reporting Services, see [Add users to team projects](#).

When a Team Foundation group account is created or changed, it must meet certain Team Foundation Server restrictions. The following table describes these restrictions.

Restriction type	Restriction
Group account name length	<ul style="list-style-type: none">• Must not contain more than 256 Unicode characters
Uniqueness (collection-level group accounts)	<ul style="list-style-type: none">• Must not match any other name of a group account in the team project collection
Uniqueness	<ul style="list-style-type: none">• Must not match any

(project-level group accounts) other group name in the same project

Reserved group names

- Must not be named \$NAMESPACE at either the project or the server level

Special character restrictions

- Must not include the following printable characters: "/ \ [] : | < > + = ; ? *
- Must not include nonprintable characters in the ASCII value range of 1-31
- Must not end in a period (.)
- Must not include commas (,)

 **Note**

You do not create user accounts in Team Foundation Server.

Restrictions for computer names

During the TFS installation process, the computer name is associated with the name of the server.

Both the operating system and Active Directory impose certain restrictions on computer names. For more information about renaming a computer, see "Rename a Computer" (

<http://go.microsoft.com/fwlink/?LinkId=76839>). For more information about Active Directory, see "Windows Server 2003 Active Directory" (<http://go.microsoft.com/fwlink/?LinkId=47541>).

Restrictions for names of team project collections

The name of a team project collection identifies a grouping of team projects and the resources that are associated with those projects. A team project collection is an organizing structure that you can use to define and control a group of team projects within TFS. Team members will use the name of the team project collection when they connect to team projects in TFS. For more information, see

[Manage team project collections.](#)

The following table describes the restrictions for names of collections.

Restriction type	Restriction
Length	<ul style="list-style-type: none">• Must not contain more than 64 Unicode characters• Must not be identical to any other collection name in your deployment of Team Foundation Server.
Uniqueness	<ul style="list-style-type: none">• If your deployment includes SharePoint Products or SQL Server Reporting Services, must not be identical to the name

and full path of an existing SharePoint site, report server, or Reporting Services Web site.

- Must not contain any Unicode control characters or surrogate characters.
- Must not contain the following printable characters: / : \ ~ & % ; @ ' " ? < > | # \$ * } { , + = []
- Must not contain an ellipsis (...) or a double period (..)
- Must not start with an underscore (_)
- Must not start or end with a period (.)
- Must not be a system-reserved name such as PRN, COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9, COM10, LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7,

Special characters

Reserved names

LPT8, LPT9, NUL,
CON, AUX, Web, or
WEB

For more information
about reserved names,
see this page on the
Microsoft Web site:
[File Names, Paths,
and Namespaces.](#)

Restrictions for project names

The project names in TFS identify a collection of work items, documents, reports, team builds, and a version control tree that make up a particular project in TFS. Team members will use this project name to connect to the project in TFS.

The following table describes the restrictions for project names.

Restriction type	Restriction
Length	<ul style="list-style-type: none">• Must not contain more than 64 Unicode characters
Uniqueness	<p>Must not be identical to any other name in the team project collection, the SharePoint Web application that supports the collection, or the instance of SQL Server Reporting Services that supports the collection</p> <ul style="list-style-type: none">• Must not contain any Unicode control characters or surrogate characters• Must not contain the following printable characters: / : \ ~ & % ;

Special characters

@ ' " ? < > | # \$ * } { ,
+ = []

- Must not contain an ellipsis (...) or a double period (..)
- Must not start with an underscore (_)
- Must not start or end with a period (.)
- Must not be a system-reserved name such as PRN, COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9, COM10, LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7, LPT8, LPT9, NUL, CON, or AUX

Reserved names

- Must not be one of the hidden segments used for IIS request filtering like App_Browsers, App_code, App_Data, App_GlobalResources, App_LocalResources, App_WebResources, bin, or web.config.
- For more information about reserved names, see

[Naming Files, Paths, and Namespaces.](#)

Restrictions for team names

TFS team names identify a group of individuals or groups that collectively work together as a team in a project. Team members will use this name to connect to the team in Team Web Access. The team name must be a name that can be rendered as part of a valid URL. In addition, each team name must be unique within a single project, but there is no restriction on using the same team name in different projects within a team project collection. For more information about working in teams, see [Add another team or a hierarchy of teams](#).

The following table describes the restrictions for team names.

Restriction type	Restriction
Length	<ul style="list-style-type: none">• Must not contain more than 64 Unicode characters
Uniqueness	Must not be identical to any other name in the project
Special characters	<ul style="list-style-type: none">• Must not contain any Unicode control characters or surrogate characters• Must not contain the following printable characters: / : \ ~ & % ; @ ' " ? < > # \$ * } { , + = []

Reserved
names

- Must not contain an ellipsis (...) or a double period (..)
- Must not start with an underscore (_)
- Must not start or end with a period (.)
- Must not be a system-reserved name such as PRN, COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9, COM10, LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7, LPT8, LPT9, NUL, CON, or AUX
- For more information about reserved names, see

[Naming Files, Paths, and Namespaces.](#)

Restrictions for work items

Microsoft Visual Studio Application Lifecycle Management tracks the progress on a project by using items such as bugs, requirements, tasks, and risks. These items are referred to generically as work items. This section describes restrictions on the data stored in the work items.

Restrictions for work item attachments

Files can be attached to work items. The following table describes the restrictions on work item attachments.

Restriction type	Restriction
File size	<ul style="list-style-type: none">• By default, the work item file attachment size limit is 4,096 kilobytes. This value can be changed by the Team Foundation Administrator. The maximum size that can be set by the Administrator is 2 gigabytes.• Default size: 4,096 kilobytes• Maximum size: 2 gigabytes

Restrictions for work item areas and iterations

Work items contain a field for Project Area and a field for Project Iteration. They are used to organize and display work items into logical groupings.

The Project Area and Project Iteration are paths made up of multiple node items separated by backslash (\) characters. Nodes are defined by the Team Foundation Server administrator to reflect the project areas and project cycle. The following table describes the restrictions on nodes and paths.

Restriction type	Restriction
Node length	<ul style="list-style-type: none">• Must not contain more than 255 characters
Special characters for nodes	<ul style="list-style-type: none">• Must not contain Unicode control characters• Must not contain any one of the following characters: \ / \$? * : " & > < # % +• Must not contain characters prohibited by the local file system. For more information about Windows character restrictions, see

[Naming Files, Paths, and Namespaces.](#)

Reserved names

- Must not consist only of a period (.) or two periods (..)
- Must not be a system-reserved name such as PRN, COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9, COM10, LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7, LPT8, LPT9, NUL, CON, or AUX
- For more information about reserved names, see [Naming Files, Paths, and Namespaces](#).

Path length

- Must not contain more than 4000 Unicode characters

Path hierarchy depth

- Must not be more than 14 levels deep

Restrictions for work item customizations

Team Foundation Server tracks the progress on a project by using items such as bugs, requirements, tasks, and risks. These items are referred to generically as work items.

Administrators of team projects can decide to change work item type definitions either at the project level or in a process template. For more information about how to customize work item types, see [Customize work tracking objects to support your team's processes](#). This section describes restrictions you will encounter when you customize work items and their associated elements.

Restrictions for work item field names

Each work item type contains one or more work item fields. These fields define the information stored in a work item type. A work item field name uniquely identifies each work item field.

The following table describes the restrictions for work item field names.

Restriction type	Restriction
Length	<p>Must not contain more than 128 Unicode characters</p> <ul style="list-style-type: none">• Must not contain leading or trailing spaces• Must not contain two

Special characters	<p>or more consecutive spaces</p> <ul style="list-style-type: none"> • Must not contain periods (.) • Must not contain opening or closing square brackets ([])
Scope	<ul style="list-style-type: none"> • Work item field names are scoped to an entire deployment of TFS. If a field name is changed, the change will be reflected in all projects and the work item types that contain that work item field.

Restrictions for work item field reference names

Each work item field has an associated field reference name. The field reference name uniquely identifies each field and cannot be changed after it is assigned. The following table describes the restrictions applied to field reference names.

Restriction type	Restriction
Length	<ul style="list-style-type: none"> • Must not contain more than 70 Unicode characters

Special characters

- Must not contain hyphens (-)
- Must contain at least one period (.)
- Must not start or end with a period (.)
- Must not start with a number
- Must not start with an underscore (_)

Uniqueness

- Must not be identical to any other field reference name in Team Foundation Server
- Must not be identical to any other field reference name after those names are processed by the computer to replace all periods (.) with underscores (_)

For example, the field reference names My._Field and My_.Field would both be processed by the computer to be the same name:

My__Field

▣ Restrictions for work item field help text

As an option, you can associate help text with work item fields by using the <HELPTEXT> tag. The system displays this text at run time to help users know what to enter into the field. For more information about work item field help text, see

[Apply a rule to a work item field.](#)

The following table describes the restrictions for work item field help text.

Restriction type	Restriction
Length	<ul style="list-style-type: none">• Must not contain more than 255 Unicode characters
Scope	Unlike the field name and field type, field Help text is scoped to a specific work item type in a specific team project.

▣ Restrictions for global lists

A global list is a set of list item values that is stored and used globally by all Team Foundation servers in a Team Foundation Server implementation. As you define work item types, you may find that some work item fields share the same set of possible values. Global lists enable you to define these values one time and share them among multiple work item types. For more information, see [Define global lists.](#)

A global list (GLOBALLIST) contains one or more list items (LISTITEM elements).

The following table describes the restrictions on list items.

Restriction type	Restriction
Length	<ul style="list-style-type: none"> • Must not contain more than 254 Unicode characters • Must not be empty • Must not contain leading or trailing white space
Special characters	<ul style="list-style-type: none"> • Must not contain two consecutive spaces • Must not contain backslash (\) characters
Scope	<ul style="list-style-type: none"> • Since global lists are available among all projects, they must not contain elements defined at the project level, such as project-specific group account definitions.

The following table describes restrictions that apply to a global list.

**Restriction
type**

Restriction

Number of items The global list must not be empty. It must contain at least one LISTITEM element.

Uniqueness

- The global list name must not be identical to any other global list name in Team Foundation Server.

Restrictions for process templates

A process template is a set of default work items, work item queries, product templates, reports, security groups, and guidance that influences the structure of a project in Team Foundation. Team Foundation Server includes two default process templates that encompass two different styles for managing the software cycle. These templates can be customized to reflect the unique needs of your organization. For more information, see [Customize a process template](#).

The following table describes restrictions on the process templates.

Restriction type	Restriction
Process template name length	Must not contain more than 256 Unicode characters.
Process template name uniqueness	<ul style="list-style-type: none">• Must be unique across the Team Foundation server.• If you upload a template with the same name as an existing template, the existing template will be overwritten.
Process template file size	The process template file size must not exceed 2 GB (gigabytes).

Restrictions for Team Foundation Build

Team Foundation Build lets you manage all the aspects of the build process on a single computer. By using Team Foundation Build, you can synchronize the sources, compile the application, run associated unit tests, perform code analysis, release builds on a file server, and publish build reports.

Build computer restrictions

Team Foundation Build is a separate installation from the Team Foundation Server application tier, data tier, or Visual Studio client. You may designate a separate computer. Otherwise, you can install the build side-by-side on the client computer or on the servers.

The following table describes restrictions for the build computer.

Restriction type	Restriction
Disk space	Must contain sufficient space for the build (insufficient space will lead to failed builds).
Build directory	Must be a local path (for example, C:\bulddirectory).
Drop location directory	Must be a UNC path (for example, \\server\share).

Each generated build is put in a new directory in the drop folder.

Drop location permissions

- The Team Foundation Server Service account (for example, Domain\TFSSERVICE) must have Full Control permission access to the UNC drop location.
- The UNC drop location must be a shared folder.

If you change the TFS Service account after the initial installation, you must make sure that the following conditions are true.

Team Foundation Build Service account

- The account is a member of the Build Services group.
- The account has read/write permissions to the temporary folders and the ASP.NET temporary folder.
- The account has Full Control permission to the build directory and drop location.

If the build computer is

Firewall issues	firewall enabled, make sure that the program tfsbuildservice is in the exceptions list.
-----------------	--

Build type names

Team Foundation Build uses build types to configure the conditions under which a single solution or a set of solutions in a team project will be built. To conduct a build, you must either create a new build type or use an existing build type. For more information about build types, see [Define your build process](#).

The following table describes the restrictions on build type names.

Restriction type	Restriction
Uniqueness	Must not be the same as any other build type name in the project
Special characters	<ul style="list-style-type: none"> Must not contain the following printable character: \$

Build quality names

The build quality lets you attach a quality level or completion state to a completed build. Team Foundation Build also lets you create new values for the build quality type. For more information, see [Create or edit a build definition](#). For a list of the default build quality values, see [Rate the quality of a completed build](#).

The following table describes the restrictions on build quality names.

Restriction type	Restriction
Length	Must not contain more than 256 Unicode characters
Uniqueness	Must not be the same as any other Build Quality name in the Team Foundation Build computer

Restrictions for version control

Team Foundation version control provides a central repository for files and the commands that are required to manage those files across a team. It also provides customizable check-in policies, branching, merging, shelving, and many other features.

Version control paths

The following table describes restrictions on a version control path.

Restriction type	Restriction
	<ul style="list-style-type: none">• Must not contain more than 259 Unicode characters for a single folder or file name.
Server source control folder path length	<ul style="list-style-type: none">• Must not contain more than 388 Unicode characters for a directory.• Must not contain more than 399 Unicode characters for a combined directory and file name.• Must not contain more than 248

Local folder path length	<p>Unicode characters for a directory.</p> <ul style="list-style-type: none"> • Must not contain more than 260 Unicode characters for a combined directory and file name.
--------------------------	--

See also: [Optimize your workspace.](#)

■ Adding files to version control

The version control system stores many different types of files. For more information about how to add existing Visual Studio projects or solutions to version control, see [Set up Team Foundation Version Control on your dev machine.](#)

The following table describes the restrictions applied to files and folders to be added to version control.

Restriction type	Restriction
Files and folders	<ul style="list-style-type: none"> • Names must not contain the following printable characters: / \ : * ? " < > ; • Folders must not reside outside the mapped directory for the active workspace

File names	<ul style="list-style-type: none"> • Must not begin with a \$ • Must not contain the following printable characters: / \ : * ? " < > ;
------------	--

Label names

In Team Foundation version control, a label is a name applied to a specific set of revisions. You can attach labels to a set of unrelated files in version control. This lets you retrieve the files or act upon them as a group. The following table describes the restrictions put on label names.

Restriction type	Restriction
Length	Must not contain more than 64 Unicode characters
Special characters	<ul style="list-style-type: none"> • Cannot end with a space () or period (.) • Must not contain the following printable characters: " / : < > \ * ? @ ;

Shelvesets

Shelvesets enable you to set aside temporarily a batch of pending changes and then, as an option, remove the pending changes from your workspace. Later, you can restore the changes in a shelveset to your workspace or put them into another

user's workspace.

The following table describes restrictions on shelveset names.

Restriction type	Restriction
Length	Must not contain more than 64 Unicode characters
Special characters	<ul style="list-style-type: none">• Must not contain the following printable characters: " / : < > \ * ? ;

▣ **Workspace names**

A workspace is a client-side copy of the files and folders in Team Foundation version control. When you create multiple workspaces, you can have different versions of the same version control folder on a client computer. For more information about workspaces, see [Create and work with workspaces](#). The following table describes restrictions on workspace names.

Restriction type	Restriction
Length	Must not contain more than 64 Unicode characters
Special	<ul style="list-style-type: none">• Cannot end with a space ()

characters

- Must not contain the following printable characters: " / : < > \ | * ? ;

See Also

Tasks

Create or edit a build definition
Rate the quality of a completed build

Concepts

Customize work tracking objects to support your team's processes
[Define global lists](#)
Customize a process template
Define your build process
Create and work with workspaces



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Naming conventions for work item tracking objects

[See Also](#) [Send Feedback](#)

In Visual Studio Team Foundation Server (TFS), all work item tracking objects are associated with one or more names. Most have friendly display names and all, except work item types and global lists, are associated with reference names. A friendly name is a unique, user-visible identifier for a field. Using friendly names ensure consistency across all team projects and work item types in a project collection. TFS uses the reference name internally and you cannot change it after it is defined.

The following table summarizes the naming requirements that must be met for each work item tracking object.

Work item tracking object	Reference name	Friendly name
Work item type	Not applicable	The name of each work item type can have up to 255 Unicode characters and must be unique within a team project.

Work item field	<p>Required. See Reference name requirements.</p>	<p>Field names can be up to 128 Unicode characters long and must be unique within a team project collection.</p>
Link type	<p>Required. See Reference name requirements.</p>	<p>You define two friendly names for each link type: Forward Name and Reverse Name. These names can be up to 128 Unicode characters long and must be unique for all link types defined for a team project collection.</p>
Category	<p>Required. See Reference name requirements.</p>	<p>Category friendly names can be up to 128 Unicode characters long and must be unique within a team project.</p>
		<p>The name of</p>

Global
list

Not
applicable

each global list
can have up to
254 Unicode
characters and
must be unique
within a team
project
collection.

▣ Friendly name requirements

In addition to the requirements summarized in the table listed previously in this topic, the friendly names you define should meet the following requirements:

- Names must not be empty.
- Names cannot have leading or trailing white spaces.
- Names cannot contain backslash (\) characters.
- Field names cannot contain the following characters: backslash (\), period (.), and opening and closing square brackets ([]).
- Names cannot contain two or more consecutive white spaces.

Reference name requirements

You must define a reference name whenever you add or create a work item field, link type, or category. All reference names can be up to 70 Unicode characters long.

You can define a reference name by using alphanumeric characters, underscore characters, and hyphen characters. Each reference name must contain at least one period (.), but no period can appear at the start or end of a name. A reference name cannot start with a number or an underscore, and it cannot have multiple consecutive hyphens, such as (--).

Field reference names and portability

The work item type definition language includes the concept of a field reference name. Field reference names can help you to port definitions between Team Foundation project collections and also to allow third party integrations to find and refer to specific fields. These names are globally unique, just as a namespace in the .NET Framework application is globally unique.

Field reference names cannot be renamed. If, for example, you changed the field name "Title" to "Header", the field reference name of that field remains the same. Integrations and internal representations of fields should use the field reference name instead of depending on the field name itself.

The System namespace is used only to define all core system fields that are mandatory for Team Foundation system functions. Team Foundation Server prevents you from creating your own System.X field because it might impede Team Foundation Server functionality.

The Microsoft namespace is used to define work item tracking fields. These fields are defined in a work item type definition of the TFS process templates. TFS does not prevent you from creating your own Microsoft.X field. However, this practice is strongly discouraged because it might impede Team Foundation Server TFS functionality or the ability for the Configure Features wizard to successfully update a team project after a TFS upgrade.

Customers and partners can create their own field namespaces for custom work item types.

For descriptions of system fields and fields defined in the TFS process templates, see

[Work item field reference for Visual Studio ALM.](#)

▣ **Examples of field reference names**

The following examples show valid field reference names, in various namespaces.

▣ **System namespace examples**

System.Id

System.Title

System.CreatedBy

System.CreationDate

System.ChangedBy

System.ChangedDate

System.State

System.Reason

▣ **Microsoft namespace examples**

Microsoft.Common.Status

Microsoft.Common.Priority

Microsoft.Scheduling.Duration

Microsoft.Scheduling.PercentComplete

Microsoft.Testing.TestCaseName

▣ **Examples in other namespaces**

Customers and partners can also define their own namespaces to support their custom work item types. For example, the fictitious company Trey Research might define the following custom work item types:

TreyResearch.Common.Severity

TreyResearch.Common.Phase

TreyResearch.RiskManagement.RiskType

TreyResearch.RiskManagement.Resolution

The fictitious software company A. Datum Corporation might define the following work item types:

A_Datum.Common.BusinessPriority

A_Datum.Bug.FoundInPhase

A_Datum.Bug.FixInPhase

See Also

Reference

[FIELD \(Definition\) element reference](#)

Concepts

[Customize work tracking objects to support your team's processes](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Configure Team Foundation Build Service to Use Team Foundation Server Proxy

[See Also](#) [Send Feedback](#)

You can configure Team Foundation Build Service to retrieve files from Team Foundation Server Proxy by modifying a registry entry on the server that is running Team Foundation Build Service. If you modify this entry, Team Foundation Build Service at the remote site can work with version control from the proxy's cache.

⚠ Important

If you improperly modify your computer's registry, you can cause your computer to become unstable. If you are not familiar with the registry, you should not modify it in any way.

Required Permissions

To complete this procedure, you must be a member of the Administrators group on the server that is running Team Foundation Build Service.

To configure Team Foundation Build Service to Use Team Foundation Server Proxy

1. On the computer that is running a controller for Team Foundation Build Service and also has the client for Team Foundation Server Proxy installed on it, choose Start, choose Run, type regedit, and then choose OK.
2. In the browser pane, expand HKEY_CURRENT_USER to HKEY_CURRENT_USER\Software\Microsoft\VisualStudio\12.0\TeamFou

If you do not see this path, you have not installed the proxy client on this computer. See [How to: Configure Team Foundation Server Proxy Using the Team Foundation Server Configuration Tool](#).

3. In the details pane, set Enabled to True, and then set URL to `http://ProxyServer:Port`.

ProxyServer is the name of the proxy server that you want Team Foundation Build Service to use, and Port is the number of the port on which the proxy server is listening for requests. (By default, this number is 8081.)

Name	Type	Data
 (Default)	REG_SZ	(value not set)
 AutoConfigured	REG_SZ	True
 Enabled	REG_SZ	True
 LastCheckTime	REG_SZ	5247036903017963333
 LastConfiguretime	REG_SZ	5247036903019889500
 Uri	REG_SZ	<code>http://FabrikamFiberProxy:8081/VersionControlProxy</code>

4. On the File menu, choose Exit to close Registry Editor.

Team Foundation Build will now use the proxy server that you specified.

See Also

Other Resources

Managing Remote Connections to Team Foundation Server Proxy
Setting up a Build Computer



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Permission reference for Team Foundation Server

[Send Feedback](#)

Permissions determine what tasks users can and can't do. For users to have access to Team Foundation Server (TFS) resources and team projects, you need to add them to a team project or TFS group. For an overview of how TFS manages membership, permissions, and access, see [Manage users and groups in TFS](#).

This topic describes TFS permissions and their default assignments to each of the built-in TFS groups. It also explains the tools you can use to set permissions. There are three categories of built-in groups, four permissions levels, and five permission states. Each user's access to a functional task depends on the explicit or inherited permission state assigned to them or to a group to which they belong.

In this topic

- [What's new in permissions?](#)
- [Tools used to set permissions](#)
- [Built-in TFS groups](#)
 - [Server-level groups](#)
 - [Collection-](#)

[level groups](#)

- [Project-level groups](#)

- [Allow, Deny, Not set, and other permission states](#)

- [Inheritance](#)

- [Do's and Don'ts when setting permissions](#)

- [Server-level permissions](#)

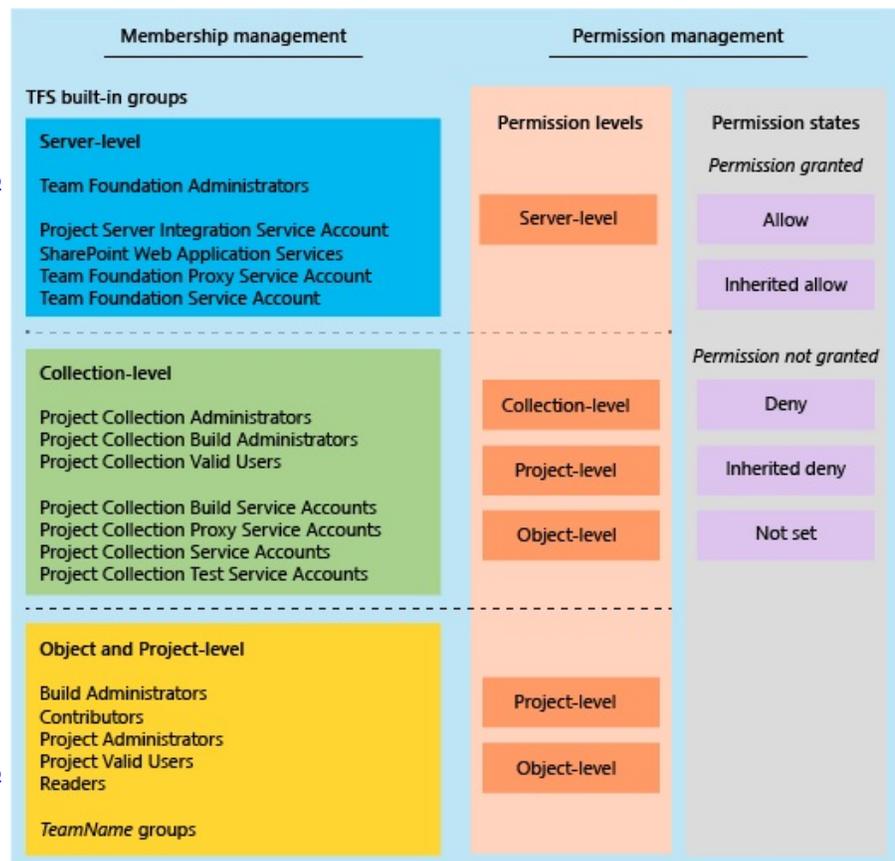
- [Collection-level permissions](#)

- [Project, test, and object-level permissions](#)

- [Project and test-level permissions](#)

- [Build-level permissions](#)

- [Work item query](#)



- [Tagging](#)
- [Area-level for work item tracking](#)
- [Iteration-level for work item tracking](#)
- [Team Foundation Version Control](#)
- [Git repository](#)
- [Lab Management](#)
- [Release Management](#)

To assign permissions for SharePoint Products or SQL Server Reporting Services, see [Add users to team projects](#) and Grant permissions to view or create reports in TFS.

■ What's new in permissions?

Additions and changes to the TFS permission model are noted in the following table. They are based on the version you have installed on your application-tier server.

TFS version	New or changed permissions
TFS 2013.3	Manage test suites permission added, and Manage test plans permission re-scoped to manage only test plans. These permissions are set for an area path. Previously, it covered permission management of both test plans and test suites.
TFS 2013.2	

[Tagging permissions](#) added.

TFS 2013

[Git repository permissions](#) added.

Tools used to set permissions

You can use the tools listed in the following table to set permissions. Different tools are used depending on whether you are setting permissions at a server, collection, or project-level. You use Team Web Access (TWA) to set most permissions for users and groups to access a team project.

Permission level	TWA administrative page or object-level security	Team Explorer (Note 1)	Team Foundation Administration Console	TFS Security command-line tool	Tf command-line tool
-------------------------	---	-------------------------------	---	---------------------------------------	-----------------------------

Server-level

- ✓
- ✓

Collection-level

- ✓
- ✓
- ✓
- ✓

Project and test-level

- ✓
- ✓
- ✓

Build-level



Work item query



Tagging



Area-level for work item tracking



Iteration-level for work item tracking



Team Foundation Version Control



Git repository



Lab Management



[Release Management \(2\)](#)

Notes

1. Some permission options that you access from Team Explorer open a user interface in Team Web Access.
2. If you add Release Management to your deployment, you can manage permissions using groups that you define in Release Management, TFS, or Active Directory. You manage permissions through the Release Management client.

Another tool that you can use to manage user membership within groups is the [TFSAdmin tool from CodePlex](#).

≡ **Command-line tools, namespaces, and permission names**

When you exercise the TFSSecurity command-line tool to manage permissions, you specify a namespace as well as the name of the permission. In the sections below, the namespace and command name are indicated. There are two namespace groups: project collection and server. Use the TFSSecurity /a command to list the namespaces. To obtain the set of permissions you can set under a namespace, use TFSSecurity /a Namespace /collection:CollectionNameURL

Project collection namespaces

Server namespaces

```
TFSSecurity /a  
/collection:CollectionNameURL
```

 **Copy Code**

```
Build  
BuildAdministration  
Chat  
Collection  
CSS  
Discussion Threads  
EventSubscription
```

```
TFSSecurity /a  
/server:ServerNameURL
```

 **Copy Code**

```
Catalog  
CollectionManagement
```

Git Repositories
Identity
Iteration
Job
Project
ProjectServerAdministration
Registry
Server
ServiceHooks
StrongBox
Tagging
TeamLabSecurity
VersionControlItems
VersionControlPrivileges
WorkItemQueryFolders
WorkItemTrackingAdministration
WorkItemTrackingProvision
Workspaces

Diagnostic
EventSubscription
Feature Availability
HostingAccount
Identity
Job
Lab
Registry
Server
StrongBox
Warehouse
WebAccess

Built-in TFS groups

When you install TFS, four groups are defined at the server level. When you create a project collection, seven groups are created at the collection-level, and for each team project that you create, six groups are created that are scoped to the team project. Each of these groups is associated with a set of default permissions. You can't remove or delete a default server-level groups, such as the Team Foundation Administrators group.

Server-level	Collection-level	Project-level
	<ul style="list-style-type: none"> Project Collection Administrators 	
	<ul style="list-style-type: none"> Project Collection Build Administrators 	
<ul style="list-style-type: none"> SharePoint Web Application Services 	<ul style="list-style-type: none"> Project Collection Build Service Accounts 	<ul style="list-style-type: none"> Build Administrators
<ul style="list-style-type: none"> Team Foundation Administrators 	<ul style="list-style-type: none"> Project Collection Service Accounts 	<ul style="list-style-type: none"> Contributors
<ul style="list-style-type: none"> Team Foundation Service Accounts 	<ul style="list-style-type: none"> Project Collection Proxy Service Accounts 	<ul style="list-style-type: none"> Project Administrators
<ul style="list-style-type: none"> Team Foundation 		<ul style="list-style-type: none"> Project Valid Users (Note 1)
		<ul style="list-style-type: none"> Readers
		<ul style="list-style-type: none"> TeamProject group (Note 2)

- | | |
|-------------|--|
| Valid Users | <ul style="list-style-type: none">• Project Collection Test Service Accounts• Project Collection Valid Users (Note 1) |
|-------------|--|

Notes:

1. To learn more about valid user groups, jump to this

[What is a Valid User? How are Valid User groups populated?](#) section.

- A team group is created with the name of the team project. For example, if you create a team project named "Code Sample," a team group will also be created with the name "Code Sample Team." You can rename this team.

In addition, when you create additional teams, a team group is created for each team.

Server-level groups are assigned server-level permissions. Collection-level groups are assigned permissions defined for the collection, project, and objects. And, permissions assigned to project-level groups include both project-level and object-level.

For example, the following picture shows the permissions assigned to the project-level Contributor group.

Code	Work	Build	Test
Object and project-level:	Area-node level	Object-level:	Area-node level
TFVC: Check in Check out Label Lock Merge Read Git repositories Administer Branch creation Contribute Note management Read Rewrite and destroy history Tag creation	Edit work View permissions View work items Shared queries Read Project-level View project-level information	Edit build quality Queue builds View build definition View builds Edit build quality Queue builds View build definition View builds	Manage test plans Manage test suites Create test runs Delete test runs Manage test configurations Manage test environments View test runs

The Project administrator group permissions include those assigned to the Contributor group and a few more.

Code	Work	Build	Test
Object and project-level:	Area node-level	Object and project-level:	Area node-level
TFVC: Administer labels Check in Check in other users' changes Check out Label Lock Manage branch Manage permissions Merge Read Revise other users' changes Undo other users' changes Unlock other users' changes Git repositories Administer Branch creation Contribute Note management Read Tag creation	Create child nodes Delete this node Edit this node Edit work View permissions View work items Shared queries Contribute Delete Manage Permissions Read Project-level Create tag definition Delete team project Edit project-level information View project-level information	Administer build permissions Delete build definition Delete builds Edit build definition Edit build quality Manage build qualities Manage build queue Queue builds Retain indefinitely Stop builds View build definition View builds	Manage test plans Manage test suites Create test runs Delete test runs Manage test configurations Manage test environments View test runs

Server-level groups

By default, the following groups exist at the server level for the application-tier when you install TFS.

Group name (prefix: Team Foundation\	Permissions	Default user accounts
Team Foundation Administrators	Can perform all operations for TFS. This group should be restricted to the smallest possible number of users who need total administrative control over TFS.	Local Administrators group (BUILTIN\Administrators) for any server that hosts Team Foundation application services. Server\ Team Foundation Service Accounts group and the members of the \ Project Server Integration Service Accounts group.
Team Foundation Valid Users	Have read access to source code, work items, and build definitions. Access to TWA features is dependent on the license or access level group they've been assigned.	Contains all users and groups that have been added anywhere within TFS.
	⚠ Important	You can't modify the

If you set the membership of this group. View instance-level information permission to Deny or Not set for this group, no users will be able to access the deployment.

Contains the service account that was supplied during installation.

Team Foundation Service Accounts

Have service-level permissions for TFS. This group should contain only service accounts and not user accounts or groups that contain user accounts. By default, this group is a member of Team Foundation Administrators.

Project Server Integration Service

Have service-level permissions for the Project Server deployments that are configured for interoperability with TFS. In This group should contain only service accounts and not user accounts or groups that contain user accounts. By default, this

Accounts	<p>addition, members of this group have some TFS service-level permissions.</p>	<p>group is a member of Team Foundation Administrators.</p>
SharePoint Web Application Services	<p>Have service-level permissions for the SharePoint Web applications that are configured for use with TFS, in addition to some service-level permissions for TFS.</p>	<p>This group should contain only service accounts and not user accounts or groups that contain user accounts. Unlike the Service Accounts group, this group is not a member of Team Foundation Administrators.</p>
Team Foundation Proxy Service Accounts	<p>Members of this group have service-level permissions for Team Foundation Server Proxy, and have some TFS service-level permissions.</p>	<p>This group should contain only service accounts and not user accounts or groups that contain user accounts.</p>

Collection-level groups

By default, the following groups exist at the collection level when you configure a collection.

Group name (prefix: TeamProjectCollectionName\	Group-level permissions	Account assignments
Project Collection Administrators	Can perform all operations for the team project collection.	Contains the Local Administrators group (BUILTIN\Administrators) for the server where the application-tier services for TFS have been installed. Also, contains the members of the TeamProjectCollectionName\ServiceAccounts group. This group should be restricted to the smallest possible number of users who need total administrative control over the collection.
Project Collection Valid Users	Can access team projects defined for the collection. ⚠ Important If you set the View collection-level	Contains all users and groups that have been added anywhere within the team project collection. You cannot modify the membership of

	<p>information permission to Deny or Not set for this group, no users will be able to access the collection.</p>	<p>this group.</p>
<p>Project Collection Service Accounts</p>	<p>Have service-level permissions for the collection and for TFS.</p>	<p>Contains the service account that was supplied during installation. This group should contain only service accounts and groups that contain only service accounts. By default, this group is a member of Team Foundation Administrators and Team Foundation Service Accounts.</p>
<p>Project Collection Build Administrators</p>	<p>Can administer build resources and permissions for the collection.</p>	<p>Limit this group to the smallest possible number of users who need total administrative control over build servers and services for this collection.</p>
<p>Project Collection Build Service Accounts</p>	<p>Have those permissions required to run build services for the collection.</p>	<p>Limit this group to service accounts and groups that contain only service accounts.</p>

Project Collection Proxy Service Accounts	Have permissions required to run the proxy service for the collection.	Limit this group to service accounts and groups that contain only service accounts.
Project Collection Test Service Accounts	Have test service permissions for the collection.	Limit this group to service accounts and groups that contain only service accounts.

Project-level groups

By default, the following groups exist when you create a team project. Their permissions are scoped to the project level.

Group (prefix: ProjectName\)	Group-level permissions	Additional notes
Project Administrators	Can administer all aspects of the team project, although they can't create	Assign to users who will manage user permissions, create teams, define area an iteration

projects. paths, or customize work item tracking.

Build Administrators

Can administer build resources and build permissions for the project. Members can manage test environments, create test runs, and manage builds.

By default, the team group created when you create a team project is added to this group, and any user you add to the team will be a member of this group. In addition,

Contributors

Can contribute fully to the project code base and

	work item tacking.	any team you create for a team project will be added to this group by default, unless you choose a different group from the list.
Readers	Can view the project but not modify it.	Assign to stakeholders who want to be able to view work in progress.
TeamName	Inherit the same permissions assigned to the Contributors group. Can contribute fully to the project code base and	The default Team group is created when you create a team project, and by default is added to the Contributors group for the team project. Any new teams you create will also

work item
tacking.

have a
group
created for
them and
added to the
Contributors
group.

Besides these project-level groups, two collection-level groups also appear in every project in TFS:

- TeamProjectCollectionName**Project Collection Administrators**
You cannot change the permissions for this collection-level group.
- TeamProjectCollectionName**Project Collection Build Service Accounts**

Do not remove or set the View project-level information permission to Deny for this group.

Allow, Deny, Not set, and other permission states

TFS uses a least-permissive model for security permissions. What that means is that if a user belongs to two groups and the same permission is assigned Allow for one group and Deny for another group, Deny takes precedence over Allow. There are a few exceptions to this rule for those who belong to the Project Collection Administrator and Team Foundation Server Administrator groups.

You can specify two explicit authorization states for permissions: Deny and Allow. In addition, there are three other states: Inherited allow, Inherited deny, and Not set. Not set is an implicit Deny state.

Permission	Authorization
Allow	Explicitly grants users to perform the task associated with the specific permission. Allow is usually inherited from group membership. For users to access a task, the associated permission must be set to Allow or Inherited allow.
Deny	Explicitly prevents users from performing the task associated with the specific permission. Deny is usually inherited from group

	membership.
Inherited allow/Inherited deny	Allows or denies a user to perform the task associated with the permission based on the permission set for a group to which the user belongs.
	Implicitly prevents users from performing the action associated with the permission.
Not set	Because the permission is neither explicitly set to Deny nor explicitly set to Allow, authorization for that permission can be inherited from other groups of which the user or group is a member.
	By default, most permissions are not set to either Deny or Allow. The permissions are left Not set.

Permission states follow these precedence settings:

- The Deny permission takes precedence over all other permission settings,

including Allow. For example, a user might belong to two groups in a project. For one group, Publish test results permission is set to Deny; the other group has that permission set to Allow. The Deny setting takes precedence and the user is not authorized to publish test results.

Exceptions to this rule are:

- The Deny permission does not take precedence if it is inherited from a hierarchical parent. These functions support hierarchical permission setting:
 - Source control folders for Team Foundation version control
 - Git repositories
 - Area and iteration nodes for work item tracking
 - Work item shared queries and query folders

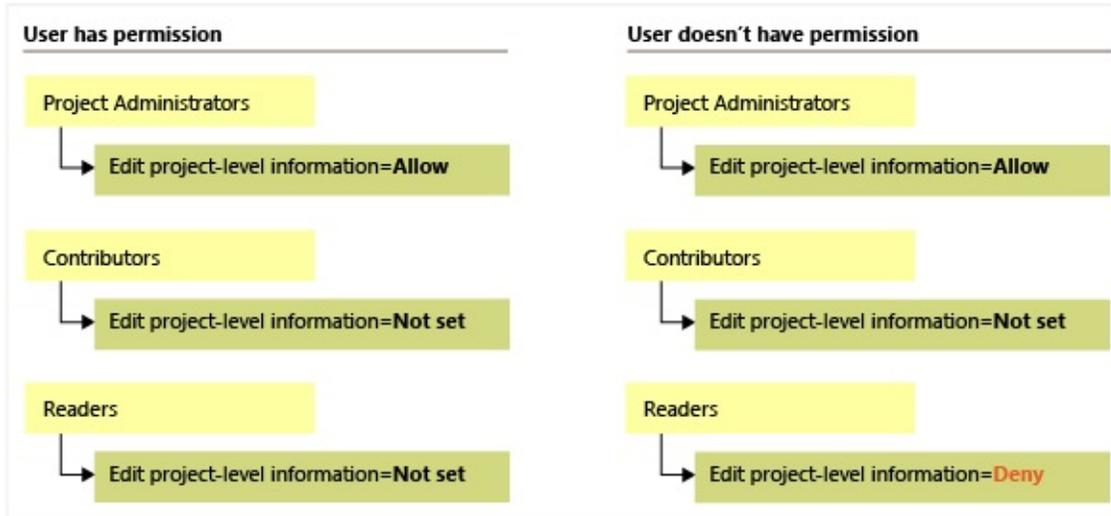
The explicit permissions that are set on a particular object—such as a source control folder, a repository, or an area child node—override those that are inherited from the parent objects. For example, the Deny set for a source control folder doesn't override an Allow set for one of its sub-folders.

- When a user belongs to an administrators group, such as the Project Collection Administrators or Team Foundation Administrators groups, unless otherwise stated in the description of the permission.
- Inherited allow takes precedence over Not set.

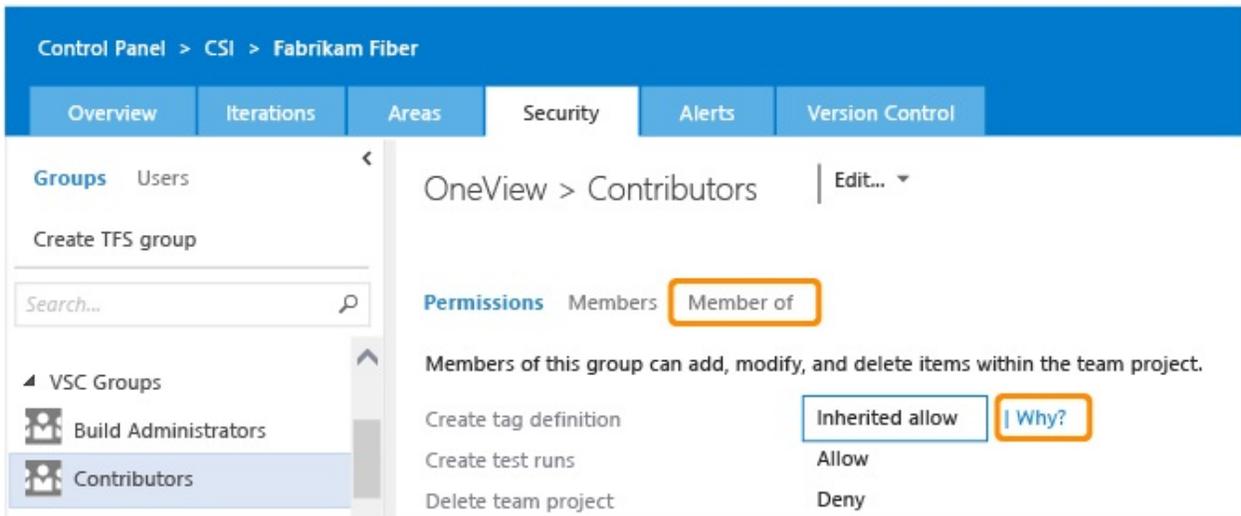
■ Inheritance

When permission is Not set for a user or group, the user or group can be affected by the explicit state for the permission for groups to which they belong because permissions in TFS are inherited. For example, when you review the permissions for a user or group, you might see both Allow and Inherited allow set for permissions. The latter permission is inherited from some other group the user or group belongs to. In this example, a user might belong to a group at the project level and a group at the collection level in a project. If one of those groups has a

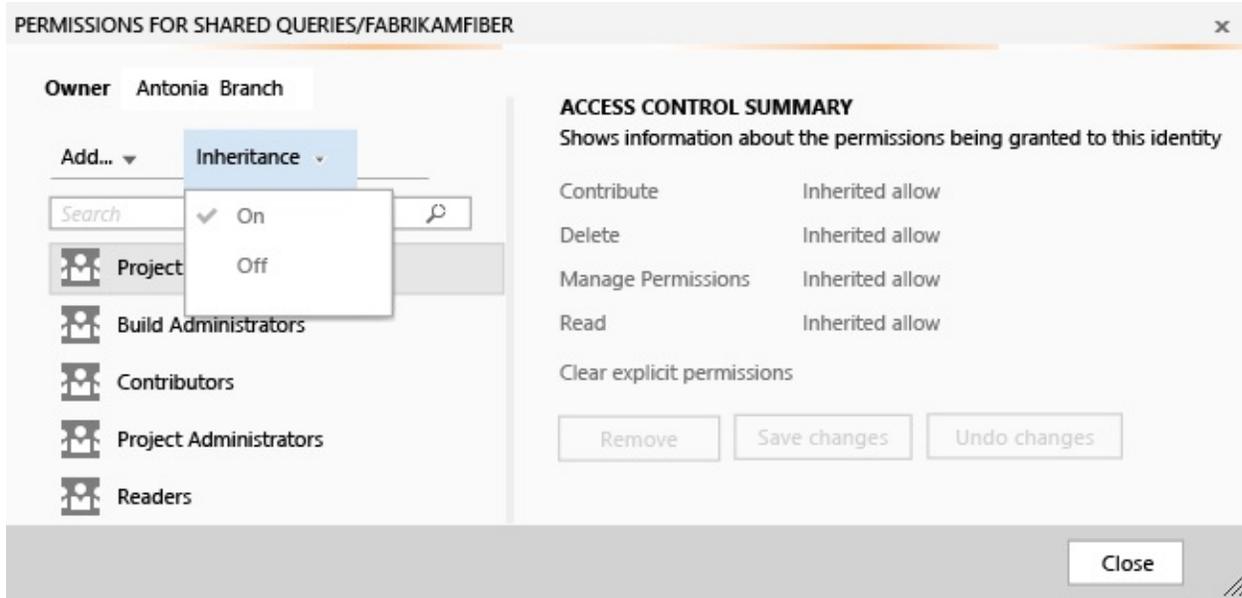
permission that is explicitly set to Allow and the other group has the same permission Not set, the user will have the Inherited allow permission to perform the actions that are controlled by that permission. The user inherits permissions from both groups, and the Allow permission takes precedence over the Not set permission.



To understand why a permission is inherited, you can pause over the permission setting, and then choose Why?. A new window will open. It displays the inheritance information for that permission.



Some object-level Security dialog boxes provide an Inheritance on/off option. Use this option to disable inheritance for folders, shared queries, and other objects.



Do's and Don'ts when assigning permissions

Do's:

- Use Windows groups when managing lots of users.
- Consider granting the Contribute permissions to users or groups that require the ability to create and share work item queries for the project.
- When adding many teams, consider creating a Team Administrators group to TFS where you allocate a subset of the permissions available to Project Administrators.
- When adding teams, consider what permissions you want to assign to team leads, scrum masters, and other team members who may need to create and modify area paths, iteration paths, and queries.

Don'ts:

- Don't add accounts to the Readers group that you've added to the Project Administrators group. Doing so causes a Deny state to be assigned to several permissions.
- Don't change the default assignments made to a valid users group. If you remove or set the View instance-level information permission to Deny for

one of the Valid Users groups, no users in the group will be able to access the team project, collection, or deployment, depending on the group you set.

- Don't assign permissions that are noted as 'Assign only to service accounts' to user accounts.

Server-level permissions

Server-level permissions grant permissions that can affect every project and collection in the deployment. You can set server-level permissions from the Team Foundation Administration Console or using the TFSSecurity command line tool.

You can set these permissions for server-level users and groups, such as Team Foundation Administrators, and for server-level custom groups that you add.

Permission name	TFSSecurity Action	TFSSecurity Namespace	Description	SharePoint Web Services
Administer warehouse (Note 1)	Administer	Warehouse	Can process or change settings for the data warehouse or SQL Server Analysis cube by using the Warehouse Control Web Service.	
Create team project collection	CreateCollection	CollectionManagement	Can create and administer team project collections.	
Delete			Can delete a	

team project collection (Note 2)	DeleteCollection	CollectionManagement	team project collection from the deployment.
Edit instance-level information (Note 3)	tf: AdminConfiguration tf: AdminConnections	Server	Can edit server-level permissions for TFS users and groups, and add or remove server-level groups from the collection.
Make requests on behalf of others	Impersonate	Server	Can perform operations on behalf of other users or services. Only assign to service accounts. ✓
Trigger events	TRIGGER_EVENT	Server	Can trigger TFS alert events. Only assign to service accounts and members of the Team Foundation Administrators group.

Use full Web Access features (Note 4)	FullAccess	Server	Can use all TWA features.
View instance-level information (Note 5)	GENERIC_READ	Server	Can view server-level group membership and the permissions of those users. ✓

Notes:

1. Additional permissions may be required to fully process or rebuild the data warehouse and Analysis cube.
2. Deleting a team project collection will not delete the collection database from SQL Server.
3. Edit instance-level information includes the ability to perform these tasks for all team projects defined in all project collections defined for the instance:
 - Add and administer teams and all team-related features
 - Create and modify areas and iterations
 - Edit check-in policies
 - Edit shared work item queries
 - Edit project-level and collection-level permission ACLs
 - Create and modify global lists

- Edit

[event subscriptions](#) (email or SOAP).

When set through the menus, the Edit instance-level information permission also implicitly allows the user to modify version control permissions. To grant all these permissions at a command prompt, you must use the `tf.exe Permission` command to grant the **AdminConfiguration** and **AdminConnections** permissions in addition to `GENERIC_WRITE`.

- If the Use full Web Access features permission is set to Deny, the user will only see those features permitted for the Limited group (see [Change access levels](#)). A Deny will override any implicit allow, even for accounts that are members of administrative groups such as Team Foundation Administrators.
- The View instance-level information permission is also assigned to the Team Foundation Valid Users group.

Collection-level permissions

Collection-level permissions grant authorization to collection-wide tasks, which you can set for these users and groups:

- Collection-level users and groups, such as Project Collection Administrators
- Project-level groups that you add to a collection
- Custom groups that you add to a collection

You can set collection-level permissions from the TWA administration page for the collection, from the Team Foundation Administration Console or using the TFSSecurity command-line tool or tf command-line tool. All permissions are scoped to the specific collection for which they are set.

Permission name	TFSSecurity Action	TFSSecurity Namespace
Administer build resource permissions	AdministerBuildResourcePermissions	BuildAdministration
Administer Project Server integration	AdministerProjectServer	ProjectServerAdministration

Administer shelved changes	AdminShelveSets tf: AdminShelveSets	VersionControlPrivileges
Administer workspaces	AdminWorkspaces tf: AdminWorkspaces	VersionControlPrivileges
Alter trace settings	DIAGNOSTIC_TRACE	Collection
Create a workspace (Note 2)	tf: CreateWorkspace	VersionControlPrivileges
Create new projects (Note 3)	CREATE_PROJECTS	Collection

Delete team project (Note 4)	Delete	Project
Edit collection-level information (Note 5)	GENERIC_WRITE tf: AdminConfiguration tf: AdminConnections	Collection VersionControlPrivileges
Make requests on behalf of others	Impersonate	Server
Manage build resources	ManageBuildResources	BuildAdministration
Manage process template	MANAGE_TEMPLATE	Collection
Manage test	MANAGE_TEST_CONTROLLERS	Collection

controllers

Trigger events
(Note 6)

TRIGGER_EVENT

Collection

Use build
resources

UseBuildResources

BuildAdministration

View build
resources

ViewBuildResources

BuildAdministration

View
collection-level
information

GENERIC_READ

Collection

View system
synchronization SYNCHRONIZE_READ
information

Collection

Notes:

1. In addition, the following default assignments are made to these TFS groups:
 - Project Collection Valid Users Group: Create a workspace, View build resources, and View collection-level information.
 - Project Collection Proxy Service Accounts: Create a workspace, View build resources, and View collection-level information.
 - Project Collection Test Service Accounts: Create a workspace, Manage test controllers, View build resources, and View collection-level information.
2. The Create a workspace permission is granted to all users as part of their membership within the Project Collection Valid Users group.
3. Additional permissions may be required depending on your deployment. Also, you must run Visual Studio or Team Explorer as an administrator to successfully complete the Create a New Team Project Wizard.
4. Deleting a team project will delete all data that is associated with the project. You cannot undo the deletion of a team project except by restoring the collection to a point before the project was deleted.
5. Edit collection-level information includes the ability to perform these tasks for all team projects defined in a collection:
 - Add and administer teams and all team-related features

- Create and modify areas and iterations
- Edit check-in policies
- Edit shared work item queries
- Edit project-level and collection-level permission ACLs
- Create and modify global lists
- Edit

[event subscriptions](#) (email or SOAP) on project or collection-level events.

When you set Edit collection-level information to Allow through TWA, users can add or remove collection-level TFS groups and implicitly allows these users to modify version control permissions. To grant all these permissions at a command prompt, you must use the `tf.exe Permission` command to grant the **AdminConfiguration** and **AdminConnections** permissions, in addition to `GENERIC_WRITE`.

- Users with this permission can't remove built-in collection-level groups such as Project Collection Administrators.
- Adding this permission to other users has the potential to allow denial-of-service attacks.

Project, test, and object-level permissions

Project-level permissions are specific to a single project's users and groups. Within a project, you can set permissions on the objects created for that project, such as areas, iterations, source control folders, queries and query folders, and build definitions. You can set project and object-level permissions for users and groups that you add to a team project or collection.

Many default permissions are assigned to these built-in project-level and collection-level groups:

- Project-level groups: Builders, Contributors, Project Administrators, and Readers
- Collection-level groups: Project Collection Administrators, Project Collection Build Service Accounts, Project Collection Proxy Service Accounts, and Project Collection Test Service Accounts

Project and test-level permissions

You can set project-level permissions from the TWA administration page for the project or by using the TFSSecurity command line tool. All project-level permissions authorize users for the specific team project for which they are set.

Permission Name	TFSSecurity Action	TFSSecurity Namespace	Description
Create tag definition	Create	Tagging	Can add tags through a work item form.

Create test runs	PUBLISH_TEST_RESULTS	Project	Can add a remove test results and or modify runs.
Delete team project	DELETE	Project	Can delete team project from TFS.
Delete test runs	DELETE_TEST_RESULTS	Project	Can delete scheduled
Edit project-level information (Note 2)	GENERIC_WRITE	Project	Can edit project-level permission for users and groups.
Manage test configurations	MANAGE_TEST_CONFIGURATIONS	Project	Can create delete test configurations
Manage test environments	MANAGE_TEST_ENVIRONMENTS	Project	Users who have this permission create and delete test environments Can view

View project-level information	GENERIC_READ	Project	project-level group membership and permission
View test runs	VIEW_TEST_RESULTS	Project	Can view plans under the team project area path.

Notes:

1. In addition, the following default assignments are made to these TFS groups:
 - Readers: Create tag definition, View project-level information, and View test runs.
 - Project Collection Administrators: Same permissions as Project Administrators, except for Delete test runs.
 - Project Collection Build Administrators: Same permissions as Project Administrators, except for Delete test runs.
 - Project Collection Build Service Accounts: Create test runs, Manage test configurations, Manage test environments, View project-level information, View test runs.
 - Project Collection Test Service Accounts: Create test runs, Manage test configurations, Manage test environments, View project-level information.
2. Edit project-level information includes the ability to perform these tasks for the team project:
 - Add and administer teams and all team-related features

- Create and modify areas and iterations
- Edit check-in policies
- Edit shared work item queries
- Edit project-level permission ACLs
- Create and modify global lists
- Edit

[event subscriptions](#) (email or SOAP) on project-level events.

Build-level permissions

You can set build-level permissions for all builds or for a build definition from the context menu for the build definition in TWA or Team Explorer, or by using the TFSSecurity command line tool.

Permission name (UI)	TFSSecurity Action	TFSSecurity Namespace	Description	Contr
Administer build permissions	AdministerBuildPermissions	Build	Can administer the build permissions for other users.	
Delete build definition	DeleteBuildDefinition	Build	Can delete build definitions for this	

			project.
Delete builds	DeleteBuilds	Build	Can delete a completed build.
Destroy builds	DestroyBuilds	Build	Can permanently delete a completed build.
Edit build definition (Note 2)	EditBuildDefinition	Build	Can create and modify build definitions for this project.
Edit build quality	EditBuildQuality	Build	Can add information about the quality of the build through ✓ Team Explorer or Team Web Access.
Manage build qualities	ManageBuildQualities	Build	Can add or remove build qualities.

Manage build queue	ManageBuildQueue	Build	Can cancel, re-prioritize, or postpone queued builds.
Override check-in validation by build (Note 3)	OverrideBuildCheckInValidation	Build	Can commit a changeset that affects a gated build definition without triggering the system to shelve and build their changes first.
Queue builds	QueueBuilds	Build	Can put a build in the queue through the interface for Team Foundation Build or at a command prompt. They can also stop the builds that they have queued. ✓
			Can mark a build so that

Retain indefinitely	RetainIndefinitely	Build	it will not be automatically deleted by any applicable retention policy.
Stop builds	StopBuilds	Build	Can stop any build that is in progress, including builds queued and started by another user.
Update build information	UpdateBuildInformation	Build	Can add build information nodes to the system, and can also add information about the quality of a build. Assign only to service accounts.
View build definition	ViewBuildDefinition	Build	Can view the build definitions that have been created ✓

		for the team project.
View builds ViewBuilds	Build	Can view the queued and completed builds for this team project. ✓

Notes:

1. In addition, the following default assignments are made to these built-in groups:
 - Readers: View build definition and View builds.
 - Project Collection Administrators: All permissions except for Update build information.
 - Project Collection Build Administrators: All permissions except for Override check-in validation by build and Update build information..
 - Project Collection Build Service Accounts: Edit build quality, Manage build queue, Update build information, Override check-in validation by build, Queue builds, View build definitions, and View builds.
 - Project Collection Test Service Accounts: Update build information, View build definitions, and View builds.
2. You turn Inheritance Off for a build definition when you want to control permissions for specific build definitions.

When inheritance is On, the build definition respects the build permissions defined at the project level for a group or user. For example, a custom Build Managers group has permissions set to manually queue a build for project Fabrikam. Any build definition with inheritance On for project Fabrikam would allow a member of the Build Managers group the ability to manually

queue a build.

However, by turning Inheritance Off for project Fabrikam, you can set permissions that only allow Project Administrators to manually queue a build for a specific build definition. This would then allow me to set permissions for that build definition specifically.

3. Assign the Override check-in validation by build permission only to service accounts for build services and to build administrators who are responsible for the quality of the code. For more information, see Check in to a folder that is controlled by a gated check-in build process.

Work item query permissions

You can set work item query permissions from the shortcut menu of Shared Queries from TWA or Team Explorer or by using the TFSSecurity command line tool. All permissions are scoped to the specific query or query folder for which they are set.

Consider granting the Contribute permissions to users or groups that require the ability to create and share work item queries for the project. To create query charts you need Advanced access.

Permission name	TFSSecurity Action	TFSSecurity Namespace	Description	Co Ad
Contribute	CONTRIBUTE	WorkItemQueryFolders	Can view and modify this query or query folder.	

Delete	DELETE	WorkItemQueryFolders	Can delete a query or query folder and its contents.
Manage Permissions	MANAGEPERMISSIONS		Can manage the permissions for this query or query folder.
Read	READ	WorkItemQueryFolders	Can view and use the query or the queries in a folder, but cannot modify the query or query folder contents. ✓
	FullControl	WorkItemQueryFolders	Can view, edit, delete, and manage permissions for a query or query folder and its contents.

Tagging permissions

Tags provide a quick way of grouping or categorizing work items. Tagging permissions are available with on-premises TFS deployments with TFS 2013.2 or later versions installed. You can set the Create tag definition from the TWA administration Security page. To set all remaining permissions, use the TFSSecurity command line tool.

Permission name	TFSSecurity Action	TFSSecurity Namespace	Description	Readers	Contributors	A
Create tag definition (Note 2)	CREATE	Tagging	Can create new tags and apply them to work items. Users without this permission can only select from the existing set of tags for the team project.	✓	✓	
Delete tag definition (Note 3, 4)	DELETE	Tagging	Can remove a tag from the list of available tags for that project.			✓
			Can view a			

Enumerate tag definition (Note 4, 5)	ENUMERATE Tagging	list of tags available for the work item within the team project. Users without this permission will not have a list of available tags from which to choose in the work item form or in the query editor.	✓	✓	✓
Update tag definition (Note 4)	UPDATE Tagging	Can rename a tag by using the REST API.		✓	✓

Notes:

1. In addition, the Project Collection Service Accounts group has all tagging permissions explicitly assigned.
2. Readers and Contributors inherit the Create tag definition permission as it is set explicitly to Allow for the Project Valid Users group.

Although the Create tag definition permission appears in the security settings at the team project level, tagging permissions are actually collection-level permissions that are scoped at the project level when they

appear in the user interface. To scope tagging permissions to a single team project when using the TFSSecurity command, you must provide the GUID for the project as part of the command syntax. Otherwise, your change will apply to the entire team project collection. Keep this in mind when changing or setting these permissions.

3. There is no UI support to delete a tag. To delete a tag, remove the assignments that are associated with the tag. TFS automatically deletes unassigned tags after 3 days of non-use.
4. Does not appear in the UI; can only be set by using the TFSSecurity command.
5. The View project-level information set to Allow for Readers and Contributors implicitly allows users in these groups to view existing tags (Enumerate tag definition permission).

■ **Area-level permissions for work item tracking**

Area-level permissions grant or restrict access to work items defined for a project based on their location with their area tree hierarchy.

You can define and set area-level permissions from the TWA administration page for Areas or by using the TFSSecurity command line tool. All permissions are scoped to the specific area-path for which they are set.

Permission name	TFSSecurity Action	TFSSecurity Namespace	Description	Contribut
			Can create area nodes.	Users who have both this

Create child nodes (Note 2)	CREATE_CHILDREN	CSS	permission and the Edit this node permission can move or re-order any child area nodes.
Delete this node (Note 2)	DELETE	CSS	Users who have both this permission and the Edit this node permission for another node can delete area nodes and reclassify existing work items from the deleted node. If the deleted node has child nodes, those nodes are also deleted.
Edit this node (Note 2)	GENERIC_WRITE	CSS	Can set permissions for this node and rename area nodes.

Edit work items in this node (Note 3)	WORK_ITEM_WRITE	CSS	Can edit work items in this area node. ✓
Manage test plans (Note 4)	MANAGE_TEST_PLANS	CSS	Can modify test plan properties such as build and test settings. ✓
Manage test suites (Note 4)	MANAGE_TEST_SUITES	CSS	Can create and delete test suites, add and remove test cases from test suites, change test configurations associated with test suites, and modify suite hierarchy (move a test suite). ✓
View permissions for this node	GENERIC_READ	CSS	Can view the security settings for this node. ✓
View work			Can view, but

items in
this node
(Note 5)

WORK_ITEM_READ

CSS

not change,
work items in ✓
this area
node.

Notes:

1. In addition, the following default assignments are made to these built-in groups:
 - Readers: View permissions for this node and View-only permissions.
 - Project Collection Test Service Accounts: View-only permissions.
 - Team Foundation Administrators, Project Collection Administrators, and Project Administrators: All CSS permissions. Any user or group that has Edit instance-level information, Edit collection-level information, or Edit project-level information permissions can create and manage area nodes.
 - Members of the Project Collection Valid Users, Project Valid Users, or any user or group that has View collection-level information or View project-level information can view permissions of any area node.
2. Consider adding this permission to any manually added users or groups that may need to delete, add, or rename area nodes.
3. Consider adding this permission to any manually added users or groups that may need to edit work items under the area node.
4. Manage test suites permission was added with the TFS 2013.3 update. Consider adding these permissions to any manually added users or groups that may need to manage test plans or test suites under this area node.
5. If you set the View work items in this node to Deny, the user will not be able to see any work items in this area node. A Deny will override any implicit allow, even for accounts that are members of administrative groups such as Team Foundation Administrators.

Iteration-level permissions for work item tracking

Iteration-level permissions grant or restrict access to create and manage iteration paths.

You can set iteration-level permissions for users and groups that you add to a team project or collection using the TWA administration page for Iterations or the TFSSecurity command line tool. All permissions are scoped to the specific iteration-path for which they are set.

Some work item tracking operations require multiple permissions. For example, you need multiple permissions to delete a node.

Consider granting team administrators, scrum masters, or team leads permissions to create, edit, or delete nodes.

Permission name	TFSSecurity Action	TFSSecurity Namespace	Description	Project Administrators (Note 1)
Create child nodes (Note 2)	CREATE_CHILDREN	Iteration	Can create iteration nodes. Users who have both this permission and the Edit this node permission can move or re-order any child iteration nodes.	✓

Delete this node (Note 2)	DELETE	Iteration	Users who have both this permission and the Edit this node permission for another node can delete iteration nodes and reclassify existing work items from the deleted node. If the deleted node has child nodes, those nodes are also deleted. ✓
Edit this node (Note 2)	GENERIC_WRITE	Iteration	Can set permissions for this node and rename iteration nodes. ✓
View permissions for this node (Note 2)	GENERIC_READ	Iteration	Can view the security settings for ✓

3)

this node.

Notes:

1. Team Foundation Administrators and Project Collection Administrators are granted all Iteration permissions. Any user or group that has Edit instance-level information, Edit collection-level information, or Edit project-level information permissions can create and manage iteration nodes.
2. Consider adding this permission to any manually added users or groups that might need to delete, add, or rename iteration nodes.
3. Members of the Project Collection Valid Users, Project Valid Users, or any user or group that has View collection-level information or View project-level information can view permissions of any iteration node.

Team Foundation Version Control (TFVC) permissions

You can set permissions on TFVC source code files and folders from the context menu for the file or folder definition in TWA or Team Explorer, or by using the [tf permission command line tool](#). These permissions appear only for a team project set up to use TFVC as the source control system.

In version control permissions, explicit deny takes precedence over administrator group permissions.

Permission name	TFS Security Action and <i>tf perm</i>	TFS Security Namespace	Description Contributor
			Can edit or delete

Administer labels
tf: LabelOther

VersionControlItems labels
created by another user.

Check in (Note 2)
tf: Checkin

VersionControlItems
Can check in items and revise any committed changeset comments. ✓
Pending changes are committed at check-in.

Check in other users' changes
tf: CheckinOther

VersionControlItems
Can check in changes that were made by other users.
Pending changes are committed at check-in.

Can check out and make a pending change to items in a folder.
Examples of pending changes

Check out (Note 2)	tf: PendChange	VersionControlItems	include adding, editing, renaming, deleting, undeleting, branching, and merging a file. Pending changes must be checked in, so users will also need the Check in permission to share their changes with the team.	✓
Label	tf: Label	VersionControlItems	Can label items.	✓
Lock	tf: Lock	VersionControlItems	Can lock and unlock folders or files.	✓
			Can convert any folder under that	

Manage
branch

tf: ManageBranch

VersionControlItems

path into a branch, and also take the following actions on a branch: edit its properties, re-parent it, and convert it to a folder. Users who have this permission can branch this branch only if they also have the Merge permission for the target path. Users cannot create branches from a branch for which they do not have the Manage Branch permission.

Can
manage

Manage permissions (Note 3) tf: AdminProjectRights VersionControlItems other users' permissions for folders and files in version control.

Merge (Note 4) tf: Merge VersionControlItems Can merge changes into this path. ✓

Read tf: Read VersionControlItems Can read the contents of a file or folder. If a user has Read permissions for a folder, the user can see the contents of the folder and the properties of the files in it, even if the user does not have permission to open the files. ✓

Revise other users' changes (Note 5)	tf: ReviseOther	VersionControlItems	Can edit the comments on checked-in files, even if another user checked in the file.
Undo other users' changes Merge (Note 5)	tf: UndoOther	VersionControlItems	Can undo a pending change made by another user.
Unlock other users' changes (Note 5)	tf: UnlockOther	VersionControlItems	Can unlock files locked by other users.

Notes:

1. In addition, all permissions are set to Inherited allow for Project Collection Administrators and Project Collection Service Accounts.

Readers group is assigned view-only permissions: Read.

2. Consider adding these permissions to any manually added users or groups that contributes to the development of the project; any users who should be able to check in and check out changes, make a pending change to items in a folder, or revise any committed changeset comments.
3. Consider adding this permission to any manually added users or groups that contributes to the development of the project and that must be able to create private branches, unless the project is under more restrictive development

practices.

4. Consider adding this permission to any manually added users or groups that contribute to the development of the project and that must be able to merge source files, unless the project is under more restrictive development practices.
5. Consider adding this permission to any manually added users or groups that are responsible for supervising or monitoring the project and that might or must change the comments on checked-in files, even if another user checked in the file.

Git repository permissions

You can set permissions on a Git project, repository, or branch from the context menu or from the administration page in TWA, or by using the TFSSecurity command line tool. These permissions appear only for a team project set up to use Git as the source control system.

You can set all permissions for a project or repository. You can set Administer, Contribute, and Rewrite and destroy history (force push) permissions for a branch. Repositories and branches inherit permissions from assignments made at the project level.

By default, the project-level and collection level Readers groups have only Read permissions.

Permission name	TFSSecurity Action	TFSSecurity Namespace	Description	Contributors
			Can rename the repository, add additional repositories, verify the database, and	

Administer
(Note 2)

Administer

GitRepositories

set permissions for the branch. Users who have this permission can delete the repository if they also have the Force permission.

At the branch level, can set permissions for the branch and delete the branch.

Can publish branches in the repository. Lack of this permission does not limit users from creating branches in their local repository; it merely prevents them from publishing local branches to the server. When a user creates a new branch on the server, they have Administer, Contribute, and

Branch
Creation

CreateBranch

GitRepositories

publishing local branches to the server. When a user creates a new branch on the server, they have Administer, Contribute, and

Force permissions for that branch by default.

Can push their changes to the repository.

Contribute GenericContribute GitRepositories

At the branch level, can push their changes to the branch. ✓

Note Management ManageNote

GitRepositories

Can push and edit Git notes to the repository. They can also remove notes from items if they have the Force permission. See

[this topic](#) for more details on notes. ✓✓✓

Read

GenericRead

GitRepositories

Can clone, fetch, pull, and explore the contents of the repository, but cannot push any changes they make to the repository.

✓✓✓

Rewrite and destroy history (force push)

ForcePush

GitRepositories

Can force an update, which can overwrite or discard commits from any user. Deleting commits changes the history. Without this permission, users cannot discard their own changes. Rewrite and destroy history is also required to delete a branch.

Because Rewrite and destroy history enables users to change the history or remove a commit from history, users who have this permission can delete a change and its history from the server. They can also modify the commit history of the server repository.

At the branch level, can rewrite and destroy history of the branch.

Tag Creation

CreateTag

GitRepositories

Can push tags to the repository, and can also edit or remove tags from items if they have the Force permission.



Notes:

1. For Project Collection Administrators and Project Collection Service Accounts, all permissions are set to Inherited allow.

Readers and Project Collection Build Service Accounts groups are assigned view-only permissions: Read.

2. Consider adding all permissions to any manually added users or groups that contribute to the development of the project.

Lab Management permissions

Visual Studio Lab Management permissions are specific to virtual machines, environments, and other resources. In addition, the creator of an object in Lab Management is automatically granted all permissions on that object. You can set these permissions by using the TFSLabConfig permissions command-line tool.

By default, the project-level and collection level Readers groups have only View lab resources (Read) permissions.

Permission name	TFSLabConfig <i>perm</i>	Description	Contributors (Note 1)	A
Delete Environment and Virtual Machines	Delete	Can delete environments and templates. The permission is checked for the object that is being deleted.		✓
Delete Lab Locations	DeleteLocation	Can delete the locations for Lab Management resources, which include collection host groups, collection library shares, project host groups, and project library shares. To delete a location,		✓

		you must have the Delete Lab Location permission for that location.		
Edit Environment and Virtual Machines	Edit	Can edit environments and templates. The permission is checked for the object that is being edited.	✓	✓
Environment Operations	EnvironmentOps	Can start, stop, pause, and manage snapshots, in addition to performing other operations on an environment.		✓
Import Virtual Machine	Create	Can import a virtual machine from a VMM library share. This permission differs from Write because it only creates an object in Lab Management and does not write anything to the Virtual Machine Manager host group or library share.	✓	✓
Manage		Can change the permissions of all the child Lab Management objects. For example, if a user has Manage		

Child Permissions	ManageChildPermissions	<p>Child Permission for a team project host group, the user can change permissions for all the environments under that team project host group.</p> <p>Can edit the locations of Lab Management resources, which include collection host groups, collection library shares, project host groups, and project library shares. To edit a specific location, you must have the Manage Lab Location permission for that location. This permission for collection-level locations (collection host groups and collection library shares) also allows you to create project-level locations (project host group and project library share).</p>	✓
Manage Lab Locations	ManageLocation	<p>Can edit the locations of Lab Management resources, which include collection host groups, collection library shares, project host groups, and project library shares. To edit a specific location, you must have the Manage Lab Location permission for that location. This permission for collection-level locations (collection host groups and collection library shares) also allows you to create project-level locations (project host group and project library share).</p>	✓
Manage	ManagePermissions	<p>Can modify the permissions for a Lab Management object. This permission is</p>	

Permissions		checked for the object whose permissions are being modified.		
Manage Snapshots	ManageSnapshots	Can perform all snapshot management tasks for an environment, which include taking a snapshot, reverting to a snapshot, renaming a snapshot, deleting a snapshot, and reading a snapshot.	✓	✓
Pause Environment	Pause	Can pause an environment.	✓	✓
Start	Start	Can start an environment.	✓	✓
Stop	Stop	Can stop an environment.	✓	✓
View Lab Resources	Read	Can view information for the various Lab Management resources, which include collection host groups, project host groups, and environment. To view information about a specific lab resource,	✓	✓

Write
Environment
and Virtual
Machines Write

you must have the
View Lab
Resources permission
for that resource.

Can create
environments for a
project host group.
Users who have this
permission for a
project library share
can store environments
and templates.



Notes:

1. The Readers group is assigned view-only permissions: Read.

Release Management permissions

In Release Management, you can assign permissions based on the role assigned to a user, explicit functional permissions assigned to groups, or permissions assigned to specific instances of a release object. In addition, you can assign approvers and validators to specific stages within a release path to ensure that the applications being deployed meet quality standards.

- **Role based:** The two roles are Release Manager and Service User. Release Managers can manage all functions, regardless of the groups they are linked to. Service User corresponds to a service account role. To limit a user's access, do not assign them to any role. Instead, have them inherit the permissions assigned to the group they are linked to.
- **Group:** To restrict access to specific functional areas, you assign the permissions allowed by that group. Members of that group inherit the permissions assigned to the group. Restricting access requires that you change the permissions granted to the Everyone group, which by default has all permissions.
- **Object:** In addition to roles and groups, you can restrict access to edit, view, and manage security of release paths and release templates. You do this through the Security tab on the release path and through the Properties page for a release template.
- **Approvers and Validators:** Approvers and validators must sign off at each step or stage of a release. You assign approvers and validators when you configure a release path. All approvers and validators must be added as users or a member of a group in Release Management.

Release Management defines a single default group, Everyone, to which all accounts that you add to Release Management belong. In addition, specific permissions are allocated to the Release Manager and Service User roles.

You manage Release Management permissions from the Release Management client. You can set these permissions by opening the sub-menu listed in the Where set column. To learn more about how to set these permissions, see Add

users and groups and control access to Release Management. To install Release Management, go

[here](#).

Permission name or user role	Where set	Description	Release Service Everyone Manager Role User Role role
Release Manager	Administration > My Profile and New User page	<p>Can administer the Release Management server, manage the connection between TFS and Release Management, and manage the following objects:</p> <ul style="list-style-type: none"> • Release paths and stage information defined in a release path. • Release templates, including adding custom tools and actions and deployment sequence and configuration variables for 	✓

all stages defined in a release template.

- Security for all functional areas.

Consider adding:
Users who will administer the Release Management server.

Can manage deployments or web application pools.

Consider adding:
Service account identities assigned to run server application pools, deployment agent's Windows Service, and Release Management monitoring of Windows Services.

Service User

Administration
> My Profile
and New User
page



Can view release templates or

View	Configure Apps > Release Template > Properties	release paths and selectively assign view access to specific release templates and release paths to specific users.	✓	✓	✓
	Configure Paths > Release Paths	Consider adding: Users or groups that need to view specific release templates or release paths, but not edit them.			
Edit	Configure Apps > Release Template > Properties	Can edit release templates or release paths and choose which users can edit specific release templates and release paths to specific users.	✓	✓	✓
	Configure Paths > Release Paths	Consider adding: Users or groups that need to edit specific release templates or release paths.			
		Can initiate a release and specify which users can initiate			

		<p>a release from those release templates that they can view.</p>			
Can Release	<p>Configure Apps > Release Template > Properties</p>	<p>Consider adding: Users or groups that will initiate a release. With this permission, you can specify which users can initiate a release from those release templates that they can view.</p>	✓	✓	✓
		<p>Can manage which groups have permissions to view, edit, or manage release templates or release paths.</p>			
Manage Security	<p>Configure Apps > Release Template > Properties</p>	<p>Consider adding: Users or groups that will manage which groups have permissions to view, edit, or manage release templates or release paths.</p>	✓	✓	✓
	<p>Configure Paths > Release Paths</p>	<p>With this permission, creators of release templates and</p>			

release paths can control who can view, edit, or release specific templates or paths.

Can define release templates.

Without this permission, the New button on the Configure Apps > Release Template tab is hidden.

Can Create Release Template

Configure Apps > Release Template



Consider adding: Users or groups that need to create, start, or approve a release.

Can define the stages, approval process, and security of release paths.

Without this permission, the New button on the Configure Paths > Release Paths tab is hidden.

Can Create Release Path

Configure Paths > Release Paths



Can Manage Environment

Configure Paths > Environments

Consider adding:
Users or groups that need to manage the release configuration used in deploying applications.

Can define the stages that comprise a release path and the servers and security for each stage.

Without this permission, the Configure Paths > Environments tab is hidden. ✓ ✓ ✓

Consider adding:
Users or groups that need to manage the servers and environments used to define the release paths.

Can define the release paths for deploying applications in your system. This

permission supports access to defining the servers use to deploy applications to test, stage, and production servers.

Can Manage Server
Configure Paths > Server

Without this permission, the Configure Paths > Server tab is hidden.



Consider adding: Users or groups that will define the release paths for deploying applications in your system. This permission supports access to defining the servers used to deploy applications to test, stage, and production servers.

Can define custom tools or actions for deploying applications in

Can Manage
Inventory

Inventory >
Actions and
Tools

your system. With this permission they can view, edit, and create actions and tools. See Release actions to deploy an app for Release Management.

Without this permission, the Inventory tab is hidden.



Consider adding: Users or groups that will define custom tools or actions for deploying applications in your system. With this permission they can view, edit, and create actions and tools used in deploying applications.

Can edit the Command and Arguments fields when No Tool is selected.

Configure

Without this

Can Use Custom Tool in Actions and Components	<p>Apps > Component > Deployment</p> <p>Configure Apps > Release Template > Component > Deployment</p>	<p>permission, users cannot edit these fields.</p> <p>Consider adding: Users or groups that will define release paths or release templates or who will initiate releases. This allows them to edit the Command and Arguments fields when No Tool is selected.</p>	✓	✓	✓
Edit Values and Target Servers	Configure Apps > Release Template	<p>Can edit deployment sequence and configuration variables for specific releases or stages.</p> <p>Without this permission, stage information is read-only.</p> <p>Consider adding: Users or groups that will define release paths or release templates or who will initiate releases.</p>	✓	✓	✓

This allows them to edit deployment sequence and configuration variables for specific releases or stages.

Can edit approvals and environments for a specific stage.

Without this permission, stage information is read-only.

Edit Approvals and Environment Stages
Configure Paths > Release Path >

Consider adding: Users or groups that will define release paths or release templates. This allows them to edit approvals and environments for a specific stage. Without this permission, stage information is read-only.



Q & A

Q: What's the difference between permissions and access levels?

A: Certain features in TFS are only available to users who have the appropriate licensing level for those features. Access to those features is not controlled by permissions but by membership in licensing groups for Team Web Access. See [Change access levels](#).

Q: What permissions are assigned to team administrators?

A: Team administrators are granted several role-based permissions that are described here.

Q: What permissions are associated with Alerts?

A: There are no UI permissions associated with alerts that you can subscribe to through TWA.

By default, all Contributors can subscribe to alerts for themselves. Project Collection Administrators and Project Administrators or users or members of groups who have either the Edit collection-level information or Edit project-level information can set alerts for others or for a team.

You can manage alert permissions using TFSSecurity at the collection-level. The

[Team Foundation Event service](#) is designed to be flexible and extensible.

TFSSecurity Action	TFSSecurity Namespace	Description	Project Collection Administrator and Project Collection Service Account
---------------------------	------------------------------	--------------------	--

CREATE_SOAP_SUBSCRIPTION	EventSubscription	Can create a SOAP-based web service subscription. ✓
GENERIC_READ	EventSubscription	Can view subscription events defined for a team project. ✓
GENERIC_WRITE	EventSubscription	Can create alerts for other users or for a team. ✓
UNSUBSCRIBE	EventSubscription	Can unsubscribe from an event subscription. ✓

Q: What additional features or tools reference groups?

A: These features reference built-in and custom (ones that you create) TFS groups:

- Work item queries: In Group, Not in Group operator
- Field (Definition) child XML element attribute: for and not attributes
- Field (Workflow) child XML element attribute: for and not attributes

- Access levels

Q: What is a Valid User? How are Valid User groups populated?

A: When you add accounts of users directly to a TFS group or through a Windows group, they are automatically added to one of the valid user groups.

- Server\Team Foundation Valid Users: All members added to server-level groups.
- ProjectCollectionName\Project Collection Valid Users: All members added to project-collection level groups.
- TeamProjectName\Project Valid Users: All members added to project-level groups.

The default permissions assigned to these groups are primarily limited to read access, such as View build resources, View project-level information, and View collection-level information.

This means that all users that you add to one team project can view the objects in other team projects within a collection. If you need to restrict view access, then you can set restrictions through the area path node. For additional methods, see

[Restrict access to functions and tasks.](#)

If you remove or set the View instance-level information permission to Deny for one of the Valid Users groups, no users in the group will be able to access the team project, collection, or deployment, depending on the group you set.

In addition, the VALIDUSER element can be used to allow or restrict access for work item tracking.

Q: How do I manage permissions to access reports or the project portal?

A: For information about how to set permissions in Reporting Services and SharePoint Products for users in TFS, see [Set administrator permissions for team project collections](#), and [Set administrator permissions for Team Foundation](#)

[Server.](#)

For step-by-step examples of how to create custom groups, configure permissions to control access to resources, and other options, see [Restrict access to functions and tasks](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Command-line tools for TFS

[Send Feedback](#)

Visual Studio Team Foundation Server (TFS) command-line tools do several types of tasks. Some tasks can be done through a user interface and others are accessible only through the tool. Command-line tool tasks include managing Team Foundation version control (TFVC) objects, work item tracking objects, objects that support Microsoft Test Manager, as well as configuring and managing TFS application-tier servers.

You might need to open an elevated Command Prompt window to run a command line tool, even if you log on with administrative credentials.

Client tools: To access client command-line tools, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace %programfiles(x86)% with %programfiles%.

Server tools: To access server command-line tools, open a Command Prompt window on the application-tier server where TFS is installed and enter:

```
cd %programfiles%\Microsoft Team Foundation Server 12.0\Tools
```

For the required permissions to exercise the command options, see the associated topic.

Area	Client tool	Server tool	Topics
-------------	--------------------	--------------------	---------------

Manage TFVC:



- `tf`: Team Foundation version control commands

Do nearly all tasks you can do in Visual Studio, and some that aren't supported in Visual Studio.

Manage work item tracking:

- `witadmin` command line tool

Manage and customize objects, such as work item types, categories, process configuration, and more.

- `TFSFieldMapping` command line tool

Upload or download the file that maps TFS fields to Microsoft Project fields.

- `TFSAdmin` ProjectServer



command line tool

Configure the integration of TFS and Project Server.

- TFSDeleteProject

Permanently delete a team project.

Manage Microsoft Test Management objects:

- TCM command line tools

Run tests, import automated tests, and view a list of test items.

- tcm bugfieldmapping

Import and export a mapping file to change the bug type used to automatically file bugs or customize the pick list for resolution states and failure types.



Manage TFS servers:

- TFSConfig



command line tool

Manage TFS configuration settings.

- TFSSecurity command line tool

Review and manage TFS groups, memberships, and permissions.

- TFSServiceControl command line tool

Stop or start all TFS services and application pools.



Manage Visual Studio Lab Management:

- TFSLabConfig command line tool

Manage the lab service.

Q & A

Q: Are there other command line tools that interface with TFS?

A: Yes. [Visual Studio Team Foundation Server Power Tools](#) provides companion tools to tf.exe that offer additional version control commands, work item tracking, and team project manipulation. Some commands invoke a graphical user interface. In addition, the power tools provide several user interface tools. Refer to the help content that ships with the power tools for usage.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSSConfig: Manage TFS server configuration

[See Also](#) [Send Feedback](#)

By using the **TFSSConfig** command-line tool, you can configure or create a script to configure many aspects of your deployment of Team Foundation Server (TFS). For example, you can use the commands in **TFSSConfig** to find information about or change the following items in Team Foundation Server:

- service accounts and their passwords.
- addresses and ports that TFS uses.
- the state of team project collections that are part of the deployment.
- the software updates that are applied to the application tier for Team Foundation and the databases for TFS.
- information about other components, such as Team Foundation Build or Visual Studio Lab Management.

Note

You can also configure a server interactively by using the administration console for Team Foundation. See [Configure and manage TFS resources](#).

You can use **TFSSConfig** to configure TFS to use credentials, addresses, and ports that you have already defined. You cannot use the tool to create accounts or change the addresses that components such as SharePoint Products use. For example, you can use **TFSSConfig** to change service accounts and passwords as long as the accounts and passwords already exist. You can also use **TFSSConfig** to help repair or recover your deployment if an unexpected event occurs. For

example, you can use it to repair your deployment if a software update does not apply to all components in the system or if a restoration of databases from a backup leaves a collection database improperly detached.

You must run **TFSConfig** from the application tier for Team Foundation. By default, **TFSConfig** is located in Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools.

Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window to run **TFSConfig**.

To operate correctly, **TFSConfig** must be able to connect to the following servers and services:

- The instance or instances of SQL Server that host any databases that this deployment of TFS uses, such as the configuration database and databases for team project collections.
- Any SharePoint Web applications that this deployment of TFS uses.
- Any instance of SQL Server Reporting Services that this deployment of TFS uses.
- The Active Directory domain controller, if Team Foundation Server is deployed in a domain.
- Internet Information Services (IIS) on the application-tier server or servers.
- Any server that is running Team Foundation Build and is used by this deployment.
- Any server that is running Visual Studio Lab Management and is used by this deployment.

You can use the global options that are listed in the following table with most **TFSConfig** commands. The exceptions are noted in the specific topics for those commands.

Option	Description
/?	Displays the command syntax and options for TFSConfig .
/NoPrompt or /i	Specifies that the administrator is not prompted before changes are committed.

In This Section

[Accounts Command](#)

[Authentication Command](#)

[Certificates Command](#)

[ChangeServerID Command](#)

[Collection command \[TFSSConfig\]](#)

[CodeIndex Command](#)

[ConfigureMail Command](#)

[Identities Command](#)

[Jobs Command](#)

[Lab Commands](#)

[License Command](#)

[Proxy Command \(TFSSConfig\)](#)

[RebuildWarehouse Command](#)

[RegisterDB Command](#)

[RemapDBs Command](#)

[Settings Command](#)

[Unattend Command](#)

See Also

Other Resources

[Whitepaper: Unattended Installation \(downloadable whitepaper\)](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Accounts Command

[Example](#) [See Also](#) [Send Feedback](#)

You use the **Accounts** command to change the accounts and passwords for service accounts that Visual Studio Team Foundation Server (TFS) uses. These accounts include the service account for Team Foundation Server (TFSService), the data sources account for SQL Server Reporting Services (TFSReports), and the service account for Team Foundation Server Proxy (TFSProxy). You can also use this command to change the ownership of the databases that Team Foundation Server uses.

The **Accounts** command supports on-premises TFS deployments. To determine the account owner of Visual Studio Online accounts, see [Change account ownership](#).

To change the user account or password interactively for the service account for Team Foundation Server (TFSService), you can use the administration console for Team Foundation. For more information, see [Change the service account or password for Team Foundation Server](#).

Requirements

To use the **Accounts** command, you must be a member of the Team Foundation Administrators security group and a member of the sysadmin security group for any SQL Server databases that Team Foundation Server uses. If you use the **/proxy** option, you must be an administrator on the proxy server. For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative credentials, you must open an elevated

Command Prompt window to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt window, click Start, right-click Command Prompt, and then click Run as Administrator. For more information, see this page on the Microsoft Web site: [User Account Control](#).

```
TFSSConfig Accounts /change|add|set|delete|updatepassword|resetowner
```

Parameters

Argument	Description
AccountName	Specifies the name of the account that is used as the service account, in Domain\UserName or Computer\UserName format.
Password	Specifies the password for the account that is used as the service account.
ServerName	Used only with /resetowner /SQLInstance . Specifies the name of the server that hosts the databases for Team Foundation Server, and, if an instance other than the default instance is used, the name of the instance. If you specify an instance, you must use the following format: ServerName\InstanceName
	Used only with /resetowner

DatabaseName /**DatabaseName**. Specifies the name of the database whose ownership will be changed.

Option

/change

/add

Cl
as
of
sp
gr
pe
th
yc
/A
of
th
ch

A
th
ac
Th
yc
gr
pe
ac
pc
w
is
Th
ne
sc
of
se

/set

/delete

/ResetOwner

be
en

Se
ac
ad
Th
of
ha
re
ne
of
sc

Re
ac
Th
th
ne
th
to
pc
w
is
M
th
th
cu
ac

Cl
da
Se
yc
cc
th

th
yc
co
us
or

/UpdatePassword

Cl
ac
ac
pa
yc
Te
th

Sp
yc
us
pe
of
co
(A

/AccountType:{ AdminConsole|ApplicationTier|ReportingDataSource|Proxy}

ac
Fc
th
Re
(T
ac
Se
or
va

Sp
ac
ch
re
/A

/Account:AccountName

/Password:Password

/SQLInstance:ServerName

Sp
fo
De
Co
wa
su
Lo
Co
Fo
to
th
to

Sp
se



T
y
o
a
S

U:
Sp
th
th
w
th
m
in

Se

U:

/DatabaseName:DatabaseName

/continue

/usesqlalwayson

Sp
da
w
cc
ov
yc
w
cc

U
av
cc
us

U
cc
an
th
A
SC
of
in

Fc
[A](#)
[S](#)

Remarks

Use the **Accounts** command to automate changes to the service accounts, databases, and service account groups of Team Foundation Server. You can configure accounts that you have already created, but you can't create accounts.

Before you change the domain or workgroup of an account, the account must have the Account is sensitive and cannot be delegated permission on the application-tier server. For more information, see this page on the Microsoft Web site: [Enabling Delegated Authentication](#).

Example

The following example shows how to change the service account of data sources for Reporting Services to a new account in the Contoso domain, Contoso\NewAccount, and the password, to Password.

Note

The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, places, or events is intended or should be inferred.

Copy Code

```
TFSSConfig Accounts /change /AccountType:ReportingDataSource /Account
```

The following example shows how to add the Network Service system account to the service account groups for Team Foundation Server. System accounts do not have passwords.

Copy Code

```
TFSSConfig Accounts /add /AccountType:ApplicationTier /Account:"NT Au
```

The following example shows how to change the ownership of the database that is named TFS_Warehouse on the server that is running ssnoversion and that is named "ContosoMain" on the named instance "TeamDatabases" to the user account under which you are running the command.

Note

You cannot specify what account to set as the owner of the databases when you use this command. The owner will be set to the account under which you are running the command.

 **Copy Code**

```
TFSSConfig Accounts /ResetOwner /SQLInstance:ContosoMain\TeamDatabase
```

See Also

Other Resources

[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Authentication Command

[Example](#) [See Also](#) [Send Feedback](#)

The **Authentication** command changes the network authentication protocol that a website for Team Foundation Server (TFS) uses.

Requirements

- You must be a member of the Team Foundation Administrators security group on the application-tier server.
- If you use the **siteType** option, you must be an administrator on the application-tier server or the proxy server.
- Even if you are logged on with administrative credentials, you must open an elevated Command Prompt.

```
TFSSConfig Authentication [/provider:NTLM|Negotiate] [/viewAll] [/sit
```

Parameters

Argument	Description
NTLM	Use with the /provider option to specify the NTLM authentication protocol.
Negotiate	Use with the /provider option to specify the Negotiate (Kerberos) authentication protocol.

Option	Description
/viewAll	Displays the current authentication settings for TFS.
/provider	Specifies the authentication provider you want to configure for the website.
/siteType	Specifies the website whose network authentication protocol you want to change. If you do not specify a name, the application tier for TFS is changed. If you specify

the proxy switch, Team
Foundation Server Proxy is
changed.

Remarks

The **Authentication** command is used by an administrator who wants to change the network authentication protocol for one or more websites on which TFS relies. The administrator runs this command from the application tier to update those websites that require a change in their network authentication protocol. The command changes the **NTAuthenticationProviders** property in the IIS metabase.

Important

Before you use the **Authentication** command to change the authentication protocol, you should run the command with the **/viewAll** option to view the existing settings.

Example

The following example displays the current value that is assigned for the network authentication protocol.

 **Copy Code**

```
>TFSSConfig Authentication /viewAll
```

See Also

Concepts

[Service accounts and dependencies in Team Foundation Server](#)

Other Resources

[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Certificates Command

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **Certificates** command to change how certificates are configured for client authentication in a deployment of Visual Studio Team Foundation Server (TFS) that utilizes HTTPS, secure sockets layer (SSL), and certificates. By default, the **Certificates** command will automatically select a client certificate from the certificate list for the current user. However, you can use the options for the command to specify a specific certificate or certificates from the current user context or from the local machine context.

Before you use the **Certificates** command, you must first configure the servers in your deployment of TFS to utilize certificates. For more information, see [Set up HTTPS with Secure Sockets Layer \(SSL\) for Team Foundation Server](#).

Required Permissions

To use the **Certificates** command, you must be a member of the Team Foundation Administrators security group and the local Administrators group on the computer from which you run the command. For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window to perform this function.

```
TFSSConfig Certificates [/machine] [/disable] [/autoSelect] [/noprompt]
```

Parameters

Placeholder	Description
Thumbprint	Specifies the hexadecimal string that contains the secure hash algorithm (SHA) in SHA-1 hash form of the certificate that you want to use. For more information about how to find the thumbprint for a certificate, see the following page on the Microsoft Web site: How to: Retrieve the Thumbprint of a Certificate .

Option	Description
/machine	Specifies that the certificate list will be from the local machine context instead of the current user context.

/disable

Specifies that the client authentication certificate setting will be disabled.

/autoSelect

Specifies that a certificate will be automatically selected from the certificate list. The Manage Client Certificates window will not open.

/noprompt

Specifies that the Manage Client Certificates window will not open when the Certificates command is run.

Specifies that the certificate that matches the specified thumbprint

/thumbprints:thumbprint

will be used.
You can
specify more
than one
certificate by
separating
individual
thumbprints
with a
comma.

Remarks

You use the **Certificates** command to configure the client certificates that are used by a deployment of TFS that has been configured to use HTTPS/SSL and certificates. If you use the Certificates command with no options, a client certificate will be automatically selected from the current user context from which you run the command.

Example

The following example shows how to specify the local machine certificate that has the thumbprint "aa bb cc dd ee" with no prompting.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
TFSSConfig Certificates /machine /thumbprint:aa bb cc dd ee /noprompt
```

The following example shows how to specify using automatic selection of a client certificate from the current user store.

Copy Code

```
TFSSConfig Certificates /autoselect
```

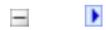
See Also

Concepts

[Team Foundation Server architecture](#)

Other Resources

[TFSSConfig: Manage TFS server configuration](#)
[Examples of Complex Topology](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

ChangeServerID Command

[Example](#) [See Also](#) [Send Feedback](#)

The **ChangeServerID** command changes the global universal identifications (GUIDs) that are associated with the databases for Visual Studio Team Foundation Server (TFS). GUIDs must be unique within a deployment of TFS. If more than one database has the same GUID, your deployment can become unstable or unusable. You can change the GUID for the configuration database, the GUIDs for all team project collection databases in the deployment, or both. Although you would not typically use this command in daily operations, you might use this command in the following circumstances:

- You restored your deployment to new hardware, the old deployment is still operational, and you want to utilize both deployments. This scenario is sometimes referred to as cloning the server.
- You want to test a software update or a hardware configuration on a duplicate deployment so that you do not risk disrupting your production environment.
- You want to fully test the restoration of databases to new hardware in a test lab or separate environment, to ensure that your deployment can be restored.
- You must reset the GUID for a collection database after moving it to another deployment for which that GUID is already reserved.

 **Note**

The ChangeServerID command is not reversible. After a GUID has been changed, you cannot undo that change except by restoring a previous version of that database.

Required Permissions

To use the **ChangeServerID** command, you must be a member of the Team Foundation Administrators security group and a member of the sysadmin security group for any SQL Server databases that Team Foundation Server uses. For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window to perform this function.

```
TFSSConfig ChangeServerID [/SQLInstance:ServerName] /DatabaseName:Conf
```

Parameters

Placeholder	Description
ServerName	<p>Specifies the name of the server that hosts the configuration database for TFS and the name of the instance if you want to use an instance other than the default instance. If you specify an instance, you must use the following format:</p> <p>ServerName\InstanceName</p>
DatabaseName	<p>Specifies the name of the configuration database. By default, the name of this database is TFS_ConfigurationDB.</p>
Option	Description
/SQLInstance:ServerName	<p>Required. Specifies the name of the server that is running SQL Server and the name of the instance if you want to use an instance other than the default instance. If you</p>

specify an instance, you must use the following format:

ServerName\InstanceName

/DatabaseName:DatabaseName Required. Specifies the name of the configuration database for TFS.

/ProjectCollectionsOnly Specifies that only the GUIDs for collections will be changed.

/ConfigDBOnly Specifies that only the GUID for the configuration database will be changed.

/usesqlalwayson Specifies that the databases are part of an AlwaysOn Availability Group in SQL Server. If configured, this option sets MultiSubnetFailover in the connection string.

For more information, see [AlwaysOn Availability Groups \(SQL Server\)](#).

Remarks

You use the **ChangeServerID** command to create a discrete duplicate of a deployment of Team Foundation Server for testing or cloning purposes. After you use the ChangeServerID command, you must direct clients to create a connection to the changed server before it can be used.

Example

The following example shows how to change the GUIDs of all databases in the Contoso1 deployment of TFS, where the configuration database is hosted on the server that is named "ContosoMain" on the named instance "TeamDatabases" in SQL Server.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
TFSSConfig ChangeServerID /SQLInstance:ContosoMain\TeamDatabases /Da1
```

See Also

Concepts

[Back up and restore TFS](#)

Other Resources

[TFSSConfig: Manage TFS server configuration](#)

[Move a team project collection](#)

[Split a team project collection](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

CodeIndex Command

[See Also](#) [Send Feedback](#)

Use the **CodeIndex** command to manage code indexing on Team Foundation Server. For example, you might want to reset the index to fix CodeLens information, or turn off indexing to investigate server performance issues.

Required Permissions

To use the **CodeIndex** command, you must be a member of the Team Foundation Administrators security group. See [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window to run this command. You must also run this command from the application tier for Team Foundation.

```
TFSSConfig CodeIndex /indexingStatus | /setIndexing:[ on | off | keep
```

Parameters

Argument	Description
CollectionName	Specifies the name of the team project collection. If the name has spaces, enclose the name with quotation marks, for example, "Fabrikam Web Site".
CollectionId	Specifies the identification number of the team project collection.
ServerPath	Specifies the path to a code file.
	Option
/indexingStatus	Description Show the status and configuration of the code indexing service. <ul style="list-style-type: none">• on: Start indexing all changesets.

/setIndexing:[on | off | keepupOnly]

- **off:** Stop indexing all changesets.
- **keepupOnly:** Stop indexing previously created changesets and start indexing new changesets only.

Specifies a list of code files and their paths that you don't want indexed.

/ignoreList:[add | remove | removeAll | view] ServerPath

You can use the wildcard character (*) at the start, end, or both ends of the server path.

- **add:** Add the file that you don't want indexed to the ignored file list.
- **remove:** Remove the file that you want indexed from the ignored file list.
- **removeAll:** Clear the ignored file list and start indexing all files.
- **view:** See all the files that aren't being indexed.

Shows the specified number of files that exceeds the specified

/listLargeFiles

[/fileCount:FileCount/minSize:MinSize]

size in KB. You can then use the **/ignoreList** option to exclude these files from indexing.

For this, you'll need [Team Foundation Server 2013 with Update 3](#).

/reindexAll

Clear previously indexed data and restart indexing.

/destroyCodeIndex [/noPrompt]

Delete the code index and remove all indexed data. Does not require confirmation if you use the **/noPrompt** option.

Control how much temporary data that CodeLens creates when processing changesets. The default limit is 6 GB (2 GB in Update 5).

- **view**: Show the current size limit.
- **SizeInGBs**: Change the size limit.
- **disable**: Remove the size limit.

/temporaryDataSizeLimit:[view | <SizeInGBs> | disable]

This limit is checked before CodeLens processes a new changeset. If temporary data exceeds this limit, CodeLens will pause processing past changesets, not new ones. CodeLens will restart processing after the data is cleaned up and falls below this limit. Cleanup runs automatically once a day. This means temporary data might exceed this limit until cleanup starts running.

For this, you'll need [Team Foundation Server 2013 with Update 4](#).

Control how long to index your change history. This affects how much history CodeLens shows you. The default limit is 12 months. This means CodeLens shows your change history from the last 12 months only.

- **view:** Show the current number of months.

/indexHistoryPeriod:[view | all |

<NumberOfMonths>]

- **all:** Index all change history.
- **NumberOfMonths:** Change the number of months used to index change history.

For this, you'll need [Team Foundation Server 2013 with Update 4](#).

/collectionName:CollectionName

Specifies the name of the team project collection on which to run the **CodeIndex** command. Required if you don't use **/CollectionId**.

/collectionId:CollectionId

Specifies the identification number of the team project collection on which to run the **CodeIndex** command. Required if you don't use **/CollectionName**.

Examples

Note

The example companies, organizations, products, domain names, email addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, places, or events is intended or should be inferred.

To see the code indexing status and configuration:

Copy Code

```
TFSSConfig CodeIndex /indexingStatus /collectionName:"Fabrikam Web Site"
```

To start indexing all changesets:

Copy Code

```
TFSSConfig CodeIndex /setIndexing:on /collectionName:"Fabrikam Web Site"
```

To stop indexing previously created changesets and start indexing new changesets only:

Copy Code

```
TFSSConfig CodeIndex /setIndexing:keepupOnly /collectionName:"Fabrikam Web Site"
```

To find up to 50 files that are larger than 10 KB:

Copy Code

```
TFSSConfig CodeIndex /listLargeFiles /fileCount:50 /minSize:10 /collectionName:"Fabrikam Web Site"
```

To exclude a specific file from indexing and add it to the ignored file list:

 **Copy Code**

```
TFSSConfig CodeIndex /ignoreList:add "$/Fabrikam Web Site/Catalog.cs'
```

To see all the files that aren't indexed:

 **Copy Code**

```
TFSSConfig CodeIndex /ignoreList:view
```

To clear previously indexed data and restart indexing:

 **Copy Code**

```
TFSSConfig CodeIndex /reindexAll /collectionName:"Fabrikam Web Site"
```

To save all changeset history:

 **Copy Code**

```
TFSSConfig CodeIndex /indexHistoryPeriod:all /collectionName:"Fabrikam Web Site"
```

To remove the size limit on CodeLens temporary data and continue indexing regardless of temporary data size:

 **Copy Code**

```
TFSSConfig CodeIndex /temporaryDataSizeLimit:disable /collectionName:"Fabrikam Web Site"
```

To delete the code index with confirmation:

 **Copy Code**

```
TFSSConfig CodeIndex /destroyCodeIndex /collectionName:"Fabrikam Web Site"
```

See Also

Other Resources

[TFSSConfig: Manage TFS server configuration](#)
[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Collection command [TFSSConfig]

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **Collection** command to attach, detach, or delete a team project collection from a deployment of Visual Studio Team Foundation Server (TFS). You can also use the **Collection** command to duplicate the database of an existing collection, rename it, and attach it to the deployment. This process is sometimes referred to as cloning a collection. However, you cannot use the **Collection** command to create a team project collection.

To manage collections interactively or to create a collection, you can use the Team Project Collections node in the administration console for Team Foundation. See [Manage team project collections](#).

Requirements

To use the **Collections** command, you must be a member of the following groups:

- the Team Foundation Administrators security group and the Administrators security group on the server or servers that are running the administration console for Team Foundation
- the sysadmin group on the server or servers that are running the instance of SQL Server that hosts the databases for Team Foundation Server
- the Farm Administrators group for the farm from which you are deleting the site collection, if your deployment uses SharePoint Products and you are using the **/delete** option.

For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window to perform this function.

```
TFSSConfig Collection {/attach | /detach | /delete} [/collectionName:
```

Parameters

Placeholder	Description
CollectionName	Specifies the name of the team project collection. If the name of the collection contains spaces, you must enclose the name in quotation marks (for example, "My Collection").
ServerName	Specifies the name of the server that hosts the configuration database for TFS, and the name of the instance if you want to use an instance other than the default instance. If you specify an instance, you must use the following format: ServerName\InstanceName
DatabaseName	Specifies the name of the configuration database. By default, the name of this database is TFS_ConfigurationDB.

Option	Description
/attach	<p>Required if neither /detach nor /delete is used. If you specify this option, you must also use the /collectionDB option. As an option, you can also use /collectionName and /clone with this option. If you use the /attach option, the specified collection database will be added to your deployment of TFS.</p>
/detach	<p>Required if neither /attach nor /delete is used. If you specify this option, you must also use the /collectionName option. If you use the /detach option, the database for the</p>

specified collection will be stopped, and the collection will be detached from your deployment of TFS.

Required if neither **/detach** nor **/attach** is used. If you specify this option, you must also use the **/collectionName** option. If you use the **/delete** option, the database for the specified collection will be stopped, and the collection will be permanently detached from TFS. You will not be able to re-attach the collection database to this or any other deployment.

 **Tip**

/delete

The **/delete** option will not delete the collection database from SQL Server. After deleting the collection database from TFS, you can delete the database manually from SQL Server.

Required if either **/detach** or **/delete** is used. If you use this option with **/detach** or **/delete**, it specifies the collection that will be detached or deleted. If you use this option with **/attach**, it specifies a new name for the collection. If you use this option with both **/attach** and **/clone**, it specifies the name for the

/CollectionName:CollectionName

duplicate
collection.

Required if
/attach is used.

This option
specifies the
name of the
server that is
running SQL
Server and the
name of the
collection
database that is
hosted on that
server.

/CollectionDB:ServerName;DatabaseName

If you specify
this option, the
original
collection
database will be
attached as a
clone in SQL
Server, and this
database will be
attached to TFS.
This option is
primarily used
as part of
splitting a team
project
collection.

/clone

Example

The following example shows how to permanently remove the "Contoso Summer Intern Projects" team project collection from a deployment of Team Foundation Server.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
TFSSConfig Collection /delete /CollectionName:"Contoso Summer Intern
```

```
TFSSConfig - Team Foundation Server Configuration Tool  
Copyright © Microsoft Corporation. All rights reserved.  
Deleting a team project collection is an irreversible operation. A d  
Yes  
Found Collection 'Contoso Summer Intern Projects' Deleting...  
The delete of collection 'Contoso Summer Intern Projects' succeeded.
```

The following example shows how to duplicate the "Contoso Summer Interns Projects" team project collection, name it "Contoso Winter Interns Projects," and attach the duplicate collection to the deployment of Team Foundation Server.

```
TFSSConfig Collection /attach /collectiondb:"ContosoMain;TFS_Contoso
```

See Also

Concepts

[Back up and restore TFS](#)

Other Resources

[TFSSConfig: Manage TFS server configuration](#)

[Move a team project collection](#)

[Split a team project collection](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

ConfigureMail Command

[Example](#) [See Also](#) [Send Feedback](#)

You can support the generation of automatic alerts when changes occur to work items and other conditions by configuring the mail server. The mail server that you specify supports all team projects defined for all project collections that are defined on the application-tier server where you run the command. For e-mail notifications to be sent to team members, you must configure the server that runs Team Foundation Server (TFS) to use an existing SMTP server by using the **ConfigureMail** command.

Required Permissions

To use the **ConfigureMail** command, you must be a member of the Team Foundation Administrators security group on the Team Foundation application-tier server. For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative permissions, you must open an elevated Command Prompt window to perform this function.

```
TFSSConfig ConfigureMail /FromEmailAddress:emailAddress /SmtpHost:SM1
```

Parameters

Option	Description
/FromEmailAddress: emailAddress	<p>Specifies the address from which to send e-mail notifications from TFS for a check in, work item assigned to you, or other notifications. This address is also checked for validity and, depending on your server configuration, might have to represent a valid e-mail account on the mail server.</p> <p>If the address does not exist or is not valid, the default e-mail</p>

address is
used.

/SmtpHost:SMTPHostName

Specifies the
name of the
server that
hosts the mail
server.

Example

The following example shows the syntax used to configure the from e-mail address to TFS@contoso.com and the SMTP mail server as ContosoMailServer:

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
TFSSConfig ConfigureMail /FromEmailAddress:TFS@contoso.com /SmtpHost:
```

▣ See Also

Other Resources

[Configure an SMTP server to support alerts and feedback requests](#)

[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Diagnose Command

[See Also](#) [Send Feedback](#)

The Diagnose command has been deprecated. Do not use it.

If you need assistance with diagnosing potential mismatches between software updates on your application-tier and data-tier servers for Team Foundation Server (TFS), contact [Microsoft Support](#).

See Also

Concepts

[Team Foundation Server architecture](#)

Other Resources

[Upgrading Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Identities Command

[See Also](#) [Send Feedback](#)

The Identities command lists or changes the security identifier (SID) of users and groups in your deployment of Team Foundation Server (TFS). You might need to change or update the SID for users and groups in one of the following scenarios:

- changing the domain of your deployment
- changing from a workgroup to a domain or from a domain to a workgroup
- migrating accounts across domains in Active Directory

Note

You do not need to run this command if you are changing domains within the same Active Directory forest. TFS will automatically handle SID changes for moves within the same forest.

Requirements

- You must be a member of the Team Foundation Administrators security group on the application-tier server for Team Foundation.
- You must also be a member of the sysadmin security group for SQL Server on the data-tier server for Team Foundation. For more information, see [Set administrator permissions for Team Foundation Server](#).
- Even if you are logged on with administrative credentials, you must open an elevated Command Prompt to perform this function.

TFSSConfig Identities [/change /fromdomain:DomainName1 /todomain:Dom

Parameters

Placeholder	Description
DomainName	Specifies the name of the domain from which (/fromdomain) or to which (/todomain) you are changing SIDs. If you must specify a workgroup, use the computer name as the domain name.
AccountName	Specifies the name of the account that you want to list or change. When you use this parameter with /change , you specify the account from which you want to change after /account , and you specify the account to which you want to change after /toaccount . Do not include a domain or computer name with the account name.
	Specifies the name of the server that hosts the configuration database for TFS and the name of the

ServerName instance if you want to use an instance other than the default instance. If you specify an instance, you must use the following format:

ServerName\InstanceName

DatabaseName Specifies the name of the configuration database. By default, the name of this database is TFS_ConfigurationDB.

Option

Description

/change

Specifies that you want to change identities instead of listing them.

/fromdomain:DomainName

Required when using **/change**. Specifies the original domain of the identities that you want to change. If you are changing from a workgroup environment, specifies the name of the computer.

Required when using **/change**. Specifies the domain to which you want

/todomain:DomainName to change identities. If you are changing to a workgroup environment, specifies the name of the computer.

/account:AccountName Specifies the name of an account for which you want to list or change identities.

/toaccount:AccountName Specifies the name of an account to which you want to change identities.

/SQLInstance:ServerName Specifies the name of the server that is running SQL Server and the name of the instance if you want to use an instance other than the default instance. If you specify an instance, you must use the following format:

ServerName\InstanceName

/DatabaseName:DatabaseName Specifies the name of the configuration database for TFS.

Specifies that the databases are part of an AlwaysOn Availability Group in SQL

/usesqlalwayson

Server. If configured, this option sets MultiSubnetFailover in the connection string.

For more information, see [AlwaysOn Availability Groups \(SQL Server\)](#).

Remarks

You can optionally specify the database to change identities before you configure an application-tier server for the deployment. For example, you might specify the database to change the service account when you clone a deployment of TFS.

When you change identities, the target account or accounts must already exist in Windows.

You must wait for the next identity synchronization with Windows before the properties of accounts that you change with this command will be updated. This requirement includes changes from group to user, user to group, and domain account to local account.

Examples

The following example shows how to list the names of all Windows users and groups that are stored in TFS and to display whether the SID for each user or group matches the SID in Windows. The Contoso1 domain administrators created domain groups such as "Contoso1\Developers" and "Contoso1\Testers" to help ease the management of permissions across TFS, SQL Server Reporting Services, and SharePoint Products.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

TFSSConfig Identities

Sample output:



```
TFSSConfig - Team Foundation Server Configuration Tool  
Copyright © Microsoft Corporation. All rights reserved.
```

```
Account Name Exists (see note 1) Matches (see note 2)  
-----
```

```
CREATOR OWNER True True  
Contoso1\hholt True True  
BUILTIN\Administrators True True  
Contoso1\Developers True True  
Contoso1\Testers True True  
Contoso1\PMs True True  
Contoso1\jpeoples True True  
Contoso1\Domain Admins True True  
Contoso1\SVACCT1 True True
```

```
9 security identifiers (SIDs) were found stored in Team Foundation S
```

The following example shows how to change the SIDs for all accounts in Team Foundation Server from the Contoso1 domain to the SIDs for accounts that have matching names in the ContosoPrime domain. Only account names that match will have their SIDs updated. For example, if the "hholt" account exists as Contoso1\hholt and ContosoPrime\hholt, the account SID will be changed to the SID for ContosoPrime\hholt. If the "ContosoPrime\hholt" account does not exist, the SID will not be updated for Contoso1\hholt.

Copy Code

```
TFSSConfig Identities /change /fromdomain:Contoso1 /todomain:ContosoPrime
```

The following example shows how to change the account for a single user account, Contoso1\hholt, to the account for another user account, ContosoPrime\jpeoples.

Copy Code

```
TFSSConfig Identities /change /fromdomain:Contoso1 /todomain:ContosoPrime
```

The following example shows how to change the SID of the "NT AUTHORITY\NETWORK SERVICE" service account that is used in the deployment of Team Foundation Server when changing the domain of the deployment from Contoso1 to ContosoPrime. To change a system account such as Network Service, you must follow a two-stage process. You first change the service account from NT AUTHORITY\NETWORK SERVICE to a domain account in the new domain (TempSVC), and then you change the account back to NETWORK SERVICE on the server in the new domain. The configuration database is hosted on the server that is named "ContosoMain" on the named instance "TeamDatabases" in SQL Server.

```
TFSSConfig Identities /change /fromdomain:"NT AUTHORITY" /todomain:ContosoPrime
```

```
TFSSConfig Identities /change /fromdomain:ContosoPrime /todomain:"NT AUTHORITY"
```

See Also

Concepts

[Service accounts and dependencies in Team Foundation Server](#)

Other Resources

[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Import Command (TFSSConfig)

[Send Feedback](#)

The Import command has been deprecated. Do not use it.

If you need assistance with upgrading data and projects from an earlier version of Team Foundation Server (TFS), see Upgrade Team Foundation Server, or contact [Microsoft Support](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Jobs Command

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **Jobs** command to create a log file that provides the details of the most recent job activity for a specific team project collection. Jobs are run at any time when an update is required to change the information or configuration for a collection. You can also use this command to retry a job for one or all team project collections.

To retry a job interactively, you can open the administration console for Team Foundation, click the Status tab for the collection, and then click Retry Job. For more information, see [Open the Team Foundation Administration Console](#).

Required Permissions

To use the **Jobs** command, you must be a member of the Team Foundation Administrators security group. For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window to perform this function.

```
TFSSConfig Jobs /retry [/CollectionName:CollectionName] [/Collection]
TFSSConfig Jobs /dumplog [/CollectionName:CollectionName] [/Collectio
```

Parameters

Placeholder	Description
CollectionName	Specifies the name of the team project collection. If the name of the collection contains spaces, you must enclose the name with quotation marks (for example, "My Collection") If you want to specify all collections, you can use an asterisk (*).
ID	Specifies the identification number of the team project collection.
Option	Description
	Required if /dumplog is not used. Specifies that the most recent job will be reattempted for the specified

/retry

team project collection. If you use this option, you must also use either the **/CollectionName** or the **/CollectionID** option.

Required if **/retry** is not used. Specifies that the most recent job activity for the collection will be sent to a log file. If you use this option, you must also use either the **/CollectionName** or **/CollectionID** option.

/dumplog

Required if **/CollectionID** is not used. Specifies the name of the collection for which the most recent job will be either retried (**/retry**) or logged (**/dumplog**). If

/CollectionName:CollectionName

you want to specify all collections, you can use an asterisk (*).

/CollectionID:ID

Required if **/CollectionName** is not used. Specifies the identification number of the collection for which the most recent job will be either retried (**/retry**) or logged (**/dumplog**).

Example

The following example shows how to create a log file that lists the most recent job activity for the "Contoso Summer Intern Projects" team project collection in Team Foundation Server.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
TFSSConfig Jobs /dumplog /CollectionName:"Contoso Summer Intern Proje
```

See Also

Concepts

[Back up and restore TFS](#)

Other Resources

[TFSSConfig: Manage TFS server configuration](#)

[Move a team project collection](#)

[Split a team project collection](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Lab Commands

[See Also](#) [Send Feedback](#)

Visual Studio Lab Management enables you to manage virtual environments used in the development, deployment, and testing of your software applications. Lab Management works with Microsoft System Center Virtual Machine Manager (SCVMM) to accomplish these tasks.

You can use the **TfsConfig Lab** commands to configure Lab Management in the Team Foundation Server Application Tier and to add SCVMM host groups and library shares to team project collections.

- A host group is a logical container that is used to manage one or more physical machines that host virtual machines. Host groups are created by an administrator in SCVMM.
- A library share is a designated share that provides access to the file-based resources such as ISO images and virtual hard disks that Lab Management uses to create and deploy virtual environments. Library shares are created in SCVMM by the SCVMM administrator.

To manage the host groups and library of individual team project from the command line, use the TfsLabConfig tool. For more information, see [Configure Lab Management with TFSLabConfig](#).

Common Tasks

Task	Related Content
<p>Set up Team Lab Management and configure network isolation. The Settings options identify the SCVMM that will be used by Lab Management and the network location that will be used for virtual machines. Other options enable you to configure Network Isolation which allows you to run multiple copies of a lab environment to run at the same time without causing network conflicts, such as conflicts in computer names and Domain Name System (DNS) registration.</p>	<p>TFSSConfig Lab /Settings Commands</p>
<p>Manage SCVMM host groups for a team project collection. The HostGroup options lets</p>	

you assign a host group that was created in SCVMM to a team project collection, remove an assignment, or modify Lab Management properties of the host group.

[TFSConfig
Lab
/HostGroup
Commands](#)

Manage SCVMM library shares for a team project collection. The

LibraryShare options lets you assign a library share to a team project collection, remove an assignment, or modify Lab Management properties of the library share.

[TFSConfig
Lab
/LibraryShare
Commands](#)

Reassign DNS records to moved or modified team project collections.

When you move a team project collection from one instance of Team Foundation Server to another, or when you change the Team Foundation Server service account, you might have to reassign DNS names that were generated by Lab Management. The **DNS** options enable this

[TFSConfig
Lab /DNS
Commands](#)

procedure.

Upgrade from a previous
version of SCVMM.

[Upgrading
SCVMM
2008 R2 to
SCVMM
2012](#)

See Also

Concepts

[Configure Lab Management with TFSLabConfig](#)

Other Resources

[TFSConfig: Manage TFS server configuration](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSTConfig Lab /Settings Commands

[Example](#) [See Also](#) [Send Feedback](#)

You can configure Visual Studio Lab Management by using the **TFSTConfig Lab /Settings** option. The Settings option

- Sets the name of the System Center Virtual Machine Manager (SCVMM) Server that controls the administration of virtual machines in your lab.
- Sets the network location, such as the network domain or workgroup, that the physical computers in all host groups can connect to.
- Sets the IP addresses and virtual DNS suffix for network-isolated networks in your lab.

TfsConfig Lab
/Settings

```
[/ScVmmServerName:VMMServerName]  
[/NetworkLocation:networkLocation]  
[/IpBlock:networkIsolationIpRange]  
[/DnsSuffix:networkIsolationDnsSuffix]  
[/NoPrompt]  
[/List]
```

Options

Option	Description
ScvmmServerName:VMMServerName	Optional. Sets the fully qualified name of the System Center Virtual Machine Manager 2008 (SCVMM) server that will be used by Team Foundation Server. This is the name of the SCVMM server that will be used to manage the virtual machines.
	For Team Foundation Server to communicate with SCVMM, you must add the account under which Team Foundation Server is running to the Administrator role in SCVMM.
	Optional. Sets the

NetworkLocation:networkLocation

fully qualified name of a network, such as a network domain or workgroup, that is available on all the hosts in your lab network. When it provisions a virtual machine, Lab Management automatically connects the virtual machine to the specified network. You can find the available network locations on a host using SCVMM Administrator Console.

Optional. Sets the range of IP addresses to be assigned to the virtual machines in an environment when an isolated network is created. Because the IP addresses are used only for internal routing among virtual machines and are not exposed beyond

IpBlock:networkIsolationIpRange

the boundaries of an environment, you can specify any IP range that is not used within the network specified by the **/NetworkLocation** option. In most cases, the default range of 192.168.23.0/24 should work. If you encounter problems connecting to network isolated environments, you might have to choose a different range.

DnsSuffix:networkIsolationDnsSuffix

Optional. Sets the DNS suffix that will be used to register the names of virtual machines on the isolated network for the virtual environment. To confirm that the suffix is configured correctly in the DNS hierarchy, contact your network

administrator.

NoPrompt

Optional. Do not prompt for confirmation. Do not use with the **/List** option.

List

Lists the current configuration values for Lab Management.

Remarks

One of the **/ScvmmServerName**, **/NetworkLocation**, **/IpBlock**, **/DnsSuffix**, or **/List** options must be specified on each **TfsConfig Lab /Settings** command line.

To set up Lab Management, you must specify both **/ScVmmServerName** and the **/NetworkLocation** options. However, you can specify these options on separate command lines.

To set up network isolation, you must specify both the **/IpBlock** and **/DnsSuffix** options. However, you can specify these options on separate command lines.

Network isolation enables you to use multiple copies of a virtual machine without machine name or IP address conflicts. You must assign both a DNS suffix and an IP range for an isolated network. Because the IP addresses are used only for internal routing among virtual machines and are not exposed beyond the boundaries of an environment, you can specify any IP range that is not used within your public network. In most cases, the default range of 192.168.1.0/24 should work. If you encounter problems connecting to network-isolated environments, you might have to choose a different range.

Example

In the first example, Lab Management is set up by using the **/ScvmmServerName** and **/NetworkLocation** options on a single command line. In the second example, network isolation is configured by using the **/IpBlock** and **/DNSSuffix** options on separate command lines.

 **Copy Code**

```
REM First example
tfscnfig lab /settings /scvmmservername:vmmserver /networklocation:

REM Second example
tfscnfig lab /settings /ipblock: 192.168.23.0/24
tfscnfig lab /settings /dnssuffix:virtual1.lab1.contoso.com
```

See Also

Reference

[TFSSConfig Lab /LibraryShare Commands](#)

[TFSSConfig Lab /HostGroup Commands](#)

Concepts

TFSSConfig Lab Command Reference

TFSSLabConfig Command Reference

[Configure and administer Lab Management](#)

Other Resources

[TFSSConfig: Manage TFS server configuration](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TfsConfig Lab /HostGroup Commands

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **TfsConfig Lab /HostGroup** commands to add, edit, or delete the assignment of a System Center Virtual Machine Manager (SCVMM) host group to a team project collection. Host groups that are assigned in this manner are managed by Visual Studio Lab Management.

```
TfsConfig Lab /hostgroup /CollectionName:collectionName
{ /Add
    /SCVMMHostGroup:vmmHostPath
    /Name:name
    [LabEnvironmentPlacementPolicy:{Conservative|Aggressive}]
    [/AutoProvision:{True|False}]
    [/DNSSuffix:dnsSuffix]
| /Delete
    /Name:name
    [/NoPrompt]
| /Edit
    /Name:name
    {[AutoProvision:{True|False}]
    [/LabEnvironmentPlacementPolicy:{Conservative|Aggressive}]
    [/DNSSuffix:dnsSuffix]}
    [/NoPrompt]
| /List
| /ListSCVmmHostGroups }
```

Parameters

	Option	Description
	CollectionName:collectionName	Required. The project collection name. The application-tier server.
Add		Adds the specified group to the team. You must specify the /SCVmmHost options with Add .
Delete		Removes the specified group from the team. You must specify the group name with Delete .
Edit		Sets one or both Management Agent LabEnvironment properties for the team. You must specify the /AutoProvision and at least one of the /LabEnvironment options with Edit .

SCVMMHostGroup:vmmHostGroupPath

Required with
Specifies the h
SCVMM host ;

Name:name

Required with
/Edit options. !
the team projec
group to add, d

AutoProvision: { **True**|**False** }

Optional with t
options. Sets (T
(**False**) the Aut
of the host grou
specifies wheth
automatically a
project in the c
host groups are
projects in a co
use the **TfsCor**
command.

Optional with t
options. Specif
Management tr
machines in a l
it deploys new
environments.

LabEnvironmentPlacementPolicy: { **Conservative**|**Aggressive** }

- **Conserva**
Consider
environme
decisions.
virtual ma
of environ
"Stopped"

- **Aggressive** non-running environment decisions.

Optional. Sets the virtual computer group.

- If the **/DN** specified value, sets the suffix of the computer's host computer group.
- If the **/DN** specified the suffix are set to their host group.
- If the **/DN** specified the suffix is not changed.

DNSSuffix:[dnsSuffix]

Optional with the options. Do not confirm.

NoPrompt

Displays the host of the team project.

List

ListSCVmmHostGroups

Displays the hc
from SCVMM

Remarks

Host groups are containers that an administrator creates within SCVMM to group a set of virtual machine hosts for easy management. Host groups are hierarchical; a host group can contain other host groups.

Each host group is identified by its host path, a sequence of host group names that specifies the location of a host or host group within the hierarchy of host groups in SCVMM. All host paths begin with the root host group. For example, the host path All Hosts\New York\Site21\VMHost05 indicates that the host VMHost05 belongs to the host group Site21, which is a child host group of the host group New York.

Use only one of the **/Add**, **/Delete**, or **/Edit** options on a command line. Use separate **TfsConfig Lab /HostGroup** command lines to assign multiple host groups to a team project collection.

You can also use **TfsConfig Lab /HostGroup** commands to set properties that are specific to Lab Management:

- AutoProvision specifies whether the host group is assigned to each team project in the team project collection. By default, AutoProvision is on. To assign a host group in a project collection to an individual team project, use the [TFS Lab Config Create Team Project Host Group Command](#).
 - **True** (Default). The host group is assigned to each team project in the team project collection.
 - **False**. The host group is not assigned to each team project in the team project collection.
- LabEnvironmentPlacementPolicy specifies whether Lab Management considers the existing virtual machines when it deploys new environments on a physical machine in a host group.
 - **Conservative** (Default). Consider non-running virtual environments in deployment decisions. This includes all virtual machines that are part

of environments and that are in "Stopped" state also.

- **Aggressive** Do not consider non-running virtual environments in deployment decisions.
- **DNSSuffix** specifies a DNS suffix to use for the virtual computers created in the host group. The following table describes how the DNS suffixes of virtual computers are affected by the /DNSSuffix setting.

DNSSuffix	/Add	/Edit
DNSSuffix: dnsValue	The DNS suffix is set to dnsValue.	The DNS suffix is set to dnsValue.
DNSSuffix:	The DNS suffix is inherited from host computer.	The existing suffix value is removed and the DNS suffix is inherited from host computer.
<Not Specified>	The DNS suffix is inherited from host computer.	DNS suffix not changed.

Example

In the following example a SCVMM host group is assigned to a team project collection. Because the **/AutoProvision** options is not specified, the host group is automatically assigned to all team projects in the collection.



```
tfssconfig lab /hostgroup /add /scvmmhostgroup:"All Hosts\Lab1\HostG
```

See Also

Reference

[TFSSConfig Lab /Settings Commands](#)

[TFSSConfig Lab /LibraryShare Commands](#)

[TFSSConfig Lab /HostGroup Commands](#)

Concepts

[TFSSConfig Lab Command Reference](#)

[TFSSLabConfig Command Reference](#)

[Configure and administer Lab Management](#)

Other Resources

[TFSSConfig: Manage TFS server configuration](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TfsConfig Lab /LibraryShare Commands

[See Also](#) [Send Feedback](#)

You can use the **TfsConfig Lab /LibraryShare** command to add, remove, or edit the assignment of a library share from System Center Virtual Machine Manager (SCVMM) to a team project collection. You can also use this command set properties that are specific to Visual Studio Lab Management and to display the library shares that are currently assigned to a collection in Lab Management or to display all the library shares in SCVMM.

```
TfsConfig Lab /LibraryShare
  /CollectionName:collectionName
  { /Add
    /SCVVMLibraryShare:librarySharePath
    /Name:name
    [/AutoProvision:{True|False}]
  | /Delete
    /Name:name
    [/NoPrompt]
  | /Edit
    /Name:name
    /AutoProvision:{True|False}
    [/NoPrompt]
  | /List
  | /ListSCVVMLibraryShares }
```

Parameters

	Option	Description
Add		Adds the specified library share to the team project collection. You must specify the /SCVMMLibraryShare and /Name options with Add .
Delete		Removes the specified library share from the team project collection. You must specify the /Name option with Delete .
		Sets or clears the AutoProvision property of the library share. You must specify the /Name and /AutoProvision options with Edit .
Edit		By default, library shares are assigned to the team projects in a collection.

 **Note**

Changing auto-provision does impact existing team projects.

-

CollectionName:collectionName

Required. Specify the name of the team project collection on the application-tier Team Foundation Server.

SCVMMLibraryShare:librarysharePath

Required with **Add**. Specifies the path to the Virtual Machine Manager library share.

Name:libraryShareName

Required with **Add**, **Delete**, and **Edit**. Specifies the name of library share in the team project collection.

AutoProvision

Optional with **Add**; required with **Edit**. Specifies whether the library shares are automatically assigned to each team project in the collection. By default, library shares are assigned to the team projects.

NoPrompt

Optional with **Add** and **Edit**. Do not prompt the user for confirmation.

List

Lists all library shares that are assigned to the specified team project collection.

ListSCVMMLibraryShares

Lists all library shares that are available in Virtual Machine Manager.

Remarks

A library share is a designated share on a Virtual Machine Manager library server. A library share provides access to file-based resources for virtual Lab Management environments that are stored on your library servers, such as ISO images and virtual hard disks. Library shares are created in Virtual Machine Manager. Visual Studio Lab Management uses library shares to provision virtual machines in the lab.

Use only one of the **/Add**, **/Edit**, or **/Delete** options on a TfsConfig Lab **/LibraryShare** command line. Use separate **TfsConfig Lab /LibraryShare** command lines to assign multiple library shares to a collection.

By default, library shares in a team project collection are automatically assigned to each of the team projects in the collection. You can use the **/AutoProvision** option with the **/Add** and **/Edit** commands to change the library shares are assigned.

- Set the **/AutoProvision** option to **False** to disable the automatic assignment of the library share to team projects. To assign or remove a library share in an individual team project, use the **TfsLabConfig** [TFSLabConfig CreateTeamProjectLibraryShare Command](#).
- Set the **/AutoProvision** option to **True** to enable the automatic assignments.

See Also

Reference

[TFSSConfig Lab /Settings Commands](#)

[TFSSConfig Lab /HostGroup Commands](#)

Concepts

TFSSConfig Lab Command Reference

TFSSLabConfig Command Reference

[Configure and administer Lab Management](#)

Other Resources

[TFSSConfig: Manage TFS server configuration](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSSConfig Lab /DNS Commands

[Example](#) [See Also](#) [Send Feedback](#)

The **TfsConfig Lab /DNS** command adds, deletes, or displays DNS records that were created by Visual Studio Lab Management for network-isolated environments.

Team Foundation Server uses the suffix you enter when it registers a unique external name with DNS for each virtual machine in a network-isolated environment. The DNS alias record makes it possible for machines and other objects outside the isolated network to communicate with machines inside the isolated network. Because Team Foundation Server goes into the DNS zone to register the alias record, the service account under which Team Foundation runs must have permissions to add or delete alias records in the specified DNS zone.

If your Team Foundation Server deployment has more than one application tier and each application tier runs under a different service account, then each application-tier service account must have permission to edit the DNS alias records created by the other application tiers.

Note DNS record management is performed automatically by Lab Management. You should use the **/DNS** command only in the following situations:

- You change the account under which Visual Studio Team Foundation Server (TFS) runs.
- You move a team project collection from one instance of Team Foundation Server to another.

In both these cases, the DNS records that were created by using the old TFS service account have to be removed, and then the same DNS records have to be

re-created by using the new TFS service account. If these steps are not performed in the previous scenarios, the new TFS service account will not be able to perform automatic management of those DNS records. As a result, users will not be able connect to virtual environments.

```
TfsConfig Lab /DNS
{/Add | /Delete | /List}
  [/CollectionName:collectionName |
  / CollectionName:collectionName /TeamProject:projectName |
  / CollectionName:collectionName /TeamProject:projectName /LabEnv
  /Name:FQDN /IP:IpAddress]
  [/NoPrompt]
```

Options

Option	Description
Add	Adds the specified DNS records. To use the /Add option, the targeted environments must be running.
Delete	Removes the specified DNS records.
List	Displays the specified DNS records.
LabEnvironment: environmentUri	Targets the /Add , /Delete , or /List options to an individual network-isolated environment that is specified by environmentUri. To use the LabEnvironment option, you must also specify the

/Collection and
/TeamProject
options.

When used without
/LabEnvironment,
targets the **/Add**,
/Delete, or **/List**
options to all of the
network-isolated
environments in
the team project
that is specified by
projectName.

TeamProject:projectName

Otherwise,
/TeamProject
specifies the team
project that
contains the
environment.

To use the
/TeamProject
option, you must
also specify the
/Collection option

When used without
/TeamProject,
targets the **/Add**,
/Delete, or **/List**
options to all of the
network-isolated
environments in
the team project
collection that is
specified by

CollectionName:collectionName

collectionName.
Otherwise,
/Collection
specifies the team
project collection
that contains the
team project.

Name:FQDN

Specifies the fully
qualified domain
name of the
network location
that contains the
environment to
target.

You must specify
the **/Name** and **/IP**
options together.

IP:IPAddress

Specifies the IP
address of the
environment to
target.

You must specify
the **/Name** and **/IP**
options together.

Remarks

Specify only one of the **/Add**, **/Delete**, or **/List** options on a **TfsConfig Lab /DNS** command line. If you do not specify any target options, the operation acts on all virtual machines of all network isolated environments that belong to all project collections in the Team Foundation Server database.

To target the DNS entries of network isolated environments of an object in the Lab Management object hierarchy, use the appropriate combination of **/Collection**, **/TeamProject**, and **/LabEnvironment** options

- The **/LabEnvironment** option targets the command to the specified network-isolated environment. You must use the **/CollectionName** and **/TeamProject** options with the **/LabEnvironment** option to specify the team project collection and the team project that contain the environment.

Use the format

vstfs:///LabManagement/LabEnvironment/environmentID to specify the environment URI. You can view the environment identifier (environmentID) in the Environment Viewer of Lab Management or from the description of the virtual machine in SCVMM Administrator Console.

- The **/TeamProject** option targets the operation to isolated network environments in the specified team project. The **/TeamProject** option must be used with the **/CollectionName** option and the **/CollectionName** option must specify the team project collection that contains the team project.
- The **/CollectionName** option targets the operation to network isolated environments in the specified team project collection.

The second way to target a network isolated environment is to use the **/Name** and **/IP** options to specify the fully qualified external name and external IP address of an individual virtual machine. You must specify both the **/Name** and **/IP** options on the command line. You can view the external name and external IP address of a virtual machine in the Environment Viewer of Lab Management or from the description of the virtual machine in SCVMM Administrator Console.

Example

In the first example, records for all network-isolated environments in a team project are added to DNS. In the second example, an individual DNS record is removed.

 **Copy Code**

```
REM First example
tfsconfig lab /dns /add /collectionname:Collection0 /teamproject:Pro

REM Second example
tfsconfig lab /dns /delete /name:0b668996-2736-46d2-88ac-0733acbd0d9
```

See Also

Concepts

TFSSConfig Lab Command Reference

TFSLabConfig Command Reference

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TfsConfig Lab /Delete Command

[Example](#) [See Also](#) [Send Feedback](#)

Use the **TfsConfig Lab /Delete** option to remove all group hosts, library shares, and environments from a specified team project collection. By default, the associated objects in the System Center Virtual Machine Manager (SCVMM) are not deleted. You can add the **/External** option to the command line to remove the objects from the project collection and from SCVMM.

Note

The **/Delete** option works on all lab assets in a team project collection only when the **/LibraryShare** and **/GroupHost** options are not specified on the command line. For more information, see [TfsConfig Lab /LibraryShare Commands](#) and [TfsConfig Lab /HostGroup Commands](#).

```
TfsConfig Lab /Delete
  /CollectionName:collectionName
  [/External]
  [/NoPrompt]
```

Options

Option	Description
CollectionName:collectionName	Required. Specifies the name of the team project collection on the application tier of Team Foundation Server.
External	Optional. When specified, removes the library shares and host groups in SCVMM in addition to the objects in the team project collection. When /External is not

NoPrompt

specified,
the
**TfsConfig
Lab /Delete**
command
only
removes the
objects in
the team
project
collection.

Optional.
When
specified,
does not
display
progress
and success
information.

Remarks

Use the **/Delete** command to remove the lab assets from a project collection when you want to detach the lab configuration of a project collection. This is useful when moving a project collection from one Team Foundation Server (TFS) instance to another, and where the new Team Foundation Server instance uses a different SCVMM server than the original one. In this case, you will have to delete all the lab assets and reconfigure lab for the project collection.

Example

In the following example, all lab objects in the DefaultCollection team project collection are removed. Because the **/External** option is not specified, the objects are retained in SCVMM.

 **Copy Code**

```
tfssconfig lab /delete /collectionName:DefaultCollection
```

See Also

Concepts

TFSSConfig Lab Command Reference

TFSLabConfig Command Reference

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

License Command

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **License** command to display, modify, or extend the licensing key for your deployment of Visual Studio Team Foundation Server.

To view, modify, or change the licensing for your deployment interactively, you can use the administration console for Team Foundation. For more information, see [Open the Team Foundation Administration Console and Locate or Change the Product Key for Team Foundation Server](#).

Required Permissions

To use the **License** command, you must be a member of the Team Foundation Administrators security group. For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt window, click Start, right-click Command Prompt, and then click Run as Administrator. For more information, see this page on the Microsoft Web site: [User Account Control](#).

```
TFSSConfig License [/ProductKey:Key] [/extend [NewTrialID]]
```

Parameters

Placeholder	Description
Key	Specifies a valid product key to use when changing your deployment from Team Foundation Server from a trial license to a full license.
NewTrialID	Specifies a second trial license key. A second trial license key can be obtained from Microsoft Corporation. The primary trial license key is good for 90 days, but the secondary trial license key will extend your deployment for only 30 days.
Option	Description
/ProductKey:Key	Specifies that the license key for the deployment will be updated with the value of Key.

/extend

Specifies that the trial licensing period for Team Foundation Server will be extended by 30 days. This option can be used only once. If a second extension is required, you must obtain a second trial license from Microsoft.

Example

The following example shows how to view the licensing information for a deployment of Team Foundation Server. In this example, the deployment is using a trial license.

Note

The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, places, or events is intended or should be inferred.

Copy Code

```
TFSSConfig License
```

```
TFSSConfig - Team Foundation Server Configuration Tool
Copyright © Microsoft Corporation. All rights reserved.
Team Foundation Server Standard license
The following features are installed:
Team Foundation Server
Build Services
Build: 21106.00
Product ID: 01234-567-8910
Trial license with 74 days remaining, expiring on 6/30/2010
Trial ID: ABCD-EFGH-IJKL
```

The following example shows how to extend the trial license for a deployment of Team Foundation Server. In this example, the trial from the previous example is extended for 30 days.

Copy Code

```
TFSSConfig License /extend
```

```
TFSSConfig - Team Foundation Server Configuration Tool
Copyright © Microsoft Corporation. All rights reserved.
Now extending trial
```

Team Foundation Server Standard license
The following features are installed:
Team Foundation Server
Build Services
Build: 21106.00
Product ID: 01234-567-7654987
Trial license with 134 days remaining, expiring on 8/30/2010
Trial ID: XYZZY-REDFG-WESDF

The following example shows how to update the trial license to a full license.

Copy Code

```
TFSSConfig License /ProductKey:abcde-fghij-klmno-pqrst-uvwxy
```

```
TFSSConfig - Team Foundation Server Configuration Tool  
Copyright © Microsoft Corporation. All rights reserved.  
Now applying Product Key 'abcde-fghij-klmno-pqrst-uvwxy'  
Team Foundation Server Standard license  
The following features are installed:  
Team Foundation Server  
Build Services  
Build: 21106.00  
Product ID: 01234-567-765456  
Retail license
```

See Also

Tasks

Locate or Change the Product Key for Team Foundation Server

Concepts

[Administer Team Foundation Server](#)

Other Resources

[Upgrading Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

PrepareClone command

[See Also](#) [Send Feedback](#)

The **PrepareClone** command removes information about scheduled backups, SharePoint, and reporting resources from a Team Foundation Server (TFS) deployment. This command is used in two circumstances: when you move a deployment to new hardware but want to keep using the old deployment, or when you clone a TFS deployment. In either case it is critical to run this command. If you don't, the original resources will be used by both the original and the new servers. If both the original and the new servers are live and point to the same SharePoint or reporting resources for any amount of time, you could end up with corrupted databases.

Important

This command should be run before configuration, whether you move or clone TFS. If you run it after configuration, you could end up with inconsistencies between content in your databases and content in your web.config file. These inconsistencies might take your server offline. If you have already configured your moved or cloned TFS deployment and realize you need to run the command, follow these steps. First, quiesce your server. Next, run PrepareClone command, ChangeServerID command, and then RemapDBs command. Finally, unquiesce your server.

Required Permissions

To use the **PrepareClone** command, you must be a member of the Team Foundation Administrators security group and a member of the sysadmin security group for any SQL Server databases that Team Foundation Server uses. For more information, see [Permission reference for Team Foundation Server](#).

☑Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window.



TFSSConfig PrepareClone /SQLInstance:ServerName /DatabaseName:TFSSConi

Argument	Description
ServerName	Specifies the name of the server that you want to map as a server that hosts one or more databases for TFS. If an instance other than the default instance hosts a database, you must also specify the name of the instance. Use this format: ServerName\InstanceName
DatabaseName	Specifies the name of the configuration database for TFS.

Option	What it does
/DatabaseName	Specifies the name of the server that hosts the database that you want to map for TFS, in addition to the name of the configuration

database itself.

/SQLInstance:ServerName

Specifies the name of the server that is running SQL Server, in addition to the name of the instance. Use this option if you want to use an instance other than the default instance.

/NotificationURL:TFSURL

Optional. Specifies The notification URL for the application-tier server.

/usesqlalwayson

Optional. Specifies that the databases are part of an AlwaysOn Availability Group in SQL Server. If configured, this option sets MultiSubnetFailover in the connection string.

For more information, see [AlwaysOn Availability Groups \(SQL Server\)](#).

Remarks

You use the **PrepareClone** command to reconfigure TFS when you move the original installation to new hardware and want to continue to use the original deployment TFS and hardware, or when you want to clone your TFS deployment for testing purposes. You use `TFSSConfig PrepareClone` in conjunction with `TFSSConfig RemapDBs` and `TFSSConfig ChangeServerID` to support the cloning configuration.

The following example shows how to use the command on a moved TFS named `NewFabrikamTFS` to remove old backup, reporting, and SharePoint resource information. If this information isn't removed, the original deployment of TFS still uses those same resources and databases become corrupt. In the example, the SQL Server supporting the moved TFS is also named `NewFabrikamTFS`, and the instance is the default instance, so no specific instance information is required, just the server name.

Copy Code

```
TFSSConfig PrepareClone /SQLInstance:NewFabrikamTFS /DatabaseName:TFS
```

See Also

Reference

[ChangeServerID Command](#)

[RemapDBs Command](#)

Other Resources

[Move or clone Team Foundation Server \(hardware move\)](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Proxy Command (TFSSConfig)

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **TFSSConfig Proxy** command to update or change the settings used by Team Foundation Server Proxy. Team Foundation Server Proxy provides support for distributed teams to use version control by managing a cache of downloaded version control files in the location of the distributed team. By configuring Team Foundation Server Proxy, you can significantly reduce the bandwidth needed across wide area connections. In addition, you do not have to manage downloading and caching of version files; management of the files is transparent to the developer who is using the files. Meanwhile, any metadata exchanges and file uploads continue to appear in Team Foundation Server (TFS). If you use the Visual Studio Online to host your development project in the cloud, you can use the Proxy command to not only manage the cache for projects in the hosted collection, but also to manage some of the settings used by that service.

For more information about installing Team Foundation Server Proxy and initial configuration of the proxy, see [How to: Install Team Foundation Proxy and set up a remote site](#). For more information about configuring proxy on client computers, see [Team Foundation Version Control Command Reference](#).

Requirements

To use the **Proxy** command, you must be a member of the Team Foundation Administrators security group and an administrator on the proxy server. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSConfig Proxy /add|delete|change [/Collection:TeamProjectCollectio
```

Options

Option	Description
/add	Adds the specified server or collection to the proxy list in the Proxy.config file. You can run /add multiple times to include more collections or servers. When using /add with a collection hosted on Visual Studio Online, you will be prompted for your credentials on Visual Studio Online.
/change	Changes the credentials stored as the service account for Visual Studio Online. The /change option is only used for Visual Studio Online; it should not be used for local deployments of TFS.
/delete	Removes the specified server or collection from the proxy list in the Proxy.config file.
	Specifies the account used as the service account for the proxy in Visual Studio

Online. This option is only used for Visual Studio
/account Online in conjunction with the /change option.

The default service account used for Visual Studio Online is "Account Service."

Continues the execution of the command even if the verification process produces warnings.
/continue

Parameters

Argument	Description
TeamProjectCollectionURL	Used with /Collection. Specifies the URL of the team project collection that is hosted on Visual Studio Online, in AccountName.DomainName/CollectionName format.
AccountName	Used with /Collection. Specifies the name of the account that is used as the service account for Visual Studio Online. If the account name has spaces, the name must be enclosed within quotation marks (""). All special characters in account names must be specified in accordance with command-line syntax.
ServerURL	Used with /Server. Specifies the URL of a TFS deployment, in ServerURL:Port/tfs format.

Remarks

You use the **Proxy** command to update the existing configuration of Team Foundation Server Proxy. You cannot use the Proxy command for initial installation and configuration of the proxy.

Example

The following example shows how to add a TFS deployment named FABRIKAM to the proxy list.

Note

Examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
TFSSConfig Proxy /add /Server:http://www.fabrikam.com:8080/tfs
```

The following example shows how to add a team project collection hosted on Visual Studio Online to the proxy list. The collection is named PhoneSaver and the account name used for Visual Studio Online is HelenaPetersen.fabrikam.com. Because the /account option is not specified, the default service account will be used.

Copy Code

```
TFSSConfig Proxy /add /Collection:https://HelenaPetersen.tfs.visualst
```

The following example shows how to change the service account used by the proxy for the team project collection hosted on Visual Studio Online. The collection is named PhoneSaver, the account name used for Visual Studio Online is HelenaPetersen.fabrikam.com, and the service account used by the proxy is being changed to "My Proxy Service Account". Because the account name contains spaces, quotation marks are used to enclose the name.

Copy

```
TFSSConfig Proxy /change /Collection:https://HelenaPetersen.tfs.visualst
```

See Also

Concepts

How to: Install Team Foundation Proxy and set up a remote site

Other Resources

[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

RebuildWarehouse Command

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **RebuildWarehouse** command to rebuild the SQL Server Reporting Services and SQL Server Analysis Services databases that Visual Studio Team Foundation Server (TFS) uses.

To start the rebuild of these databases interactively, you can use the Reporting node in the administration console for Team Foundation. For more information, see [Open the Team Foundation Administration Console](#).

Required Permissions

To use the **RebuildWarehouse** command, you must be a member of the following groups:

- the Team Foundation Administrators security group and the Administrators security group on the server or servers that are running the administration console for Team Foundation
- the sysadmin group on the server or servers that are running the instance of SQL Server that hosts the databases for TFS

For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window to perform this function.

```
TFSSConfig RebuildWarehouse /analysisServices | /all [/ReportingDataS
```

Parameters

Placeholder	Description
Password	Specifies the password that the data sources account for SQL Server Reporting Services uses. This account is referred to in the documentation as TFSReports.

Option	Description
/analysisServices	Required if /all is not used. Specifies that only the database for Analysis Services will be rebuilt. If no database exists for Analysis Services, you must also use the /reportingDataSourcePassword option.
/all	Required if /analysisServices is not used. Specifies that all reporting and analysis databases that TFS uses will be rebuilt.
	Required if the TFS_Analysis database does not exist. Specifies the password of the account that

/reportingDataSourcePassword:Password

is used as the data sources account for SQL Server Reporting Services (TFSReports). For more information, see [Service accounts and dependencies in Team Foundation Server](#).

Remarks

You might use this command when moving or splitting a team project collection, restoring data, or otherwise changing the configuration of your deployment.

Example

The following example shows how to rebuild the Analysis Services database for a deployment of TFS.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
TFSSConfig RebuildWarehouse /analysisServices
```

```
TFSSConfig - Team Foundation Server Configuration Tool  
Copyright © Microsoft Corporation. All rights reserved.  
The Analysis Services database was successfully rebuilt.
```

See Also

Concepts

[Team Foundation Server architecture](#)

Understanding SQL Server and SQL Server Reporting Services

[Add reports to a team project](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

RegisterDB Command

[Example](#) [Send Feedback](#)

Use **RegisterDB** to update name of the server that hosts the configuration database in Visual Studio Team Foundation Server (TFS). You might use this command when restoring the configuration database to new hardware or when changing the domain of a deployment.

Important

For the RegisterDB command to succeed, the following application pools and programs must be running:

- Team Foundation Server Application Pool (application pool)
- ReportServer (application pool)
- SQL Server Reporting Services (program)

In addition, the following components must not be open or running:

Required Permissions

To use the **RegisterDB** command, you must be a member of the Team Foundation Administrators group on the application-tier server for Team Foundation and a member of the sysadmin group in SQL Server on the data-tier server for Team Foundation. For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you are logged on with administrative credentials, you must open an elevated Command Prompt to perform this function.

```
TFSSConfig RegisterDB /SQLInstance:ServerName /databaseName: Database
```

Parameters

Placeholder	Description
ServerName	<p>Specifies the name of the server that hosts the configuration database for TFS and the name of the instance if you want to use an instance other than the default instance. If you specify an instance, you must use the following format:</p> <p>ServerName\InstanceName</p>
DatabaseName	<p>Specifies the name of the configuration database that you want to register with TFS.</p>
Argument	Description
/SQLInstance:ServerName	<p>Required. Specifies the name of the server that is running SQL Server and the name of the instance if you want to use an instance other than the default instance. If you specify an instance, you</p>

must use the following
format:

ServerName\InstanceName

/databaseName:DatabaseName

Required. Specifies the name of the configuration database. By default, this is Tfs_Configuration.

/usesqlalwayson

Optional. Specifies that the databases are part of an AlwaysOn Availability Group in SQL Server. If configured, this option sets MultiSubnetFailover in the connection string.

For more information, see [AlwaysOn Availability Groups \(SQL Server\)](#).

Remarks

Important

Back up the databases for TFS before you use this command.

You must provide the exact name or address of the configuration database for this command to operate correctly. If you must change the server on which this database is stored, you must ensure that TFS points to the new location.

Example

The following example redirects TFS to a configuration database that is located on the server ContosoMain in the SQL Server instance TeamDatabases.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
TFSSConfig RegisterDB /SQLInstance:ContosoMain\TeamDatabases /databas
```



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

RemapDBs Command

[Example](#) [See Also](#) [Send Feedback](#)

The **RemapDBs** command redirects Visual Studio Team Foundation Server (TFS) to its databases when they are stored on more than one server and you are restoring, moving, or otherwise changing the configuration of your deployment. For example, you must redirect TFS to any databases for team project collections if they are hosted on a separate server or servers from the configuration database. You must also redirect TFS to the server or servers that are running SQL Server Analysis Services or SQL Server Reporting Services if those databases are hosted on a separate server or instance from the configuration database.

Required Permissions

To use the **RemapDBs** command, you must be a member of the Team Foundation Administrators security group and a member of the sysadmin security group for any SQL Server databases that Team Foundation Server uses. For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative credentials, you must open an elevated Command Prompt window to perform this function.

```
TFSSConfig RemapDBs /DatabaseName:ServerName;DatabaseName /SQLInstanc
```

Parameters

Argument	Description
ServerName	Specifies the name of the server that you want to map as a server that hosts one or more databases for TFS. If an instance other than the default instance hosts a database, you must also specify the name of the instance in the following format: ServerName\InstanceName
DatabaseName	Specifies the name of the database that you are mapping to TFS.
Option	Description
/DatabaseName	Specifies the name of the server that hosts the database that you want to map for TFS, in addition to the name of the database itself.

Specifies the name of the server that is running SQL Server, in addition to the name of the instance if you want to use an instance other than the default instance.

/SQLInstances:ServerName1,ServerName2

If you are specifying more than one server, you must use a comma to separate multiple pairs of server and instance names.

Optional. Specifies the name of the server and instance that hosts SQL Server Analysis Services. Use this option to specify the server and instance that hosts the Analysis Services database.

/AnalysisInstance:ServerName

Optional. Specifies the name of the Analysis Services database that you want to use with TFS if you have

/AnalysisDatabaseName:DatabaseName

more than one such database on the server that you specified with the **/AnalysisInstance** option.

/preview

Optional. Displays the actions that you must take to update the configuration.

/continue

Optional. Specifies that the **RemapDB** command should continue even if an error occurs during the attempt to locate one or more databases. If you use the **/continue** option, any collections whose databases are not found on the server or servers that you specify will be reconfigured to use the server and instance that hosts the configuration database.

Optional. Specifies that the databases are part of an

/usesqlalwayson

AlwaysOn
Availability Group
in SQL Server. If
configured, this
option sets
MultiSubnetFailover
in the connection
string.

For more
information, see
[AlwaysOn
Availability Groups
\(SQL Server\)](#).

Remarks

You use the **RemapDBs** command to reconfigure TFS to use different servers and instances of SQL Server from the servers and instances in the original installation.

Example

The following example shows how to redirect TFS to its configuration database TFS_Configuration. This database is hosted on ContosoMain on the named instance TeamDatabases. Its project collection databases are stored on both ContosoMain\TeamDatabases and the default instance on Contoso2.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

 **Copy**

```
TFSSConfig RemapDBs /DatabaseName:ContosoMain\TeamDatabases;TFS_Conf:
```

See Also

Other Resources

[TFSSConfig: Manage TFS server configuration](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Settings Command

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **Settings** command to automate changes to the uniform resource locator (URL) that is used by the notification interface and for the server address for Visual Studio Team Foundation Server (TFS). By default, the notification interface URL and the server URL match in TFS, but you can configure separate URLs. For example, you might want to use a different URL for the automatic e-mails that TFS generates. You must run this tool from the application tier to update all servers so that they use the new URLs.

To change these URLs interactively or to just view the current settings, you can use the administration console for Team Foundation. See [Configure and manage TFS resources](#).

Requirements

- You must be a member of the Team Foundation Administrators security group and the Administrators group on the application-tier server. For more information, see [Set administrator permissions for Team Foundation Server](#).
- Even if you log on with administrative credentials, you must open an elevated Command Prompt window.

```
TFSConfig Settings [/ServerURL:URL] [/NotificationURL:URL]
```

Parameters

Placeholder	Description
-------------	-------------

URL	Use to set the uniform resource locator (URL) with the appropriate option.
-----	--

Option	Description
--------	-------------

/ServerURL:URL	Specifies the URL of the application-tier server for Team Foundation. This value is stored in the configuration database for Team Foundation.
-----------------------	---

/NotificationURL:URL	Specifies the URL to use in the text of e-mail alerts, if that URL differs from the URL of the application-tier server for
-----------------------------	--

Team
Foundation. This
value is stored in
the
configuration
database.

Remarks

The **Settings** command changes connection information for server-to-server communication in a deployment of TFS. The URL that is specified in **/ServerURL** must be available to all servers within the deployment.

Example

The following example changes the value of NotificationURL to `http://contoso.example.com/tfs` and the value of ServerURL to `http://contoso.example.com:8080/tfs`.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

 **Copy Code**

```
TFSSConfig Settings /NotificationURL:http://contoso.example.com/tfs ,
```

See Also

Other Resources

[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TCM Command

[Example](#) [Send Feedback](#)

When upgrading to the latest version of TFS, the system automatically attempts to upgrade the Tests Management components, including test plans and suites. If the automatic migration fails, use the **TCM** command to upgrade your test plans and test suites manually to their respective work item types (WITs).

Before you can use the TCM command, you must first import the test plan work item definition and the test plan category into the project. To learn more about the migration and when to use this command, see [Manual updates to support test management](#).

The TCM command is applied to individual team projects. If you need to upgrade test plans in more than one team project, you will have to run it against each team project individually.

You must run the **TCM** command from the tools directory for TFS. By default, that location is:

 **Copy Code**

```
drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools
```

Requirements

- Your application-tier server must be upgraded to the latest version of TFS.
- You must be a member of the Team Foundation Administrators security group, and an administrator on the application-tier server. See [Set administrator permissions for Team Foundation Server](#).

TFSSConfig TCM /upgradeTestPlans|upgradeStatus /CollectionName:TeamPr

Parameters

Argument	Description
	Required unless /upgradeStatus is used.
/upgradeTestPlans	Converts existing test plan and test suites to point to the work item-based test plans and test suites. It also updates existing test management data and links between other existing test artifacts such as test points, test runs, and test results.
	Required unless /upgradeTestPlans is used.
/upgradeStatus	Reports the migration status of test data for the specified team project. It will also indicate whether

the specified project has any test plans.

Required. Specifies the project collection that contains the team project for which you want to migrate test data or check the migration status.

CollectionName

If there are spaces in the name of the team project collection, enclose the name in quotation marks, for example, "Fabrikam Fiber Collection".

Required. Specifies the team project for which you want to migrate test data or check the migration status. This team project must be defined in the collection that you specified by using the

TeamProjectName

/collectionName parameter.

If there are spaces
in the name of the
team project,
enclose the name in
quotation marks,
for example, "My
Team Project".

Remarks

You use the **TCM** command only in the event that automatic migration of existing test data fails. To learn more about the migration and when to use this command, see Manual updates to support test management.

If you can't access test plans or test suites that were defined before the server upgrade occurred, run **TFSSConfig TCM upgradeStatus** to determine the status of the migration.

You run the **TCM** command for an individual team project. If you need to upgrade more than one team project, you will have to run it against each team project in turn.

Example

The following example shows how to check the status of test plan upgrade on the Fabrikam Fiber project hosted on the default team project collection (DefaultCollection):

 **Copy Code**

```
tfsconfig tcm /upgradeStatus /CollectionName:DefaultCollection /Team
```

A sample return of the command for a project that has been successfully upgraded:

 **Copy Code**

```
Logging sent to file C:\ProgramData\Microsoft\Team Foundation\Server
Microsoft (R) TfsConfig - Team Foundation Server Configuration Tool
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
Command: tcm
Microsoft (R) TfsConfig - Team Foundation Server Configuration Tool
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
Test plan migration job has successfully completed.
```

The following example shows how to upgrade the test plans for the Fabrikam Fiber project, hosted on the default team project collection (DefaultCollection):

 **Copy Code**

```
tfsconfig tcm /upgradeTestPlans /CollectionName:DefaultCollection /1
```

A sample return of the command, including the two questions listing the tasks you must complete before you can run the command:

 **Copy Code**

Logging sent to file C:\ProgramData\Microsoft\Team Foundation\Server
Microsoft (R) TfsConfig - Team Foundation Server Configuration Tool

Command: tcm
Microsoft (R) TfsConfig - Team Foundation Server Configuration Tool
Copyright (c) Microsoft Corporation. All rights reserved.

In order to upgrade the test plans, you have to complete the followi
1. Import the test plan work item definition into the project. Have
2. Import the test plan category into the project. Have you complete
Test plan migration job successfully scheduled.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Unattend Command

[Example](#) [See Also](#) [Send Feedback](#)

The **Unattend** command is designed for users who are familiar with Team Foundation Server (TFS) and the configuration process, and who need to install TFS on different machines.

For example, if you use Team Foundation Build, you can use the **Unattend** command to install multiple build servers using the same configuration file.

You use the Unattend /create option to create an unattend file. This file defines all configuration parameters for a TFS installation. Next, you use the Unattend /configure option to actually perform the configuration.

This process allows you to configure TFS from start to finish without having to provide input during the installation process. In the case of multiple installations, this also helps ensure that the exact same configuration parameters are used across multiple servers.

Requirements

- You must be a member of the Administrators group on the computer where you are installing the software.
- Depending on the type of installation, you might also require additional administrator permissions. For example, if you are using the **Unattend** command to install Team Foundation Server, you must be a member of the sysadmin group on the instance of SQL Server that will support TFS. For more information, see the Q & A section of [Set administrator permissions for Team Foundation Server](#).
- Even if you log on with administrative credentials, you must open an elevated Command Prompt window.

- Before you can use the **Unattend** command to configure TFS, you must [create the service accounts](#) you will use as part of your deployment. You must also install any prerequisite software for your chosen installation type. This includes TFS itself. You must install TFS but you don't have to configure it because the **Unattend** command will do that for you.

```
TFSSConfig unattend /create /type:InstallType /unattendfile:Configura
```

```
TFSSConfig unattend /configure /type:InstallType /unattendfile:Config
```

Parameters

Parameter	Description
InstallType	<p data-bbox="570 562 883 642">Specifies the type of configuration to use.</p> <ul data-bbox="607 688 1015 1879" style="list-style-type: none"><li data-bbox="607 688 1015 905">• Basic: Configures TFS in the basic configuration, including SQL Server Express.<li data-bbox="607 947 1015 1115">• Standard: Configures TFS in the standard single-server configuration.<li data-bbox="607 1157 1015 1325">• ATOnly: Configures an additional application-tier for an existing TFS deployment.<li data-bbox="607 1367 1015 1493">• Build: Configures the Team Foundation Build Service.<li data-bbox="607 1535 1015 1661">• Proxy: Configures Team Foundation Server Proxy.<li data-bbox="607 1703 1015 1879">• SPInstall: Installs and configures SharePoint Foundation 2013 for use with a TFS

deployment.

- Upgrade: Upgrades a previous version of Team Foundation Server to the latest version of the software.

You must have downloaded and installed that version locally before running this command with /configure.

- SPExtensions: Configures the SharePoint Extensions for Team Foundation Server.

ConfigurationFileName Specifies the name of the unattend file to create or use in configuration, for example, "ConfigureMyTFS.ini".

Key1=Value1;
Key2=Value2; Specifies the parameters and values for configuration variables, such as the account to use as the service account. For example, the key and value pair to specify the Network Service account as the service account for TFS is "ServiceAccountName="NT

Authority\Network
Service"".

Option	Description
/create	Required if /configure is not used. Creates the unattend file with the name and parameters you specify.
/configure	Required if /create is not used. Configures TFS using the unattend file and parameters you specify.
/type	Specifies the configuration type. When you use /configure, either /type or /unattendfile are required, but you cannot use both.
/unattendfile	Specifies the unattend file to create or use, depending on whether the initial parameter is /create or /configure. When you use /configure, either /unattendfile or /type is required.

/inputs

Optional. If you use /create, specifies settings and values to use when creating the unattend file. If you use /configure specifies additional settings and values to use in conjunction with the unattend file.

As an alternative to using /inputs, you can manually edit the unattend file in any plain-text editor. This is necessary for certain input types, such as ServiceAccountPassword, as the password cannot be set using the /inputs parameter.

/verify

Optional. Specifies a configuration run that only completes verification checks based on the unattend file, inputs, and configuration type. This is an alternative to performing a complete configuration.

/continue

Optional. Specifies that /create or /configure should continue regardless of warnings from verification checks.

/?

Optional. Provides
command-line help for
the unattend command.

Remarks

The Unattend command configures TFS components. It does not perform the initial installation of the software. It configures the software according to your specifications after the bits are present on the computer.

Example

The following example shows how to create an unattend file for a basic installation of Team Foundation Server.

```
TFSConfig Unattend /create /type:basic /unattendfile:configTFSSBasic.ini
```

In this example, the unattend file is created in the same directory as the command. A log file is created as part of the command, and the location of the file is returned in the command as part of executing the command.

The following example shows how to create an unattend file for the configuration of Team Foundation Build on a server using "FabrikamFiber\BuildSVC" as the build service account, and then configure Team Foundation Build using that unattend file.

Important

In this example, after creating the unattend file, the administrator manually edits the file to specify the password for the build service account. Adding the password as an input using "ServiceAccountPassword=Password" doesn't add the password information to the file.

```
TFSConfig Unattend /create /type:build /unattendfile:configTFSSBuild.ini  
/inputs:IsServiceAccountBuiltIn=false;ServiceAccountName=FabrikamFiber\Bui  
Unattend /configure /unattendfile:configTFSSBuild.ini
```

The first command returns the following:

 Copy

```
Microsoft (R) TfsConfig - Team Foundation Server Configuration Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
Command: unattend  
Logging sent to file C:\ProgramData\Microsoft\Team Foundation\Server
```

The second command returns the following information, including the name of the server where Team Foundation Build was configured (FabrikamFiberTFS) and the team project collection associated with the controller (DefaultCollection):

 **Copy**

```
Microsoft (R) TfsConfig - Team Foundation Server Configuration Tool
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
Command: unattend
```

```
-----
Inputs:
-----
```

```
Feedback
```

```
    Send Feedback: True
```

```
Build Resources
```

```
    Configuration Type: create
```

```
    Agent Count: 1
```

```
    New Controller Name: FabrikamFiberTFS - Controller
```

```
    Clean Up Resources: False
```

```
Team Project Collection
```

```
    Collection URL: http://FabrikamFiberTFS:8080/tfs/defaultcoll
```

```
Windows Service
```

```
    Service Account: FabrikamFiber\BuildSVC
```

```
    Service Password: *****
```

```
Advanced Settings *
```

```
    Port: 9191
```

```
-----
Running Readiness Checks
-----
```

```
[1/2] System Verifications
```

```
[2/2] Build Service Verifications
```

```
-----
Configuring
-----
```

```
root
```

```
[1/4] Install Team Foundation Build Service
      Installing Windows services ...
      Adding service account to groups ...
      Setting ACL on a windows service
[2/4] Enable Event Logging
      Adding event log sources ...
      Token replace a config file
      RegisterBuildEtwProvider
      Configuring ETW event sources ...
[3/4] Register with Team Foundation Server
      Registering the build service
[4/4] Start Team Foundation Build Service
      StartBuildHost
      Starting Windows services ...
      Marking feature configured status
[Info] [Register with Team Foundation Server] Firewall exception add
9191
```

TeamBuild completed successfully.
Logging sent to file C:\ProgramData\Microsoft\Team Foundation\Server

See Also

Other Resources

[MSDN Blog Article: Unattended installation of TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Updates Command

[See Also](#) [Send Feedback](#)

The Updates command has been deprecated. Do not use it.

If you need assistance with installing any software updates that are missing from the databases for Team Foundation Server (TFS), contact [Microsoft Support](#).

See Also

Other Resources

[Upgrading Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Configure Lab Management with TFSLabConfig

[See Also](#) [Send Feedback](#)

Team Foundation Server includes a command-line tool to help you configure and manage the lab service provide by Visual Studio Lab Management.

The **TFSLabConfig** command-line tool is located in Drive:\Program Files\Microsoft Team Foundation Server 2010\Tools on Team Foundation Server application tier. It is also located in Drive:\Program Files\Microsoft Visual Studio 10.0\Common7\IDE on the client machine where Microsoft Test Manager is installed.

Required Permissions

To use TFSLabConfig, you must have the appropriate permissions for the operation that you want to perform. The required permissions are described for each command in the command reference topic.

Commands

This tool provides commands to perform the following tasks.

Task	Command
Configure the SCVMM server that will be used by Lab Management	<ul style="list-style-type: none">• TFSLabConfig Settings Command
Manage host groups in a team project. Add or remove a host group in an individual team project.	<ul style="list-style-type: none">• TFSLabConfig CreateTeamProjectHostGroup Command• TFSLabConfig DeleteTeamProjectHostGroup Command• TFSLabConfig ListTeamProjectHostGroups Command• TFSLabConfig ListTeamProjectCollectionHostGroups Command• TFSLabConfig TPCHostGroup Command• TFSLabConfig TPHostGroup Command

Manage library shares in a team project. Add or remove a library share in an individual team project in the collection.

- [TFSLabConfig CreateTeamProjectLibraryShare Command](#)
- [TFSLabConfig DeleteTeamProjectLibraryShare Command](#)
- [TFSLabConfig ListTeamProjectLibraryShares Command](#)
- [TFSLabConfig ListTeamProjectCollectionLibraryShares Command](#)
- [TFSLabConfig TPCLibraryShare Command](#)
 1. [TFSLabConfig TPLibraryShare Command](#)

Manage permissions for users on specified Lab Management objects. Add or remove permissions for users and user groups to an object in a team project or team project collection.

- [TFSLabConfig Permissions Command](#)

See Also

Concepts

[Configure TFS using the administration console \[redirected\]](#)

[Configure and administer Lab Management](#)

TFSSConfig Lab Commands

Other Resources

[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig CreateTeamProjectHostGroup Command

[Example](#) [See Also](#) [Send Feedback](#)

Use the **CreateTeamProjectHostGroup** command to assign a host group from a team project collection to an individual team project in the collection. Host groups specify one or more physical machines that are the deployment targets for virtual environments in Visual Studio Lab Management. Host groups are created in System Center Virtual Machine Manager (SCVMM) and assigned to a project collection by Visual Studio Lab Management. Use separate **CreateTeamProjectHostGroup** commands to assign multiple host groups to a team project.

Note

You can automatically assign a host group to all the projects in a team project collection when you assign the host group to the team project collection. See [TFSSConfig Lab /HostGroup Commands](#) and [How to: Change the Host Groups for Your Team Project Collections](#).

Required Permissions

To use the **CreateTeamProjectHostGroup** command, you must have the **Manage Lab Locations** permission at the Team Project Collection Host Group level. By default, the members of the Team Foundation Server Administrators and Project Collection Administrators groups have this permission. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSLabConfig CreateTeamProjectHostGroup
    /Collection:collectionUrl
    /TeamProject:{* |teamProjectName}
    /TeamProjectCollectionHostGroup:(* |teamProjectCollectionHostC
```

```
/Name:teamProjectHostGroupName  
[/Description:teamProjectHostGroupDescription]  
[/NoPrompt]
```

Parameters

Option	Description
Collection: collectionUrl	Required. The URL of the team on the application tier of Team Foundation Server that contains the team project. For example, /collection:http://abc:8080/TFS/
TeamProject: {* teamProjectName>}	Required. The name of the team project. Use quotation marks if there are spaces in the name. Use an asterisk (*) to assign the group to all team projects in the collection.
TeamProjectCollectionHostGroup: {* teamProjectCollectionHostGroupName}	Required. The name of the host group in the project collection. Use quotation marks if there are spaces in the name. Use an asterisk (*) to assign all host groups in the collection to the specified team project. Use quotation marks if there are spaces in the name.
Name: teamProjectHostGroupName	Required. The name to assign to the team project.
Description: teamProjectHostGroupDescription	Optional. A description of the team project group.
NoPrompt	Optional. Do not prompt the user for confirmation.

Example

For increased readability in the example, command options are listed on separate lines. In a command prompt window, type all options for a command on the same line.

In the first example, all the host groups in the team project collection are assigned to each team project in the collection. In the second example, one host group in the team project collection is assigned to a specific team project.

Copy Code

```
REM First example
TFSLabConfig CreateTeamProjectHostGroup
    /collection:http://abc:8080/TFS/Collection0
    /teamProject:*
    /teamProjectCollectionHostGroup:*
REM Second example
TFSLabConfig CreateTeamProjectHostGroup
    /collection:http://abc:8080/TFS/Collection0
    /teamProject:Project1
    /teamProjectCollectionHostGroup:tpchg1
    /name:hg1
```

See Also

Reference

[TFSLabConfig ListTeamProjectCollectionHostGroups Command](#)

[TFSLabConfig ListTeamProjectHostGroups Command](#)

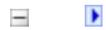
[TFSLabConfig DeleteTeamProjectHostGroup Command](#)

Concepts

[TFSLabConfig Command Reference](#)

[TFSSConfig Lab Command Reference](#)

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig CreateTeamProjectLibraryShare Command

[Example](#) [See Also](#) [Send Feedback](#)

Use the **CreateTeamProjectLibraryShare** command to assign a library share from a team project collection to an individual team project in the collection. A library share provides access to file-based resources for virtual environments such as ISO images and virtual hard disks. Library shares are created in System Center Virtual Machine Manager (SCVMM) and assigned to project collection by Visual Studio Lab Management. Use separate **CreateTeamProjectLibraryShare** commands to assign multiple library shares to a team project.

Note

You can automatically assign a library share to all projects in a team project collection by using the [TFSSConfig Lab /LibraryShare Commands](#) and [How to: Change the Library Share for Your Team Project Collections](#).

Required Permissions

To use the **CreateTeamProjectLibraryShare** command, you must have **Manage Lab Locations** permission at the Team Project Collection Library Share level. By default, members of the Team Foundation Server Administrators and Project Collection Administrators groups have this permission. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSLabConfig CreateTeamProjectLibraryShare
Collection:collectionUrl
  /TeamProject:{* |teamProjectName}
  /TeamProjectCollectionLibraryShare:{* |teamProjectCollectionLi
  /Name:teamProjectLibraryShareName
```

[/Description:teamProjectLibraryShareDescription]
[/NoPrompt]

Parameters

Option	Description
Collection: collectionUrl	Required. The URL of the team project on the application tier of Team Foundation Server that contains the team project. example, /collection:http://abc:8080/TFS/Default
TeamProject: {* teamProjectName}	Required. The name of the team project. Use quotation marks if there are spaces in the name. Use an asterisk (*) to assign all libraries in the collection to the specified team project.
TeamProjectCollectionLibraryShare: {* teamProjectCollectionLibraryShareName}	Required. The name of the library share in the team project collection. Use an asterisk (*) to create a team project library share for all library shares in the team project collection.
Name: teamProjectLibraryShareName	Required. The name to assign to the library share in the team project.
Description: teamProjectLibraryShareDescription	Optional. A description of the team project library share.
NoPrompt	Optional. Do not prompt the user for confirmation.

Example

For increased readability in the example, command options are listed on separate lines. In a command prompt window, type all options for a command on the same line.

In the first example, all the library shares in the team project collection are assigned to each of the team projects in the collection. In the second example, one library share in the team project collection is assigned to a specific team project.

Copy Code

```
REM First example
TFSLabConfig CreateTeamProjectLibraryShare
    /collection:http://abc:8080/TFS/Collection0
    /TeamProject:*
    /TeamProjectCollectionLibraryShare:*

REM Second example
TFSLabConfig CreateTeamProjectLibraryShare

    /collection:http://abc:8080/TFS/Collection0
    /TeamProject:Project1
    /TeamProjectCollectionLibraryShare:tpcls1
    /name:ls1
```

See Also

Reference

[TFSLabConfig ListTeamProjectCollectionLibraryShares Command](#)

[TFSLabConfig ListTeamProjectLibraryShares Command](#)

[TFSLabConfig DeleteTeamProjectLibraryShare Command](#)

Concepts

TFSSConfig Lab Command Reference

[Configure TFS using the administration console \[redirected\]](#)

TFSLabConfig Command Reference

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig DeleteTeamProjectHostGroup Command

[Example](#) [See Also](#) [Send Feedback](#)

Use the **DeleteTeamProjectHostGroup** command to remove the assignment of a host group from an individual team project. Host groups specify one or more physical machines that are the deployment targets for virtual environments in Visual Studio Lab Management.

Host groups are created in System Center Virtual Machine Manager (SCVMM). In Lab Management, host groups are assigned to one or more team project collections and then to one or more team projects in the collections. The **DeleteTeamProjectHostGroup** command does not remove the assignment of the host group to the team project collection.

Required Permissions

To use the **DeleteTeamProjectHostGroup** command, you must have Delete Lab Locations permission for the Team Project host group. By default, Team Foundation Server Administrators, Team Project Collection Administrators and Team Project Administrators have this permission. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSLabConfig DeleteTeamProjectHostGroup
  /Collection:collectionUrl
  /TeamProject:{* |teamProjectName>}
  /Name:{* |teamProjectHostGroupName}
  [/NoPrompt]
```

Parameters

Option	Description
Collection: collectionUrl	Required. The URL of the team project collection on the application tier of Team Foundation Server that contains the team project. For example, /collection:http://abc:8080/TFS/DefaultCollection
TeamProject: {* teamProjectName}	Required. The name of the team project from which you want to remove the host group. Use quotation marks if there are spaces in the name. Use an asterisk (*) to specify all team projects in the team project collection.
Name: {* teamProjectHostGroupName}	Required. The name of the host group to delete from a team project. Use quotation marks if there are spaces in the name. Use an asterisk (*) to specify all host groups in the team project.
NoPrompt	Optional. Do not prompt the user for confirmation.

Example

For increased readability in the example, command options are listed on separate lines. In a command prompt window, type all options for a command on the same line.

In the first example, all the host groups assigned to each of the team project in the team project collection are removed. In the second example, one host group is removed from a specific team project.

Copy Code

```
REM First example
TFSLabConfig DeleteTeamProjectLibraryShare
    /collection:http://abc:8080/TFS/DefaultCollection
    /teamProject:*
    /name:*
```

```
REM Second example
TFSLabConfig DeleteTeamProjectLibraryShare
    /collection:http://abc:8080/TFS/DefaultCollection
    /teamProject:Project1
    /name:hg1
```

See Also

Reference

[TFSLabConfig ListTeamProjectCollectionHostGroups Command](#)

[TFSLabConfig ListTeamProjectHostGroups Command](#)

[TFSLabConfig CreateTeamProjectHostGroup Command](#)

Concepts

[TFSLabConfig Command Reference](#)

[TFSSConfig Lab Command Reference](#)

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig DeleteTeamProjectLibraryShare Command

[Example](#) [See Also](#) [Send Feedback](#)

Use the **DeleteTeamProjectLibraryShare** command to remove the assignment of a library share from an individual team project. A library share provides access to file-based resources for virtual environments such as ISO images and virtual hard disks., Library shares are created by System Center Virtual Machine Manager (SCVMM). In Visual Studio Lab Management, library shares are assigned to one or more team project collections and then to one or more team projects in the collections. The **DeleteTeamProjectLibraryShare** command does not remove the assignment of the library share to the team project collection.

Required Permissions

To use the **DeleteTeamProjectLibraryShare** command, you must have **Delete Lab Locations** permission for that Team Project Library Share. By default, members of the Team Foundation Server Administrators, Project Collection Administrators and Team Project Administrators groups have this permission. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSLabConfig DeleteTeamProjectLibraryShare
Collection:collectionUrl
    /TeamProject:{* |teamProjectName}
    /Name:{* |teamProjectLibraryShareName}
    [/NoPrompt]
```

Parameters

Argument	Description
Collection: collectionUrl	Required. The URL of the team project collection that contains the team project. For example, /collection:http://abc:8080/TFS/DefaultCollecti
TeamProject: {* teamProjectName}	Required. The name of the team project that contains the library share that you want to delete. Use quotation marks if there are spaces in the name. Use an asterisk (*) to specify all team projects in the team project collection.
Name: (* teamProjectLibraryShareName)	Required. The name of the library share to delete from a team project. Use quotation marks if there are spaces in the name. Use an asterisk (*) to specify all library shares of the team project n.
NoPrompt	Optional. Do not prompt the user for confirmation.

Example

For increased readability in the example, command options are listed on separate lines. In a command prompt window, type all options for a command on the same line.

In the first example, all the library shares assigned to each of the team project in the team project collection are removed. In the second example, one library share is removed from a specific team project.

Copy Code

```
REM First example
TFSLabConfig DeleteTeamProjectLibraryShare
  /collection:http://abc:8080/TFS/DefaultCollection
  /teamProject:*
  /name:*
```

```
REM Second example
TFSLabConfig DeleteTeamProjectLibraryShare
  /collection:http://abc:8080/TFS/DefaultCollection
  /teamProject:Project1
  /name:ls1
```

See Also

Reference

[TFSLabConfig ListTeamProjectCollectionLibraryShares Command](#)

[TFSLabConfig ListTeamProjectLibraryShares Command](#)

[TFSLabConfig CreateTeamProjectLibraryShare Command](#)

Concepts

TFSLabConfig Command Reference

TFSSConfig Lab Command Reference

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig ListTeamProjectCollectionHostGroups Command

[Example](#) [See Also](#) [Send Feedback](#)

Use the **ListTeamProjectCollectionHostGroups** command to list the System Center Virtual Machine Manager (SCVMM) host groups that are assigned to a team project collection and that you have read access to. These collection host groups can be added to a team project by using the **CreateTeamProjectHostGroup** command. For more information, see [TFSLabConfig CreateTeamProjectHostGroup Command](#)

Required Permissions

To use the **ListTeamProjectCollectionHostGroups** command, you must have **View Lab Resources** permission at the Team Project collection level. By default, members of the TFS Administrators and Team Project Collection Administrators groups have this permission. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSLabConfig ListTeamProjectCollectionHostGroups  
/Collection:collectionUrl
```

Parameters

Option	Description
Collection: collectionUrl	Required. The URL of the team project collection on the application tier of Team Foundation Server. For example, /collection:http://abc:8080/TFS/DefaultCollection.

Example

For increased readability in the example, command options are listed on separate lines. In a command prompt window, type all options for a command on the same line.

 **Copy Code**

```
TFSLabConfig ListTeamProjectCollectionHostGroups  
/collection:http://abc:8080/TFS/Collection0
```

See Also

Reference

[TFSLabConfig ListTeamProjectHostGroups Command](#)

[TFSLabConfig CreateTeamProjectHostGroup Command](#)

[TFSLabConfig DeleteTeamProjectHostGroup Command](#)

Concepts

[TFSLabConfig Command Reference](#)

[TFSSConfig Lab Command Reference](#)

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig ListTeamProjectCollectionLibraryShares Command

[Example](#) [See Also](#) [Send Feedback](#)

Use the **ListTeamProjectCollectionLibraryShares** command to list the library shares that have been assigned to a team project collection and that you have read access to. Library shares are created in System Center Virtual Machine Manager (SCVMM), and are assigned to a team project collection by using the [TFSLabConfig Lab /LibraryShare Commands](#). Team project collection library shares can be added to a team project by using the **CreateTeamProjectLibraryShare** command. For more information, see [TFSLabConfig CreateTeamProjectLibraryShare Command](#).

Required Permissions

To use the **ListTeamProjectCollectionLibraryShares** command, you must have **View Lab Resources** permission at the Team Project collection level. By default, members of the TFS Administrators and Team Project Collection Administrators groups have this permission. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSLabConfig ListTeamProjectCollectionLibraryShares  
Collection:collectionUrl
```

Parameters

Argument	Description
Collection: collectionUrl	Required. The URL of the team project collection on the application tier of Team Foundation Server. For example, /collection:http://abc:8080/TFS/DefaultCollection.

Example

For increased readability in the example, command options are listed on separate lines. In a command prompt window, type all options for a command on the same line.

 **Copy Code**

```
TFSLabConfig ListTeamProjectCollectionLibraryShares  
/collection:http://abc:8080/TFS/Collection0
```

See Also

Reference

[TFSLabConfig ListTeamProjectLibraryShares Command](#)

[TFSLabConfig CreateTeamProjectLibraryShare Command](#)

[TFSLabConfig DeleteTeamProjectLibraryShare Command](#)

Concepts

[TFSLabConfig Command Reference](#)

[TFSSConfig Lab Command Reference](#)

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig ListTeamProjectHostGroups Command

[Example](#) [See Also](#) [Send Feedback](#)

Use the **ListTeamProjectHostGroups** command to list the host groups that are assigned to a team project and that you have read access to.

Required Permissions

To use the ListTeamProjectHostGroups command, you must have **View Lab Resources** permissions at the Team Project level. By default, members of the Team Foundation Server Administrators, Team Project Collection Administrators, Team Project Administrators, Team Project Contributors, and Team Project Readers groups have this permission. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSLabConfig ListTeamProjectHostGroups  
Collection:collectionUrl  
/TeamProject:teamProjectName
```

Parameters

Argument	Description
Collection: collectionUrl	Required. The URL of the team project collection on the application tier of Team Foundation Server that contains the team project. For example, /collection:http://abc:8080/TFS/DefaultCollectio
TeamProject: teamProjectName	Required. The name of the team project. Use quotation marks if there are spaces in the name

Example

For increased readability in the example, command options are listed on separate lines. In a command prompt window, type all options for a command on the same line.

 **Copy Code**

```
TFSLabConfig ListTeamProjectHostGroups  
  /collection:http://abc:8080/TFS/DefaultCollection  
  /teamProject:Project1
```

See Also

Reference

[TFSLabConfig ListTeamProjectCollectionHostGroups Command](#)

[TFSLabConfig CreateTeamProjectHostGroup Command](#)

[TFSLabConfig DeleteTeamProjectHostGroup Command](#)

Concepts

[TFSLabConfig Command Reference](#)

[TFSSConfig Lab Command Reference](#)

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig ListTeamProjectLibraryShares Command

[See Also](#) [Send Feedback](#)

Use the **ListTeamProjectLibraryShares** command to list all library shares that are assigned to a team project and that you have access to.

Required Permissions

To use the ListTeamProjectLibraryShares command, you must have the **View Lab Resources** permission at Team Project level. By default, members of the TFS Administrators, Team Project Collection Administrators, Team Project Administrators, Team Project Contributors, and Team Project Readers groups have this permission. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSLabConfig ListTeamProjectLibraryShares  
Collection:collectionUrl  
/TeamProject:teamProjectName
```

Parameters

Option	Description
Collection: collectionUrl	Required. The URL of the team project collection on the application tier of Team Foundation Server that contains the team project. For example, /collection:http://abc:8080/TFS/DefaultCollectio
TeamProject: teamProjectName	Required. The name of the team project. Use quotation marks if there are spaces in the name.

See Also

Reference

[TFSLabConfig ListTeamProjectCollectionLibraryShares Command](#)

[TFSLabConfig DeleteTeamProjectLibraryShare Command](#)

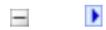
[TFSLabConfig CreateTeamProjectLibraryShare Command](#)

Concepts

[TFSLabConfig Command Reference](#)

[TFSSConfig Lab Command Reference](#)

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig Permissions Command

[See Also](#) [Send Feedback](#)

Use the **TFSLabConfig Permissions** command to set and get permissions set for a specified user or for multiple users on a specified object in Visual Studio Lab Management. For more information about individual permissions, see the Lab Management Permissions section of [Permission reference for Team Foundation Server](#).

Required Permissions

To query permissions on an object, you must have Read permissions for the object. To change permissions on an object the **Permissions** command, you must have the **Manage Permissions** permission. By default, the creator of the object has this permission. For more information, see [Permission reference for Team Foundation Server](#).

TFSLabConfig Permissions

```
/Collection:collectionUrl  
[objectSpec]  
{[/User:userName1[,userName2][, ...]]  
[/Group:groupName1[,groupName2][, ...]]}  
[/Allow:{* |perm1[,perm2][, ...] }]  
[/Deny:{* |perm1[,perm2][, ...]}]  
[/Remove:{* |perm1[,perm2][, ...]}]  
[/Inherit:Yes|No]
```

Parameters

Option	Description
Collection: collectionUrl	Required. The URL of the team project collection on the application tier of Team Foundation Server. For example, /collection:http://abc:8080/TFS/DefaultCollection
objectSpec	Optional. Specifies the target object such as team project or library share to which the permissions are applied. For information about how to specify objects, see objectSpec below.
User: userName1[,userName2][,...]	Optional. Specifies one or more users to which permissions are applied. Use commas to separate multiple user names.
Group: groupName1[,groupName2][,...]	Optional. Specifies one or more groups to which the permissions are applied. Use commas to separate multiple group names.
Allow: {* perm1[,perm2][,...]}	Optional. Enables the specified permissions for the specified users or groups. Use an asterisk to specify all permissions. To specify an individual permission, use the identifiers in the Name at command line column of the table in the Lab Management Permissions section of the Permission reference for Team Foundation Server . Use commas to separate multiple permissions.

Deny:{* |perm1[,perm2][,...]}

Optional. Denies the specified permissions to the specified users or groups. Use an asterisk (*) to specify all permissions. To specify an individual permission, use the identifiers in the **Name at command line** column of the table in the Lab Management Permissions section of [Permission reference for Team Foundation Server](#). Use commas to separate multiple permissions.

Remove:{* |perm1[,perm2][,...]}

Optional. Unsets the specified permissions that were previously granted or denied to the user or group. To specify an individual permission, use the identifiers in the **Name at command line** column of the table in the Lab Management Permissions section of [Permission reference for Team Foundation Server](#). Use commas to separate multiple permissions.

Inherit: Yes|No

Optional. If you specify **Yes**, all permissions associated with a parent ACL are inherited by this item. Cannot be combined with the **/remove**, **/user**, or **/group** options.

objectSpec

You can specify the objects that you want to include in the **TFSLabConfig Permissions** command in two ways:

- Use one or more **locations** options to specify the object in the Lab Management hierarchy.
- Use the **/Url** option to specify the object as a Uri.

If the **objectSpec** parameter option is not specified, the permissions are applied to all objects in the team project collection.

Object type options

The following table lists the valid combination of options that you can use to specify an object as the **objectSpec** parameter of a **TFSLabConfig permissions** command.

To set permissions on	Use these options
A specific host group in a team project collection	/TeamProjectCollectionHostGroup: teamProjectCollectionHostGr
A specific library share in a team project collection	/TeamProjectCollectionLibraryShare: teamProjectCollectionLibr

All group
hosts in a **/TeamProject:projectName /TeamProjectHostGroup:***
team project

A group
host in a **/TeamProject:projectName**
team project **/TeamProjectHostGroup:teamProjectHostGroupName**

A lab
environment **/TeamProject:projectName**
in a host **/TeamProjectHostGroup:teamProjectHostGroupName**
group for a **/LabEnvironment:labEnvironmentName**
team project

All library
shares in a **/TeamProject:projectName /TeamProjectLibraryShare:***
team project

A library
share in a **/TeamProject:projectName**
team project **/TeamProjectLibraryShare:teamProjectLibraryShareName**

A lab
template in **/TeamProject:projectName**
a library **/TeamProjectLibraryShare:teamProjectLibraryShareName**
share of a **/LabTemplate:labTemplateName**
team project

A lab
environment **/TeamProject:projectName**
in a library **/TeamProjectLibraryShare:teamProjectLibraryShareName**
share of a **/LabEnvironment:labEnvironmentName**

team project

≡Url

Use the following syntax to specify the objectSpec target object of a **TFSLabConfig permissions** command by using the **/Url** option:

/url:VSTFS:///LabManagement/objectType/objectId

The **objectId** is the unique numeric identifier of the object.

The following table lists the valid keywords for the **objectType** keyword:

Object Type	Description
TeamProjectCollectionHostGroup	A host group of a team project collection
TeamProjectCollectionLibraryShare	A library share of a team project collection
TeamProject	A team project
TeamProjectHostGroup	A host group of a team project

TeamProjectLibraryShare

A library
share of a
team project

LabTemplate

A virtual
machine or
template in
a team
project
library share

LabEnvironment

An
environment
that is
deployed on
a team
project host
group or
stored in a
team project
library
share.

Permissions

You can specify one or more of Lab Management permissions as the target of the **/Allow**, **/Deny**, or **/Remove** options. For a lists of available permissions, see the **Lab Management Permissions** section of the

[Permission reference for Team Foundation Server](#) topic.

- Use an asterisk (*) to specify all lab permissions.
- Use commas to separate multiple permissions.

Remarks

If neither the **/User** or **/Group** option is specified, the current permissions of the specified object is displayed.

If the `objectSpec` parameter option is not specified, the permissions are applied to all objects in the team project collection.

See Also

Concepts

TFSLabConfig Command Reference

[Permission reference for Team Foundation Server](#)

TFSSConfig Lab Command Reference

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TfsLabConfig TPCHostGroup Command

[Example](#) [See Also](#) [Send Feedback](#)

Note

These commands only work on SCVMM 2012 server, and are not supported on SCVMM 2008 R2 server.

Use the **TPCHostGroup** command to add or remove a host group from a team project collection, or to modify the Lab Management settings of the host group.

To run these commands, you must be a member of Team Project Collection Administrators group in Team Foundation Server for the collection you specify. In addition, you must be a member of Administrator or Delegated Administrator role in the SCVMM Server from which you are adding the host groups. Furthermore, you must ensure that no other team project collection in any Team Foundation Server is already using the same SCVMM host group.

```
TfsLabConfig TPCHostGroup
  /collection:collectionUrl
  [/add
    /scvmmhostgroup:scvmmHostGroupPath
    /name:name
    [/dnssuffix:dnsSuffixForNetworkIsolation]
    [/autoprovision:{True|False}]]
  [/delete
    /name:name
    [/noprompt]]
  [/edit
    /name:hostGroupName
    [/autoprovision:{True|False}]
    [/dnssuffix:dnsSuffixForNetworkIsolation]
    [/noprompt]]
  [/list]
```

[/listscvmmhostgroups]

Parameters

Option	Description
collection: collectionUrl	Required. Specifies the URL of the team project collection on the application-tier of the Team Foundation Server.
add	Adds the specified host group to the team project collection. You must specify the /collection , /scvmmhostgroup , and name options with /add .
scvmmhostgroup: scvmmHostGroupPath	Required with /add . Specifies the fully qualified domain name (FDQN) of the VMM host group. The FQDN path can be found by using the VMM Admin Console.

name:hostGroupName

Required with **/add**. Specifies the name of the host group in the team project collection.

dnsSuffix:dnsSuffixForNetworkIsolation

Optional with **/add** or **/edit**. Specifies the DNS suffix that will be used to register the names of virtual machines on the isolated network for the virtual

environments within this host group. To confirm that the suffix is configured correctly in the DNS hierarchy, contact your network administrator.

Optional with **/add** or **/edit**. Specifies whether the host group is automatically assigned to each team project in the collection. By default, **/autoProvision** is

autoProvision:True|False

set to **True**, and host group is automatically assigned to every team project in the collection. Note: The **/autoProvision** option affects existing team projects.

Delete

Removes the specified host group from the team project collection. You must specify the **/collection** and **/name** options.

noPrompt

Optional with **/add**, **/edit**, or **/delete**. Suppresses display progress and result data from the command window.

edit

Changes the Lab Management settings of the host group. You must specify the **/collection** and **/name** options.

list

Lists all host groups that are assigned to the specified team project collection.

listVmmHostGroups

Lists all host groups that are available in VMM.

Example

In this example, a host group is added to a team project collection:

 **Copy Code**

```
TFSLabConfig TPCHostGroup  
  /collection:http://abc:8080/TFS/Collection0  
  /name:HostGroup1
```

See Also

Reference

[TFSLabConfig TPCLibraryShare Command](#)

Concepts

[Configure Lab Management with TFSLabConfig](#)
[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TfsLabConfig TPCLibraryShare Command

[Example](#) [See Also](#) [Send Feedback](#)

Note

These commands only work on SCVMM 2012 server, and are not supported on SCVMM 2008 R2 server.

Use the **TPCLibraryShare** command to add or remove a library share from a team project collection, or to modify the lab management settings of the library share.

To run these commands, you must be a member of Team Project Collection Administrators group in Team Foundation Server for the collection you specify. In addition, you must be a member of Administrator or Delegated Administrator role in the SCVMM Server from which you are adding the host groups. Furthermore, you must ensure that no other team project collection in any Team Foundation Server is already using the same SCVMM library share.

```
TfsLabConfig TPCLibraryShare
  /collection:collectionUrl
  [/add
    /scvmmLibraryShare:vmmLibrarySharePath
    /name:libraryShareName
    [/autoprovision:{True|False}]]
  [/delete
    /name:libraryShareName
    [/noprompt]]
  [/edit
    /name:libraryShareName
    [/autoprovision:{True|False}]
    [/noprompt]]
  [/list]
  [/listscvmmlibraryshares]
```

Parameters

Option	Description
collection: collectionUrl	Required. Specifies the URL of the team project collection on the application-tier of the Team Foundation Server.
add	Adds the specified library share to the team project collection. You must specify the /collection , /scvmmlibraryshare , and /name options with /add .
scvmmLibraryShare: vmmLibrarySharePath	Required with /add . Specifies the fully qualified domain name (FQDN) of the VMM share. The FQDN path can be retrieved by using the VMM Admin Console.
	Required with /add ,

name:libraryShareName

edit, and **delete**.

Specifies the name of the library share in the team project collection.

autoProvision:True|False

Optional with **/add** or **/edit**. Specifies whether the library share is automatically assigned to each team project in the collection. By default,

/autoProvision is set to **True**, and library share is automatically assigned to every team project in the collection. The **/autoProvision** option only affects existing team projects.

Delete

Removes the specified library share from the team project collection. You must specify the **/collection** and **/name** options.

Optional with **/add**, **/edit**, or **/delete**.

Suppresses display

noPrompt

progress and result data from the command window.

edit

Changes Lab Management settings of the library share. You must specify the **/collection** and **/name** options.

list

Lists all library shares that are assigned to the specified team project collection.

listscvmmlibraryshares

Lists all library shares that are available in VMM.

Example

In this example, a library share is added to a team project collection.

 **Copy Code**

```
TFSLabConfig TPCLibraryShare
  /collection:http://abc:8080/TFS/Collection0
  /add
  /scvmmLibraryShare:"LibraryShares\LibraryShare1"
  /name:LibraryShare1
```

See Also

Reference

[TFSLabConfig TPCHostGroup Command](#)

Concepts

[Configure Lab Management with TFSLabConfig](#)
[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Tfslabconfig TPHostGroup Command

[Example](#) [See Also](#) [Send Feedback](#)

Note

These commands only work on SCVMM 2012 server, and are not supported on SCVMM 2008 R2 server.

Use the **TPHostGroup** command to assign or unassign a host group from a team project collection to individual team projects in the collection. Host groups specify one or more physical machines that are the deployment targets for virtual environments in Visual Studio Lab Management..

To run these commands, you must be a member of Team Project Collection Administrators group in Team Foundation Server for the collection you specify. In addition, you must be a member of Administrator or Delegated Administrator role in the SCVMM Server from which you are adding the host groups.

```
Tfslabconfig TPHostGroup
  /collection:collectionUrl
  /teamProject:* | teamProjectName
  [/add
    /teamProjectCollectionHostGroup:* | teamProjectCollectionName
    /name:teamProjectHostGroupName
    [/description:description]
    [/NoPrompt]]
  [/delete
    /name:* | teamProjectHostGroupName
    [/noprompt]]
  [/list]
```

Parameters

Option	Description
collection: collectionUrl	Required. Specifies the URL of the team project collection on the application tier of the Team Foundation Server.
teamProject: * teamProjectName	The name of the team project to which to add or delete host groups. Use quotation marks if there are spaces in the name. Use an asterisk (*) to assign the

add

specified
host group
to all team
projects in
the
collection

Adds the
specified
host group
to the team
projects.

The name
of the hos
group in tl
team
project
collection
Use
quotation
marks if
there are
spaces in
the name.

teamProjectCollectionHostGroup:teamProjectCollectionHostGroup

Use an
asterisk (*
to assign a
host group
in the
collection
to the
specified
team
project.

name:hostGroupName

The name of the host group in the team project collection

Delete

Removes the host group from the team project.

noPrompt

Suppresses display of progress and result data from the command window.

list

Lists all host groups that are assigned to the specified team project.

Example

In this example, all the host groups in the team project collection are assigned to all the team projects in the collection:

 **Copy Code**

```
TFSLabConfig TPHostGroup /add
  /collection:http://abc:8080/TFS/Collection0
  /teamProject:*
  /teamProjectCollectionHostGroup:NORTHAMERICA\hostgroup1
  /name:HostGroup1
```

See Also

Reference

[TFSLabConfig TPCLibraryShare Command](#)

[TFSLabConfig TPLibraryShare Command](#)

Concepts

[Configure Lab Management with TFSLabConfig](#)

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TfsLabConfig TPLibraryShare Command

[See Also](#) [Send Feedback](#)

Note

These commands only work on SCVMM 2012 server, and are not supported on SCVMM 2008 R2 server.

Use the **TPLibraryShare** command to assign or unassign a library share from a team project collection to an individual team project in the collection. A library share provides access to file-based resources for virtual environments such as ISO images and virtual hard disks. Library shares are created in System Center Virtual Machine Manager (SCVMM) and assigned to project collection by Visual Studio Lab Management.

To run these commands, you must be a member of Team Project Collection Administrators group in Team Foundation Server for the collection you specify. In addition, you must be a member of Administrator or Delegated Administrator role in the SCVMM Server from which you are adding the host groups.

```
TfsLabConfig TPLibraryShare
  /collection:collectionUrl
  /teamProject:* | teamProjectName
  [/add
    /teamProjectCollectionLibraryShare:* | teamProjectCollec
    /name:teamProjectLibraryShareName
    [/noprompt]]
  [/delete
    /name:* | teamProjectLibraryShareName
    [/noprompt]]
  [/list]
```

Parameters

Option

collection:collectionUrl

add

teamProjectCollectionLibraryShare:teamProjectCollectionLibraryShareName

name:libraryShareName

Delete

noPrompt

list

See Also

Reference

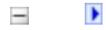
[TFSLabConfig TPCLibraryShare Command](#)

[TFSLabConfig TPHostGroup Command](#)

Concepts

[Configure Lab Management with TFSLabConfig](#)

[Configure and administer Lab Management](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSLabConfig Settings Command

[Send Feedback](#)

Assign the System Center Virtual Machine Manager (VMM) that will be used by Lab Management.

To enable TFS to communicate with SCVMM, add the account under which TFS is running to the Administrator role in SCVMM.

TFSLabConfig Settings

```
/collection:teamProjectCollectionUrl  
/scvmmServerName:VMM server name  
[/ipblock:range of addresses for network isolation]  
[/dnssuffix:dns suffix for network isolation]  
[/noPrompt]  
[/Force]  
[/list]
```

Parameters

Parameter	Description
collection: collectionUrl	Required. The URL of the team project collection on the application-tier of the Team Foundation Server. Example: http://team.fabrikam.com:8080/tfs/DefaultCollection/
scvmmServerName: VMM server name	Required. The fully qualified domain name of the SCVMM server that will be used by Lab Management. Example: scvmm.fabrikam.com
ipblock: range of addresses for network isolation	If you want to isolate each lab environment in its own private network, set this to specify the addresses to assign to its virtual machines. Because these IP addresses are used only for internal routing among the virtual machines in an environment, and are not exposed beyond the boundaries of an environment, you can specify any IP range that is not used within your public network. Example: 192.168.1.0/63
dnssuffix: dns suffix for network isolation	If you want to isolate each lab environment in its own private network, specify the domain name suffix that you want to give the virtual machines in each lab environment.

Example: fabrikam.com

noPrompt

Don't display progress and results.

Force

Update the server name even if it has already been set.

list

Report the SCVMM server that is currently configured for a specified collection.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Change groups and permissions with TFSSecurity

[See Also](#) [Send Feedback](#)

You can use the **TFSSecurity** command-line tool to create, modify, and delete groups and users in Visual Studio Team Foundation Server (TFS), in addition to modifying permissions for groups and users. For information about how to perform these tasks in the user interface, see [Manage users or groups in TFS](#).

This server-level tool is located in Drive:\%programfiles%\Microsoft Team Foundation Server 12.0\Tools on the TFS application-tier server.

Even if you are logged on with administrative credentials, you must open an elevated Command Prompt to perform this function.

You can use the options in the following table for all commands in this utility.

Option	Description
/?	Displays the command syntax and options for TFSSecurity .

In This Section

Command set	Command
Add, remove, or view permissions assigned to a user or group	<ul style="list-style-type: none"><li data-bbox="461 604 797 821">• /a+: Add permissions for a user or a server, collection, or project-level group<li data-bbox="461 863 797 1121">• /a-: Remove a user or group from membership in a server, collection, or project-level group<li data-bbox="461 1163 797 1331">• /acl: Display the access control list that applies to a specified object<li data-bbox="461 1423 797 1591">• /g: List the groups in a team project, project collection, or across TFS<li data-bbox="461 1633 797 1759">• /g+: Add a user or another group to an existing group<li data-bbox="461 1801 797 1890">• /g-: Remove a user or a group from an

Create, list, delete, rename, and add users to groups

existing group

- /gc: Create a project-level group
- /gcg: Create a server or collection-level group
- /gd: Delete a server or collection-level group
- /gud: Change the description for a server or collection-level group
- /gun: Rename a server or collection-level group

View information about identities and membership within groups

- /i: Display identity information for a specified group
- /im: Display information about the identities that compose the direct membership of a group that you specify
- /imx: Display information about

Check
explicit and
implicit group
membership
information

the identities that
compose the
expanded
membership of a
specified group

- /m: Check explicit and implicit group membership information for a specified group or user

See Also

Concepts

[TFS Security Identity and Output Specifiers](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSSecurity Identity and Output Specifiers

[See Also](#) [Send Feedback](#)

The input and output for the **TFSSecurity** command-line utility follows a standard format. The tables later in this topic describe valid identity and output specifiers for this command. These specifiers apply to all of the TFSSecurity command-line utilities.

 **Note**

Even if you are logged on with administrative credentials, you must open an elevated Command Prompt to perform this function.

 **Note**

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Identity Specifiers

You can reference an identity by using one of the notations in the following table.

Identity specifier	Description	Example
sid: Sid.	References the identity that has the specified security identifier (SID).	sid:S-1-5-21-2127521184-1604012920-1887927527-588340
n: [Domain\]Name	References the identity that has the specified name. For Windows, Name is the account name. If the referenced identity is in a domain, the domain name is required. For application groups, Name is the group display name, and Domain is the URI or GUID of the containing	To reference the identity of the user "John Peoples" in the domain "Datum1" at the fictitious company "A. Datum Corporation:" n:DATUM1\jpeoples To reference application groups: n:"Full-time Employees" n:00a10d23-7d45-4439-981b-d3b3e0b0b1ee\Vendors

project. In this context, if Domain is omitted, the scope is assumed to be at the collection level.

References the administrative application group for the scope, such as "Team Foundation Administrators" for the server level or "Project Collection Administrators" at the collection level. The optional parameter

adm:[Scope]

Scope is a project URI or URL, including its GUID and connection string. If scope is omitted, the server or collection scope is assumed based on whether the

adm:vstfs:///Classification/TeamProject/Gl

	<p>/instance or /server parameter is used. In either case, the colon is still required.</p>	
srv:	<p>References the application group for service accounts.</p>	Not applicable
all:	<p>References all groups and identities.</p>	Not applicable
String	<p>References an unqualified string. If String starts with S-1-, it is identified as a SID. If String starts with CN= or LDAP:// it is identified as a distinguished name. Otherwise, String is identified as a name.</p>	"Team testers"

Identity Type Markers

Identity Type Markers

The following table lists identity type markers that are used in output messages.

Identity type marker	Description
U	Windows user.
G	Windows group.
A	Team Foundation Server (TFS) application group.
a [A]	Administrative application group.
s [A]	Service account application group.
X	Identity is not valid.
?	Identity is unknown.

Access Control Entry Markers

The following table lists access control entry markers that are used in output messages.

Access control entry marker	Description
+	ALLOW access control entry.
-	DENY access control entry.
* []	Inherited access control entry.

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/a+ command: Add permissions

[See Also](#) [Send Feedback](#)

Use /a+ to add permissions for a user or a group in a server-level, collection-level, or project-level group. To add users to groups from the user interface, see [Manage users or groups in TFS](#).

Required Permissions

To use the /a+ command, you must have the View collection-level information or the View instance-level information permission set to Allow, depending on whether you are using the /**collection** or /**server** parameter, respectively. If you are changing permissions for a team project, you must also have the Edit project-level information permission for the team project set to Allow. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /a+ Namespace Token Action Identity (ALLOW | DENY) [/co]
```

Parameters

Argument	Description
Namespace	The namespace that contains the group to which you add permissions for a user or group. You can also use TFSSecurity /a command to view a list of namespaces on server, collection, and project level.
Token	<p>The name or GUID of the object on which you want to add permissions.</p> <p> Note</p> <p>Tokens vary depending on the namespace you specify. Some namespaces do not have tokens that apply for this command.</p>
Action	The name of the permission for which you are granting or denying access. For a list of valid IDs, see Permissions for Team Foundation Server , or use the TFSSecurity /a command to view a list of valid actions for a namespace that you specify.
Identity	<p>The identity of the user or the group. For more information about identity specifiers, see TFS Security Identity and Specifiers.</p> <ul style="list-style-type: none">• ALLOW <p>The group or user can perform the operation that the Action specifies.</p>

- **DENY**

The group or user cannot perform the operation
Action specifies.

/collection:CollectionURL Required if **/server** is not used. Specifies the URL of project collection in the following format:
http://ServerName:Port/VirtualDirectoryName/Colle

/server:ServerURL Required if **/collection** is not used. Specifies the URL application-tier server in the following format:
http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

Access control entries are security mechanisms that determine which operations a user, group, service, or computer is authorized to perform.

Examples

The following example displays what namespaces are available at the server level for the application-tier server that is named ADatumCorporation.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
>tfssecurity /a /server:ServerURL
```

Sample output:

Copy Code

```
TFSSecurity - Team Foundation Server Security Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
The target Team Foundation Server is http://ADatumCorporation:8080/.
```

```
The following security namespaces are available to have permissions
```

```
Registry  
Identity  
Job  
Server  
CollectionManagement  
Warehouse  
Catalog  
EventSubscription  
Lab
```

```
Done.
```

The following example displays what actions are available for the Server namespace at the collection level.

Copy Code

```
>tfssecurity /a Server /collection:CollectionURL
```

Sample output:

Copy Code

```
TFSecurity - Team Foundation Server Security Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
The target Team Foundation Server is http://ADatumCorporation:8080/.
```

```
The following actions are available in the security namespace Server  
GenericRead  
GenericWrite  
Impersonate  
TriggerEvent
```

```
Done.
```

The following example grants the server-level "View instance-level information" permission to the ADatumCorporation deployment for the Datum1 domain user John Peoples (Datum1\jpeoples).

Copy Code

```
>tfssecurity /a+ Server FrameworkGlobalSecurity GenericRead n:Datum:
```

Sample output:

Copy Code

```
TFSecurity - Team Foundation Server Security Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
The target Team Foundation Server is http://ADatumCorporation:8080/.  
Resolving identity "n:Datum1\jpeoples" ...  
[U] Datum1\jpeoples (John Peoples)  
Adding the access control entry ...  
Verifying ...
```

```

Effective ACL on object "FrameworkGlobalSecurity":
[+] GenericRead [INSTANCE]\Team Foundation
[+] GenericRead [INSTANCE]\SharePoint Web /
[+] Impersonate [INSTANCE]\SharePoint Web /
[+] GenericRead [INSTANCE]\Team Foundation
[+] GenericWrite [INSTANCE]\Team Foundation
[+] Impersonate [INSTANCE]\Team Foundation
[+] TriggerEvent [INSTANCE]\Team Foundation
[+] GenericRead [INSTANCE]\Team Foundation
[+] GenericWrite [INSTANCE]\Team Foundation
[+] TriggerEvent [INSTANCE]\Team Foundation
[+] GenericRead DATUM1\jpeoples

```

Done.

The following example grants the collection-level "View collection-level information" permission to the Collection0 team project collection for Datum1 domain user John Peoples (Datum1\jpeoples).

```
>tfssecurity /a+ Server FrameworkGlobalSecurity GenericRead n:Datum1\jpeoples
```

Sample output:

 **Copy Code**

```

TFSSecurity - Team Foundation Server Security Tool
Copyright (c) Microsoft Corporation. All rights reserved.
The target Team Foundation Server is http://ADatumCorporation:8080/
Resolving identity "n:Datum1\jpeoples"...
[U] DATUM1\jpeoples (John Peoples)
Adding the access control entry...
Verifying...

```

```

Effective ACL on object "FrameworkGlobalSecurity":
[+] GenericRead [Collection0]\Project Colle
[+] GenericRead [Collection0]\Project Colle
[+] GenericWrite [Collection0]\Project Colle
[+] Impersonate [Collection0]\Project Colle
[+] TriggerEvent [Collection0]\Project Colle
[+] GenericRead [Collection0]\Project Colle
[+] GenericWrite [Collection0]\Project Colle
[+] TriggerEvent [Collection0]\Project Colle

```

[+] GenericRead
[+] Impersonate
[+] GenericRead
[+] GenericRead

[INSTANCE]\SharePoint Web /
[INSTANCE]\SharePoint Web /
[Collection0]\Project Colle
DATUM1\jpeoples

Done.

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)

How to: Create a Global Group

Team Project Groups

Default Groups



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/a- command: Remove a user or a group from membership in a group

[See Also](#) [Send Feedback](#)

Use the **/a-** command to remove a user or a group from membership in a server-level, collection-level, or project-level group. To add users to groups from the user interface, see [Manage users or groups in TFS](#).

Required Permissions

To use the **/a-** command, you must have the View collection-level information or the View instance-level information permission set to Allow, depending on whether you are using the **/collection** or **/server** parameter, respectively. If you are changing permissions for a team project, you must also have the Edit project-level information permission for the team project set to Allow.

```
TFSSecurity /a- Namespace Token Action Identity (ALLOW | DENY) [/co]
```

Parameters

Argument	Description
Namespace	<p>The namespace that contains the group from which you remove the user or group. You can also use the TFSS command to view a list of namespaces at the server level, collection level, and the project level.</p>
Token	<p>The name or GUID of the object on which you want to set permissions.</p> <p> Note</p> <p>Tokens vary depending on the namespace that you specify. Some namespaces do not have tokens that apply for the TFSS command.</p>
Action	<p>The name of the permission that for which access is granted or denied. For a list of valid IDs, see Permission references for Team Foundation Server, or use the TFSSecurity /a command to view a list of valid actions for a namespace that you specify.</p>
Identity	<p>The identity of the user or the group. For more information about the identity specifiers, see TFSSecurity Identity Output Specifiers.</p> <ul style="list-style-type: none">• ALLOW <p>The group or user can perform the operation that the Action specifies.</p>

- **DENY**

The group or user cannot perform the operation
Action specifies.

/collection:CollectionURL Required if **/server** is not used. Specifies the URL of project collection in the following format:
http://ServerName:Port/VirtualDirectoryName/Colle

/server:ServerURL Required if **/collection** is not used. Specifies the URL application-tier server in the following format:
http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

Access control entries are security mechanisms that determine which operations a user, group, service, or computer is authorized to perform on a computer or server.

Examples

The following example displays what namespaces are available at the server level for the application-tier server that is named ADatumCorporation.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
>tfssecurity /a /server:ServerURL
```

Sample output:

Copy Code

```
TFSSecurity - Team Foundation Server Security Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
The target Team Foundation Server is http://ADatumCorporation:8080/.
```

```
The following security namespaces are available to have permissions
```

```
Registry  
Identity  
Job  
Server  
CollectionManagement  
Warehouse  
Catalog  
EventSubscription  
Lab
```

```
Done.
```

The following example displays what actions are available for the Server namespace at the collection level.

Copy Code

```
>tfssecurity /a Server /collection:CollectionURL
```

Sample output:

Copy Code

```
TFSSecurity - Team Foundation Server Security Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
The target Team Foundation Server is http://ADatumCorporation:8080/.
```

```
The following actions are available in the security namespace Server  
GenericRead  
GenericWrite  
Impersonate  
TriggerEvent
```

```
Done.
```

The following example removes the server-level "View instance-level information" permission to the ADatumCorporation deployment for the Datum1 domain user John Peoples (Datum1\jpeoples).

Copy Code

```
>tfssecurity /a- Server FrameworkGlobalSecurity GenericRead n:Datum:
```

Sample output:

Copy Code

```
TFSSecurity - Team Foundation Server Security Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
The target Team Foundation Server is http://ADatumCorporation:8080/.  
Resolving identity "n:Datum1\jpeoples"...  
[U] Datum1\jpeoples (John Peoples)  
Removing the access control entry...  
Verifying...
```

```

Effective ACL on object "FrameworkGlobalSecurity":
[+] GenericRead [INSTANCE]\Team Foundation
[+] GenericRead [INSTANCE]\SharePoint Web /
[+] Impersonate [INSTANCE]\SharePoint Web /
[+] GenericRead [INSTANCE]\Team Foundation
[+] GenericWrite [INSTANCE]\Team Foundation
[+] Impersonate [INSTANCE]\Team Foundation
[+] TriggerEvent [INSTANCE]\Team Foundation
[+] GenericRead [INSTANCE]\Team Foundation
[+] GenericWrite [INSTANCE]\Team Foundation
[+] TriggerEvent [INSTANCE]\Team Foundation

```

Done.

The following example removes the collection-level "View collection-level information" permission to the Collection0 team project collection for Datum1 domain user John Peoples (Datum1\jpeoples).

```
>tfssecurity /a+ Server FrameworkGlobalSecurity GenericRead n:Datum1\jpeoples
```

Sample output:

 **Copy Code**

```

TFSSecurity - Team Foundation Server Security Tool
Copyright (c) Microsoft Corporation. All rights reserved.
The target Team Foundation Server is http://ADatumCorporation:8080/
Resolving identity "n:Datum1\jpeoples"...
[U] DATUM1\jpeoples (John Peoples)
Removing the access control entry...
Verifying...

```

```

Effective ACL on object "FrameworkGlobalSecurity":
[+] GenericRead [Collection0]\Project Colle
[+] GenericRead [Collection0]\Project Colle
[+] GenericWrite [Collection0]\Project Colle
[+] Impersonate [Collection0]\Project Colle
[+] TriggerEvent [Collection0]\Project Colle
[+] GenericRead [Collection0]\Project Colle
[+] GenericWrite [Collection0]\Project Colle
[+] TriggerEvent [Collection0]\Project Colle
[+] GenericRead [INSTANCE]\SharePoint Web /

```

[+] Impersonate
[+] GenericRead

[INSTANCE]\SharePoint Web /
[Collection0]\Project Colle

Done.

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)

How to: Create a Global Group

Team Project Groups

Default Groups



Visual Basic C#

Visual C++

F#

HLSL

JScript

Visual Studio Application Lifecycle Management

/acl command: Display the access control list

[See Also](#) [Send Feedback](#)

Use **/acl** to display the access control list that applies to a particular object.

Required Permissions

To use the **/acl** command, you must have the View collection-level information or the View instance-level information permission set to Allow, depending on whether you are using the **/collection** or **/server** parameter, respectively. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /acl Namespace Token [/collection:CollectionURL] [/serve
```

Parameters

Argument	Description
Namespace	The namespace that contains the group for which you view permissions for a user or group.
Token	<p>The name or GUID of the object on which you want permissions.</p> <p><input checked="" type="checkbox"/> Note</p> <p>Tokens vary depending on the namespace that you specify. Some namespaces do not have tokens that apply for this command.</p>
/collection: CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/server: ServerURL	Required if /collection is not used. Specifies the URL of application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

Access control entries are security mechanisms that determine which operations a user, group, service, or computer is authorized to perform on a computer or server.

Examples

The following example displays what users and groups have access to the FrameworkGlobalSecurity token in the Server namespace within the ADatumCorporation deployment.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
> tfssecurity /acl Server FrameworkGlobalSecurity /server:ServerURL
```

Sample output:

Copy Code

```
TFSSecurity - Team Foundation Server Security Tool
Copyright (c) Microsoft Corporation. All rights reserved.
The target Team Foundation Server is http://ADatumCorporation:8080/.
Retrieving the access control list for object "Server"...
```

```
Effective ACL on object "FrameworkGlobalSecurity":
[+] GenericRead [INSTANCE]\Team Foundation
[+] GenericRead [INSTANCE]\SharePoint Web /
[+] Impersonate [INSTANCE]\SharePoint Web /
[+] GenericRead [INSTANCE]\Team Foundation
[+] GenericWrite [INSTANCE]\Team Foundation
[+] Impersonate [INSTANCE]\Team Foundation
[+] TriggerEvent [INSTANCE]\Team Foundation
[+] GenericRead [INSTANCE]\Team Foundation
[+] GenericWrite [INSTANCE]\Team Foundation
[+] TriggerEvent [INSTANCE]\Team Foundation
[+] GenericRead DATUM1\jpeoples
```

Done.

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)

How to: Create a Global Group

Team Project Groups

Default Groups



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/g command: List the groups

[Example](#) [See Also](#) [Send Feedback](#)

Use **/g** to list the groups in a team project, in a team project collection, or across Team Foundation Server.

Required Permissions

To use the **/g** command, you must have the View collection-level information or the View instance-level information permission set to Allow, depending on whether you are using the **/collection** or **/server** parameter, respectively. To use the **/g** command within the scope of a single team project, you must have the View project-level information permissions set to Allow. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /g [scope] [/collection:CollectionURL] [/server:ServerUF
```

Parameters

Argument	Description
scope	Optional. Specifies the URI of the team project for which you want to display groups. To obtain the URI for a team project, open Team Explorer, right-click the team project, click Properties, and copy the entire entry for URL.
/ collection :CollectionURL	Required if /server is not used. Specifies the URL of the project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/ server :ServerURL	Required if /collection is not used. Specifies the URL of the application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

The **/g** command of the **TFSecurity** command-line utility displays information about every group within the selected scope. This scope can be the team project collection (**/server**) or the application-tier server (**/instance**). If used with the scope of a team project, it will display information only about the groups associated with that team project.

Example

The following example displays information for all the groups within a team project collection.

 **Copy Code**

```
>TFSSecurity /g /collection:CollectionURL
```

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)

How to: Create a Global Group

Team Project Groups



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/g+ command: Add a user or another group to an existing group

[See Also](#) [Send Feedback](#)

Use /g+ to add a user or another group to an existing group.

Required Permissions

To use the /g+ command, you must have the View collection-level information and Edit collection-level information or the View instance-level information and Edit instance-level information permissions set to Allow, depending on whether you are using the /collection or /server parameter, respectively. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /g+ groupIdentity memberIdentity [/collection:Collection
```

Parameters

Argument	Description
groupIdentity	Specifies the group identity. For more information on identity specifiers, see TFSSecurity Identity and Output Specifiers .
memberIdentity	Specifies the member identity. For more information on identity specifiers, see TFSSecurity Identity and Output Specifiers .
/collection: CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/server: ServerURL	Required if /collection is not used. Specifies the URI application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

You can also add users and groups to an existing group using Team Explorer. For more information, see [How to: Add Users to a Global Group](#).

Examples

The following example adds the Datum1 domain user John Peoples (Datum1\jpeoples) to the Team Foundation Administrators group.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

 **Copy Code**

```
>tfssecurity /g+ "Team Foundation Administrators" n:Datum1\jpeoples
```

Sample output:

 **Copy Code**

```
TFSSecurity - Team Foundation Server Security Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
The target Team Foundation Server is http://ADatumCorporation:8080/.  
Resolving identity "Team Foundation Administrators"...  
a [A] [INSTANCE]\Team Foundation Administrators  
Resolving identity "n:Datum1\jpeoples"...  
[U] DATUM1\jpeoples (John Peoples)  
Adding John Peoples to [INSTANCE]\Team Foundation Administrators...  
Verifying...
```

```
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1
```

```
DN:
```

```
Identity type: Team Foundation Server application group  
Group type: AdministrativeApplicationGroup  
Project scope: Server scope  
Display name: [INSTANCE]\Team Foundation Administrators  
Description: Members of this group can perform all operations on t
```

```
4 member(s):  
[U] Datum1\hholt (Holly Holt)
```

[U] Datum1\jpeoples (John Peoples)
[G] BUILTIN\Administrators (BUILTIN\Administrators)
s [A] [INSTANCE]\Team Foundation Service Accounts

Member of 2 group(s):

a [A] [Collection0]\Project Collection Administrators
e [A] [INSTANCE]\Team Foundation Valid Users

Done.

See Also

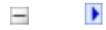
Other Resources

[Change groups and permissions with TFSecurity](#)

How to: Create a Global Group

Team Project Groups

Default Groups



Visual Basic C#

Visual C++

F#

HLSL

JScript

Visual Studio Application Lifecycle Management

/g- command: Remove a user or group

[See Also](#) [Send Feedback](#)

Use /g- to remove a user or a user group from an existing group.

Required Permissions

To use the /g- command, you must have the View collection-level information and Edit collection-level information or the View instance-level information and Edit instance-level information permissions set to Allow, depending on whether you are using the /**collection** or /**server** parameter, respectively. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /g- groupIdentity memberIdentity [/collection:Collection
```

Parameters

Argument	Description
groupIdentity	Specifies the group identity. For more information about identity specifiers, see TFSSecurity Identity and Output Specifiers .
memberIdentity	Specifies the member identity. For more information about valid identity specifiers, see TFSSecurity Identity and Output Specifiers .
/collection:CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/server:ServerURL	Required if /collection is not used. Specifies the URL of application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

You can also add users and groups to an existing group using Team Explorer. For more information, see [How to: Remove Users from a Team Project Group](#), [How to: Remove Users from a Default Group](#) or [How to: Remove Users from a Server-Level Group](#).

Examples

The following example removes the Datum1 domain user John Peoples (Datum1\jpeoples) from the Team Foundation Administrators group.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

 **Copy Code**

```
>tfssecurity /g- "Team Foundation Administrators" n:Datum1\jpeoples
```

Sample output:

 **Copy Code**

```
TFSSecurity - Team Foundation Server Security Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
The target Team Foundation Server is http://ADatumCorporation:8080/.  
Resolving identity "Team Foundation Administrators"...  
a [A] [INSTANCE]\Team Foundation Administrators  
Resolving identity "n:Datum1\jpeoples"...  
  [U] DATUM1\jpeoples (John Peoples)  
Removing John Peoples from [INSTANCE]\Team Foundation Administrators  
Verifying...
```

```
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1
```

```
DN:
```

```
Identity type: Team Foundation Server application group  
  Group type: AdministrativeApplicationGroup  
Project scope: Server scope  
  Display name: [INSTANCE]\Team Foundation Administrators  
  Description: Members of this group can perform all operations on t
```

```
3 member(s):  
  [U] Datum1\hholt (Holly Holt)
```

[G] BUILTIN\Administrators (BUILTIN\Administrators)
s [A] [INSTANCE]\Team Foundation Service Accounts

Member of 2 group(s):

a [A] [Collection0]\Project Collection Administrators
e [A] [INSTANCE]\Team Foundation Valid Users

Done.

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/gc command: Create a project-level group

[Example](#) [See Also](#) [Send Feedback](#)

Use **/gc** at a command prompt to create a project-level group. To create a project-level group from the user interface, see [Manage users or groups in TFS](#).

Required Permissions

To use the **/gc** command, you must have the Edit Project-Level Information permission for that team project set to Allow. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /gc Scope GroupName [GroupDescription] [/collection:Col]
```

Parameters

Argument	Description
Scope	The URI of the team project to which you want to add a new level group. To obtain the URI for a team project, connect to the project and open Team Explorer, hover over the name of the project, and read the address. Alternatively, connect to the project in Web Access and copy the URL.
GroupName	The name of the new group.
GroupDescription	A description of the project group. Optional.
/collection:CollectionURL	The URL of the team project collection. Required. The URL will be created within the team project collection. The format for the URL is http://ServerName:Port/VirtualDirectoryName/CollectionName .

Remarks

Run this command on an application-tier server for Team Foundation.

A project-level group is a security group for your team project. You can use project groups to grant read, write, and administrative permissions that meet the security requirements of your organization.

Example

The following example creates a group that is specific to the project that the URI "vstfs://Classification/TeamProject/00000000-0000-0000-0000-000000000000" specifies. The group is named "Test Group" and has the description "This group is for testing."

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

You must replace the placeholder GUID with the URI of the team project for which you want to create this group. To obtain the URI for a team project, open Team Explorer, right-click the team project, click Properties, and copy the entire value of the URL property.

After you run the command, you can verify the group in Team Explorer. Right-click the team project that you used in the command, click Team Project Settings, and then click Group Memberships. In the Project Groups on TeamProjectName dialog box, the Groups list includes Test Group .

Note

You can use the `/gc` command to create groups but not to add any users to the groups or assign any permissions. To change the membership of the group, see [/g+ command: Add a user or another group to an existing group](#) and [/g- command: Remove a user or group](#). To change the permissions for the group, see [/a+ command: Add permissions](#) and [/a- command: Remove a user or a group from membership in a group](#).

```
>TFSsecurity /gc "vstfs:///Classification/TeamProject/00000000-0000-
```

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/gcg command: Create a server or collection-level group

[Example](#) [See Also](#) [Send Feedback](#)

Use the **/gcg** command to create a server-level or collection-level group. To create a server-level or collection-level group from the user interface, see [Manage users or groups in TFS](#).

Required Permissions

To use the **/gcg** command, you must have the Edit project-level information permission for that team project set to Allow. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /gcg GroupName [GroupDescription] [/collection:Collectio
```

Parameters

Argument	Description
GroupName	The group name.
GroupDescription	A description of the group. Optional.
/collection: CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/Colle
/server: ServerURL	Required if /collection is not used. Specifies the URI application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

Server-level groups are created directly on the application tier and apply to all team project collections. Collection-level are created at the team project collection level. They apply to that collection and have implications for all team projects within the collection. In contrast, team project groups apply to a specific project within a collection but not any other projects in that collection. You can assign permissions to server-level groups so that members of those groups can perform tasks in Team Foundation Server (TFS) itself, such as creating team project collections. You can assign permissions to collection-level groups so that members of those groups can perform tasks across a team project collection, such as administering users.

Note

You can use the **/gcg** command to create groups, but you cannot use it to add any users to the groups or assign any permissions. For information about how to change the membership of a group, see [/g+ command: Add a user or another group to an existing group](#) and [/g- command: Remove a user or group](#). For information about how to change the permissions for the group, see [/a+ command: Add permissions](#) and [/a- command: Remove a user or a group from membership in a group](#).

Example

The following example creates a collection-level group that is named "Datum Testers" with the description "A. Datum Corporation Testers."

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
>TFSSecurity /gcg "Datum Testers" "A. Datum Corporation Testers" /cc
```

The following example creates a server-level group that is named "Datum Auditors" with the description "A. Datum Corporation Auditors."

Copy Code

```
>TFSSecurity /gcg "Datum Auditors" "A. Datum Corporation Auditors" ,
```

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/gd command: Delete a server or collection-level group

[Example](#) [See Also](#) [Send Feedback](#)

Use **/gd** to delete a server-level or collection-level group.

Required Permissions

To use the **/gd** command, you must have the View collection-level information and Edit collection-level information or the View instance-level information and Edit instance-level information permissions set to Allow, depending on whether you are using the **/collection** or **/server** parameter, respectively. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /gd groupIdentity [/collection:CollectionURL] [/server:s
```

Parameters

Argument	Description
groupIdentity	Specifies the group identity. For more information about identity specifiers, see Change groups and permissions in TFS Security .
/collection: CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/server: ServerURL	Required if /collection is not used. Specifies the URL of application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

You can also remove groups on Team Explorer. For more information, see [How to: Remove a Global Group](#) and [How To: Remove a Team Project Group](#).

Example

The following example deletes a group from the team project collection. The group is identified by "S-1-5-21-2127521184-1604012920-1887927527-588340", the security identifier (SID). For more information about finding the SID of a group, see [/im command: Display information about identities that compose direct membership](#). You can also use the friendly name to delete a group.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
>TFSSecurity /gd S-1-5-21-2127521184-1604012920-1887927527-588340 /c
```

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)

How to: Create a Global Group

Team Project Groups

Default Groups



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/gud command: Change the description for a server or collection-level group

[Example](#) [See Also](#) [Send Feedback](#)

Use **/gud** to change the description for a server-level or collection-level group.

Required Permissions

To use the **/gud** command, you must have the Edit project-level information permission set to Allow. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /gud GroupIdentity GroupDescription [/collection:Collect
```

Parameters

Argument	Description
GroupIdentity	Specifies the group identity. For more information about identity specifiers, see TFSSecurity Identity and Output Specifiers .
GroupDescription	Specifies the new description for the group.
/collection: CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/server: ServerURL	Required if /collection is not used. Specifies the URI application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

Example

The following example associates the description "The members of this group test the code for this project" with the group "Datum Testers."

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

 **Copy Code**

```
>TFSSecurity /gud "Datum Testers" "The members of this group test t
```

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)

Managing Groups

Default Groups

Global Groups

Team Project Groups



Visual Basic C#

Visual C++

F#

HLSL

JScript

Visual Studio Application Lifecycle Management

/gun command: Rename a group

[Example](#) [See Also](#) [Send Feedback](#)

Use **/gun** to rename a server-level or collection-level group.

Required Permissions

To use the **/gun** command, you must have the View collection-level information and Edit collection-level information or the View instance-level information and Edit instance-level information permissions set to Allow, depending on whether you are using the **/collection** or **/server** parameter, respectively. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /gun GroupIdentity GroupName [/collection:CollectionURL]
```

Parameters

Argument	Description
GroupIdentity	Specifies the group identity. For more information about identity specifiers, see TFSSecurity Identity and Output Specifiers .
GroupName	Specifies the new name of the group.
/collection: CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/server: ServerURL	Required if /collection is not used. Specifies the URI application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

Example

The following example renames the collection-level group "A. Datum Corporation Testers" to "A. Datum Corporation Test Engineers."

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

 **Copy Code**

```
>tfssecurity /gun "A. Datum Corporation Testers" "A. Datum Corporati
```

See Also

Concepts

[TFSSecurity Identity and Output Specifiers](#)

Other Resources

[Change groups and permissions with TFSSecurity](#)

How To: Rename a Team Project Group

How to: Rename a Global Group



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/i command: Display identity information for a specified group

[See Also](#) [Send Feedback](#)

Use */i* to display identity information for a specified group in a deployment of Team Foundation Server.

Required Permissions

To use the */i* command, you must have the View collection-level information or the View instance-level information permission set to Allow, depending on whether you are using the */collection* or */server* parameter, respectively. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /i Identity [/collection:CollectionURL] [/server:ServerURL]
```

Parameters

Argument	Description
Identity	The identity of the user or the application group. For information about identity specifiers, see TFSSecurity and Output Specifiers .
/ collection :CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/Colle
/ server :ServerURL	Required if /collection is not used. Specifies the URL application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

The **/i** command of the **TFSSecurity** command-line utility displays information about each group within the team project collection (/server) or the application-tier server (/instance). It does not display any membership information.

Examples

The following example displays identity information for the "Team Foundation Administrators" group.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
>tfssecurity /i "Team Foundation Administrators" /server:ServerURL
```

Sample output:

Copy Code

```
Resolving identity "Team Foundation Administrators"...
```

```
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1
```

```
DN:
```

```
Identity type: Team Foundation Server application group
```

```
Group type: AdministrativeApplicationGroup
```

```
Project scope: Server scope
```

```
Display name: Team Foundation Administrators
```

```
Description: Members of this application group can perform all pr
```

The following example displays identity information for the Project Collection Administrators group using the adm: identity specifier.

Copy Code

```
>tfssecurity /i adm: /collection:CollectionURL
```

Sample output:

 **Copy Code**

```
Resolving identity "adm:"...
```

```
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1
```

```
DN:
```

```
Identity type: Team Foundation Server application group
```

```
Group type: AdministrativeApplicationGroup
```

```
Project scope: Server scope
```

```
Display name: [DatumOne]\Project Collection Administrators
```

```
Description: Members of this application group can perform all pr
```

The following example displays identity information for the Project Administrators group for the "Datum" project by using the adm: identity specifier.

 **Copy Code**

```
>tfssecurity /i adm:vstfs:///Classification/TeamProject/ProjectGUID
```

Sample output:

 **Copy Code**

```
Resolving identity "adm:vstfs:///Classification/TeamProject/ProjectC
```

```
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1
```

```
DN:
```

```
Identity type: Team Foundation Server application group
```

```
Group type: AdministrativeApplicationGroup
```

```
Project scope: Datum
```

```
Display name: [Datum]\Project Administrators
```

```
Description: Members of this application group can perform all ope
```

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)

How to: Create a Global Group

Team Project Groups



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/im command: Display information about identities that compose direct membership

[See Also](#) [Send Feedback](#)

Use **/im** to display information about the identities that compose the direct membership of a group that you specify.

Required Permissions

To use the **/im** command, you must have the View collection-level information or the View instance-level information permission set to Allow, depending on whether you are using the **/collection** or **/server** parameter, respectively. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /im Identity [/collection:CollectionURL] [/server:Server]
```

Parameters

Argument	Description
Identity	The identity of the user or the group. For more information about identity specifiers, see TFSSecurity Identity and Specifiers .
/ collection :CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/ server :ServerURL	Required if /collection is not used. Specifies the URL of application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

The **/im** command of **TFSecurity** displays the direct members of the specified group only. This list includes other groups that are members of the specified group. However, the actual members of the member groups are not listed.

Examples

The following example displays direct membership identity information for the "Team Foundation Administrators" group in the domain "Datum1" at the fictitious company "A. Datum Corporation".

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
>tfssecurity /im "Team Foundation Administrators" /server:ServerURL
```

Sample output:

Copy Code

```
Resolving identity "Team Foundation Administrators"...
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1
DN:
Identity type: Team Foundation Server application group
Group type: AdministrativeApplicationGroup
Project scope: Server scope
Display name: Team Foundation Administrators
Description: Members of this application group can perform all privi

3 member(s):
  [U] Datum1\hholt (Holt, Holly)
  [G] BUILTIN\Administrators (BUILTIN\Administrators)
s [A] [InstanceName]\Team Foundation Service Accounts

Member of 2 group(s):
a [A] [DatumOne]\Project Collection Administrators ([DatumOne]\Proje
e [A] [InstanceName]\Team Foundation Valid Users

Done.
```

The following example displays identity information for the Project Collection Administrators group in the "DatumOne" team project collection in the domain "Datum1" at the fictitious company "A. Datum Corporation" by using the adm: identity specifier.

Copy Code

```
>tfssecurity /im adm: /collection:CollectionURL
```

Sample output:

Copy Code

```
Resolving identity "adm: "...
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1
DN:
Identity type: Team Foundation Server application group
Group type: AdministrativeApplicationGroup
Project scope: Server scope
Display name: [DatumOne]\Project Collection Administrators
Description: Members of this application group can perform all privi
5 member(s):
  [U] Datum1\jpeoples (Peoples, John)
  [U] Datum1\hholt (Holt, Holly)
  [G] BUILTIN\Administrators (BUILTIN\Administrators)
a [A] [InstanceName]\Team Foundation Administrators
s [A] [DatumOne]\Project Collection Service Accounts ([DatumOne]\Pro
Member of 1 group(s):
e [A] [DatumOne]\Project Collection Valid Users ([DatumOne]\Project
Done.
```

The following example displays identity information for the Project Administrators group for the "Datum" project in the "DatumOne" team project collection in the domain "Datum1" at the fictitious company "A. Datum Corporation" using the adm: identity specifier.

Copy Code

```
>tfssecurity /im adm:vstfs:///Classification/TeamProject/ProjectGUII
```

Sample output:

Copy Code

```
Resolving identity "adm:vstfs:///Classification/TeamProject/ProjectC
```

```
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXX
```

```
DN:
```

```
Identity type: Team Foundation Server application group
```

```
Group type: AdministrativeApplicationGroup
```

```
Project scope: Datum
```

```
Display name: [Datum]\Project Administrators
```

```
Description: Members of this application group can perform all opera
```

```
2 member(s):
```

```
  [U] Datum1\jpeoples (Peoples, John)
```

```
  [U] Datum1\hholt (Holt, Holly)
```

```
Member of 1 group(s):
```

```
e [A] [DatumOne]\Project Collection Valid Users ([DatumOne]\Project
```

```
Done.
```

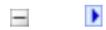
See Also

Other Resources

[Change groups and permissions with TFSecurity](#)

How to: Create a Global Group

Team Project Groups



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

`/imx` command: Display information about the identities that the expanded membership

[See Also](#) [Send Feedback](#)

Use `/imx` to display information about the identities that compose the expanded membership of a specified group.

Required Permissions

To use the `/imx` command, you must have the View collection-level information or the View instance-level information permission set to Allow, depending on whether you are using the `/collection` or `/server` parameter, respectively. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSSecurity /imx Identity [/collection:CollectionURL] [/server:ServerName]
```

Parameters

Argument	Description
Identity	The identity of the user or the group. For more information about identity specifiers, see TFSSecurity Identity and Specifiers .
/ collection :CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/ server :ServerURL	Required if /collection is not used. Specifies the URL of application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on an application-tier server for Team Foundation.

The **/imx** command of **TFSecurity** displays the expanded members of the specified group only. This list includes not only other groups that are members of the specified group but also the members of the member groups.

Examples

The following example displays expanded membership identity information for the "Team Foundation Administrators" group in the domain "Datum1" at the fictitious company "A. Datum Corporation".

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

Copy Code

```
>tfssecurity /imx "Team Foundation Administrators" /server:ServerURI
```

Sample output:

Copy Code

```
Resolving identity "Team Foundation Administrators"...
```

```
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1
```

```
DN:
```

```
Identity type: Team Foundation Server application group
```

```
Group type: AdministrativeApplicationGroup
```

```
Project scope: Server scope
```

```
Display name: Team Foundation Administrators
```

```
Description: Members of this application group can perform all priv:
```

```
10 member(s):
```

```
[U] Datum1\hholt (Holly Holt)
```

```
[U] Datum1\jpeoples (John Peoples)
```

```
[U] Datum1\tommyh (Tommy Hartono)
```

```
[U] Datum1\henrieta (Henriette Andersen)
```

```
[U] Datum1\djayne (Darcy Jayne)
```

```
[U] Datum1\aprilr (April Reagan)
```

```
[G] Datum1\InfoSec Secure Environment
```

```
[U] Datum1\nbento (Nuno Bento)
```

```
[U] Datum1\crisp (Cristian Petculescu)
```

```
[G] BUILTIN\Administrators (BUILTIN\Administrators)
s [A] [InstanceName]\Team Foundation Service Accounts
```

Member of 3 group(s):

```
a [A] [DatumOne]\Project Collection Administrators ([DatumOne]\Project
e [A] [DatumOne]\Project Collection Valid Users ([DatumOne]\Project
e [A] [InstanceName]\Team Foundation Valid Users
```

Done.

The following example displays identity information for the Project Collection Administrators group in the "DatumOne" team project collection in the domain "Datum1" at the fictitious company "A. Datum Corporation" using the adm: identity specifier.

Copy Code

```
>tfssecurity /imx adm: /collection:CollectionURL
```

Sample output:

Copy Code

```
Resolving identity "adm: "...
```

```
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-0-0-0-0-1
```

```
DN:
```

```
Identity type: Team Foundation Server application group
```

```
Group type: AdministrativeApplicationGroup
```

```
Project scope: Server scope
```

```
Display name: [DatumOne]\Project Collection Administrators
```

```
Description: Members of this application group can perform all priv:
```

```
6 member(s):
```

```
[U] Datum1\jpeoples (Peoples, John)
[U] Datum1\hholt (Holt, Holly)
[G] BUILTIN\Administrators (BUILTIN\Administrators)
a [A] [InstanceName]\Team Foundation Administrators
s [A] [InstanceName]\Team Foundation Service Accounts
s [A] [DatumOne]\Project Collection Service Accounts ([DatumOne]\Pro
```

```
Member of 1 group(s):
```

```
e [A] [DatumOne]\Project Collection Valid Users ([DatumOne]\Project
Done.
```

The following example displays identity information for the Project Administrators group for the "Datum" project in the "DatumOne" team project collection in the domain "Datum1" at the fictitious company "A. Datum Corporation" using the adm: identity specifier.

Copy Code

```
>tfssecurity /imx adm:vstfs:///Classification/TeamProject/ProjectGU
```

Sample output:

Copy Code

```
Resolving identity "adm:vstfs:///Classification/TeamProject/Project(
SID: S-1-9-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXXXXX-XXXXXXX
DN:
Identity type: Team Foundation Server application group
Group type: AdministrativeApplicationGroup
Project scope: Datum
Display name: [Datum]\Project Administrators
Description: Members of this application group can perform all opera
2 member(s):
  [U] Datum1\jpeoples (Peoples, John)
  [U] Datum1\hholt (Holt, Holly)
Member of 2 group(s):
e [A] [DatumOne]\Project Collection Valid Users ([DatumOne]\Project
e [A] [InstanceName]\Team Foundation Valid Users
Done.
```

For more information about the output specifiers, such as [G] and [U], see [TFS Security Identity and Output Specifiers](#).

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)

How to: Create a Global Group

Team Project Groups



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

/m command: Check explicit and implicit group membership

[See Also](#) [Send Feedback](#)

Use **/m** to check explicit and implicit group membership information for a specified group or user.

Required Permissions

To use the **/m** command, you must be a member of the Team Foundation Administrators security group. For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you are logged on with administrative credentials, you must open an elevated Command Prompt to perform this function.

```
TFSSecurity /m GroupIdentity [MemberIdentity] [/collection:Collectio
```

Parameters

Argument	Description
GroupIdentity	Specifies the group identity. For more information on identity specifiers, see TFSSecurity Identity and Output Specifiers .
MemberIdentity	Specifies the member identity. By default, the value of this argument is the identity of the user who is running the command. For more information on valid identity specifiers see TFSSecurity Identity and Output Specifiers .
/collection: CollectionURL	Required if /server is not used. Specifies the URL of project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName
/server: ServerURL	Required if /collection is not used. Specifies the URL of application-tier server in the following format: http://ServerName:Port/VirtualDirectoryName

Remarks

Run this command on the local application-tier computer.

The **/m** command of the **TFSsecurity** command-line utility checks both direct and extended memberships.

Examples

The following example verifies whether the user "Datum1\jpeoples" belongs to the Team Foundation Administrators server-level group.

Note

The examples are for illustration only and are fictitious. No real association is intended or inferred.

 **Copy Code**

```
>TFSSecurity /m "Team Foundation Administrators" n:Datum1\jpeoples /
```

Sample output:

 **Copy Code**

```
TFSSecurity - Team Foundation Server Security Tool  
Copyright (c) Microsoft Corporation. All rights reserved.
```

```
The target Team Foundation Server is http://ADatumCorporation:8080/.  
Resolving identity "Team Foundation Administrators"...  
a [A] [INSTANCE]\Team Foundation Administrators  
Resolving identity "n:Datum1\jpeoples"...  
[U] DATUM1\jpeoples (John Peoples)  
Checking group membership...
```

```
John Peoples IS a member of [INSTANCE]\Team Foundation Administrator
```

```
Done.
```

See Also

Other Resources

[Change groups and permissions with TFSecurity](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Use Team Foundation version control commands

[Send Feedback](#)

You can use version control commands to do nearly all tasks you can do in Visual Studio, and also several tasks that can't be done in Visual Studio. You can use the **tf.exe** tool to run version control commands from a command prompt or within a script.

What do you want to do?

- [Run a command](#)
 - [Set up your dev machine](#)
 - [Develop your app](#)
 - [Suspend your work](#)
 - [Contribute your work](#)
 - [Manage files and solve problems](#)
 - [Isolate risk](#)
- [Understand command syntax](#)
- [Specify the items affected by a command](#)
 - [Use an itemspec argument to specify affected items](#)
 - [Use a versionspec argument to specify affected versions of items](#)
- [Use options to modify how a command functions](#)

- [Use the /noprompt option to suppress requests for data input and redirect output data to the command prompt window](#)
- [Use /login option to specify credentials when running a command](#)
- [Use the /lock option to apply or remove a lock](#)
- [Use option shortcuts](#)
- [Understand exit codes](#)

Run a command

To launch the Visual Studio command prompt, from Windows Start, choose Microsoft Visual Studio 2012, Visual Studio Tools, and then choose one of the Command Prompt shortcuts.

In most cases, you run the version control command in the context of a directory that is mapped in the workspace. For example, `$/SiteApp/Main/` is mapped to `c:\code\SiteApp\Main\`. To get the latest version of all items in the workspace:

Copy Code

```
c:\code\SiteApp\Main\SolutionA>tf get
```

Note

We are not currently republishing some of the topics. However, you can read the Visual Studio 2010 version of these topics.

Set up your dev machine and manage workspaces

Your workspace is a local copy of your team's codebase. Because it is a local copy on your dev machine, you can develop and test your code in isolation until you are ready to check in your work. Here are some commands to manage your workspace:

Proxy Command (Visual Studio 2010)

WorkFold Command (Visual Studio 2010)

Workspace Command (Visual Studio 2010)

Workspaces Command (Visual Studio 2010)

See also: [Set up Team Foundation Version Control on your dev machine, Create and work with workspaces](#)

■ **Develop your app**

Use these commands to develop your app under version control with your team:

[Add command](#)

Adds files and folders to version control.

[Checkout \(or Edit\) command](#)

Checks out a file and changes its pending change status to "edit".

Delete Command (Visual Studio 2010)

[Get command](#)

Gets (downloads) the latest or a specified version of one or more files or folders from Team Foundation Server to the workspace.

Rename Command (Visual Studio 2010)

[Status command](#)

Displays information about pending changes to files and folders items in one or more workspaces, or in a shelve set.

[Undo command](#)

Discards one or more pending changes to files or folders.

Undelete Command (Visual Studio 2010)

See also: Develop your app in Team Foundation version control

■ **Suspend your work**

For a variety of reasons, sometimes you need to set aside some or all of your work in progress. To suspend and resume your work, and to manage your shelve sets, use these commands:

Shelve Command (Visual Studio 2010)

Shelvesets Command (Visual Studio 2010)

Unshelve Command (Visual Studio 2010)

See also: Suspend your work and manage your shelvesets.

☐ **Contribute your work**

Here's how to check in your code to the team's codebase:

[Checkin command](#)

Checks in pending changes to files or folders to the server.

See also: Check in your work to the team's codebase

☐ **Manage files and solve problems**

View and Manage Version Control Files and Folders
Properties Command (Visual Studio 2010)

Property Command (not documented)

Dir Command (Visual Studio 2010)

Destroy Command (Visual Studio 2010)

LocalVersions Command (Visual Studio 2010)

See also: Use Source Control Explorer to manage files under version control

View and Manage Past Versions
Changeset Command (Visual Studio 2010)

[History command](#)

Displays the revision history of one or more files or folders.

Label Command (Visual Studio 2010)

Labels Command (Visual Studio 2010)

Rollback Command (Visual Studio 2010)

Unlabel Command (Visual Studio 2010)

View Command (Visual Studio 2010)

See also: View and manage past versions

Compare Folders and Files

 Difference Command (Visual Studio 2010)

 Folderdiff Command (Visual Studio 2010)

 See also: View and manage past versions

Resolve File Conflicts

 Resolve Command (Visual Studio 2010)

 See also: Resolve Team Foundation Version Control conflicts.

Work with Version Control Locks

 Lock Command (Visual Studio 2010)

 See also: Work with version control locks.

☐ **Isolate risk**

Use the following commands to isolate risk using branches:

Branch Command (Visual Studio 2010)

Branches Command (Visual Studio 2010)

Merge Command (Visual Studio 2010)

Merges Command (Visual Studio 2010)

See also: Use branches to isolate risk in Team Foundation Version Control.

☐ **Administer Version Control**

Use the following commands to administer your version control system:

Configure Command (Visual Studio 2010)

Permission Command (Visual Studio 2010)

See also: Managing Team Foundation Version Control (Visual Studio 2010).

☐ **Get Help on Version Control Commands**

Use the following commands to get more information about version control commands:

Help Command (Visual Studio 2010)

Msdn Command (Visual Studio 2010)

▣ Understand command syntax

The syntax of each command appears at the top of each reference topic.

▣ Required and optional arguments

Non-bracketed arguments are required. **[Brackets]** indicate optional arguments that are not required to complete a command. However, some optional arguments have defaults that are applied to the command even if you do not specify the option.

▣ Exclusive arguments

When options are separated by a pipe (`|`), you can specify one of the options.

▣ Verbatim and replaceable arguments

Bold items are options that you include verbatim. Italicized items are arguments that you must replace with actual characters to perform a command.

▣ Command Shortcuts and Aliases

Some commands support shortcuts. For example, you can call the Delete command with either **tf delete** or **tf del**.

▣ Example

For example, the Checkout command:

 **Copy Code**

```
tf checkout [/lock:( none|checkin|checkout)] [/recursive] itemspec |
```

Let's review the arguments from this example:

- `itemspec`: You must replace this argument with an [itemspec](#) that specifies the items you are checking out.
- **`/lock:(none|checkin|checkout)`**: You are not required to specify the **`/lock`** option. If you do not specify it, then the system by default specifies **`/lock:none`**. Otherwise, you can specify one of the lock options.
- The following arguments are optional and if you do not supply them, none of their effects apply to the command:
 - **`/recursive`**: If you want to recursively check out multiple items in a folder, you must specify this option verbatim.
 - **`/login:username,password`**: If you want to run the command as another user, you must specify the **`/login`** option verbatim, replace username with the name of the user, and if necessary, you can supply the password.

Specify the items affected by a command

You can use itemspecs and versionspecs to specify which items are affected by a command.

Use an itemspec argument to specify affected items

You use an itemspec (item specification) to specify the items affected by a command. You can specify items either on a client machine or on your Team Foundation Server. You can use wildcard characters such as * and ?.

Client itemspec arguments

A client itemspec argument specifies a path to items on a client machine such as a folder (for example, `c:\code\SiteApp\Main\SolutionA\`) a file (for example, `c:\code\SiteApp\Main\SolutionA\Project1\program.cs` or multiple files (for example, `c:\code\SiteApp\Main\SolutionA*.cs`). You can also specify UNC paths such as `\\myshare\code\SiteApp\Main`.

Server itemspec arguments

A server itemspec argument specifies a path to items on your Team Foundation Server such as a folder (for example, `$/SiteApp/Main/SolutionA`) a file (for example, `$/SiteApp/Main/SolutionA/Project1/program.cs` or multiple files (for example, `$/SiteApp/Main/SolutionA/*.cs`).

You typically use server itemspec arguments when you need run a command on items not on the client machine. For example, you are working on a dev machine and need to get some revision history data about some items that are in a team project collection you don't work in:

Copy Code

```
c:\>tfs history /collection:http://fabrikam-3:8080/tfs/DefaultCollect
```

```
$/SiteApp/Main/SolutionA/Project1/* /recursive /noprompt
```

Multiple itemspec arguments

For some commands, you can specify multiple itemspec arguments. For example:

Copy Code

```
c:\code\SiteApp\Main\SolutionA\Project1\>tf checkout program1.cs pro
```

Checks out program.cs and program2.c.

Use a versionspec argument to specify affected versions of items

You use a versionspec (version specification) to specify the version of items affected by a command. To provide a versionspec you can:

- Use the **/version** option. For example: **/version:C44**.
- Append the versionspec to an itemspec with a semicolon. For example: **program1.cs;C44**.

When you use the

[History command](#) or the Difference Command, you can specify a range of versions by separating the versions with a tilde (~). For example:

Copy Code

```
c:\code\SiteApp\Main\SolutionA>tf history /noprompt * /recursive /v:
```

Use the following syntax to specify a versionspec.

Type	Syntax	Description
------	--------	-------------

Specifies items based on a changeset number. If a scope was not modified in the specified changeset, it takes the latest version of the item that occurred before the specified changeset.

Tip

You can omit the C if you specify only a number.

Examples

Copy Code

```
c:\code\SiteApp\Main>tf get readme.txt /v
```

Changeset [C]n

-- or --

Copy Code

```
c:\code\SiteApp\Main>tf get readme.txt /v
```

-- or --

Copy Code

```
c:\code\SiteApp\Main>tf get readme.txt;8
```

If readme.txt was modified in changeset 8, gets that version of the file. Otherwise, gets the most recent version of the file before version 8.

Specifies items to which label was applied.

Examples

Copy Code

		<code>c:\code\SiteApp\Main>tf get readme.txt;LJ</code>
Label	Llabel	Gets the version of readme.txt that was labeled Llabel  Copy Code
		<code>c:\code\SiteApp\Main>tf get /version:LLas</code>
		Retrieves the version of all labeled items (and del not labeled) in the workspace as they existed whe labeled as LastKnownGood was created, for exan part of an automated build process.
		Specifies a changeset created on a specified date .
	Dyyyy-mm-ddTxx:xx	Examples
	-or-	 Copy Code
	Dmm/dd/yyyy	<code>c:\code\SiteApp\Main>tf get /version:D200</code>
	-or-	Updates the workspace to match the codebase as 3/22/2004 at 00:00 (midnight).
Date and time	Any .NET Framework-supported format.	 Copy Code
	-or-	<code>c:\code\SiteApp\Main>tf get /version:D200</code>
	Any of the date formats supported on the local machine.	Updates the workspace to match the codebase as 3/22/2004 at 09:00 (9 AM). For more information about .NET Framework-su and time formats see DateTime and Standard Dat Format Strings.

Workspace W (current)	Specifies the version in your workspace.
Workspace W workspacename; (specified) workspaceowner	Specifies the version in a specified workspace. For example: WResolveRIFConflicts;PeterW
Tip T	Specifies the most recent version.

▣ Use options to modify how a command functions

You can use some common options to modify how a command functions.

▣ Use the **/noprompt** option to suppress requests for data input and redirect output data to the command prompt window

Use the **/noprompt** option to suppress requests for data input and redirect output data to the command prompt window. This option can be useful when you need to use version control commands in a script because the command proceeds without intervention by a user, and the data is available for the script to perform operations such as parsing or capturing.

When you use this option, the system:

- Suppresses all requests for input:
 - Questions are not asked in the command prompt window. For example, when you use the

[Undo command](#) with this option, the system does not prompt you to confirm if you want to proceed with undoing the changes.

- Windows and dialog boxes are not presented. For example, you use this option with the [Checkin command](#). Instead of displaying the Check In dialog box for you to confirm your options (which items you want to check in or which work items to associate), the system proceeds with the check in without confirmation.
- Redirects output data to the command prompt. For example, you use this option with the [History command](#). The data is displayed in the command prompt window instead of the History window.

▣ Use **/login** option to specify credentials when running a command

Use the **/login** option to specify the Team Foundation Server user account to run a command. This option can be useful when you are working at the machine of another team member.

For example, Julia is working with Peter at his dev machine. She uses the Lock command to unlock a file that she locked earlier:

Copy Code

```
c:\code\SiteApp\Main> tf lock /lock:none program.cs /login:JuliaI, Ju
```

If she wants to avoid having her password appear in the command prompt, she can enter the command without the password:

Copy Code

```
c:\code\SiteApp\Main> tf lock /lock:none program.cs /login:JuliaI
```

After she enters this command, the system then prompts her to type her password in a dialog box that masks her input.

Use the **/lock** option to apply or remove a lock

Important

As a best practice, use the **/lock** option with discretion and notify your teammates why you are locking an item, and when you plan to remove the lock.

Use the **/lock** option to apply or remove a lock at the same time you run another command such as Add or Edit.

Copy Code

```
/lock:(none|checkin|checkout)
```

- **None:** No lock is placed on an item. If a lock is already in place, it is removed.

- **Checkin** or **Checkout**: Applies a lock. See Understand lock types.

📌 **Note**

In a few cases, the lock operation can fail:

- If any other users have locked any of the specified items, the lock operation will fail.
- The system ignores this switch if there is already a pending change to the item. In this case, you must use the Lock Command to change a lock on an item.

☰ **Use option shortcuts**

You can abbreviate the following options.

Option	Option Alias
/comment	-C
/computer	-M
/delete	-D
/force	-P
/format	-F
/help	-?, -H

/lock	-K
/login	-Y
/newname	-N
/noprompt	-I
/owner	-O
/recursive	-R
/server	-S
/slotmode	-X
/template	-T
/user	-U
/version	-V
/workspace	-W

Understand exit codes

Version control commands return the following exit codes:

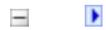
Exit Code	Definition
0	Success.
1	Partial success; this means at least something, or possibly everything, failed to succeed.
2	Unrecognized command.
100	Nothing succeeded.

For example:

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA\Project1\>tf checkout program1.cs pro
```

If one of the files you are trying to check out does not exist on the server, the command returns **1** to indicate partial success.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Add command

[Send Feedback](#)

Adds files and folders to version control.

Tip

 Before you add files to version control, you should first set up the workspace on your dev machine. See [Workspace Command](#) or [Set up Team Foundation Version Control on your dev machine](#).

Requirements: See [Permission reference for Team Foundation Server](#).

```
tf add itemspec [/lock:(none|checkin|checkout)] [/encoding:filetype:  
[/noprompt] [/recursive] [/noignore] [/login:username,[password]]
```

Parameters

Parameter	Description
/encoding: filetype	Ignore this parameter.
itemspec	Specifies the scope of the items to add. You can specify more than one itemspec argument. For syntax, see Use Team Foundation version control commands .
/lock	Applies or removes a lock. See Use Team Foundation version control commands .
/login: username, [password]	Specifies the user account to run the command. See Use Team Foundation version control commands .
	By default certain

/noignore

types of files (for example, .dll files) are ignored by version control. The rules in a .tfigure files apply to the **Add** command when you specify a wildcard in your itemspec. To override the application of the rules in this case, specify **/noignore**.

You can configure which kinds of files are ignored using a .tfigure file (see Add Files: .tfigure file).

/noprompt

Suppresses the display of windows and dialog boxes and redirects output data to the command prompt. See [Use Team Foundation version control commands](#).

/recursive

Recursively adds items in the specified directory and any subdirectories.

Examples

In all the following examples, assume that `$/SiteApp/Main/` is mapped to `c:\code\SiteApp\Main\` in the workspace.

Add all new files in a local workspace

New files in a local workspace are automatically detected. You can promote these newly detected files to your pending changes.

Copy Code

```
c:\code\SiteApp\Main\SolutionA\Project1>tf add
```

Adds the latest versions of all items (except those that are ignored) in a local workspace.

Copy Code

```
c:\code\SiteApp\Main\SolutionA\Project1>tf add /noignore
```

Adds the latest versions of all items in a local workspace.

Add individual items

Copy Code

```
c:\code\SiteApp\Main>tf add program1.cs program2.c
```

Adds the files `program1.cs` and `program2.c`.

Recursively add all items of a specific type

Copy Code

```
c:\code\SiteApp\Main>tf add *.cs /recursive
```

Adds all C# code files (.cs) in the current directory and any subdirectories.

Work in Visual Studio

- Add files to the server Use Visual Studio to add files to the server.

☐ Tips

- 💡 The results of this command are queued as pending changes (see [Status command](#)) and do not take effect on the server until you check in (see [Checkin command](#)).
 - 💡 If you need to set aside changes (and perhaps also want to clean your workspace for another task), use the Shelve Command.
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Checkin command

[Send Feedback](#)

Checks in your pending changes to files or folders to the server.

Almost every change that you make to the files on your dev machine is stored in your workspace as a pending change until you check it in. When you check in your changes, they are stored as a changeset on the server. Although the **Checkin** command provides a different user interface than the one you can use in Visual Studio (see Check in your work to the team's codebase), the process is fundamentally the same.

Requirements: See [Permission reference for Team Foundation Server](#).

```
tf checkin [/author:author name] [/comment:("comment"@comment file)  
[/noprompt] [/notes:("Note Name"="note text"@notefile)]  
[/override:(reason"@reasonfile)] [/recursive] [/saved] [/validate] |
```

```
tf checkin /shelveset:shelvesetname[;shelvesetowner] [/bypass] [/no
```

Parameters

Parameter	Description
/author :author name	Identifies the author of the pending changes so that one user can check in changes on behalf of another user. Requires the CheckinOther permission. See Permission reference for Team Foundation Server
/bypass	Bypasses a gated check-in requirement. For more information, see Check in to a folder that is controlled by a gated check-in build process.
/collection :TeamProjectCollectionUrl	If you use the /shelveset option, the /collection option specifies the URL of the team project collection that contains the shelveset. For example: <code>http://myserver:8080/tfs/DefaultCollection</code>
	By default, the team project collection is presumed to be the one that contains the workspace that maps the current directory.
/comment	Associates a comment with the changeset using one of the following arguments: <ul style="list-style-type: none">• Comment: A user-provided comment about the check-in.

- **@comment file:** The path to a file on disk that contains the comment for the check-in.

/force

Forces a check-in on items with pending edits even when there are no content changes in the file.

itemspec

Specifies the scope of the items to check in from the user's workspace. You can specify more than one Itemspec argument. For syntax, see [Use Team Foundation version control commands](#).

/login:username,[password]

Specifies the user account to run the command. See [Use Team Foundation version control commands](#).

/new

The selected state of each pending change (as shown in the Check In dialog box), the comment, associated work items, check-in notes, and check-in policy override reason are stored on your dev machine as pending changes until you check them in. The **/new** option clears this check-in metadata before you check in.

This option and the behavior it modifies have no effect when you use the **/noprompt** option.

/noautoresolve

By default, the system automatically attempts to AutoResolve All (see Resolve Team Foundation Version Control

conflicts). Specify this option to disable this default behavior.

/noprompt

Suppresses the display of windows and dialog boxes (such as the Check In dialog box) and redirects output data to the command prompt. See [Use Team Foundation version control commands](#).

Provides one or more check-in notes to associate with the changeset using one of the following arguments:

- NoteFieldName=NoteFieldValue: Sets the value of the check-in note field. You can provide multiple, semicolon-separated "field=value" expressions.
- Notefile: The user-provided path of a file on disk that contains check-in note field names and values in the format of "field=value". A semicolon separated note tile can span multiple lines, for example:

/notes

Field1=Value1;

Field2=First line of Value2

Second line Value2;

Field3=Value3;

Overrides a check-in policy using one of the following arguments:

/override

- **reason:** A user-provided reason why the check-in policy is being ignored.
- **Reasonfile:** The path to a file that contains a user-provided description of the reason why the check-in policy is being ignored.

/recursive

Recursively checks in items in the specified directory and any subdirectories.

/saved

Ignore this parameter.

/shelveset:shelvesetname[;owner]

Specifies a shelveset to check in. The optional owner argument is used to specify a shelveset that the current user does not own.

 **Note**

After you check in the shelveset, the system deletes it.

Tests whether the check in would succeed without checking in the files. The system evaluates check-in policies, check-in notes and lists conflicts.

 **Note**

/validate

If you do not specify the **/noprompt** option, you must choose the Check In button on the Check In dialog box to validate the check in. After you choose this button, the system will not check in

the files.

Examples

Check in all pending changes in the current workspace

Copy Code

```
c:\code\SiteApp\Main>tf checkin
```

Displays the Check In dialog box, which displays all pending changes in the current workspace. You can use the Check In dialog box to select or clear the pending changes you want to check in, add a comment, associate work items, and perform other tasks and then choose the Check In button when you are ready to proceed.

Check in all pending changes with a comment

Copy Code

```
c:\code\SiteApp\Main>tf checkin /comment:"Re-implemented Pi calculat
```

Checks in all pending changes in the current workspace and provides a comment to help your teammates understand the purpose of your changes.

Check in a change to a single item without using the Check In dialog box

Copy Code

```
c:\code\SiteApp\Main>tf checkin program.cs /noprompt
```

Checks in your pending changes to program.cs. The Check In dialog box is not displayed, and if any conflicts block the check in, the system does not display the conflicts window.

Work in Visual Studio

- Check in your work to the team's codebase Use Visual Studio to check in your changes to the server.

☐ Tips

- 💡 To set aside changes (and perhaps also want to clean your workspace for another task), use the **Shelve Command**.
 - 💡 If conflicts block your check-in, you can use the **Resolve Command** to resolve them.
 - 💡 If a machine and user account do not have a workspace mapped to the Team Project Collection that contains the shelve set, you can use the **/shelve set** and **/collection** options to check in a shelve set.
-



Visual Basic C#

Visual C++

F#

HLSL

JScript

Visual Studio Application Lifecycle Management

Checkout (or Edit) command

[Send Feedback](#)

Checks out a file and changes its pending change status to Edit. You can call this command using either **Checkout** or **Edit**.

Requirements: See [Permission reference for Team Foundation Server](#).

```
tf checkout [/lock:(none|checkin|checkout)] [/recursive] [/encoding:
```

Parameters

Parameter	Description
/encoding	Ignore this parameter.
itemspec	Specifies the scope of the items to check in. For syntax, see Use Team Foundation version control commands .
/lock	Applies or removes a lock. See Use Team Foundation version control commands .
/login:username, [password]	Specifies the user account to run the command. See Use Team Foundation version control commands .
/recursive	Recursively checks out items in the specified directory and any subdirectories.

Examples

Check out a single item

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA\Project1\>tf checkout program.cs
```

Checks out program.cs.

Check out two items

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA\Project1\>tf checkout program1.cs pro
```

Checks out the files program1.cs and program2.c.

Work in Visual Studio

- Check out and edit files Use Visual Studio to check out files.

☐ Tips

- 💡 If you are beginning a new task, it's probably a good idea for you to download the latest files from the server before you check out files and begin your work. See

[Get command](#).

- 💡 When you begin editing a file in a local workspace, it is automatically checked out for you.
 - 💡 You can view a list of your pending changes:
 - And work with related data such as Comments and Associated Work Items in the Check In dialog box using the [Checkin command](#).
 - At the command prompt by using the [Status command](#).
 - 💡 When you are ready to check in your changes to the team's codebase on the server, use the [Checkin command](#).
 - 💡 If you need to set aside changes (and perhaps also want to clean your workspace for another task), use the Shelve Command.
 - 💡 If you use a server workspace, files that you have not checked out are read-only.
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Get command

[Send Feedback](#)

Gets (downloads) either the latest version or a specified version of one or more files or folders from Team Foundation Server to the workspace. Although the **Get** command provides a different user interface than Visual Studio (see Download (get) files from the Server), the process is fundamentally the same.

Requirements: See [Permission reference for Team Foundation Server](#).

```
tf get [itemspec] [/version:versionspec] [/all] [/overwrite] [/force]
[/recursive] [/preview] [/noautoreresolve] [/noprompt]
[/login:username, [password]]
```

Parameters

Parameter	Description
/all	<p>If you use a local workspace (recommended), then it's unlikely you will have to use this option.</p>
	<p>You can use this option to restore an item that you have accidentally deleted from a server workspace.</p>
	<p>Your Team Foundation Server maintains an internal record of all the items the workspace contains, including the version of each. By default, when you get files, if the internal record on the server indicates the workspace already has the version you are</p>

getting, then it does not retrieve the item. This option gets the items regardless of the data contained in this internal record.

/force

Combines **/all** and **/overwrite**.

itemspec

Specifies the scope of the items to get. You can specify more than one itemspec argument. If no itemspec is provided, the system recursively gets all items in the current workspace.

For syntax, see [Use Team Foundation version control commands](#).

/login:username,

Specifies the user account to run the command. See [Use Team](#)

[password]

[Foundation
version control
commands.](#)

/noautoreresolve

By default, the system automatically attempts to AutoResolve All (see Resolve Team Foundation Version Control conflicts). Specify this option to disable this default behavior.

/noprompt

Suppresses the display of windows and dialog boxes and redirects output data to the command prompt. See [Use Team Foundation version control commands.](#)

If you use a local workspace (recommended), then it is unlikely you will have to

use this option.

/overwrite

By default, the system does not retrieve an item if it is writable (that is, if its read-only attribute is cleared) on the client machine. This option overrides the default behavior and overwrites a writable item, unless the item is checked out.

/preview

Displays what would occur, without actually performing the **Get** operation.

/recursive

Recursively gets items in the specified directory and any subdirectories. If you do not specify an itemspec, then this option is implied.

See phkelley's

/remap

blog: tf get
/remap.

Specifies the maximum version, or the minimum and the maximum versions, to display in the history data. The

/version:version-spec default is /version:T (the latest version).

For syntax, see [Use Team Foundation version control commands](#).

Examples

In all the following examples, assume that `$/SiteApp/Main/` is mapped to `c:\code\SiteApp\Main\` in the workspace.

Get the latest version of all items in a workspace

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA>tf get
```

Gets the latest versions of all items in the workspace. For example, the above command would recursively get all files in `$/SiteApp/Main/` including all its child folders.

Recursively get the latest version of items of a certain type in a folder

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA\Project1>tf get *.cs /recursive
```

Gets the latest version of all C# (.cs) files in `c:\code\SiteApp\Main\SolutionA\Project1`.

Get the latest version of a file

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA\Project1>tf get program.cs
```

Gets the latest version of `program.cs` in `Project1`.

Get a specific version of a file

Copy Code

```
c:\code\SiteApp\Main\SolutionA\Project1>tf get program.cs;8
```

Gets version 8 of program.cs in Project1.

▣ **Get the latest version of two files**

Copy Code

```
c:\code\SiteApp\Main\SolutionA\Project1>tf get file1.cs file2.cs
```

Gets the latest version of file1.cs and file2.cs in Project1.

▣ **Synchronize a workspace to match a version of the team's codebase**

Copy Code

```
c:\code\SiteApp\Main>tf get /v:15
```

Synchronizes the workspace to match the codebase as it existed when changeset 15 was created:

- The name and content of every item in the workspace is changed to match the state it was in on the server.
- Items that were deleted after that changeset are restored to the workspace.
- Items that were added after that changeset are deleted from the workspace.

▣ **Synchronize a workspace to match a labeled version of the team's codebase**

Copy Code

```
c:\code\SiteApp\Main>tf get /v:LLastKnownGood
```

Synchronizes the workspace to match the items in the codebase that are labeled LastKnownGood:

- The name and content of every labeled item in the workspace is changed to match the state it was in on the server.
- Labeled items that were deleted are restored to the workspace.
- Items that are not labeled on the server are deleted from the workspace.

Work in Visual Studio

- Download (get) files from the Server Use Visual Studio to get files and folders.

☐ Tips

- 💡 If you are beginning a new task, it's probably a good idea for you to run `tf get` (you can do so from any directory in your workspace) to download the latest files from the server before you check out files and begin your work.
- 💡 As needed, this command creates folders on disk to contain the child items that the command is downloading.
- 💡 You can view information about the history of an item using the

[History command](#) and the Changeset Command.

- 💡 If you are concerned about what changes might occur to the files in the workspace, you can use the `/preview` option to see the changes that would occur without actually implementing them.
 - 💡 Conflicts could block your `get`. A typical cause of conflicts is trying to get an item on which you have pending changes. You can use the Resolve Command to resolve these conflicts.
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

History command

[Send Feedback](#)

Displays the revision history of one or more files or folders. The data is displayed in either the History window, or if **/noprompt** option is specified, at the command prompt.

Requirements: See [Permission reference for Team Foundation Server](#).

```
tf hist[ory] itemspec [/version:versionspec]
[/stopafter:number] [/recursive] [/user:username]
[/format:(brief|detailed)] [/slotmode] [/itemmode] [/noprompt]
[/login:username,[password]] [/sort:ascending,descending]
[/collection:TeamProjectCollectionUrl]
```

Parameters

Parameter	Description
/collection: TeamProjectCollectionUrl	<p>Specifies the URL of the team project collection that contains the items. For example: http://myserver:8080/tfs/DefaultCollection.</p> <p>By default, the team project collection is presumed to be the one that contains the workspace that maps the current directory.</p>
/format	<p>Specifies how much detail to display about each changeset when the /noprompt option is specified:</p> <ul style="list-style-type: none">• Brief (default): Displays one line about each changeset that includes: ID number, changes made, user who made the changes, date, and comment. Some of the data may be truncated.• Detailed: Displays a full description of each changeset. In addition to the above information, this option displays additional data such as date with time, items changed, check-in notes, and check-in policy warnings. <p>You cannot combine this option with the /slotmode option. See Matt Mitrik:</p>

/itemmode	Changing to Slot Mode in TFS 2010 Version Control.
itemspec	Specifies the items for which to display history. For syntax, see Use Team Foundation version control commands .
	This option:
	<ul style="list-style-type: none">• Suppresses the display of windows and dialog boxes (such as the History window) and redirects output data to the command prompt. See Use Team Foundation version control commands.• Does not display the history of revisions that occurred before an item was moved, renamed, branched, or merged.
/login:username, [password]	Specifies the user account to run the command. See Use Team Foundation version control commands .
/recursive	Recursively retrieves historical data on items in the specified directory and any subdirectories.
/slotmode	You cannot combine this option with the /itemmode option. See Matt Mitrik: Changing to Slot Mode in TFS 2010 Version Control.

Sorts the revision history for one or more files or folders in either of the following orders:

/sort

- **Ascending:** from the oldest to the most recent revision.
- **Descending** (default): from the most recent to the oldest revision.

You must use the **/noprompt** option with this option.

/stopafter:number

Specifies the maximum number of changesets to display in the history data.

/user:username

Filters the historical data to show changes made by the specified user. An asterisk (*) symbol includes data on changes from all users (the default).

Specifies one of the following limits on the history data:

- The maximum version
- The minimum and the maximum versions using the range ~ syntax.

/version:versionspec

The default is **/version:W** (the version in the workspace).

For syntax, see [Use Team Foundation version control commands](#).

You cannot combine this option with the

| **/slotmode** option.

Examples

Get history of a single file

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA\Project1>tf history program2.cs
```

Displays all changes made to program.cs in the History window.

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA\Project1>tf history program2.cs /nop
```

Displays all changes made to program.cs in the command prompt window. For example:

 **Copy Code**

Changeset	Change	User	Date	Co
29	edit	Jamal Hartnett	4/23/2012	Fi
20	add	Raisa Pokrovskaya	4/12/2012	Ac

Get history of all items in a folder

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA>tf history * /recursive
```

Displays all changes made to all items in SolutionA (including those in subfolders) in the History window.

Get history of the last five changes to all items in a folder

Copy Code

```
c:\code\SiteApp\Main\SolutionA>tf history * /noprompt /recursive /st
```

Displays the latest 5 changes made to items in SolutionA (including those in subfolders):

Copy Code

Changeset	User	Date	Comment
31	Raisa Pokrovskaya	5/15/2012	
30	Raisa Pokrovskaya	4/23/2012	
29	Jamal Hartnett	4/23/2012	Fix bug in new method
20	Raisa Pokrovskaya	4/12/2012	Add new method, add program2.
15	Raisa Pokrovskaya	4/8/2012	

Get history from version x and earlier

Copy Code

```
c:\code\SiteApp\Main\SolutionA>tf history /noprompt * /recursive /v:
```

Displays changes made to all items in SolutionA (including those in subfolders) in version 30 and earlier:

Copy Code

Changeset	User	Date	Comment
30	Raisa Pokrovskaya	4/23/2012	
29	Jamal Hartnett	4/23/2012	Fix bug in new method
20	Raisa Pokrovskaya	4/12/2012	Add new method, add program2.

Get history from date D and earlier

Copy Code

```
c:\code\SiteApp\Main\SolutionA>tf history /noprompt * /recursive /v:
```

-- Or --

Copy Code

```
c:\code\SiteApp\Main\SolutionA>tf history /noprompt * /recursive /v:
```

Displays changes made to all items in SolutionA (including those in subfolders) on 4/23/12 or earlier:

Copy Code

Changeset	User	Date	Comment
30	Raisa Pokrovskaya	4/23/2012	
29	Jamal Hartnett	4/23/2012	Fix bug in new method
20	Raisa Pokrovskaya	4/12/2012	Add new method, add program2.

■ Get history from version x to version y

Copy Code

```
c:\code\SiteApp\Main\SolutionA>tf history /noprompt * /recursive /v:
```

Displays changes made to all items in SolutionA (including those in subfolders) between 4/12/2012 and 4/23/12:

Copy Code

Changeset	User	Date	Comment
30	Raisa Pokrovskaya	4/23/2012	
29	Jamal Hartnett	4/23/2012	Fix bug in new method
20	Raisa Pokrovskaya	4/12/2012	Add new method, add program2.

■ Get detailed history

Copy Code

```
c:\code\SiteApp\Main\SolutionA>tf history /noprompt * /recursive /v:
```

Displays details about changes made to all items in SolutionA (including those in subfolders) between 4/12/2012 and 4/23/12:

Copy Code

```
-----  
Changeset: 30  
User: Raisa Pokrovskaya (Fabrikam)  
Date: Monday, April 23, 2012 1:23:05 PM
```

```
Comment:  
    Much better name for this file
```

```
Items:  
    rename                $/SiteApp/Main/SolutionA/Project1/programBig  
    delete, source rename $/SiteApp/Main/SolutionA/Project1/program3.c
```

```
-----  
Changeset: 29  
User: Raisa Pokrovskaya (Fabrikam)  
Date: Monday, April 23, 2012 1:03:13 PM
```

```
Comment:  
    Fix bug in new method
```

```
Items:  
    edit $/SiteApp/Main/SolutionA/Project1/program1.cs  
    edit $/SiteApp/Main/SolutionA/Project1/program2.cs
```

```
-----  
Changeset: 20  
User: Raisa Pokrovskaya (Fabrikam)  
Date: Thursday, April 12, 2012 5:09:35 PM
```

```
Comment:  
    Add new method, add program2.cs to Project1
```

```
Items:  
    add $/SiteApp/Main/SolutionA/Project1/program2.cs
```

```
Check-in Notes:  
    Documentation:  
        An important new part of our codebase.
```

Policy Warnings:

Override Reason:

Jamal agrees with me that we can bypass for this check-in.

Messages:

The Code Analysis Policy requires files to be checked in through Studio with an open solution.

■ **Get the non-recursive history of a folder**

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA>tf history .
```

Displays the history of the SolutionA folder in the History window, which enables you to explore earlier changes to the folder. For example, if the most recent change to the folder was a rename, you can expand the changeset to see changes that occurred before the rename.

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA>tf history . /noprompt
```

Displays the most recent change to the SolutionA folder in the command prompt window.

Work in Visual Studio

- Get the history of an item Use Visual Studio to get revision history.

☐ Tips

- 💡 For information about how to use the History window, see [Get the history of an item](#).
 - 💡 To get more detailed information about a changeset (for example, you want to see associated work items):
 - In the History window, double-click the changeset or open its context menu and choose [Changeset Details](#).
 - From the command prompt, use the [Changeset Command](#).
 - 💡 For more information about changesets, see [Find and view changesets](#).
 - 💡 The **/collection** option is useful for running this command from a machine and user account that does not have a workspace mapped to the Team Project Collection that contains the items.
 - 💡 See [View and manage past versions](#).
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Status command

[Send Feedback](#)

Displays information about pending changes to files and folders in one or more workspaces. Or, when you use the **/shelveset** option, displays information about pending changes in a shelveset.

Requirements: See [Permission reference for Team Foundation Server](#).

```
tf stat[us] itemspec [/collection:TeamProjectCollectionUrl]
[/login:username, [password]]
([/workspace:workspacename[;workspaceowner]]
| [/shelveset:shelvesetname[;shelvesetowner]])
[/format:(brief|detailed)] [/recursive][/user:(*|username)]
[/nodetect]
```

Parameters

Argument	Description
/collection: TeamProjectCollectionUrl	<p>When you use the /workspace option, this option specifies the URL of the team project collection that contains the workspace that contains the pending changes. For example: http://myserver:8080/tfs/DefaultCollection If not specified, by default the team project collection is presumed to be the current collection that contains the workspace that maps to the current directory.</p>
/format	<p>Ignored if you do not use the /workspace option.</p> <p> Note</p> <p>See Remarks, below, for the limitations of this option.</p> <p>Specifies how much detail to display for each pending change:</p> <ul style="list-style-type: none">• Brief (default): Displays one line of information about each pending change. The information includes: file name, change number, whether the item is locked (indicated by an asterisk (*) symbol), local user name (if using the /collection or /workspace options). Some

	<p>data might be truncated.</p> <ul style="list-style-type: none"> • Detailed: Displays a full de of each pending change. In to the above information, tl displays additional data suc and time, and lock.
<p>itemspec</p>	<p>Specifies the items for which yo pending change data. For syntax Team Foundation version contr commands. You can specify mo itemspec argument.</p>
<p>/login</p>	<p>Specifies the user account to use command. See Use Team Found version control commands.</p>
<p>/recursive</p>	<p>Recursively retrieves data about changes to items in the specifiec and any subdirectories.</p>
<p>/shelveset:shelvesetname[;shelvesetowner]</p>	<p>Specifies the shelveset that cont changes you want to list.</p> <p>This option cannot be combined /workspace option.</p>
<p>/user</p>	<p>Lists all pending changes made specified user. An asterisk (*) sy includes data about changes from The default is the current user.</p> <p> Note</p>

▣ **Remarks**

You can use the **Status** command to view pending changes in the current workspace (for example, the workspace that maps the current directory in the command prompt window) regardless of whether it is a local workspace or a server workspace. You can also use this command to view pending changes in a remote server workspace (for example, changes made by another user on another dev machine) by using the **/collection**, **/user**, and **/workspace** options. However, you cannot view pending changes in a remote local workspace.

Also see: Decide between using a local or a server workspace.

Examples

In all the following examples, assume that `$/SiteApp/Main/` is mapped to `c:\code\SiteApp\Main\` in the workspace.

List all changes in the current workspace

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA\>tf stat
```

Lists all pending changes in the workspace.

List all changes in a folder

 **Copy Code**

```
c:\code\SiteApp\Main>tf stat SolutionA\*
```

Lists all pending changes to all items in the SolutionA folder.

List all changes in a folder and its subfolders

 **Copy Code**

```
c:\code\SiteApp\Main>tf stat SolutionA\* /recursive
```

Lists pending changes to all items in the SolutionA folder, including those in its subfolders).

Work in Visual Studio

- Develop code and manage pending changes Use Visual Studio to view and manage pending changes.

☐ Tips

- 💡 Most changes you make to files under version control are queued as pending changes in your workspace. See [Develop code and manage pending changes](#) and [Create and work with workspaces](#).
- 💡 You can use the [Difference Command](#) to get details about edit changes (changes to the content) in a file.
- 💡 If you need to set aside changes (and perhaps also want to clean your workspace for another task), use the [Shelve Command](#). For more information about shelve sets, see [Suspend your work and manage your shelve sets](#).
- 💡 When you are ready to check in changes to the server, use the

[Checkin command](#).



Visual Basic C#

Visual C++

F#

HLSL

JScript

Visual Studio Application Lifecycle Management

Undo command

[Send Feedback](#)

Discards one or more pending changes to files or folders.

Requirements: See [Permission reference for Team Foundation Server](#).

```
tf undo [/workspace:workspacename[;workspaceowner]]  
[/recursive] itemspec [/noprompt] [/login:username,[password]]  
[/collection:TeamProjectCollectionUrl]
```

Parameters

Parameter	Description
/collection : TeamProjectCollectionUrl	Specifies the URL of the team project collection that contains the items. For example: http://myserver:8080/tfs/DefaultCollection
itemspec	If you do not use the /workspace parameter by default the team project collection is presumed to be the one that contains the workspace that maps the current workspace.
/login	Specifies the scope of the items. You can specify more than one itemspec parameter. For syntax, see Use Team Foundation version control commands .
/noprompt	Specifies the user account to use for the command. See Use Team Foundation version control commands .
/recursive	Suppresses the display of window dialog boxes and redirects output to the command prompt. See Use Team Foundation version control commands .
/recursive	Recursively undoes changes to the specified directory and any subdirectories.

Specifies the name of the workspace which you want to undo pending changes in. If not specified, the workspace is the workspace that maps the current directory.

You can specify workspaceowner to indicate a pending change in a workspace belongs to a specific user. If not specified, the workspace is presumed to be owned by the current user, or if specified, the user specified by the /login:username. You must have the UndoOther permission set to All to undo changes in another user's workspace.

/workspaceworkspaceName[;workspaceOwner] undo changes in another user's workspace.

Note

If you use the **undo** command to undo a pending change in a remote workspace that is still in use, then before you can work in that workspace, a user must log on to the machine that hosts the workspace and then get (and in some cases get /all) the items affected by the undo.

Remarks

For each item on which there is a pending edit change, the **undo** command determines if the file has been modified on disk. If the file has been modified and the **/noprompt** option has not been specified, the system prompts you to confirm that you want to proceed. Choose the N key to leave the change in place, the Y key to proceed with only the current change, or the A key to proceed with this and any other modified files that are subsequently detected.

The **undo** command removes any locks on the items.

Examples

Remove pending changes to a file

 **Copy Code**

```
c:\code\SiteApp\Main\SolutionA\Project1>tf undo program.cs
```

Removes all pending changes to program.cs.

Recursively remove pending changes to all items in a folder

 **Copy Code**

```
c:\code\SiteApp\Main>tf undo * /recursive
```

Removes all pending changes in the c:\code\SiteApp\Main folder and all its subfolders.

Remove pending changes to a file in a remote workspace

 **Copy Code**

```
c:\>tf undo /collection:http://fabrikam-3:8080/tfs/DefaultCollection  
/workspace:FABRIKAM-1;JuliaI $/SiteApp/Main/SolutionA/Project1/progr
```

Removes all pending changes to program.cs in the specified collection and workspace.

Work in Visual Studio

- Develop code and manage pending changes Use Visual Studio to undo pending changes.

☐ Tips

- 💡 To view a list of pending changes in the current or in a remote workspace, use the

[Status command](#).

- 💡 You can use the **/workspace** option (and as needed, the **/collection** option) to undo changes on a remote dev machine. This capability is especially useful in cases when, for example, a file has been checked out and possibly locked on a dev machine that you cannot access. See the above explanation of the **/workspace** for information about how this works.
 - 💡 If you need to clean your workspace (for example, because your work is interrupted by a more urgent task) and want to preserve the pending changes instead of undoing them, you can suspend them. See [Shelve Command](#). You can also preserve the position of your open windows, breakpoints, and other important cues. See [Suspend your work and manage your shelvesets](#).
 - 💡 You can also discard changes that have already been checked in. See [Undelete Command](#) and [Rollback Command \(Team Foundation Version Control\)](#).
-



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

TFSServiceControl Command

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **TFSServiceControl** command to stop or start all of the services and application pools Team Foundation Server (TFS) uses. For example, you use this command when backing up or restoring databases, or when you are moving your deployment from one machine to another.

Note

You must use the **TFSServiceControl** command to ensure that all necessary operations, services, and application pools are stopped for maintenance tasks such as backup and restore. You cannot manually perform all of the tasks carried out by the **TFSServiceControl** command.

Requirements

- You must be a member of the Team Foundation Administrators security group, a member of the Administrators group on the application-tier server, and a member of the sysadmin security group for any SQL Server databases that TFS uses. See [Set administrator permissions for Team Foundation Server](#).
- Even if you log on with administrative credentials, you must open an elevated Command Prompt window to perform this function.

```
TFSServiceControl [quiesce|unquiesce]
```

Parameters

Option	Description
quiesce	Stops or pauses all of the services, application pools, and operations in your deployment of TFS. This is required for certain maintenance tasks, such as restoring databases.
unquiesce	Starts or restarts all of the services, application pools, and operations in your deployment of TFS. This is required to return your server to operation after you run the command with the quiesce option.

Remarks

You use the **TFSServiceControl** command as part of specific maintenance tasks. After you specify the **quiesce** option, the server will not operate until you specify the **unquiesce** option. By default, the **TFSServiceControl** command is located in the %programfiles%\Microsoft Team Foundation Server 12.0\Tools directory.

Example

The following example shows how to stop a deployment of Team Foundation Server.

 **Copy Code**

```
TFSServiceControl quiesce
```

The following example shows how to start a deployment of Team Foundation Server.

 **Copy Code**

```
TFSServiceControl unquiesce
```

See Also

Tasks

[Restore data to the same location](#)

Concepts

[Back up and restore TFS](#)

[Configure and manage TFS resources](#)

Other Resources

[Restore a deployment to new hardware](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Delete a team project [TFSDeleteProject]

[Example](#) [See Also](#) [Send Feedback](#)

You can remove a team project from Team Foundation Server(TFS) when the project is no longer required by using **TFSDeleteProject**. In addition, if there are components that remain undeleted after an unsuccessful team project creation, you can use **TFSDeleteProject** to remove them.

To delete a team project from Visual Studio Online or from TFS using the admin console, see Delete a team project.

Caution

TFSDeleteProject permanently destroys the team project, after which it cannot be recovered. You should backup all important project data before using **TFSDeleteProject**.

To access the **TFSDeleteProject** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

Required permissions

To use the **TFSDeleteProject** command, you must be a member of the Team Foundation Administrators security group or the Project Administrators security

group. For more information, see [Set administrator permissions for Team Foundation Server](#).

```
TFSDeleteproject [/q] [/force] [/excludewss] /collection:URL TeamPr
```

Parameters

Option	Description
/q	Optional. Use the quiet mode. Do not prompt the user for confirmation.
/force	Optional. Specifies that the deletion process should continue even if some components cannot be deleted.
/excludewss	Optional. Specifies to not delete the SharePoint site that is associated with the team project. Specify this option to maintain the existing site so that other team projects can continue using it.
/collection:URL	Required. Specifies the URI of the team project collection. You must use the following format for the URI: http://ServerName:Port/VirtualDirectoryName/CollectionName If you do not specify a virtual directory, you must use the following format for the URI: http://ServerName:Port/CollectionName.
TeamProjectName	Required. The name of the project. If the name includes space, enclose it in quotation marks.

Remarks

When you create a team project, Team Foundation Server creates data objects on the server that hosts Team Foundation Server, and may create data objects on the server that hosts SharePoint Products, and the server that hosts SQL Server Reporting Services. When you remove a team project, the reports are automatically removed from SQL Server Reporting Services.

When you remove a team project, you can choose whether or not to remove the objects that were created to support the SharePoint site. However, an error can prevent Team Foundation Server from creating or deleting all the objects. To troubleshoot these problems, the following sections provide background information, links to other resources, and specific steps that help you determine the cause of the problem, fix the problem, and when necessary delete data objects that remain after running **TFSDelProject**.

- [TFSDelProject Process](#)
- [Data That May Remain Undeleted](#)
- [Verify Team Project Components Are Deleted](#)
- [Remove Remaining Components After Partial Project Deletion](#)
- [Increase the Time-Out Period](#)

TFSDelProject Process

When you use the **TFSDelProject** command-line tool, it first deletes project data and then project Web sites.

Phase 1: TFSDelProject Deletes Project Data

In the first phase, **TFSDelProject** automatically performs the following steps to remove team project data:

1. **TFSDelProject** creates an inventory of all the components that are

candidates for deletion. This includes components that integrate with Test Manager, Team Foundation Build, and Team Foundation version control.

2. **TFSDeleteProject** deletes the component that displays the project node in Team Explorer.
3. **TFSDeleteProject** flags the version control information for deletion, but does not immediately delete this information. The information includes all version control branches in the specified project, but no other branches outside of the project.
 - If a parent branch and a child branch are both in the project, **TFSDeleteProject** flags both for deletion.
 - If parent and child branches are in different projects, **TFSDeleteProject** flags only the specified branch.
 - If another project is a branch from the specified project, **TFSDeleteProject** flags only the specified project. When the specified project is deleted, the branch project becomes an orphan.
4. **TFSDeleteProject** immediately deletes build data, including information and core data, build definitions, build agents, and test results associated with the team project. The tool does not delete build drop locations. You do not need to delete the build drop location of an old team project before creating a team project that uses the same build drop location.

If the specified project contains a large amount of build data, the deletion might not finish within the timeout period. To work around this problem, see

[Increase the Time-Out Period](#), and then run **TFSDeleteProject** again.

- **TFSDeleteProject** immediately deletes work items and work item fields that belong to the specified project, and it deletes all non-shared metadata.

If the specified project contains a large amount of work item data, the deletion might not finish within the timeout period. To solve this problem, see [Increase the Time-Out Period](#), and then run **TFSDeleteProject** again.

Phase 2: TFSDeleteProject Deletes Project Web Sites

In the second phase, **TFSDeleteProject** deletes the following data:

⚠️ Important

These steps can take a long time to complete, and during that time they can degrade server performance.

- **TFSDeleteProject** Uses the Reporting Services APIs to delete reports on the server that hosts Reporting Services.
- **TFSDeleteProject** Deletes the project portal Web site from the server that hosts SharePoint Products. This step occurs only if the project owns the site and site deletion is not excluded in the command line. (consider that multiple projects may point to a single site, but only one of them can be the owner where reports/dashboards are by default using this project).

📌 Note

Prior to deleting a team project, you can confirm that Reporting Services and SharePoint Products are using the correct project URLs by verifying the portal settings. For more information, see [How to: Add a Team Project Portal](#).

If **TFSDeleteProject** successfully deletes all of the above data elements, it returns the message Done. To verify this result, see [Verify Team Project Components Are Deleted](#).

If one or more components are not removed, you can rerun **TFSPortDelete** by using the **/force** option to continue the deletion process even if it is unable to delete all data elements. With this option **TFSDeleteProject**, skips a component that it cannot delete, returns an error message, deletes the next component, and leaves the team project metadata and security settings intact.

Data That May Remain Undeleted

The following data might remain in the deployment after **TFSDeleteProject** successfully completes:

- Team project data in the cube. Team project data remains in the cube until the cube is rebuilt, at which time the warehouse controller service removes all of the historic build data that has been deleted from the Team Foundation databases.
- Build drop files and folders. Build binaries, build log files, and log files containing test results are published during the build process. The locations for these files are not deleted. If you want to remove these files, you must remove them manually.
- Work item tracking metadata that is shared. **TFSDeleteProject** does not delete any metadata for work item tracking that is shared between team projects.
- Version control shelvesets containing shared code. Version control shelvesets are not deleted if there is code in the shelveset from multiple team projects.

■ **Verify Project Deletion**

You can verify the success of a project deletion by confirming that the team project node no longer appears in Team Explorer and that its project portal Web site and reports folders no longer exist.

To verify that a team project has been successfully deleted

1. Open Team Explorer and verify that the project does not appear as a project node.
2. Open Internet Explorer and type the URL of the project portal Web site. Verify that the site no longer exists.
3. In Internet Explorer, in the Address box, type the URL of the Reporting Services Web site using one of the following URL formats:
 - <http://ReportingServices/Reports>
 - http://ReportingServices/Reports_TFSInstance
4. In Report Manager, choose Show Details.
5. Verify that the folder for the deleted team project no longer appears. Choose the root folder TfsReports, and then choose the folder named for the team project collection. There should no longer be a folder with the name of the deleted project.
6. If either the reports or the Web site remain, see the next procedure.

Remove Remaining Components After Partial Project Deletion

If the project portal Web site and reports folder remain after you delete a team project, remove the site and folder manually.

To manually remove reports and the project portal Web site of a deleted team project

1. Log on to the server that hosts Reporting Services for the team project that you deleted.
2. Open Internet Explorer, and in the Address box type the URL of the Reporting Services Web site using one of the following URL formats:
 - `http://localhost/Reports`
 - `http://localhost/Reports_TFSInstance`
3. In Report Manager, choose Show Details.
4. Choose the root folder TfsReports, and then choose the folder named for the team project collection.
5. Select the check box for the team project that was deleted.
6. Choose Delete.
7. Choose OK to confirm that you want to delete the reports folder for the team project.
8. To remove the project portal Web site of a deleted team project, see the following page on the Microsoft Web site:

[How to: Create, Edit, and Delete Windows SharePoint Services Sites.](#)

■ Increase the Time-Out Period

By default, each Web service call that the **TFSDeleteProject** command issues to delete a component must complete within 10 minutes. If there are six calls, then the process could take up to an hour. If you want to delete a team project that is associated with a large amount of data, you can temporarily increase this time-out period.

■ Note

When you increase the time-out period, the change affects all Web service calls. In general, you want to keep the time-out period to within 10 minutes to prevent Web service calls from degrading server performance and causing users to be locked out from using the user interface for long periods of time. Therefore, after the project is successfully deleted, you should change the time-out period back to 10 minutes.

Required Permissions

To complete these procedures, you must be a Windows Administrator on the application-tier server.

⚠ Important

Improperly modifying your computer's registry can cause your computer to become unstable. If you are not familiar with the registry, you should not add or remove entries, or modify it in any way.

To increase the time-out period for the **TFSDeleteProject** command

1. Log on to the application-tier server.
2. Choose Start, Run, type regedit, and then choose OK.

3. In the browser pane, expand **HKEY_LOCAL_MACHINE**:
 - If the server runs a 32-bit operating system, expand:
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\VisualStudio\11.
 - If the server runs a 64-bit operating system, expand:
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432\Microsoft\Visua
4. If the TeamFoundation\RequestSettings key does not exist, follow these steps to create it:
 1. Open the context menu for TeamFoundation, point to New, and choose Key.
 2. Name the key RequestSettings.
 3. Open the context menu for RequestSettings, point to New, and choose DWORD Value.
 4. Name the new value DefaultTimeout.
5. Open the context menu for DefaultTimeout and choose Modify.
6. In Value Data, type the time-out period in milliseconds, and then choose Decimal.

For example, to increase the time-out period to 30 minutes, type 1800000.
To change the time-out period back to 10 minutes, type 600000.
7. Choose OK.
8. On the File menu, choose Exit.

Example

The following command removes all components associated with the team project StoreFront on the Team Foundation Server AdventureWorks1 server in project collection Collection1 and from Team Explorer.

 **Copy Code**

```
TFSDeleteProject /force /collection:http://AdventureWorks1:8080/tfs,
```

See Also

Concepts

Connect to team projects in Team Foundation Server

Other Resources

[Create a team project](#)

[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Upload or download the Microsoft Project Mapping file [TFSFieldMapping]

[See Also](#) [Send Feedback](#)

Use **TFSFieldMapping** to upload and download the Microsoft Project mapping file. This file controls how Microsoft Project publishes tasks to a specified team project that was created in Visual Studio Team Foundation Server 2013. For example, you can modify the file to support fields you added to work item types or modify the way in which existing fields are published and refreshed.

You can specify which fields in Team Foundation are mapped to Microsoft Project fields. You can also control publishing behavior through the PublishOnly attribute, and designate special fields such as the context field. For the complete XML syntax of the mapping file, see [Customize the Microsoft Project field mapping file](#). This file is the same file that was defined in the process template that was used to create the team project. See [Map Microsoft Project fields to Team Foundation fields](#)

To access the **TFSFieldMapping** command-line tool, open a Command Prompt window where you have installed Project and either Team Explorer or TFS and enter:

 **Copy Code**

```
cd %programfiles%\Common Files\microsoft shared\Team Foundation Serv
```

Note

Even if you are logged on with administrative permissions, you must open an elevated Command Prompt to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt, choose Start, open the context menu for Command Prompt, and then choose Run as Administrator. For more information, see the [Microsoft Web site](#).

Required Permissions

To use the **TFSFieldMapping** command, you must be a member of the Team Foundation Administrators or the Project Administrators security group. For more information, see [Permission reference for Team Foundation Server](#).

```
TFSFieldMapping upload | download /collection:CollectionURL /teamprc
```

Parameters

Argument	Description
upload download	Specifies whether to download or upload the field mapping.
/collection:CollectionURL	<p>Specifies the URI of the team project collection in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName</p> <p>If you do not specify a virtual directory is used, you must use the following format: http://ServerName:Port/CollectionName</p> <p>If you have previously connected to the server, you can use the friendly name for the server instead of the URI.</p>
/teamproject:ProjectName	Specifies the name of the team project whose mapping parameters you want to modify. If the name contains spaces, you must enclose it in quotation marks.
/mappingfile:MappingFile	Specifies the name and file path of the mapping file.

■ To change how work item fields are mapped to Project

1. Open a Command Prompt window where you have installed Project and either Team Explorer or TFS and enter:

Copy Code

```
cd %programfiles%\Common Files\microsoft shared\Team Foundation
```

Tip

Project isn't required to download the mapping file, however, it is required to upload it.

2. Download the mapping file by entering the following command:

Copy Code

```
TFSFieldMapping download /collection:CollectionURL /teamproject
```

Tip

If a name contains spaces, enclose the name in quotes.

3. Open the mapping file in a text editor or XML editor.
4. Add new mappings or edit existing mappings in the following format:

Copy Code

```
<Mapping WorkItemTrackingFieldReferenceName="System.Id"  
  ProjectField=""  
  ProjectName=""  
  ProjectUnits=""
```

```
PublishOnly=""  
IfSummaryRefreshOnly="" />
```

For example, to add additional scheduling fields to a team project created using the Scrum process template, add the following mappings:

```
<Mapping WorkItemTrackingFieldReferenceName="Microsoft.VSTS.Sch  
<Mapping WorkItemTrackingFieldReferenceName="Microsoft.VSTS.Sch  
<Mapping WorkItemTrackingFieldReferenceName="Microsoft.VSTS.Sch  
<Mapping WorkItemTrackingFieldReferenceName="Microsoft.VSTS.Sch
```

The default Scrum template only maps the Remaining Work field to Project.

Note

For a full list of all the Office Project field values for the ProjectField attribute, see [Field mappings in Microsoft Project](#).

5. (Optional) Indicate if the field is to be published but not refreshed. Add the PublishOnly attribute to the mapping section for that field and set the attribute to true. This attribute can be used to allow for team members to see a field value but not be able to change it in Team Foundation.
6. (Optional) Indicate how summary tasks are to be refreshed in Project. Add IfSummaryRefreshOnly="true" attribute to indicate that the field is never published to the work item database but is refreshed from the work item database when the row for the field is a summary task in Project, the summary task has Publish and Refresh=Yes, and the summary task contains at least one child task that is published to TFS.
7. Save and upload the file:

 **Copy Code**

```
TFSFieldMapping upload /collection:CollectionURL /teamproject:Pi
```

See Also

Reference

[Field mappings in Microsoft Project](#)

Concepts

[Customize the Microsoft Project field mapping file](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Customize and manage the test experience [tcm and Microsoft Test Manager]

[See Also](#) [Send Feedback](#)

You can customize three of the features that are supported in Microsoft Test Manager using the tcm fieldmapping command. This command allows you to change the bug type used to automatically file bugs in Test Manager, and to customize the drop-down menu or pick lists for resolution states and failure types.

To access the **tcm** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

Also, you can use tcm command options to import and run automated tests for a test plan.

Requirements

For the team project where the work item types are defined, you must be a member of the Team Foundation Administrators security group or the Project Administrators security group. See [Set administrator permissions for team project collections](#).

```
tcm fieldmapping /export /collection:CollectionURL /teamproject:Pro:
```

tcm fieldmapping /import /collection:CollectionURL /teamproject:Pro

Parameters

Parameter	Description
/export	Specifies export of the field mapping file of the type For the syntax of supported file types, see the Remark
/import	Specifies import of the field mapping file.
/mappingfile:Path	The path and file name of the XML definition file that the field mappings.
/collection:CollectionURL	Specifies the uniform resource identifier (URI) of the project collection. The format for the URI is as follows: http://ServerName:Port/VirtualDirectoryName/CollectionName If no virtual directory is used, the format for the URI follows: http://ServerName:Port/CollectionName
/teamproject:Project	Specifies the name of the team project for which the mappings are to be applied. This team project must be in the team project collection that is specified by the parameter.
/type:resolutiontype bug failuretype	The type of file to import or export. For the syntax see each file type, see bug , resolution states , and failure

/login:UserName,Password

Optional. Specifies the name and password of a user logged on to the application-tier server for Team Fou and who has permissions to run the command.

You would use this option when your Windows cred not have the appropriate permissions, you are using l authentication, or you are not on a domain.

/? or **help**

Displays help about the command in the Command I window.

Remarks

Each time that you run the **tcm fieldmapping import** command, the contents of the existing field mapping file are replaced with the contents that you import.

Mapping file for bug work item type

When Test Manager creates a bug, the mapping file defines the type of work item to create and fills in three data fields: reproducible steps, system information, and the build in which the defect was found. When a tester runs a test and finds a defect, they can create a bug in which these three fields are automatically filled in.

If your team project was created with one of the default process templates provided with Team Foundation Server, the bug work item type is already enabled. However, if you're adding another type of work item to the bug category or you're working with a customized process template, you might have to use the **tcm** command.

You can define only one type of work item to be created when you run tests that use Test Manager.

The following XML syntax lists the default contents of the bug field mappings file. All three fields must be specified when you import a bug field mappings file.

Copy Code

```
<?xml version="1.0" encoding="utf-16"?>
<BugFilerMappings workitemtypetocreate="Bug">
  <ReproSteps>Microsoft.VSTS.TCM.ReproSteps</ReproSteps>
  <SystemInformation>Microsoft.VSTS.TCM.SystemInfo</SystemInformation>
  <BuildFoundIn>Microsoft.VSTS.Build.FoundIn</BuildFoundIn>
</BugFilerMappings>
```

For more information about fields that are used to track information that Test Manager finds, see

[Build and test integration field reference.](#)

Mapping file for resolution states

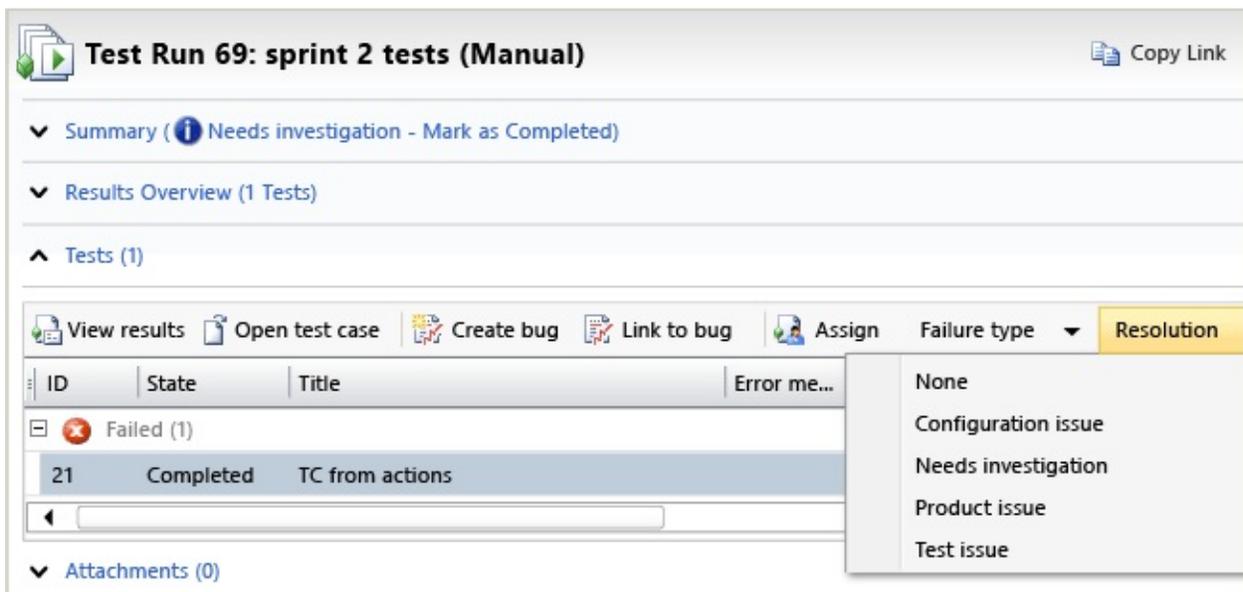
The following XML syntax lists the default contents of the resolution states defined for the default process templates.

Copy Code

```
<?xml version="1.0" encoding="utf-8"?>
<TestResolutionStates>
  <TestResolutionState name="Needs investigation" />
  <TestResolutionState name="Test issue" />
  <TestResolutionState name="Product issue" />
  <TestResolutionState name="Configuration issue" />
</TestResolutionStates>
```

Before you create a team project, you can customize the resolution states defined in the process template.

When you analyze failed tests, you assign the failure type and resolution state.



The screenshot shows a test run interface for "Test Run 69: sprint 2 tests (Manual)". The interface includes a summary section with a "Needs investigation" status, a "Results Overview" section with 1 test, and a "Tests" section with 1 test. A table of test results is displayed, with a dropdown menu open for the "Resolution" column. The table has columns for ID, State, Title, and Error message. The dropdown menu lists the following resolution states: None, Configuration issue, Needs investigation, Product issue, and Test issue.

ID	State	Title	Error me...
21	Completed	TC from actions	

Mapping file for failure types

The following XML syntax lists the default failure types defined for a team

project.

 **Copy Code**

```
<?xml version="1.0" encoding="utf-16"?>
<TestFailureTypes>
  <TestFailureType name="Regression" />
  <TestFailureType name="New Issue" />
  <TestFailureType name="Known Issue" />
  <TestFailureType name="Unknown" />
</TestFailureTypes>
```

Examples

Unless otherwise specified, the following values apply in each example:

- URI for the team project collection:
http://AdventureWorksServer:8080/tfs/Collection1
- Project name: AdventureWorks
- Bug field mapping file name: bugfieldmappings.xml
- Folder location: "C:\Users\AdminUser\Documents\"

Import the bug field mappings file

The following command imports the contents of the bugfieldmappings.xml file to the AdventureWorks team project that is defined in Collection1 on the server that is named AdventureWorksServer.

```
tcm fieldmapping /import /type:bug /mappingfile:"C:\Users\AdminUser\
```

See Also

Concepts

Using TCM from the command line

Other Resources

[Command-line tools for TFS](#)



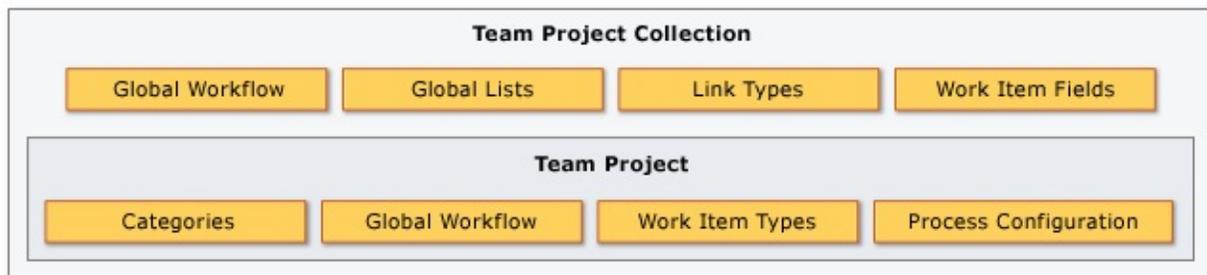
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

witAdmin: Customize and manage objects for tracking work

[See Also](#) [Send Feedback](#)

You can change how you track your team's progress by creating and customizing objects that track work items. By using the **witadmin** command-line tool, you can create, delete, import, and export objects such as categories, global lists, global workflow, types of links, and types of work items. You can also permanently delete types of work item types, and you can delete, list, or change the attributes of fields in work item.

As the following illustration shows, you manage categories and types of and work items for each team project. You manage global lists, types of links, and fields in work item for each team project collection. You can customize global workflow for a team project or a collection.



Note

You can create and modify objects for tracking work items by using Process Editor, a power tool for Visual Studio. This tool is not supported. For more information, see the following page on the Microsoft website:

[Team Foundation Server PowerTools.](#)

Global parameters

You can use the following parameters to display help text for **witadmin**.

Parameter	Description
/help or /?	Displays the syntax and parameters for the witadmin command-line tool.
command /help or command /?	Displays the syntax and parameters for the witadmin command that you specify.

Command index

To access the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

 **Copy Code**

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

Tasks	Related content
<p>Create and modify work item types. Work item types define the fields and behavior that you use to create and manage work items. You can create and modify types of work items to support the process objectives of your team and your team project.</p>	
<p>You can manage work item types for a team project by using the following witadmin commands:</p>	
<ul style="list-style-type: none">• destroywitd: Deletes a work item type.	<ul style="list-style-type: none">•
<ul style="list-style-type: none">• exportwitd: Exports the	

XML definition of a work item type.

- **importwitd:** Imports the XML definition of a work item type.
- **listwitd:** Lists the names of the work item types.
- **renamewitd:** Changes the display name of a work item type.

[Import, export, and manage work item types \[witadmin\]](#)

- Customize work tracking objects to support your team's processes

Customize process configuration. You can customize the Agile experience for Team Web Access.

- **exportprocessconfig:** Exports the XML definition for process configuration .
- **importprocessconfig:** Imports the XML definition for process configuration.
- [Import and export process configuration \[witadmin\]](#)
- Configure and customize Agile planning tools for a team project
- [Process configuration XML element reference](#)

Import and export categories. By using categories, you can group work item types and then perform operations on them as a set. For example, you can specify a category when you define the criteria of a work item query to find all work items in that category, regardless of whether the items are different types.

You can manage work item categories for a team project by using the following **witadmin** commands:

- **exportcategories:** Exports the XML definitions of all categories.
- **importcategories:** Imports the XML definitions of all categories.
- [Import and export categories \[witadmin\]](#)
- Use categories to group work item types

Manage global lists for work item types. By using global lists, you can share list items among multiple types of work items.

You can manage global lists for a collection by using the following **witadmin** commands:

- **destroygloballist:** Deletes a global list.
- **exportgloballist:** Exports the XML definition file that contains one or more global lists.
- **importgloballist:** Imports the XML file that defines one or more global lists.
- **listgloballist:** Lists the name of each global list that is defined for the collection.
- [Manage global lists for work item types \[witadmin\]](#)
- [Define global lists](#)

Import and export global workflow. By using global workflow, you can minimize the work required to define and update global lists and fields in work items that many team projects and types of work items share. You can share definitions of fields and list items among multiple types of work items.

You manage global workflow for each team project or collection by using the following **witadmin** commands:

- **exportglobalworkflow:** Exports the XML file that define a global workflow.
- **importglobalworkflow:** Imports the XML file that defines a global

workflow.

- [Import and export global workflow \[witadmin\]](#)
- [Customize global workflow](#)

Manage link types. By using link types, you can associate work items based on a set of link rules. If you create custom types of links, you can track work items and their dependencies in more ways.

You can manage types of links for a collection by using the following **witadmin** commands:

- **deactivatelinktype:** Deactivates a specified type of link.
- **deletelinktype:** Permanently removes the specified type of link from the database.
- **exportlinktype:** Exports the XML definition files of one or more types of links.
- **importlinktype:** Imports the XML definition files of one or more types of links.
- **listlinktypes:** Lists the set of link types on a server.
- **reactivatelinktype:** Reactivates the type of link that you specify. You can also rename the type of link.
- [Manage link types \[witadmin\]](#)
- Define a custom link type
- Link work items to support traceability

Manage work item fields. You use fields to find work items and to track information in them.

You can manage fields for a collection by using the following **witadmin** commands:

- **deletefield:** Deletes the field that you specify.
- **listfields:** Lists the attributes for a field that you specify or for all fields.
- **indexfield:** Turns indexing on or off for a field that you specify.
- **changefield:** Changes one or more attributes of a field that you specify.
- [Manage work item fields \[witadmin\]](#)
- Modify or add a field to support queries, reports, and workflow
- Add or modify work item fields to support reporting

Destroy work items. You can permanently remove work items from a team project collection by using the **witadmin destroywi** command.

- [Remove work items permanently \[witadmin\]](#)

Force a rebuild of the client cache. After certain maintenance operations, you can force a rebuild of the cache of all client computers the next time that they connect to a team project collection. For example, you might force a rebuild after you move, rename, restore, or fail over a data-tier or application-tier server. By forcing a rebuild, you help prevent workspace errors during version control or build operations. To perform this operation, you use the **witadmin rebuildcache** command.

- [Rebuild the client cache \[witadmin\]](#)

See Also

Concepts

Configure and customize Agile planning tools for a team project
Customize work tracking objects to support your team's processes

Other Resources

[Command-line tools for TFS](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Import, export, and manage work item types [witadmin]

[See Also](#) [Send Feedback](#)

You can manage work item types for a team project by using the following **witadmin** commands:

- **destroywitd**: Destroys a work item type, and destroys every work item of that type permanently without recovery.
- **exportwitd**: Exports the definition of a work item type to an XML file, or to the Command Prompt window.
- **importwitd**: Imports work item types from an XML definition file into a team project on a server that runs Team Foundation Server. If a work item type with the same name already exists, the new work item type definition overwrites the existing one. If the work item type does not already exist, this command creates a new work item type. To validate the XML that defines a work item type, but not import the file, you use the **/v** option.
- **listwitd**: Displays the names of the work item types in the specified project in the Command Prompt window.
- **renamewitd**: Changes the display name of a work item type within a specific project. After you run this command, work items of this type show the new name.

To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

 **Copy Code**

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

For more information about work item types, see [Add type definitions for work items to a process template](#).

Note

You can create and modify work item types by using Process Editor, a power tool for Visual Studio. This tool is not supported. For more information, see the following page on the Microsoft Web site: [Team Foundation Server Power Tools](#).

Requirements

For the team project where the work item types are defined, you must have the following permissions set:

- To export or list work item types, you must be a member of the Project Administrators group or have your View project-level information permission set to Allow.
- To destroy, import, or rename work item types, you must be a member of the Team Foundation Administrators security group or the Project Administrators security group.

For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative permissions, you must open an elevated Command Prompt window to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt window, choose Start, open the shortcut menu for the Command Prompt, and then choose Run as Administrator. For more information, see the Microsoft Web site: [User Access Control](#).

```
witadmin destroywitd /collection:CollectionURL /p:Project /n:TypeName
```

witadmin exportwitd /collection:CollectionURL /p:Project /n:TypeName

witadmin importwitd /collection:CollectionURL [/p:Project] /f:FileName

witadmin listwitd /collection:CollectionURL /p:Project

witadmin renamewitd /collection:CollectionURL /p:Project /n:TypeName

Parameters

Parameter	Description
/collection: CollectionURL	<p>Specifies the URI of the team project collection. The the URI is the following: http://ServerName:Port/VirtualDirectoryName/Colle</p> <p>If no virtual directory is used, then the format for the following: http://ServerName:Port/CollectionName</p>
/p: Project	<p>The team project for which the types of work items are managed. This team project must be defined in the pr collection specified by the /collection parameter.</p> <p>The /p parameter is required unless you run the impo command with the /v option.</p>
/n: TypeName	<p>The name of the work item type to destroy, export, in rename.</p>
/f: FileName	<p>The path and file name of the XML definition file that the types of work items to be exported or imported. If this parameter when you use the exportwitd command XML appears in the Command Prompt window.</p> <p>Note</p> <p>If you are using Windows Vista you might not have permissions to certain folders. If you try to export the</p>

item type to a location where you do not have permis registry virtualization technology automatically redir exported file and saves it to the virtual store. To avoi redirection, you can export the file to a location wher have permissions. For more information about regist virtualization, see the Microsoft Web site: [Registry Virtualization](#) and [Common file and registry virtualiz issues in Windows Vista](#).

/e:Encoding

The name of a .NET Framework 2.0 encoding format command uses the specified encoding to export or im XML data. For example, /e:utf-7 specifies Unicode (U encoding. If you omit this parameter, **witadmin** tries the encoding, and if detection fails, **witadmin** uses U

/exportgloballists

Exports the definitions of global lists referenced by th item type. The definitions for global lists will be emb the work item type definition XML. When not specifi definitions for global lists are omitted.

Validates the XML that defines the work item type, b import the XML definition file.

/v

 **Note**

You can validate the type definition without specifyin project. References to project-scoped groups is ignor

/new:NewName

The new name of the work item type.

/noprompt

Disables the prompt for confirmation.

Displays help about the command in the Command P

/? or **help**

window.

Remarks

When you use the **destroywitd** command, it destroys all the following objects:

- The work item type
- All work items of that type
- Corresponding entries in the work item tables, the long text tables, and the link tables
- Objects in the work item type metadata cache

Examples

Unless otherwise specified, the following values apply in each example:

- URI for the team project collection:
http://AdventureWorksServer:8080/tfs/DefaultCollection
- Project name: AdventureWorks
- Input or output file name: myworkitems.xml
- Work item type name: myworkitem
- Default encoding: UTF-8

Export the definition of a WIT

The following command exports the definition for myworkitem to the file, myworkitems.xml.

```
witadmin exportwitd /collection:http://AdventureWorksServer:8080/tfs
```

The following example exports the work item by using Unicode (UTF-7) encoding.

```
witadmin exportwitd /collection:http://AdventureWorksServer:8080/tfs
```

Export the definition of a WIT and its referenced global lists

The following example exports the work item type and its referenced global lists.

```
witadmin exportwitd /collection:http://AdventureWorksServer:8080/tfs
```

▣ **List the definition of a WIT**

The following example displays the definition of the work item type the Command Prompt window.

 **Copy Code**

```
witadmin exportwitd /collection:http://AdventureWorksServer:8080/tfs
```

▣ **Import the definition of WITs**

The following example imports the work item definition from the XML file.

 **Copy Code**

```
witadmin importwitd /collection:http://AdventureWorksServer:8080/tfs
```

▣ **Validate the XML definition of a WIT**

The following example validates the XML that defines the work item type but does not import the definition.

 **Copy Code**

```
witadmin importwitd /collection:http://AdventureWorksServer:8080/tfs
```

Q & A

Q: What customizations can I make and still use the Configure Features Wizard to update my team project after a TFS upgrade?

A: You can add custom WITs and change the form layout. The Configure Features Wizard will update your team projects and you'll get access to the latest features.

Changing the workflow or renaming a WIT might require you to perform some manual operations when updating your team project. To learn about which customizations you can safely make and which you should avoid, see [Customize the work tracking experience: Before you customize, understand the maintenance and upgrade implications](#).

Q: How do I change the color associated with a WIT?

A: In Team Web Access, work items appear in query results and on the backlog and board pages of the Agile planning tools. To change the color associated with an existing WIT or add the color to use for a new WIT, edit the process configuration.

Sprint 1

Backlog Board Capacity

Create Query Column Options

Title	State	Assigned To
<ul style="list-style-type: none"> <ul style="list-style-type: none"> Hello World Web Site <ul style="list-style-type: none"> Welcome Screen Change background color About Screen Slow response on welcome page <ul style="list-style-type: none"> Rework opening animation Change initial view Add an information form <ul style="list-style-type: none"> Auto-complete user's name in form if logged in Auto-save Welcome back <ul style="list-style-type: none"> Add interactive text to Welcome Back page 	<ul style="list-style-type: none"> New To Do To Do In Progress New To Do New New In Progress To Do New To Do 	<ul style="list-style-type: none"> Nicole Zamora (Fabr... Johnnie McLeod (Fabr... Christie Church (Fab... Nicole Zamora (Fabr... Johnnie McLeod (Fab... Jamal Hartnett (Fabr... Raisa Pokrovskaya (F... Jamal Hartnett (Fabrik... Christie Church (Fab... Raisa Pokrovskaya (Fa...

Q: How do I deactivate or disable a WIT? How do I restrict users from creating work items of a certain type?

A: If you have a WIT that you want to retire, but maintain the work items that have been created based on that type, you can add a rule that disables all valid users from saving the work item type.

Copy Code

```
<TRANSITION from=" " to="New">
  <FIELDS>
    <FIELD refname="System.CreatedBy">
      <VALIDUSER not="[Team Project Name]Project Valid Users" />
    </FIELD>
  </FIELDS>
</TRANSITION>
```

If you want to restrict creation of a specific WIT to a group of users, there are two ways to restrict access:

- Add the WIT to the Hidden Categories group to prevent the majority of contributors from creating them. If you want to allow a group of users access, you can create a hyperlink to a template that opens the work item form and share that link with those team members who you do want to create them.
- Add a field rule to the workflow for the System.CreatedBy field to effectively restrict a group of users from creating a work item of a specific type. As the following example shows, the user who creates the work item must belong to the Allowed Group in order to save the work item.

Copy Code

```
<TRANSITION from=" " to="New">
  <FIELDS>
    <FIELD refname="System.CreatedBy">
      <VALIDUSER for="Allowed Group" not="Disallowed Group" .
    </FIELD>
  </FIELDS>
</TRANSITION>
```

Q: How do I delete a WIT?

A: To prevent team members from using a specific WIT to create a work item, you can remove it from the team project. When you use `witadmin destroywitd`, you permanently remove all work items that were created using that WIT as well as the WIT itself. For example, if your team doesn't use "Impediment," you can delete the WIT labeled "Impediment" from the Fabrikam Web Site project.

Copy C

```
witadmin destroywitd /collection:"http://FabrikamPrime:8080/tfs/Defa
```

When you delete a WIT that belongs to a category, you must update the categories definition for the team project to reflect the new name. In particular, the Agile planning tools will not work until you update the categories definition.

For more information, see

[Import and export categories \[witadmin\]](#).

See Also

Tasks

Create a work item type

Concepts

Customize work tracking objects to support your team's processes

Other Resources

[witAdmin: Customize and manage objects for tracking work](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Import and export categories [witadmin]

[See Also](#) [Send Feedback](#)

You can import and export categories defined for a team project by using the following **witadmin** commands:

- **exportcategories:** Exports the XML definition of categories defined on a server that runs Team Foundation Server.
- **importcategories:** Imports a work item type XML definition file into a team project on a server that runs Team Foundation Server.

If a category with the same name already exists, this command overwrites the existing definition. If the work item type does not already exist, this command creates a new category.

To learn about the default categories and how they are used, go [here](#).

To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

Note

You can create and modify categories by using Process Editor, a power tool for Visual Studio. This power tool is not supported. For more information, see the

following page on the Microsoft Web site: [Team Foundation Server Power Tools](#).

Requirements

For the team project for which the categories are defined, you must have the following permissions set:

- To export categories of work item types, you must have your View project-level information permission set to Allow.
- To import categories of work item types, you must be a member of the Project Administrators security group or have the Edit project-level information permission set to Allow.

```
witadmin exportcategories /collection:CollectionURL /p:Project [/f:F
```

```
witadmin importcategories /collection:CollectionURL /p:Project /f:F:
```

Parameters

Parameter	Description
<code>/collection:CollectionURL</code>	<p>Specifies the URI of the team project collection. The the URI is the following: http://ServerName:Port/VirtualDirectoryName/CollectionName</p> <p>If no virtual directory is used, then the format for the following: http://ServerName:Port/CollectionName.</p>
<code>/p:Project</code>	<p>The name of the team project from which the categories are exported or to which the categories are imported.</p>
<code>/f:FileName</code>	<p>The path and file name of the XML definition file that contains the categories to be exported or imported. If you omit this parameter when you use the exportcategories command, the command lists the categories in the Command Prompt.</p>
<code>/e:Encoding</code>	<p>The name of a .NET Framework 2.0 encoding format. The specified encoding will be used to export or import the data. For example, <code>/e:utf-7</code> specifies Unicode (UTF-7) encoding. If you omit this parameter, witadmin attempts to detect the encoding, and if detection fails, witadmin uses UTF-8.</p>
<code>/? or help</code>	<p>Displays help about the command in the Command Prompt window.</p>

Remarks

Importing a categories file creates the categories if they do not already exist. If categories already exist, the **witadmin importcategories** command will warn you that the current list will be overwritten. You can write your own program to update an existing set of categories, or you can update the categories yourself with new data.

Important

Changes you make to categories can impact process configuration. See [Configure and customize Agile planning tools for a team project](#).

For information about the categories XML file structure, see [Categories XML element reference](#).

Examples

Unless otherwise specified, the following values apply in each example:

- URI for the team project collection:
http://AdventureWorksServer:8080/tfs/DefaultCollection
- Project name: AdventureWorks
- Input or output file name: myCategories.xml
- Default encoding: UTF-8

List categories

The following command lists the work item type categories defined for the AdventureWorks project.

 **Copy Code**

```
witadmin exportcategories /collection:http://AdventureWorksServer:80
```

Add a category to the Hidden Types categories

You add a category to the Hidden Types categories to remove support for users to create work item types in that category.

1. Export the definition file for categories for your team project.

```
witadmin exportcategories /collection:http://AdventureWorksServ
```

2. Open the Categories file in your preferred text editor, and locate the CATEGORY element for the "Hidden Types Category".

Copy Code

```
<CATEGORY name="Hidden Types Category" refname="Microsoft.Hidder
  <DEFAULTWORKITEMTYPE name="Code Review Request" />
  <WORKITEMTYPE name="Code Review Response" />
  <WORKITEMTYPE name="Feedback Request" />
  <WORKITEMTYPE name="Feedback Response" />
  <WORKITEMTYPE name="Shared Steps" />
</CATEGORY>
```

3. Add a WORKITEMTYPE element that specifies the friendly name of the work item type that you want to add.

For example, the following syntax adds the custom work item type of "Integration Build" to the Hidden Types Category.

Copy Code

```
<CATEGORY name="Hidden Types Category" refname="Microsoft.Hidder
  <DEFAULTWORKITEMTYPE name="Code Review Request" />
  <WORKITEMTYPE name="Code Review Response" />
  <WORKITEMTYPE name="Feedback Request" />
  <WORKITEMTYPE name="Feedback Response" />
  <WORKITEMTYPE name="Shared Steps" />
  <WORKITEMTYPE name="Integration Build" />
</CATEGORY>
```

4. Import the definition file for categories to your team project.

```
witadmin importcategories /collection:http://AdventureWorksServi
```

See Also

Concepts

Configure and customize Agile planning tools for a team project

Other Resources

[Use categories to group work item types](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Manage global lists for work item types [witadmin]

[See Also](#) [Send Feedback](#)

Global lists are pick lists that you can include in one or more fields and types of work items. Use them to minimize the work that is required to update a list that is shared by multiple types of work items. Because global lists are defined for a team project collection, they can be included in work item types for all team projects within the collection.

You can define a global list within its own definition file, in the definition file for a type of work item, or in global workflow. The global-list definition manages the global lists that are defined for a collection. The global-list definition uses the following commands in the **witadmin** command-line tool:

- **destroygloballist:** Destroys a global list.
- **exportgloballist:** Exports the global lists to an XML file, or to the Command Prompt window. It exports a single file, which contains all global lists that have been defined for the team project collection.
- **importgloballist:** Imports global lists from an XML file. If you try to import a global list that already exists on the server, a warning asks if you want to overwrite the global list that is on the server. If you try to import a global list that does not already exist, the command creates a new global list.
- **listgloballist:** Displays the name of each global list defined on the server.

See [Define global lists](#) and [Customize global workflow](#).

To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 64-bit edition of Windows, replace **%programfiles%** with **%programfiles(x86)%**.

Note

You can create and modify global lists by using Process Editor, a power tool for Visual Studio. For more information, see [Team Foundation Server Power Tools](#).

Requirements

For the team project collection where the global lists are defined, you must have the following permissions set:

- To export or list global lists, you must be a member of the Project Collection Valid Users group or have your View collection-level information permission set to Allow.
- To import global lists, you must be a member of the Project Administrators security group or have your Edit project-level information permission set to Allow.
- To destroy a global list using **witadmin destroygloballist**, you must be a member of the Project Collection Administrators security group or have your Edit collection-level information permission set to Allow.

See [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative permissions, you must open an elevated Command Prompt window to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt window, choose Start, open the Command Prompt shortcut window, and choose Run as Administrator. For more information, see this page on the Microsoft Web site:

User Access Control.

witadmin destroygloballist /collection:CollectionURL /n:GlobalListName

witadmin exportgloballist /collection:CollectionURL [/f:FileName] [/o:OutputFile]

witadmin importgloballist /collection:CollectionURL /f:FileName [/e:Encoding]

witadmin listgloballist /collection:CollectionURL

Parameters

Parameter	Description
/c:collection: CollectionURL	<p>Specifies the URI of the team project collection. The the URI is the following: http://ServerName:Port/VirtualDirectoryName/Colle</p> <p>If no virtual directory is used, then the format for the following: http://ServerName:Port/CollectionName.</p>
/n: GlobalListName	<p>The name of the global list to destroy.</p> <p>The path and the name of the global list XML definiti export or import.</p> <p>Note</p> <p>If the client computer runs Windows Vista, you might permissions to certain folders. If you try to export the list to a location where you do not have permissions, registry virtualization technology automatically redir exported file and saves it to the virtual store. For mor information, see the following pages on the Microsoft site: Registry Virtualization and Common file and re virtualization issues in Windows Vista. To avoid this redirection, you can export the file to a location wher have permissions.</p>
/f: FileName	<p>The name of a .NET Framework 2.0 encoding format</p>

/e:Encoding

specified encoding will be used to export or import the data. For example, `/e utf-7` specifies Unicode (UTF-7) encoding. If you omit this parameter, **witadmin** attempts to detect the encoding, and if detection fails, **witadmin** uses the default encoding.

/noprompt

Disables the prompt for confirmation.

/? or help

Displays help about the command in the Command Prompt window.

Remarks

Importing a global list creates a list if one does not exist. If the list already exists, the **witadmin importgloballist** command will warn you that the current list will be overwritten. You can write your own program to update an existing global list, or you can update the global list yourself with the new data.

To create a new global list, start with the following code and modify it as needed. This example defines a global list of disciplines that you can assign to tasks.

Copy Code

```
<?xml version="1.0" encoding="utf-8"?>
<gl:GLOBALLISTS xmlns:gl="http://schemas.microsoft.com/VisualStudio,
  <GLOBALLIST name="Disciplines">
    <LISTITEM value="Architecture" />
    <LISTITEM value="Requirements" />
    <LISTITEM value="Development" />
    <LISTITEM value="Release Management" />
    <LISTITEM value="Project Management" />
    <LISTITEM value="Test" />
  </GLOBALLIST>
</gl:GLOBALLISTS>
```

Don't include project-scoped security groups within a global list, because global lists are scoped to a collection and not a project.

To add a global list to a field, export the definition for the work item type that contains the field and add it to the field definition, as shown in the following example:

Copy Code

```
<FIELD name="Discipline" refname="Microsoft.VSTS.Common.Discipline"
  <HELPTEXT>The discipline to which the task belongs</HELPTEXT>
  <ALLOWEDVALUES>
    <GLOBALLIST name="Disciplines" />
  </ALLOWEDVALUES>
</FIELD>
```

To view the changes, import the type definition and refresh your browser or client cache. You might need to close any work items that reference the field and reopen them.

For information about export and import of type definitions, see [Import, export, and manage work item types \[witadmin\]](#).

Examples

Unless otherwise specified, the following values apply in each example:

- URI for the team project collection:
`http://AdventureWorksServer:8080/tfs/DefaultCollection`
- Server Web site port number: 8080

Display the names of global lists

The following example shows the global lists defined on the server. The example exports the global lists using the default UTF-8 encoding:

 **Copy Code**

```
witadmin listgloballist /collection:http://AdventureWorksServer:8080
```

Export the definition of global lists

The following example exports the global lists:

 **Copy Code**

```
witadmin exportgloballist /collection:http://AdventureWorksServer:8080
```

The following example exports the same global lists to the same server, but uses Unicode (UTF-7) encoding:

 **Copy Code**

```
witadmin exportgloballist /collection:http://AdventureWorksServer:8080
```

Import the definition of global lists

The following example imports global lists:

 **Copy Code**

```
witadmin importgloballist /collection:http://AdventureWorksServer:80
```

The following example imports the same global lists to the same server, but uses Unicode (UTF-7) encoding:

 **Copy C**

```
witadmin importgloballist /collection:http://AdventureWorksServer:80
```

See Also

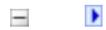
Concepts

[Define global lists](#)

Other Resources

[Customize global workflow](#)

[witAdmin: Customize and manage objects for tracking work](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Import and export global workflow [witadmin]

[See Also](#) [Send Feedback](#)

By using global workflow, you can minimize the work required to define and update work item fields and global lists that multiple team projects and types of work items share. With global workflow, you can define and update fields and global lists. You can manage the global workflow for a team project collection or a team project by using the following commands in the **witadmin** command-line tool:

- **exportglobalworkflow**: Exports the global workflow to an XML file or the Command Prompt window.
- **importglobalworkflow**: Imports global workflow from an XML file.

To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

For more information about global workflow, see [Customize global workflow](#).

Requirements

- To export a global workflow, you must be a valid user of the team project or team project collection.

- To import a global workflow to a team project, you must be a member of the following security groups: Team Foundation Administrators, Project Collection Administrators, or Project Administrators.
- To import a global workflow to a team project collection, you must be a member of either the Team Foundation Administrators or Project Collection Administrators.

See [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative permissions, you must open an elevated Command Prompt window to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt window, choose Start, open the Command Prompt shortcut menu, and then choose Run as Administrator. For more information, see the following page on the Microsoft website: [User Access Control](#).

```
witadmin exportglobalworkflow /collection:CollectionURL [/p:ProjectName]
```

```
witadmin importglobalworkflow /collection:CollectionURL [/p:ProjectName]
```

Parameters

Parameter	Description
<code>/collection:CollectionURL</code>	<p>Specifies the URI of the collection. You must specify in the following format: http://ServerName:Port/VirtualDirectoryName/CollectionName</p> <p>If you do not specify a virtual directory, you must specify the URI in the following format: http://ServerName:Port/CollectionName.</p>
<code>/p:ProjectName</code>	<p>Optional. The team project for which you want to export the global workflow. This team project must be in the collection that you specified by using the <code>/collection</code> parameter. If you do not specify a team project, the global workflow is imported or exported for the collection.</p>
<code>/f:FileName</code>	<p>The path and the name of the XML definition file for workflow to export or import.</p> <p>Note</p> <p>If the client computer is running Windows Vista, you may not have permissions to certain folders. If you try to export a global list to a location where you do not have permissions, registry virtualization technology automatically redirects the exported file and saves it to the virtual store. For more information, see the following pages on the Microsoft site: Registry Virtualization and Common file and registry virtualization issues in Windows Vista. To avoid this redirection, you can export the file to a location where you have permissions.</p>

have permissions.

/e:Encoding

Optional. The name of a .NET Framework 2.0 encoding. The specified encoding will be used to export or import XML data. For example, /e utf-7 specifies Unicode (UTF-7) encoding. If you omit this parameter, **witadmin** attempts to detect the encoding and uses UTF-8 if detection fails.

/v

Optional. Validates the XML that defines the global workflow but does not import the definition file.

/exportgloballists

Optional. Exports the definitions of global lists that are used in workflow references. The definitions for global lists are embedded into the XML definition of the global workflow. If you do not specify this parameter, the definitions for global lists are omitted.

/? or help

Displays help about the command in the Command Prompt window.

Remarks

You can define work item fields by importing them through a global workflow. However, you cannot change the properties of existing fields by using global workflow. If you import a global workflow that does not contain a **FIELDS** element, all previously imported rules for global workflow will be deleted. Field definitions will not be affected.

Examples

Unless otherwise specified, the following values apply in each example:

- URI for the collection:
http://AdventureWorksServer:8080/tfs/DefaultCollection
- Team project: Contoso
- Port number for the server website: 8080

Export the definition of a global workflow for a team project

The following example exports the global workflow for a team project:



```
witadmin exportglobalworkflow /collection:http://AdventureWorksServe
```

Import the definition of a global workflow to a team project

The following example imports the global workflow to the Contoso team project:

```
witadmin importglobalworkflow /collection:http://AdventureWorksServe
```

Import the definition of a global workflow to a collection

The following example imports the global workflow to the collection:



```
witadmin importglobalworkflow /collection:http://AdventureWorksServe
```

See Also

Concepts

[Define global lists](#)

Other Resources

[Customize global workflow](#)

[witAdmin: Customize and manage objects for tracking work](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Import and export process configuration [witadmin]

[Send Feedback](#)

You customize the process configuration to modify the display of the Agile planning tool pages that you view through Team Web Access (TWA). A few additional tools require that you map workflow states to metastate mappings.

For an overview of how to customize the Agile planning tool pages, see [Configure and customize Agile planning tools for a team project](#).

Note

If you receive error message TF400917, an invalid configuration has occurred in TWA. Re-import the process configuration file to your team project using **witadmin importcommonprocessconfig**. You'll get the messages you need to resolve the error.

To manage the process configuration for a team project, use the **witadmin** command line tool to export and import the process configuration XML definition file. To learn about process configuration, see [Process configuration XML element reference](#).



- **exportprocessconfig**: Exports the process configuration definition to an XML file or the Command Prompt window.
- **importprocessconfig**: Imports the process configuration definition XML file.

To run the **witadmin** command-line tool, open a Command Prompt window

where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

Requirements

- To export process configuration definitions, you must be a valid user of the team project or collection.
- To import process configuration definitions, you must be a member of the following security groups: Team Foundation Administrators or Project Administrators.

For more information, see [Permission reference for Team Foundation Server](#).

```
witadmin exportprocessconfig /collection:CollectionURL /p:ProjectName
```

```
witadmin importprocessconfig /collection:CollectionURL [/p:ProjectName]
```

Parameters

Parameter	Description
<code>/collection:CollectionURLx</code>	<p>Required. Specifies the URL of the collection. You must specify the URI in the following format:</p> <p>http://ServerName:Port/VirtualDirectoryName/CollectionName</p> <p>If you do not specify a virtual directory, you must specify the URI in the following format:</p> <p>http://ServerName:Port/CollectionName.</p>
<code>/p:ProjectName</code>	<p>Required. Specifies the team project for which you want to export or import the process configuration. This team project must be defined in the collection that you specified in the <code>/collection</code> parameter. You do not need to specify a virtual directory when you specify the <code>/v</code> switch.</p>
<code>/f:FileName</code>	<p>The path and the name of the XML definition file for the process configuration to export or import.</p> <p> Note</p> <p>If the client computer is running Windows Vista, you may not have permissions to certain folders. If you try to export the global list to a location where you do not have permissions, registry virtualization technology automatically redirects the exported file and saves it to the virtual store. For more information, see the following pages on the Microsoft Support website: Registry Virtualization and Common file and registry virtualization issues in Windows Vista. To avoid this redirection, you can export the file to a location where you have permissions.</p>

have permissions.

/e:Encoding

Optional. The name of a .NET Framework 2.0 encoding. The specified encoding will be used to export or import XML data. For example, `/e utf-7` specifies Unicode (UTF-7) encoding. If you omit this parameter, **witadmin** attempts to detect the encoding and uses UTF-8 if detection fails.

/v

Optional. Validates the XML that defines the process configuration but does not import the definition file.

/? or help

Displays help about the command in the Command window.

Remarks

Note

Installing VS TFS 2013 upgrades existing team projects. They will use the process configuration file, which combines the agile configuration files and the common configuration files that are supported in TFS 2012. The following **witadmin** commands have been deprecated along with their corresponding configuration files:

- **exportagileprocessconfig**
- **exportcommonprocessconfig**
- **importagileprocessconfig**
- **importcommonprocessconfig**

If you are updating an upgraded team project, see Configure features after a TFS upgrade.

If you encounter problems accessing existing test plans or test suites after an upgrade to TFS 2013.3, see Update a team project manually to support test management.

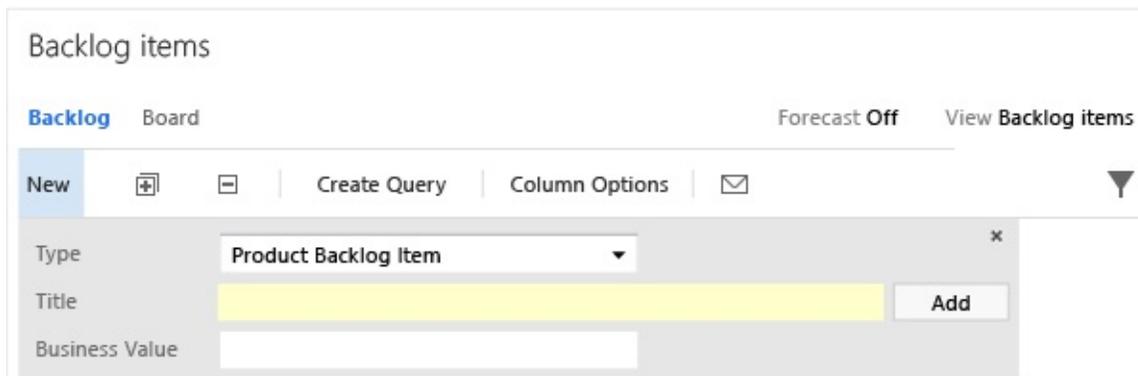
Examples

The following values apply in each example:

- URL for the collection:
http://AdventureWorksServer:8080/tfs/DefaultCollection
- Team project: Contoso
- Port number for the server website: 8080

To add a field to the quick add panel

You can add fields for any quick add panel. For example, the following example adds Business Value to the product backlog panel.



The screenshot shows the 'Backlog items' quick add panel. At the top, there are tabs for 'Backlog' and 'Board', and buttons for 'Forecast Off' and 'View Backlog items'. Below the tabs is a 'New' button and a toolbar with a plus icon, a minus icon, and links for 'Create Query', 'Column Options', and an envelope icon. The main area of the panel is a form with three fields: 'Type' (a dropdown menu set to 'Product Backlog Item'), 'Title' (a text input field highlighted in yellow with an 'Add' button to its right), and 'Business Value' (an empty text input field).

The panel only displays fields that are included in the FIELDS section of the WIT definition for the WIT selected. For example, if you select the bug WIT, then only Title displays, as Business Value isn't defined for bugs. To add another WIT to the panel, you add it to the Requirements Category as described here.

1. If you don't have administrative permissions for your team project, get them.
2. Open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace %programfiles(x86)% with %programfiles%. You can

[download Team Explorer](#) for free.

- Export the process configuration file.

Copy Code

```
witadmin exportprocessconfig /collection:CollectionURL /p:ProjectName
```

An example of a CollectionURL is
<http://MyServer:8080/tfs/TeamProjectCollectionName>.

- Add the field reference name to the AddPanel section.

For example, the following syntax adds the Priority element to the panel.

Copy Code

```
<AddPanel>  
  <Fields>  
    <Field refname="System.Title" />  
    <Field refname="Microsoft.VSTS.Common.BusinessValue" />  
  </Fields>  
</AddPanel>
```

Tips:

- You can look up the reference name for a field using this index.
- Add all required fields for work item types defined for the Requirements Category. That way, you avoid having to open the work item form to fill them in when you add backlog items through the panel.
- Import the process configuration file.

 **Copy Code**

```
witadmin importprocessconfig /collection:CollectionURL /p:ProjectName
```

- Refresh your backlog page to view the updated panel.

■ Update metastate mappings to support Test Manager

If you customize the WORKFLOW section of the test plan or test suite, you must map the states to metastates. This supports Test Manager clients that are at Visual Studio 2013.2 or earlier versions.

In the following example, the test plan workflow has been updated to use the Design, Testing, and Signed Off states. To support backward compatibility, the TestPlanWorkItems is added to the ProjectProcessConfiguration section of the process configuration definition.

WORKFLOW

 **Copy Code**

```
<WORKFLOW>
  <STATES>
    <STATE value="Design" />
    <STATE value="Testing" />
    <STATE value="Signed Off" />
  </STATES>
  <TRANSITIONS>
    <TRANSITION from="" to="Design">
      <REASONS>
        <DEFAULTREASON value="New test plan" />
      </REASONS>
    </TRANSITION>
    <TRANSITION from="Design" to="Testing">
      <REASONS>
        <DEFAULTREASON value="Authoring complete" />
      </REASONS>
    </TRANSITION>
    <TRANSITION from="Testing" to="Signed Off">
      <REASONS>
        <DEFAULTREASON value="Signed Off testing" />
      </REASONS>
    </TRANSITION>
  </TRANSITIONS>
</WORKFLOW>
```

```
</TRANSITION>
<TRANSITION from="Signed Off" to="Design">
  <REASONS>
    <DEFAULTREASON value="Reactivating to authoring phase"
  </REASONS>
</TRANSITION>
<TRANSITION from="Signed Off" to="Testing">
  <REASONS>
    <DEFAULTREASON value="Reactivating to run tests" />
  </REASONS>
</TRANSITION>
<TRANSITION from="Testing" to="Design">
  <REASONS>
    <DEFAULTREASON value="Back to authoring" />
  </REASONS>
</TRANSITION>
</TRANSITIONS>
</WORKFLOW>
```

If you modify the test suite workflow, then you have to make a similar update if you want to map new states. You would add it within a TestSuiteWorkItems section. See

[Process configuration XML element reference.](#)

Q & A

Q: What customizations can I make and still use the Configure Features Wizard to update my team project after a TFS upgrade?

A: You can customize the quick add panel. The Configure Features Wizard will update your team projects and you'll get access to the latest features.

Other changes might require you to perform some manual operations when updating your team project. To learn about which customizations you can safely make and which you should avoid, see [Customize the work tracking experience: Before you customize, understand the maintenance and upgrade implications.](#)

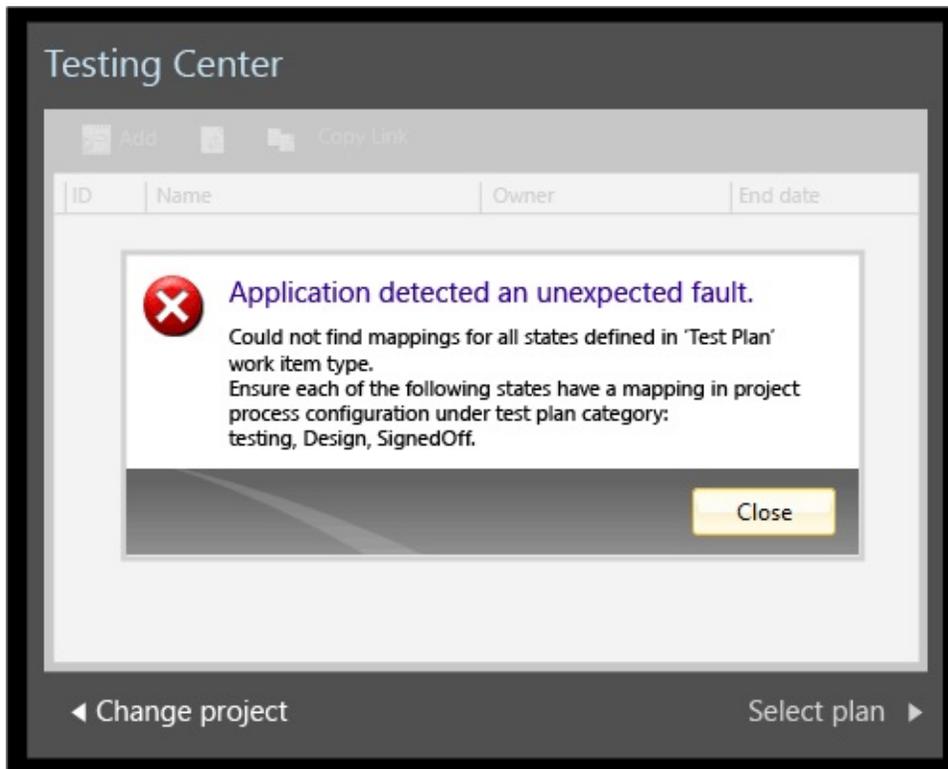
Q: When do I need to map workflow states to metastates?

A: When you add or remove workflow states to the following WITs, you should consider updating the process configuration to add or remove corresponding metastate mappings.

- WITs that belong to the Requirement Category or Task Category: Metastate mappings support the display of the Agile planning tools.
- WITs that belong to the Bug Category: Metastate mappings to support My Work tool (Agile and CMMI-based team projects).
- Test Plan and Test Suite: Updates to the workflow of these WITs must be mapped only when you support team members connecting to TFS from a version of Test Manager that is based on Visual Studio 2013.2 or earlier version.

Test Plan and Test Suite WITs are only available with the upgrade to TFS 2013.3, available from this [download page](#).

Update the metastate mappings if you receive an Application detected an unexpected fault error when you connect to your team project.



How do I resolve process configuration errors?

A: When you make one of the following changes to a team project, you'll need to update the definitions for the WIT, categories, or process configuration. To avoid errors, always make your changes in this order: (1) WITs, (2) categories, and (3) process configuration..

Customization

Update or verify the WIT definition

Update or verify the process configuration definition

To include the following fields:

- Backlog Priority (Scrum) or Stack Rank (Agile or CMMI)

To contain the necessary metastate mappings:

Add a WIT to the Requirements Category
(A WIT can belong to the Requirements Category or the Task Category, but not both.)

(must match the field value assigned to type=Order in the process configuration file)

- Effort (Scrum), Story Points (Agile), or Size (CMMI)

(must match the field value assigned to type=Effort in the process configuration file)

- Area path or the field value assigned to type=Team in the process configuration file

- All fields that are included in the AddPanel section of the process configuration file (fields must be defined within the FIELDS section but don't have to be included within the FORM section.

- Map the start of each workflow state to type="Proposed"

- Map each intermediate workflow state you want to have show up on the Kanban board to type="InProgress"

- Map the end of each workflow state to type="Complete"

You can have only one State mapped to type="Complete"

To contain an entry to define the color codes associated with the WIT. For example:

```
<WorkItemColor  
primary="FF009CCC"  
secondary="FFD6ECF2"  
name="Product Backlog  
Item" />
```

Change the workflow of a WIT in the N/A

To contain the necessary metastate mappings as described above for

Requirements Category

adding a WIT to the Requirements Category.

To include the following fields:

- Backlog Priority (Scrum) or Stack Rank (Agile or CMMI)
- Activity (Scrum or Agile) or Discipline (CMMI)

(must match the field value assigned to type=Activity in the process configuration file)

- Remaining Work
(must match the field value assigned to type=RemainingWork in the process configuration file)

- Area path or the field value assigned to type=Team in the process configuration file

- (Optional) Original Work and Completed Work (Agile and CMMI only)

To contain the necessary metastate mappings:

- Map the start of each workflow state to type="Proposed"
- Map each intermediate workflow state that you want to have show up on the task board to type="InProgress"
- Map the end of each workflow state to type="Complete"

You can have only one State mapped to type="Complete"

To contain an entry to define the color codes associated with the WIT. For example:

```
<WorkItemColor  
primary="FFF2CB1D"  
secondary="FFF6F5D2"  
name="Task" />
```

Add a WIT to the Task Category

Change the workflow of a WIT in the Task Category N/A

To contain the necessary metastate mappings as described above for adding a WIT to the Task Category.

Add a WIT to the Bug Category (Agile and CMMI only)

To contain the necessary metastate mappings:

- Map the start of each workflow state to type="Proposed"
- Map each intermediate workflow state you want to have show up for My Work to type="InProgress"
- Map the end of each workflow state type="Complete"

Change the workflow of a WIT in the Bug Category (Agile and CMMI only) N/A

You can have only one State mapped to type="Complete"

To learn more, see Support bug update status using My Work.

Remove a WIT from the Requirements Category or N/A

To remove any metastate mappings that are only associated with that WIT

Task Category

Remove a WIT from a team project

To remove the WIT from the categories file.

To remove any metastate mappings that are only associated with the WIT that you removed and the WorkItemColor element that defines the color codes for the WIT you removed.

Q: How do I customize other functions that appear on the Agile planning tools in TWA?

A: Some customizations can be done through the user interface. Others require editing process configuration or other team project objects. For an overview, see [Configure and customize Agile planning tools for a team project](#).

Q: Do you want to work with two or more portfolio backlogs?

A: The default experience supports one level of portfolio backlog. You can add up to five levels as described in [Add a backlog to Agile portfolio management](#).

Q: Do you want to add or change the WITs that appear on your task board or product backlog?

A: If you've added a custom WIT and want to add that to either the backlog or task board, you can. You just can't have them appear in both places. Learn how by reading [Add bugs or other work item types to backlogs or boards](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Manage link types [witadmin]

[See Also](#) [Send Feedback](#)

Similar to work item types, you can define and modify custom link types. However, you can't make any changes to system-defined link types. Before you add a new link type to your team project collection, review the link types available for your use. See [Link work items to support traceability](#).

You manage the link types defined for a team project collection by using the following **witadmin** commands:

- **deactivatelinktype:** Deactivates the specified link type. Users will no longer be able to create links of this type. Existing links of this type will continue to function correctly.
- **deletelinktype:** Permanently removes the specified link type from the database. All links defined with this link type are also removed.
- **exportlinktype:** Exports the definitions of link types. You can export the definition of a single link type or all link types defined for the server.
- **importlinktype:** Imports the definitions of link types from an XML file. If a link type with the same reference name already exists, it will be updated to match the imported link type. If the link type does not already exist, a new link type will be created.
- **listlinktypes:** Lists the available set of link types on a server.
- **reactivatelinktype:** Reactivates the specified link type, and optionally assigns it a new name.

To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

Requirements

For the team project collection where the link types are defined, you must have the following permissions set:

- To list or export link types, you must have your View project-level information permission for a team project in the collection set to Allow.
- To delete, import, or reactive link types, you must be a member of the Project Collection Administrators security group or have your Manage work item link types permission set to Allow.

For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative permissions, you must open an elevated Command Prompt window to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt window, choose Start, open the Command Prompt shortcut menu, and then choose Run as Administrator. For more information, see the Microsoft Web site: [User Access Control](#).

```
witadmin deactivatelinktype /collection:CollectionURL /n:LinkName
```

```
witadmin deletelinktype /collection:CollectionURL /n:LinkName [/nopr
```

```
witadmin exportlinktype /collection:CollectionURL [/n:LinkName] [/f:
```

```
witadmin importlinktype /collection:CollectionURL /f:FileName [/e:Er
```

```
witadmin listlinktypes /collection:CollectionURL
```

```
witadmin reactivatelinktype /collection:CollectionURL /n:LinkName
```

Parameters

Parameter	Description
<code>/collection:CollectionURL</code>	<p>Specifies the URI of the team project collection. The the URI is the following: http://ServerName:Port/VirtualDirectoryName/CollectionName</p> <p>For example: http://ServerName:8080/tfs/Collection0</p> <p>If a virtual directory is not used, then the format for the the following: http://ServerName:Port/CollectionName</p>
<code>/n:LinkName</code>	<p>The name or the reference name of the link type to delete.</p>
<code>/f:FileName</code>	<p>The XML file of link types. Required for import, optional for export. If you omit this parameter, the command output will be displayed on the display.</p>
<code>/e:encoding</code>	<p>The name of a .NET Framework 2.0 encoding format specified encoding will be used to export or import the data. For example, <code>/e:utf-7</code> specifies Unicode (UTF-7) encoding. If you omit this parameter, witadmin attempts to detect the encoding, and if detection fails, witadmin uses UTF-8.</p>

/v	Validates the link type XML without importing it.
/noprompt	Disables the prompt for confirmation.
/? or help	Displays help about the command in the Command P window.

Remarks

For the structure of the link type definition schema, see [Link type element reference](#).

For information about how you can restrict the types of link relationships that team members can make, see [Link controls, restrictions, and field reference](#) and [LinksControlOptions Elements](#).

Examples

Unless otherwise specified, the following values apply in each example:

- URI for the team project collection:
`http://AdventureWorksServer:8080/tfs/DefaultCollection`
- Server name: AdventureWorksServer
- Input or output file name: myLinkTypes.xml
- Link type name: mylinktype
- Default encoding: UTF-8

List Link Types

The following command displays the custom link types defined for Team Foundation AdventureWorksServer.

 **Copy Code**

```
witadmin listlinktypes /collection:http://AdventureWorksServer:8080/
```

List the Definition of a Link Type

The following example displays the definition of the link type in the Command Prompt window:

 **Copy Code**

```
witadmin exportlinktype /collection:http://AdventureWorksServer:8080/
```

Deactivate and then Reactivate a Link Type

The following examples deactivate the link type, mylinktype, and then reactivate

it:

 **Copy Code**

```
witadmin deactivatelinktype /collection:http://AdventureWorksServer:  
witadmin activatelinktype /collection:http://AdventureWorksServer:8080
```

Export the Definition of a Link Type

The following command exports the definition of all link types to the file, mylinktype.xml:

 **Copy Code**

```
witadmin exportlinktype /collection:http://AdventureWorksServer:8080
```

Import the Definition of Link Types

The following example imports the definition of the link types defined in the XML file:

 **Copy Code**

```
witadmin importlinktype /collection:http://AdventureWorksServer:8080
```

See Also

Concepts

[Link type element reference](#)

[Link controls, restrictions, and field reference](#)

Other Resources

Link work items to support traceability

[witAdmin: Customize and manage objects for tracking work](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Manage work item fields [witadmin]

[See Also](#) [Send Feedback](#)

You can manage the work item type fields that are defined for a team project collection by using the following **witadmin** commands:

- **changefield:** Changes one or more attributes of a field. When you change one of the following attributes, you change it for all work item types and team projects within the team project collection:
 - Data type for PlainText or HTML fields.

⚠Important

When you upgrade Team Foundation Server from an earlier version to the current version, the type assignment for the Description (System.Description) field is automatically converted from PlainText to HTML. With the changefield command, you can restore the content of this field to display plain text.

- Friendly name that displays in the work item query. This name may differ from that displayed on the work item form.
 - Reporting attributes which includes the name of the field as it appears in a report, the reference report name, and the reporting type.
 - Synchronization with Active Directory - you can enable/disable synchronization of person name fields.
- **deletefield:** Deletes the specified field.
 - **indexfield:** Turns indexing on or off for the specified field. When you

enable indexing for a field, you may increase the performance of finding work items whose queries specify that field. If you add a custom field that you use in many of your work item queries, you may want to enable indexing for that field.

- **listfields:** Lists the attributes for all fields or a specified field.

To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

For an overview of the fields defined within the default process templates provided with Team Foundation Server, see [Work item field reference for Visual Studio ALM](#).

Note

You can create and modify work item fields by using Process Editor, a power tool for Visual Studio. This tool is not supported. For more information, see the following page on the Microsoft Web site: [Team Foundation Server Power Tools](#).

Requirements

- To list fields, you must have your View project-level information permission for the team project in the collection set to Allow.
- To delete or rename fields or change an attribute of a field, you must be a member of the Team Foundation Administrators security group or the Project Collection Administrators security group.

For more information, see [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative permissions, you must open an elevated Command Prompt window to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt window, choose Start, open the Command Prompt shortcut menu, and then choose Run as Administrator. For more information, see the Microsoft Web site: [User Access Control](#).

```
witadmin changefield /collection:CollectionURL /n:RefName  
    [/name:NewName]  
    [/syncnamechanges:true | false]  
    [/reportingname:ReportingName]  
    [/reportingrefname:ReportingRefName]  
    [/reportingtype:Type]  
    [/reportingformula:Formula]  
    [/type:PlainText | HTML]  
    [/noprompt]
```

```
witadmin deletefield /collection:CollectionURL /n:RefName [/noprompt]
```

```
witadmin indexfield /collection:CollectionURL /n:Name /index:on|off
```

```
witadmin listfields /collection:CollectionURL /n:RefName [/unused]
```

Parameters

Parameter	Description
/collection: CollectionURL	Specifies the URI of the team project collection. If the URI is the following: http://ServerName:Port/VirtualDirectory If no virtual directory is used, then the following: http://ServerName:Port/CollectionName
/n: RefName	The reference name of a work item type
/n: Name	
/index	Specifies to enable or disable indexing for the field. Specify on to enable indexing and off to disable indexing.
/name: NewName	Specifies the new name for the field.
/syncnamechanges	Specifies to use the work item field to synchronize the name of the field as changes are made in Active Directory. This option is valid only when the type of String is specified for the type name. Specify true to enable synchronization for the field and false to disable synchronization for the field.

/reportingname:ReportingName Specifies the name of the field in the data for reporting.

/reportingrefname:ReportingRefName Specifies the reference name of the field to be used for reporting.

/reportingtype:Type Specifies how the field is used in the way. The following values are valid:

- **dimension:** Used for the Integer, String fields.
- **detail:** Used for the Integer, Double fields.
- **measure:** Used for the Integer and default aggregation type is sum. You can change aggregation type by using the **form**
- **none:** Used to disable reportability

For more information, see [Define and manage fields](#)

/reportingformula:Formula Specifies the aggregation formula to be used when reported as a measure. The only supported

/type:HTML | PlainText Specifies to convert the contents of the field from HTML or from HTML to PlainText. You can use this option only for fields whose type is assigned as HTML. See [FIELD \(Definition\) element](#)

/unused Lists all fields that are not used by any tool in the team project collection.

/noprompt

Disables prompt for confirmation.

/? or help

Displays help about the command in the window.

Remarks

Indexed fields

A query index is created based on those fields that have indexing enabled. This index improves the response time when running queries that include indexed fields.

By default, the following fields are indexed: Assigned To, Created Date, Changed By, State, Reason, Area ID, Iteration ID, and Work Item Type. If there are other fields that your team frequently uses in their queries, you can add them to the query index.

Synchronizing person names with Active Directory

You must manually enable synchronization of any custom work item fields that are used to assign person names that reference Active Directory. You must enable synchronization for each field for each team project collection that contains the custom fields.

All system reference fields that show person-names have the attribute `syncnamechanges` set to true. Such fields include `System.AuthorizedAs`, `System.AssignedTo`, `System.ChangedBy` and `System.CreatedBy`. Synchronization is enabled for each person name field that is defined in one of the default process templates. For more information, see [Assignments and workflow field reference](#).

After synchronization is enabled, the field no longer shows a static string. Instead, the field shows the name associated with a user account. When you change the user name in Active Directory or in Workgroup, a field with `syncnamechanges` set to true automatically shows the new name.

When you assign the `syncnamechanges` attribute to a String field, the field always accepts valid user names. However, the field does not allow group names that are stored in Team Foundation Server or in Active Directory if any one of the following conditions is true:

- The VALIDUSER rule is specified across all work item types
- The VALIDUSER rule is specified for a work item type
- The ALLOWEDVALUES rule is specified for a work item type, and that rule has a filter criteria that excludes groups

For more information, see [All FIELD XML elements reference](#).

▣ **Attributes that you can change for each work item type**

You change the following attributes or values defined for a field by changing the work item type definition in which the field appears:

- Name that displays on the work item form. See [Control XML element reference](#).
- Help text. See [Apply a rule to a work item field](#).
- Allowed values or items within a pick list or drop-down menu. See [Define pick lists](#).

Examples

Unless otherwise specified, the following values apply in each example:

- URI for the team project collection:
`http://AdventureWorksServer:8080/tfs/DefaultCollection`
- Work item field name: `AdventureWorks.Field`
- Default encoding: UTF-8

List fields

Use **witadmin listfields** to see the set of fields in use, to select one to add to a work item type. Also, you can list the attribute assignments defined for a specific field and determine which fields are used by which team projects.

View the attributes of a work item field

- Enter the following command to list the attributes defined for a specified field, such as `Microsoft.VSTS.Common.Issue`.

 **Copy**

```
witadmin listfields /collection:http://AdventureWorksServer:8080
```

Field and attribute information appears for the named field, as shown in this example.

 **Copy Code**

```
Field: Microsoft.VSTS.Common.Issue
Name: Issue
Type: String
Reportable As: dimension
Use: Adventure Works (Shared Steps), AW Future (Shared Steps), /
Indexed: False
```

The Use parameter indicates the name of each project and the work item type where the field is used. For more information about field attributes, see [Work item field reference for Visual Studio ALM](#).

List all fields in a team project collection

- Enter the following command to list all fields defined for a team project collection.

Copy Code

```
witadmin listfields /collection:http://AdventureWorksServer:8080
```

Field information for all the fields for the named project collection appears. See [Work item field reference for Visual Studio ALM](#).

List fields that are not being used

- Enter the following command to list the fields that are no longer being used in the team project collection by any work item type.

Copy Code

```
witadmin listfields /collection:http://AdventureWorksServer:8080
```

Field and attribute information appears for each field that is not being used, as shown in this example.

Copy Code

```
Field: Microsoft.VSTS.CMMI.TaskType  
Name: Task Type  
Type: String  
Reportable As: dimension  
Use: Not In Use  
Indexed: False
```

```
Field: Microsoft.VSTSUE.Common.Flag  
Name: Flag  
Type: String
```

Reportable As: dimension
Use: Not In Use
Indexed: False

Field: Microsoft.VSTSUE.Common.Progress
Name: Progress
Type: String
Reportable As: dimension
Use: Not In Use
Indexed: False

▣ Rename a field

You can change the friendly name of a work item field to meet the naming conventions that your team uses. Note that the new name is applied to all work item types that reference the changed field in all team projects in the project collection. The friendly name displays when you define filter criteria in a work item query. The name that appears on a work item form may be different than the friendly name defined for the field.

To rename a work item field

1. Enter the following command to rename the friendly name assigned to Microsoft.VSTS.Common.Rank to Important Rank.

```
witadmin changefield /collection:http://AdventureWorksServer:80:
```

2. At the confirmation prompt type **y** and then press ENTER. Wait until you receive the following message:

Field renamed.

To verify changes imported to a single project

1. In Team Explorer, choose  Refresh.

The latest updates are downloaded from the server, which include the changes that you just made to the field name. Wait several seconds for the

refresh to finish.

2. Choose New Query to create a query.
3. In the Query Editor, choose the Click here to add a clause link to add a row, select the blank Field cell, and in the cell, type Rank. The following message that appears above the results list. This message indicates that the Rank cannot be found.

Run the query to see the query results. TF51005: The query references a field that does not exist. The error is caused by <<Rank>>.

4. Delete the value Rank from the Field cell, and type Important Rank into the cell.
5. Select <> in the Operator cell, and type 1 into the Value cell.
6. On the Query toolbar, choose ► Run.
7. Open the shortcut menu for any row in the results and select Column Options. Scroll down in the Available columns list. Notice that the Rank field is no longer present but the Important Rank field is present.
8. Select Important Rank in the Available columns box, and then choose the > button (add selected columns). Choose OK.

Notice that the friendly name for Microsoft.VSTS.Common.Rank has been renamed from Rank to Important Rank throughout the query builder and results list.

9. Close the query. Choose No when you are prompted to save the query.
10. Create a new Task work item. Choose the New Work Items link, and then choose Task.

This work item is created from the work item type that you changed and imported.

11. Notice, in the Status box, that the label for the renamed field, Rank, has not changed. This is because the field labels on the work item forms are scoped

to the parent team project and are independent of the server-wide field name just specified.

Note

For more information about how to change field labels on work item forms, see [Control XML element reference](#).

12. Close the new Task and choose No when you are prompted to save the work item.

Change the report as value for a field

The following command specifies the ability to report the type of the DateTime field AdventureWorks.CreatedOn to dimension. This field's data enters the warehouse and Analysis Services databases so that it can be used to filter reports.

```
witadmin changefield /collection:http://AdventureWorksServer:8080/t1
```

The following command specifies the ability to report the type of the Double field AdventureWorks.Field to measure. All measures are aggregated by sum.



```
witadmin reportfield /collection:http://AdventureWorksServer:8080/t1
```

Enable synchronization of a custom person-name field

The following command enables synchronization for the work item field AW.CustomerName defined for Collection1 on the AdventureWorksServer.

Verify the data type of the field that you want to convert

- Verify the data type assigned to the field, such as

MyCompany.CustomerName, that you want to synchronize by entering the following command:



```
witadmin listfields /collection:http://AdventureWorksServer:8080
```

Enable synchronization

1. To enable synchronization for a person-named field, type the following command, substituting your data for the arguments that are shown here:

```
witadmin changefield /collection:http://AdventureWorksServer:8080
```

2. The following confirmation prompt appears:

This will change properties for field {0} on the Team Foundation Server. Do you want to continue?

3. Type **0** to confirm that you want to change the field, or **1** to cancel the request.

If the change request succeeds, the following confirmation message appears:

The field was updated.

If the change request fails, an error message appears. The most common mistakes that can be made are trying to change a system reference field, or trying to change a field of a data type other than String. These operations are not supported.

Delete a field

Before you delete a field, verify that the field is not in use. If the field is in use, you must first remove the field from the work item types that use it prior to

deleting it from the team project collection. The following command deletes the field AdventureWorks.Field from Collection1:

 **Copy Code**

```
witadmin deletefield /collection:http://AdventureWorksServer:8080/tf
```

Verify a field is not in use

1. Enter the following command, specifying the reference name for the work item field, such as MyCompany.MyProcess.MyField.

 **Copy**

```
witadmin listfields /collection:http://AdventureWorksServer:8080
```

In the information displayed for the field, verify that the value for **Use** is "**Not In Use**" as follows in this example.

 **Copy Code**

```
Field: MyCompany.MyProcess.MyField  
Name: MyField  
Type: String  
Reportable As: dimension  
Use: Not In Use  
Indexed: False
```

2. If the **Use** field indicates that the field is in use, then you must delete it from each work item type for each team project that is listed. For example, the Microsoft.VSTS.TCM.SystemInfo field indicates that it is being used by the Bug and Code Defect work item types for four projects: Arroyo, Desert, Palm, and Springs.

 **Copy Code**

```
Field: Microsoft.VSTS.TCM.SystemInfo  
Name: System Info  
Type: Html
```

Reportable As: None

Use: Arroyo (Bug), Desert (Bug), Palm (Bug), Springs (Bug, Code
Indexed: False

Before you can delete this field, you must remove it from each of the work item types listed for each team project for which it is defined. To remove the field, you modify the definition for the work item type by deleting the FIELD and Control elements that contain the field reference name. See [Import, export, and manage work item types \[witadmin\]](#), [FIELD \(Definition\) element reference](#), and [Control XML element reference](#).

Delete a field from a team project collection

- Enter the following command to delete the MyCompany.MyProcess.MyField field, and then choose Enter.

 **Copy Code**

```
witadmin deletefield /collection:http://AdventureWorksServer:80:
```

Enter **y** at the confirmation prompt to complete this step.

Q & A

Q: What customizations can I make and still use the Configure Features Wizard to update my team project after a TFS upgrade?

A: You can add custom fields, customize a pick list, and add rules to a field. The Configure Features Wizard will update your team projects and you'll get access to the latest features.

Changing field attributes is not recommended. To learn about which customizations you can safely make and which you should avoid, see [Customize the work tracking experience](#): Before you customize, understand the maintenance and upgrade implications.

See Also

Concepts

Resolve schema conflicts that are occurring in the data warehouse

Add or modify work item fields to support reporting

[Work item field reference for Visual Studio ALM](#)

[Reportable fields reference for Visual Studio ALM](#)

Other Resources

[witAdmin: Customize and manage objects for tracking work](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Remove work items permanently [witadmin]

[See Also](#) [Send Feedback](#)

You can permanently remove one or more work items from the Team Foundation database for a team project collection by using the **witadmin destroywi** command. Work items whose state is set to Closed remain in the database and can be reactivated. Permanently removed work items are removed from the database and cannot be restored nor reactivated.

Each work item represents an object that is stored in the Team Foundation database and that is assigned a unique identifier, which is referred to as a work item ID. Work item IDs are unique within a project collection.

To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

Requirements

- You must be a member of the Team Foundation Administrators security group or the Project Administrators security group for the team project collection. See [Permission reference for Team Foundation Server](#).

Note

Even if you log on with administrative permissions, you must open an elevated

Command Prompt window to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt window, choose Start, open the shortcut menu for Command Prompt, and choose Run as Administrator. For more information, see the [Microsoft Web site](#).

```
witadmin destroywi /collection:CollectionURL /id:id [/noprompt]
```

Parameters

Parameter	Description
/collection: CollectionURL	Specifies the URI of the team project collection. The the URI is the following: http://ServerName:Port/VirtualDirectoryName/Colle
/id: id	If no virtual directory is used, then the format for the following: http://ServerName:Port/CollectionName. The ID of a work item to destroy. To specify multiple items, separate IDs using only commas, without whit
/noprompt	Disables the prompt for confirmation.
/? or help	Displays help about the command in the Command P window.

Examples

Permanently Remove Work Items From the Database

The following example deletes the work item 2003 from the database for Collection1 on the AdventureWorksServer server:

 **Copy Code**

```
witadmin destroywi /collection:http://AdventureWorksServer:8080/tfs,
```

The following example deletes the work items with IDs, 12, 15, and 23 from the database for Collection1 on the AdventureWorksServer server:

 **Copy Code**

```
witadmin destroywi /collection:http://AdventureWorksServer:8080/tfs,
```

See Also

Other Resources

Resolve, close, or reactivate a work item [redirected]

[witAdmin: Customize and manage objects for tracking work](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Rebuild the client cache [witadmin]

[See Also](#) [Send Feedback](#)

You can force a rebuild of the cache on each client computer the next time it connects to a team project collection by using the **witadmin rebuildcache** command.

To prevent workspace errors from occurring during version control or build operations in Team Foundation, the data cache on client computers must be updated after certain maintenance operations. After you move, restore, rename, or fail over a data-tier or application-tier server, you must refresh the cache for tracking work items and users must refresh the version control cache on client computers.

Important

To avoid server performance issues, you should not run this command during normal operating hours.

To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

Requirements

- To use the **witadmin rebuildcache** command, you must be a member of the Team Foundation Administrators security group or the Project Administrators security group for the project collection that you want to manage. [Permission reference for Team Foundation Server](#).

☑ **Note**

Even if you log on with administrative permissions, you must open an elevated Command Prompt window to perform this function on a server that is running Windows Server 2008. To open an elevated Command Prompt window, choose Start, open the shortcut menu for Command Prompt, and choose Run as Administrator. For more information, see the [Microsoft Web site](#).

```
witadmin rebuildcache /collection:CollectionURL [/noprompt]
```

Parameters

Parameter	Description
/collection: CollectionURL	Specifies the URI of the team project collection. The the URI is the following: http://ServerName:Port/VirtualDirectoryName/Colle
/noprompt	If a virtual directory is not used, then the format for tl the following: http://ServerName:Port/CollectionName.
/? or help	Disables the prompt for confirmation.
	Displays help about the command in the Command P window.

Remarks

The **witadmin rebuildcache** command invalidates cached data on all clients for a specified team project collection. This causes the cache for each client to be refreshed the next time the client connects to the project collection.

Examples

The following command invalidates the metadata cache for all clients that connect to DefaultCollection that is defined on the server that is named AdventureWorksServer. The client caches are updated the next time they connect to the project collection.

 **Copy Code**

```
witadmin rebuildcache /collection:http://AdventureWorksServer:8080/1
```

See Also

Tasks

Refresh your Team Foundation client

Concepts

Choose the Team Foundation client to support your tasks

Other Resources

[witAdmin: Customize and manage objects for tracking work](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Work item field and database schema reference

[Send Feedback](#)

This reference helps you understand how fields are used to track progress and build reports. You can look up a description of each field used to track work items and learn which fields are set to be reportable by default.

Also, you can understand the structure of the relational warehouse and Analysis Services cube for Team Foundation Server.

Index of work item fields

- Titles, IDs, descriptions, and history
- Areas and iterations
- Assignments and workflow
- Planning, ranking, and priorities
- Effort, schedules, and tracking work
- Build integration
- Link controls,



restrictions, and fields

- Attachment controls and fields
- Fields that support integration with test, build, and version control

Reportable field reference

- Data warehouse components
- [Table reference for the relational warehouse database for Visual Studio ALM](#)
- Perspectives and measure groups provided in the Analysis Services cube
 - Shared dimensions
 - Builds
 - Code churn and code coverage
 - Test results

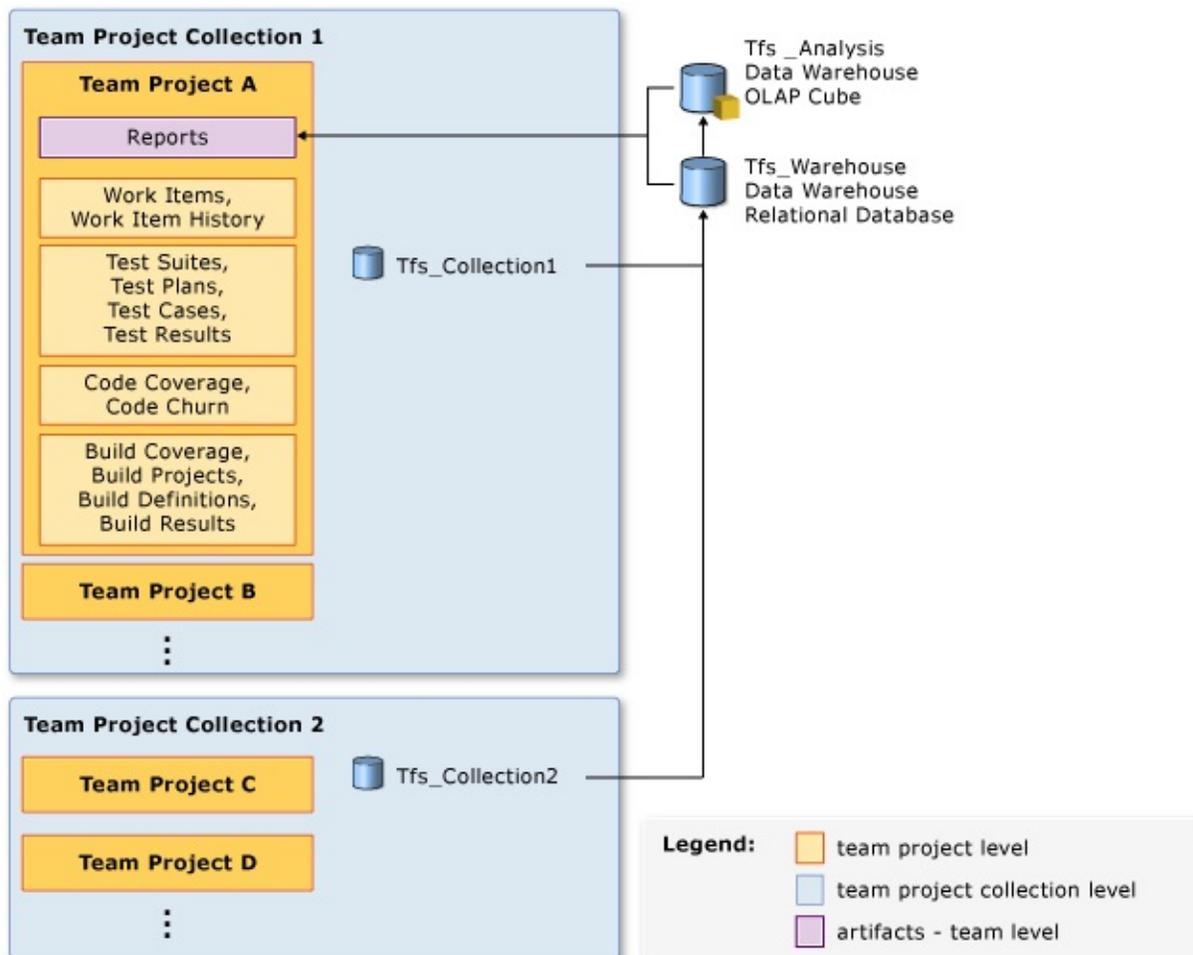


- Test cases
and work
item tracking

Required activities to monitor progress and generate useful reports

For a list of out-of-the-box (OOB) reports and activities team members must perform to generate useful reports, see Reports (SQL Server Reporting Services).

OOB reports, customized reports, and dashboards are built from data written to the relational warehouse database and OLAP cube. In addition to work item data, the warehouse contains data about builds, source code, test results and code coverage. All data captured for all team projects is written to the data store for the team project collection. All data for all team project collections is written to the relational warehouse database and OLAP cube.



For information about customizing reports, see [Create, customize, and manage reports for Visual Studio ALM](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Work item field reference for Visual Studio ALM

[See Also](#) [Send Feedback](#)

By using the index of fields in this topic, you can look up a description of each field used to track work items. This reference includes all fields defined within the default process templates for Team Foundation Server (TFS). You use two main types of fields to track data for a type of work item. System fields are defined for all work item types, and all other fields are created by adding FIELD element definitions for them to a work item type. When the work item type is uploaded to Team Foundation, either when a team project is created or by other means, all new fields are added to the set of fields that are defined for the team project collection. For example, when you create a team project that uses the process template for Microsoft Solutions Framework (MSF) for Agile Software Development, all fields that are defined in each Agile work item type are added to the team project collection of data fields.

You should use as many of the system and existing project collection fields as possible to make your work item type more portable between project collections. To support additional tracking needs, you can define your own custom work item fields. For more information, see

[Define and modify work item fields.](#)

Note

You can list the attributes of fields using the **witadmin listfields** command. Also, you can change the field name, the index, and the report attributes for any field except system fields by using the **witadmin** command-line tool. For more information, see [Manage work item fields \[witadmin\]](#).

You can view the attribute assignments made to all fields defined in a team project collection using the Process Editor, a power tool for Visual Studio. For more information, see the following page on the Microsoft website: [Team](#)

[Foundation Server Power Tools.](#)

Alphabetical index

Values in parenthesis indicate fields that are either System fields used only by one of the default process templates: Scrum, Agile, or CMMI, or are associated with test case management (TCM). For a list of fields that are written to the relational database or data warehouse cube, see

[Reportable fields reference for Visual Studio ALM.](#)

- Acceptance Criteria (Scrum)
- Accepted By
- Accepted Date
- Activated By
- Activated Date
- Activity
- Changed By (System)
- Changed Date (System)
- Closed By (System)
- Closed Date (System)
- Closed Status
- Closed Status Code
- Closing Comment
- Comments (CMMI)
- Committed (CMMI)
- Completed
- Integrated in Build (TCM)
- Issue (TCM)
- Iteration ID (System)
- Iteration Path (System)
- Justification (CMMI)
- Link Comment (System)
- Link Description (System)
- Local Data Source
- Reviewed By
- Reviewed Date
- Rev (System)
- Risk (Agile)
- Root Cause (CMMI)
- Severity
- Size (CMMI)
- Stack Rank
- Start Date

• Actual Attendee 1-8 (CMMI)	Work (TCM)	• State (System)
• Analysis (CMMI)	• Contingency Plan (CMMI)	• Meeting Type (CMMI)
• Application Launch Instructions	• Corrective Action Actual Resolution (CMMI)	• State Change Date
• Application Start Information	• Corrective Action Plan (CMMI)	• State Code
• Application Type	• Created By (System)	• Steps (TCM)
• Area ID (System)	• Created Date (System)	• Steps to Reproduce (TCM)
• Area Path (System)	• Discipline (CMMI)	• Story Points (Agile)
• Assigned To	• Description (System)	• Subject Matter Expert 1-3 (CMMI)
• Associated Context	• Due Date (Agile)	• Symptom (CMMI)
• Associated Context Code	• Effort (Scrum)	• System Info (TCM)
• Associated Context Owner	• Escalate (CMMI)	• Tags
• Associated Context Type	• Finish Date	• Target Date
• Attached File Count	• Found In	• Target Resolve Date (CMMI)
• Authorized As (Not used)		
• Automated Test Id (TCM)		
• Automated Test Name (TCM)		

- Automated Test Storage (TCM)
- Automated Test Type (TCM)
- AutomatedTestId (TCM)
- AutomatedTestName (TCM)
- Automation Status (TCM)
- Backlog Priority (Scrum)
- Blocked
- Business Value
- Called By (CMMI)
- Called Date (CMMI)
- Build (TCM)
- Found In Environment (CMMI)
- History (System)
- How Found (CMMI)
- Hyperlink Count
- ID (System)
- Impact Assessment (CMMI)
- Impact on Architecture (CMMI)
- Impact on Development (CMMI)
- Impact on Technical Publications (CMMI)
- Impact on Test (CMMI)
- Impact on User Experience
- Rating
- Reason (System)
- Related Link Count (System)
- Remaining Work
- Repro Steps
- Required Attendee 1-8 (CMMI)
- Requirement Type (CMMI)
- Requires Review (CMMI)
- Requires Test (CMMI)
- Resolution (Scrum)
- Resolved By
- Resolved Date
- Resolved Reason
- Task Type (CMMI)
- Team Project (System)
- Test Suite Audit (TCM)
- Test Suite Type (TCM)
- Test Suite Type ID (TCM)
- Title (System)
- Triage (CMMI)
- User Acceptance Test (CMMI)
- Work Item Type (System)

(CMMI)

By using the system fields or other fields you have added to your project collection, you can enable meaningful cross-team project reports and queries. In addition, any non-system field that is referenced in the workflow or forms section of the work item type definition must have a FIELD element that defines it in the FIELDS section of the work item type definition XML file. Also, you must specify any non-system field that you might want to use to generate a query or report in the FIELDS section.

Field reference topics

The following topics describe fields that used in common by all work item types, or those that are specific to just a one or a few work item types.

Fields common to many work types:

- Titles, IDs, Descriptions, and History
- Areas and Iterations
- Assignments and Workflow
- Planning, Ranking, and Priorities
- Effort, Schedules, and Tracking Work
- Build Integration
- Link Controls, Restrictions, and Fields

Fields used by specific work item types:

- Code Review Request
- Code Review Response
- Feedback Request
- Feedback Response
- Shared Steps
- Test Case

Fields used to track CMMI work items:

- Requirements
- Bugs
- Change Requests
- Issues
- Review Meetings
- Risks

- Attachment
Controls and
Fields

See Also

Reference

[Manage work item fields \[witadmin\]](#)

Concepts

[Reportable fields reference for Visual Studio ALM](#)

[Define and modify work item fields](#)

[Work with team project artifacts, choose a process template](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Reportable fields reference for Visual Studio ALM

[See Also](#) [Send Feedback](#)

A default set of fields appears in the relational warehouse database or the cube. The following tables list those reportable fields defined in the default process templates provided with the current release of Team Foundation Server. These fields have a reportable attribute value of Detail, Dimension, or Measure. See [Add or modify work item fields to support reporting](#).

For a complete list of fields that are defined in the default process templates, see

[Work item field reference for Visual Studio ALM](#). If you upgraded a team project, you may need to perform additional tasks before you can work with some of these fields. For more information, see [Configure features after a TFS upgrade](#).

Detail fields

Detail fields are written to the relational warehouse database, but not to the cube.

Field name	Description	Reference name	Data type
Backlog Priority	A number that represents the relative priority for a bug, product backlog item, or task. (Scrum process template only.)	Microsoft.VSTS.Common.BacklogPriority	Double
Revised Date	The date and time stamp when a test case or shared step is revised. <input checked="" type="checkbox"/> Note This field	System.RevisedDate	DateTime

is not
included
in the
work item
form and
is not
populated
with any
data.

Dimension fields

Dimension fields are written both to the relational warehouse database and to the cube.

Field name	Description	Reference name	Data
Accepted By	The name of the person who responded to the code review.	Microsoft.VSTS.CodeReview.AcceptedBy	String
Accepted Date	The date and time when the person responded to the code review.	Microsoft.VSTS.CodeReview.AcceptedDate	Date
Activated By	The name of the team member who activated or reactivated the work item.	Microsoft.VSTS.Common.ActivatedBy	String
Activated Date	The date and time when the work item was activated or reactivated.	Microsoft.VSTS.Common.ActivatedDate	Date
	The type of activity that is		

Activity	required to perform a task.	Microsoft.VSTS.Common.Activity	String
Area Path	Groups the work items into product feature or team areas. The area must be a valid node in the project hierarchy.	System.AreaPath	Tree
Assigned To	The name of the team member who currently owns the work item.	System.AssignedTo	String
Automation Status	The status of a test case. You can specify the following values: <ul style="list-style-type: none"> • Not Automated • Planned 	Microsoft.VSTS.TCM.AutomationStatus	String
Blocked	Indicates that progress toward resolving a bug, change request, requirement, or risk is suspended. Allowed values are Yes and No.	Microsoft.VSTS.CMMI.Blocked	String

(CMMI process template only)

Changed By

The name of the team member who modified the work item most recently.

System.ChangedBy

String

Changed Date

The date and time when the work item was modified.

System.ChangedDate

Date

Closed By

The name of the person who closed a work item.

Microsoft.VSTS.Common.ClosedBy

String

Closed Date

The date and time when a work item was closed.

Microsoft.VSTS.Common.ClosedDate

Date

The status selected by the reviewer when closing the code review request. The number is stored in the system and written to the data warehouse as follows:

Closed Status	<ul style="list-style-type: none"> • 0 - Not Reviewed • 1 - Looks Good • 2 - With Comments • 3- Needs Work • 4 - Declined • 5 - Removed 	Microsoft.VSTS.CodeReview.ClosedStatus	String
Created By	The name of the team member who created the work item.	Microsoft.VSTS.Common.CreatedBy	String
Created Date	The date and time when a work item was created.	Microsoft.VSTS.Common.CreatedDate	Date
Due Date	The forecasted due date by which an issue will be resolved. (Agile process template only)	Microsoft.VSTS.Scheduling.DueDate	Date
	The date and time when the		

Finish Date	schedule indicates that the task will be completed.	Microsoft.VSTS.Scheduling.FinishDate	Date
Found In	The product build number, also known as a revision, in which a bug was found.	Microsoft.VSTS.Build.FoundIn	String
ID	The unique identifier that is assigned to a work item. Work item IDs are unique across all team projects and work items that are defined in a team project collection.	System.Id	Integer
Integration Build	The product build number that incorporates the code or fixes a bug.	Microsoft.VSTS.Build.IntegrationBuild	String
	Indicates that the shared step is associated with an expected result. Allowed values are Yes		

Issue	and No.	Microsoft.VSTS.Common.Issue	Strin
Iteration Path	<p> Note</p> <p>This field is not included in the work item form and is not populated with any data.</p> <p>Groups the work items by named sprints or time periods. The iteration must be a valid node in the project hierarchy.</p> <p>A subjective rating of the bug, issue, task, or test case as it relates to the business. You can specify the following values:</p> <ul style="list-style-type: none"> • 1: Product cannot ship without the successful resolution of the work item, and it should be 	System.IterationPath	Tree

Priority

addressed as soon as possible.

Microsoft.VSTS.Common.Priority

Integ

- 2: Product cannot ship without the successful resolution of the work item, but it does not have to be addressed immediately.
- 3: Resolution of the work item is optional, based on resources, time, and risk.

The number of stars that an item receives from a reviewer in a star-based ranking system. (Feedback Response)

The number is stored in the system and

Rating	<p>written to the data warehouse as follows:</p> <ul style="list-style-type: none"> • 0 - Not Rated • 1 - Poor • 2 - Fair • 3- Good • 4- Very Good • 5 - Excellent 	Microsoft.VSTS.Common.Rating	Strin
Reason	<p>The reason that the work item is in the current state. Values are specific to both the state and the type of work item. The field is not tracked for test cases or shared steps.</p>	System.Reason	Strin
Resolved By	<p>The name of the team member who resolved the bug or user story.</p>	Microsoft.VSTS.Common.ResolvedBy	Strin
	<p>The date and time</p>		

Resolved Date	when the bug or user story was resolved.	Microsoft.VSTS.Common.ResolvedDate	Date
Resolved Reason	The reason that the bug was resolved (for example, it was fixed).	Microsoft.VSTS.Common.ResolvedReason	String
Rev	A number that is assigned to the historical revision of a work item.	System.Rev	Integer
Risk	<p>A subjective rating of the relative uncertainty about the successful completion of the user story. You can specify the following values:</p> <ul style="list-style-type: none"> • 1 - High • 2 - Medium • 3 - Low <p>A subjective rating of the effect of a bug, issue, or risk on</p>	Microsoft.VSTS.Common.Risk	String

Severity	<p>the project. You can specify the following values:</p> <ul style="list-style-type: none"> • 1 - Critical • 2 - High • 3 - Medium • 4 - Low 	Microsoft.VSTS.Common.Severity	String
Stack Rank	<p>A subjective rating of the user story, task, issue, or bug compared to other work items of the same type. An item that is assigned a lower number should be fixed before an item that is assigned a higher number.</p>	Microsoft.VSTS.Common.Rank	Double
Start Date	<p>The date and time when the schedule indicates that the task will start.</p>	Microsoft.VSTS.Scheduling.StartDate	Date
State	<p>The current state of the work item. The valid values for state are</p>	System.State	String

	specific to each type of work item.		
Team Project	The team project to which this work item belongs.	System.TeamProject	String
Title	A short description that summarizes what the work item is and helps users to distinguish it from other work items in a list.	System.Title	String
Work Item Type	The name of the work item type.	System.WorkItemType	String

Measure fields

Measure fields are written to the data warehouse and contain pre-calculated sum values.

Field name	Description	Reference name	Data type
Business Value	A subjective unit of measure that captures the relative business value of a product backlog item (Scrum process template only) or feature compared to other items of the same type. An item that is assigned a higher number	Microsoft.VSTS.Common.BusinessValue	Integer

should be considered as having more business value than an item that is assigned a lower number.

Completed Work A measure of the amount of work spent on a task. Microsoft.VSTS.Scheduling.CompletedWork Double

Effort The relative rating for the amount of work that a product backlog item will require to implement. (Scrum process template only) Microsoft.VSTS.Scheduling.Effort Double

Original A measure of the amount of work that is Microsoft.VSTS.Scheduling.OriginalEstimate Double

Estimate	required to complete a task.		
Remaining Work	A measure of the amount of work that remains to finish a task.	Microsoft.VSTS.Scheduling.RemainingWork	Double
Size	The relative rating for the amount of work that a requirement will require to implement. (CMMI process template only) A subjective unit of measure that captures the size of a user story. If you	Microsoft.VSTS.Scheduling.Size	Double

Story
Points

assign more Microsoft.VSTS.Scheduling.StoryPoints
points to a
user story,
you
indicate
that more
work is
required to
implement
it. (Agile
process
template
only)

Double

See Also

Concepts

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Titles, IDs, Descriptions, and History field reference

[See Also](#) [Send Feedback](#)

You use the Title and ID fields to uniquely identify work items in a list. You use the Description and History fields to provide additional information that is needed to implement the work and to track changes. These fields track information for all work item types defined in the process templates provided with Team Foundation Server.

After a work item is created, you can modify each of these fields except for ID. When you create and save a work item, the ID is assigned by Team Foundation and cannot be changed.

The Description, History, Steps to Repro and Title fields are automatically indexed for full-text search. See [Query fields, operators, values, and variables](#).

Fields that appear on work item forms

The following table describes the fields that are used in tracking detailed information and historical revisions made for a work item. For information about data types and default field attributes, see

[Define and modify work item fields.](#)

Field name	Description	Reference name	Data type	Default value of the reportable type attribute
ID	The unique identifier that is assigned to a work item. Work item IDs are unique across all team projects and within a team project collection.	System.Id	Integer	Dimension
	A short description that			

Title	<p>summarizes what the work item is and helps team members distinguish it from other work items in a list.</p>	System.Title	String Dimension
Description	<p>The long description of a work item. This field provides more details about the work item than the title provides.</p> <p> Note</p> <p>Upon upgrade to Team Foundation Server 2012, the Description field was changed from a field type of PlainText to HTML.</p>	System.Description	HTML None

Using the **witadmin changefield** command you can revert the data type for this field. See [Manage work item fields \[witadmin\]](#).

The record of changes that were made to the work item after it was created. Every time that the work item is updated, information is appended to the history, which specifies the date of the change, who made the changes, and which fields were changed.

History

System.History

History None

You can also add formatted text to the history field.

Repro Steps

The steps that are required to reproduce unexpected behavior. (bugs only)

Microsoft.VSTS.TCM.ReproSteps HTML None

System Info

Information about the software and system configuration that is relevant to the bug, code review, or feedback.

Microsoft.VSTS.TCM.SystemInfo HTML None

Fields that support revision tracking

You can use the fields in the following table to filter queries and create reports. These fields are updated with information every time that a work item is modified. These fields do not appear on the work item form, but they are tracked for all types of work items.

Field name	Description	Reference name	Data type	Default value of the reportable type attribute	Default value of the index attribute
Changed By	The name of the team member who modified the work item most recently.	System.ChangedBy	String	Dimension	True
Changed Date	The date and time when the work item was modified most recently.	System.ChangedDate	DateTime	Dimension	False
	A number				

Rev

that is
assigned to
the
historical
revision of
a work
item.

System.Rev

Integer

Dimension False

Additional fields that support query and reporting

You can use the fields in the following table to filter queries and create reports. The following fields do not appear on work item forms, but they are tracked for all types of work items.

Field name	Description	Reference name	Data type	Default value of the reportable type attribute	Default value of the index attribute
Team Project	The team project to which this work item belongs.	System.TeamProject	String	Dimension	False
Work Item Type	The name of the work item type.	System.WorkItemType	String	Dimension	True

See Also

Concepts

Query for work items

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Areas and iterations field reference

[See Also](#) [Send Feedback](#)

Area and iteration fields support categorizing work into product areas and iteration or sprint cycles. These fields display entries in a hierarchical or tree structure. These structures are useful for assigning work to specific project areas and time durations, as well as filtering project information in charts and reports.

You can assign the area and iteration fields to most work item types (WITs) that are provided with the default process templates for Team Foundation Server. You specify values for the area and iteration fields when you create a work item or during a review of the product or iteration backlog. If you defer a work item to a later time, you should change its iteration accordingly.

Classification fields that appear on work item forms

These fields appear on the forms for all WITs that TFS process templates provide. For information about data types and default field attributes, see

[Define and modify work item fields.](#)

You can restrict who can modify work items under an area path. To define area and iteration paths or set permissions, see [Add and modify area and iteration paths.](#)

Field name	Description	Reference name	Data type	Default value of the reportable type attribute	Default value of the index attribute
Area Path	Groups work items into product feature or team areas. The area must be a valid node in the project hierarchy.	System.AreaPath	TreePath	Dimension	True
	Groups work items				

Iteration Path	<p>by named sprints or time periods. The iteration must be a valid node in the project hierarchy.</p>	System.IterationPath TreePath Dimension True
----------------	---	--

If you define a path name that is longer than 256 characters, you will not be able to specify it in Microsoft Project. To avoid this problem, define path names of no more than 10 characters, and do not nest nodes more than 14 levels deep.

You can't apply most field rules to the System.AreaPath and System.IterationPath fields. To learn more, see [Apply a field rule](#).

For additional restrictions, see [restrictions in terms of naming and structuring child nodes](#).

Additional fields tracked for query and reporting purposes

The following fields do not appear on work item forms but are tracked for each work item type. These fields provide a numeric value for each classification value that is defined for a team project. You can use these fields to filter queries and create reports.

Field name	Description	Reference name	Data type	Default value of the reportable type attribute	Default value of the index attribute
Area ID	The unique ID of the area to which this work item is assigned.	System.AreaId	Integer	None	True
Iteration ID	The unique ID of the iteration to which this work item is assigned.	System.IterationId	Integer	None	True
Node Name	The name of the leaf node of an area path. For example, if the area path is Project\A1\B2\C3, the node name is	System.NodeName	String	None	False

|

C3.

See Also

Concepts

Schedule sprints

Add and modify area and iteration paths

[Define the initial areas and iterations in the classification plug-in](#)

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Assignments and workflow field reference

[See Also](#) [Send Feedback](#)

You use status and assignment fields to track the progress of user stories, tasks, bugs, and other types of work items. These fields support many team queries and reports. Most work items move from a New, Active, or Proposed state to a Done or Closed state. As each work item moves from one state to another, the item might also be reassigned to various members of the team. For example, a tester might create a bug that is assigned to another team member during triage. When the other team member resolves the bug, it is reassigned to the tester who created it. Workflow state diagrams differ among the default process templates provided with Team Foundation Server (TFS), see [Work with team project artifacts](#), choose a process template.

Tip

You can view state diagrams for a workflow using the Process Editor, a power tool for Visual Studio available here: [Team Foundation Server Power Tools](#).

The Assigned To and State fields are tracked for each type of work item that is provided with the default TFS process templates.

Fields that appear on work item forms

The following table describes the fields that track assignments and workflow progress. For information about data types and default field attributes, see

[Define and modify work item fields.](#)

Field name	Description	Reference name	Data type	required
	The name of the team member who currently owns the work item.			
	The default list of names available in the drop-down menu for this system-defined, person-name field contains all accounts added to TFS, referred to as the valid users group.			

Assigned
To

However, some shortcut menus that support assignment from the backlog or board pages in Team Web Access are automatically limited to team members. Also, these names are automatically synchronized with Active Directory when it is configured as part of the deployment. See [Synchronization and person-name fields](#) and [Drop-down menu of the Assigned To field](#).

System.AssignedTo

String Di

The reason why the work item is in the current state.

Values are defined within the WORKFLOW

Reason	<p>section of the WIT definition using the REASON element. To modify the defined reasons, see Change the workflow for a work item type.</p> <p>This field is not tracked for Test Cases or Shared Steps.</p> <p>The reason why a work item was resolved. For example, the user story is code complete or the bug is fixed.</p>	System.Reason	String Di
Resolved Reason	<p>This field is read-only and only valid for Agile and CMMI work item types.</p> <p>The current state of the work item. This field allows you to update the</p>	Microsoft.VSTS.Common.ResolvedReason	String Di

status of a work item as it progresses from new or active to a done or closed state.

State

Values are defined within the WORKFLOW section of the WIT definition using the STATE element. To modify the states, see [Change the workflow for a work item type](#).

System.State

String Di

■ Synchronization of person-name fields

By default, TFS synchronizes system-defined person-name fields with Active Directory. These fields include: Activated By, Assigned To, Closed By, Created By, and Resolved By. You can grant access to TFS by adding security groups that you created in Active Directory or by adding accounts to existing or custom groups defined in TFS. See

[Set up groups for use in TFS deployments.](#)

You can enable or disable synchronization for a person-name field by using the **witadmin changefields** command-line tool. You can also synchronize custom person-name fields by specifying the **syncnamechanges** attribute. See [Manage work item fields \[witadmin\]](#) and [FIELD \(Definition\) element reference](#).

Drop-down menu of the Assigned To field

By default, the drop-down menu of the Assigned To field displays all users that have been added to TFS, whether as a member of a group or an individual member. Also, TFS shows the display name and adds the account name when required to disambiguate identical display names.

To minimize the list of names that appear in the drop-down menus of person-name fields, you can scope the field to only those TFS groups that you want to appear in the menu. You do this by adding one or more of the following child elements to the FIELD definition in the work item type definition: ALLOWEDVALUES, PROHIBITEDVALUES, and VALIDUSER. See

[All FIELD XML elements reference](#) and [Define pick lists](#).

Additional fields that support query and reporting

Additional fields are populated with information as a work item progresses from one state to another. These fields do not appear on the work item form, but they are tracked for each type of work item as the following table indicates. You can use these fields to filter queries and create reports.

Field name	Description	Reference name	Data type	Display name
Activated By	The name of the team member who created the work item or changed its status from closed, completed, or done state to a new or active state.	Microsoft.VSTS.Common.ActivatedBy	String	Display name
	The date and time			

Activated Date	when the work item was created or when its status was changed from closed, completed, or done to a new or active state.	Microsoft.VSTS.Common.ActivatedDate	DateTime	Di
Closed By	The name of the team member who set the state to closed, completed, or done.	Microsoft.VSTS.Common.ClosedBy	String	Di
Closed Date	The date and time when a work item was closed.	Microsoft.VSTS.Common.ClosedDate	DateTime	Di
Created By	The name of the team member who created the work item.	Microsoft.VSTS.Common.CreatedBy	String	Di

Created Date	The date and time when a work item was created.	Microsoft.VSTS.Common.CreatedDate	DateTime	Di
Resolved By	The name of the team member who resolved the work item.	Microsoft.VSTS.Common.ResolvedBy	String	Di
Resolved Date	The date and time when the work item was moved into a resolved or done state.	Microsoft.VSTS.Common.ResolvedDate	DateTime	Di
State Change Date	The date and time when the value of the State field changed.	Microsoft.VSTS.Common.StateChangeDate	DateTime	N

See Also

Concepts

Change the workflow for a work item type

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Planning, ranking, and priorities field reference

[See Also](#) [Send Feedback](#)

You use planning, ranking, and priority fields to specify which work the team should complete first. By ranking and prioritizing work items, all team members gain an understanding of the relative importance of the work that they must accomplish. Ranking and priority fields are used to build several reports.

You rank and prioritize work items when you create the product backlog.

Tracking fields that appear on work item forms

The following table describes the fields that you can use to plan and prioritize work and that are common to all default process templates. For information about data types and default field attributes, see

[Define and modify work item fields.](#)

Field name	Description	Reference name	Data type
Business Value	A subjective unit of measure that captures the relative business value of a product backlog item or feature compared to other items of the same type. An item that is assigned a higher number should be considered as having more business value than an item that is assigned a lower number.	Microsoft.VSTS.Common.BusinessValue	Integer

A subjective rating of the bug, issue, task, or test case as it relates to the business. You can specify the following values:

- 1: Product cannot ship without the successful resolution of the work item, and it should be addressed as soon as possible.

Priority

- 2: Product cannot ship without the successful resolution of the work item, but it does not need to be addressed immediately.

Microsoft.VSTS.Common.Priority

Integer D

- 3: Resolution of the work item is optional based on resources,

time, and risk.

See Note 1.

A subjective rating of the impact of a bug on the project. You can specify the following values:

Severity	<ul style="list-style-type: none">• 1 - Critical• 2 - High• 3 - Medium• 4 - Low	Microsoft.VSTS.Common.Severity	String D
----------	--	--------------------------------	----------

See Note 1.

Stack Rank (Note 2)	A subjective rating of the user story, task, issue, or bug compared to other work items of the same type. An item that is assigned a lower number should be fixed before an item that is assigned a higher number. (Agile and CMMI process templates)	Microsoft.VSTS.Common.StackRank	Double D
---------------------	---	---------------------------------	----------

See Note 3.

1. To change the menu selection, see [Define pick lists](#).
2. The Stack Rank field is defined for the following work item types in the Agile process template: Bug, Feature, Issue, Task, and User Story.

The Stack Rank field is defined for the following work item types in the CMMI process template: Bug, Feature, Requirement, Risk, and Task.

Depending on the process template version used to create your team project and the work item type, the Stack Rank field may not appear on the work item form. In most process template versions starting with the TFS 2013.3 update, the Stack Rank field was removed from most work item forms as described in this blog post: [Where is the field on the work item form to order the backlog?](#)

To add the field to the form, modify the work item type to add the following control element:

```
<Control FieldName="Microsoft.VSTS.Common.BacklogPriority"  
Type="FieldControl" Label="Backlog Priority" LabelPosition="Left" />
```

To learn more, see [Import, export, and manage work item types \[witadmin\]](#).

3. The sequence of items on the product backlog page is determined according to where you have added the items or dragged the items on the page. As you drag items, a background process updates this field which is assigned to type="Order" in the ProcessConfiguration file. See [Configure and customize Agile planning tools for a team project](#).

■ Scrum-specific fields

The following table describes the fields that you can use to plan and prioritize work when you use the work item types provided with the Visual Studio Scrum process template.

Field name	Description	Reference name	Data type
Acceptance Criteria	<p>A description of the criteria to be met before the bug or product backlog item can be closed.</p> <p>Before work begins on a bug or product backlog item, the criteria for customer acceptance should be described as clearly as possible. Conversations between the team and customers to define the acceptance criteria will help ensure that your team</p>	Microsoft.VSTS.Common.AcceptanceCriteria	HTML

understands your customers' expectations. The acceptance criteria can be used as the basis for acceptance tests so that you can more effectively evaluate whether an item has been satisfactorily completed.

Backlog
Priority
(Note 1)

A subjective rating of the bug, product backlog item, or task as it relates to the business.

Microsoft.VSTS.Common.BacklogPriority Double

See Note 12.

Blocked

Indicates whether a team member is prevented from making progress toward implementing

Microsoft.VSTS.CMMI.Blocked

String

a task.

You can specify Yes or leave it blank.

Resolution	Describes how the impediment was resolved.	Microsoft.VSTS.Common.Resolution	HTML
------------	--	----------------------------------	------

1. The Stack Rank field is defined for the following work item types in the Agile process template: Bug, Feature, Issue, Task, and User Story.

The Stack Rank field is defined for the following work item types in the CMMI process template: Bug, Feature, Requirement, Risk, and Task.

Depending on the process template version used to create your team project and the work item type, the Stack Rank field may not appear on the work item form. In most process template versions starting with the TFS 2013.3 update, the Stack Rank field was removed from most work item forms as described in this blog post: [Where is the field on the work item form to order the backlog?](#)

To add the field to the form, modify the work item type to add the following control element:

```
<Control FieldName="Microsoft.VSTS.Common.StackRank"
Type="FieldControl" Label="Stack Rank" LabelPosition="Left" />
```

To learn more, see [Import, export, and manage work item types \[witadmin\]](#).

2. The sequence of items on the product backlog page is determined according to where you have added the items or dragged the items on the page. As you drag items, a background process updates the Stack Rank field which is assigned to type="Order" in the ProcessConfiguration file. See Configure and customize Agile planning tools for a team project.

Agile specific fields

The following table describes the fields that you can use to plan and prioritize work when you use the work item types defined for the Agile process template.

Field name	Description	Reference name	Data type	Default value of the reportable type attribute	Default value of the index attribute
Risk	<p>A subjective rating of the relative uncertainty around the successful completion of a user story.</p> <p>Defined allowed values are:</p> <ul style="list-style-type: none"> • 1 - High • 2 - Medium • 3 - Low 	Microsoft.VSTS.Common.Risk	String	Dimension	False

Formal or CMMI-specific fields

The following table describes the fields that you can use to plan and prioritize

work when you use the work item types provided with the MSF formal or CMMI process template.

Field name	Description	Reference name	Data type Default value of the reportable type attribute
Blocked	<p>Indicates whether a team member is prevented from making progress toward implementing a requirement or task or resolving a bug, change request, or risk. If an issue has been opened to track a blocking problem, a link should be made to the issue.</p> <p>You can specify Yes or No.</p>	Microsoft.VSTS.CMMI.Blocked	String Dimension

Indicates whether the requirement is committed in the project or not. You can specify Yes or

Committed No.

Microsoft.VSTS.CMMI.Committed String None

 **Note**

You can specify this field only for requirements.

Indicates whether the issue is affecting the critical path of the project plan. You can specify Yes or No.

Escalate

Microsoft.VSTS.CMMI.Escalate

String None

 **Note**

You can specify this field only for issues.

Indicates the type of triage decision that

is pending for the work item. You use this field when the work item is in the Proposed state.

You can specify one of the following values:

Triage

- Pending Microsoft.VSTS.Common.Triage String Dimension (default)
- More Info
- Info Received
- Triaged

Note

You can specify this field only for requirements, tasks, change requests, bugs, and issues.

Work item fields and work item types

You can use the tables provided below to look up which fields are tracked for the main work item types provided with each of the default TFS process templates.

Scrum

The following table indicates which data fields you can specify for work item types that are provided with the Visual Studio Scrum process template.

Field Name	Product				
	Feature	Backlog Item	Bug	Task	Impediment
Acceptance Criteria	✓	✓	✓	✗	✗
Blocked	✗	✗	✗	✓	✗
Business Value	✓	✓	✗	✗	✗
Effort	✗	✓	✓	✗	✗
Resolution	✗	✗	✗	✗	✓

Agile

The following table indicates which data fields you can specify for work item types that are provided with the MSF Agile process template.

Field Name	Feature	User Story	Task	Issue	Bug
Business Value	✓	✗	✗	✗	✗
Due Date	✗	✗	✗	✓	✗
Priority	✓	✗	✓	✓	✓
Risk	✓	✓	✗	✗	✗
Severity	✗	✗	✗	✗	✓
Story Points	✗	✓	✗	✗	✗

Formal or CMMI

The following table indicates which data fields you can specify for work item types that are provided with the MSF formal process template.

Field Name	Feature	Requirement	Task	Bug	Change Request	Issue	Risk
------------	---------	-------------	------	-----	----------------	-------	------

Business Value	✓	✗	✗	✗	✗	✗	✗
Blocked	✗	✓	✓	✓	✓	✗	✓
Committed	✗	✓	✗	✗	✗	✗	✗
Escalate	✗	✗	✗	✗	✗	✓	✗
Priority	✓	✓	✓	✓	✓	✓	✓
Probability	✗	✗	✗	✗	✗	✗	✓
Severity	✗	✗	✗	✓	✗	✓	✓
Triage	✓	✓	✓	✓	✓	✓	✗

See Also

Concepts

[Effort, schedules, and time tracking field reference](#)

[Work item field reference for Visual Studio ALM](#)

Other Resources

[Modify or add a field to support queries, reports, and workflow](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Effort, schedules, and time tracking field reference

[See Also](#) [Send Feedback](#)

You use the fields described in this topic to track estimated, completed, and remaining work for a backlog item, bug, or task and to report on the progress toward completing tasks. By tracking work, you gain more detailed insight into which user stories, requirements, or backlog items are almost finished and what the team burn rate is. Several reports and dashboards provide charts that display the data that these fields contain.

You specify the level of effort for the tasks that you are assigned. For more information about how to track work, see [Work in sprints](#).

Fields used to estimate and track work

The following table describes the fields that you can use to track work. This information is tracked only for task work items. For information about data types and default field attributes, see

[Define and modify work item fields.](#)

Field name	Description	Reference name
Activity	<p>The type of activity that is required to perform a task. (Agile and Scrum process templates)</p> <p>Allowed values (See note 1):</p> <ul style="list-style-type: none">• Deployment• Design• Development• Documentation• Requirements• Testing	Microsoft.VSTS.Common.Activity

Note

This field is also used to calculate capacity by activity. It is assigned to type="Activity" in the ProcessConfiguration file. (2)

Completed Work The amount of work that has been spent implementing a task. You can specify work in hours or in days. There are no inherent time units associated with this field.

Microsoft.VSTS.Scheduling.CompletedWork

The type of activity or discipline that is assigned to a task. (CMMI process template)

Allowed values (1):

- Analysis
- Development
- Test
- User Education
- User Experience

Discipline

Microsoft.VSTS.Common.Discipline

Note

This field is also used to calculate capacity by activity. It is assigned to type="Activity" in the ProcessConfiguration file. (2)

Due Date The forecasted due date for an issue to be resolved. (Agile process template only) Microsoft.VSTS.Scheduling.DueDate

Target Date The date by which a feature should be completed. (Feature work item type) Microsoft.VSTS.Scheduling.TargetDate

Effort A subjective unit of measure that captures the size of a bug or product backlog item. If you assign more effort to an item, you indicate that more work is required to implement it. (Scrum process template only) Microsoft.VSTS.Scheduling.Effort

This field is also used to calculate team velocity and forecasting. It is assigned to

type="Effort" in the
CommonConfiguration
file. (2, 3)

A subjective unit of
measure that captures
the size of a user story.
If you assign more
points to a user story,
you indicate that more
work is required to
implement it. (Agile
process template only)

Story
Points

Microsoft.VSTS. Scheduling.StoryPoints

This field is also used
to calculate team
velocity and
forecasting. It is
assigned to
type="Effort" in the
ProcessConfiguration
file. (2, 3)

A subjective unit of
measure that captures
the size of a
requirement. The larger
the size, the more work
is required to
implement it. (CMMI
process template only)

Size

Microsoft.VSTS. Scheduling.Size

This field is also used
to calculate team
velocity and
forecasting. It is
assigned to

type="Effort" in the ProcessConfiguration file. (2, 3)

Original Estimate

The amount of work required to complete a task. You can specify work in hours or in days. There are no inherent time units associated with this field. (Agile and CMMI process templates)

Microsoft.VSTS.Scheduling.OriginalEstimate

Remaining

The amount of work that remains to finish a task. You can specify work in hours or in days. There are no inherent time units associated with this field.

Microsoft.VSTS.Scheduling.RemainingWork

This field is also used to calculate burn down. It is assigned to type="RemainingWork" in the ProcessConfiguration file. (2)

Specifies the kind of task to implement. (CMMI process template)

Task Type Allowed values (1): Microsoft.VSTS.CMMI.TaskType

- Corrective Action
- Mitigation Action
- Planned

Notes:

1. To change the menu selection, see [Define pick lists](#).
2. To change the field assignment, see Configure and customize Agile planning tools for a team project.

Note: The values displayed in the Agile planning tool Capacity page reflect a union of all values defined for the field in all team projects within the project collection instance. Therefore, to restrict the values that appear for Capacity on the sprint backlog pages, you must make the values match in all the team projects for the field assigned to type="Activity".

3. For additional guidance, see the white paper about Estimating.

Fields calculated by Office Project

The following table lists fields that are calculated if you create a project plan in Microsoft Project and then synchronize that plan with tasks that are stored in TFS. These fields do not appear on the work item form, but they are calculated for those backlog items and tasks that are linked to backlog items. You can view their read-only values in results from a query or from Microsoft Excel and Microsoft Project. For more information, see [Create your backlog and tasks using Project](#).

Field name	Description	Reference name	Data type	Default value of the reportable type attribute
Finish Date	The date and time when the schedule indicates that the task will be completed.	Microsoft.VSTS.Scheduling.FinishDate	DateTime	Dimension
Start Date	The date and time when the schedule indicates that the task will start.	Microsoft.VSTS.Scheduling.StartDate	DateTime	Dimension

See Also

Concepts

Add or change how Project fields map to TFS fields

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Build and test integration field reference

[See Also](#) [Send Feedback](#)

You use Team Foundation Server (TFS) build and test data fields to perform the following actions:

- Associate bugs with the builds where they were found or fixed.
- Mark test cases as either manual or automated, and store information to support automated test cases.
- For test cases and shared steps, define the action and validation steps and the data that are used to perform tests.

Test Manager and the test work item types (WITs) use the following fields to track test plans, progress, and results. The availability of the WITs is based on the version of TFS installed on your application-tier. To learn more about using these WITs, see [Plan Manual Tests using Team Web Access](#).

TFS 2013.0	TFS 2013.2	TFS 2013.3
		<ul style="list-style-type: none">• Bug
<ul style="list-style-type: none">• Bug	<ul style="list-style-type: none">• Bug	<ul style="list-style-type: none">• Shared Parameters
<ul style="list-style-type: none">• Shared Steps	<ul style="list-style-type: none">• Shared Parameters	<ul style="list-style-type: none">• Shared Steps
<ul style="list-style-type: none">• Test	<ul style="list-style-type: none">• Shared Steps	<ul style="list-style-type: none">• Test Case

- Case
 - Test Case
 - Test Plan
 - Test Suite

To learn more about upgrading an existing team project to get WITs that your team project currently doesn't have, go [here](#).

Build and test data fields that appear on work item forms

The following table describes the fields that are defined in one or more of the test WITs. For information about data types and field attributes, see

[Define and modify work item fields.](#)

To customize a field or pick list, see [Modify or add a field to support queries, reports, and workflow.](#)

Field name	Description	Reference name
Automation Status (See Note 1)	<p>The status of a test case. You can specify the following values:</p> <ul style="list-style-type: none">• Not Automated• Planned	Microsoft.VSTS.TCM.AutomationStatus S
	<p>To run automated tests, see Automate a test case in Microsoft Test Manager.</p>	

Found In (Note 2)	Product build number, also known as a revision, in which a bug was found.	Microsoft.VSTS.Build.FoundIn	Si
Integration Build (Note 2)	Product build number that incorporates the code or fixes a bug.	Microsoft.VSTS.Build.IntegrationBuild	Si
Issue	Indicates that the Shared Steps is associated with an expected result. Allowed values are Yes and No. (Shared Steps only)	Microsoft.VSTS.Common.Issue	Si
Parameters(Note 3)	Contains the parameters to use when running a manual test. (Shared Parameters, Shared Steps, and Test Case)	Microsoft.VSTS.TCM.Parameters	H
Steps	The action and validation steps that are required to perform the test.	Microsoft.VSTS.TCM.Steps	Ti

System Info	Information about the software and system configuration that is relevant to the test.	Microsoft.VSTS.TCM.SystemInfo	H
-------------	---	-------------------------------	---

Steps to Reproduce	The steps that are required to reproduce unexpected behavior.	Microsoft.VSTS.TCM.ReproSteps	H
--------------------	---	-------------------------------	---

The test suite category. Allowed values are:

- Query Based:
Use to group together test cases that have a particular characteristic - for example, all the tests that have Priority=1.
The suite will automatically include every test case that is returned by the query that you define.

Test Suite Type
(Notes 1 and 4)

- **Static:** Use to group together test cases designed to track the test status of backlog items. Each test case that you add to a requirement-based test suite is automatically linked to the backlog item.
- **Requirement Based:** Use to group together test cases with any characteristics or test suites.

Microsoft.VSTS.TCM.TestSuiteType

Si

For more information, see Plan Manual Tests using Team Web Access.

Notes

1. Do not customize the pick list for these fields. The system accepts only those values listed.

2. By adding a GLOBALLIST element to the FIELD definition, you can provide a drop-down menu of builds that users can choose from. To learn how, see [Fields that support integration with test, build, and version control](#).
3. Requires TFS 2013.2 or TFS 2013.3 to be installed on the application-tier server and existing team projects to be updated to support Shared Parameters. To learn more, see [Configure features after a TFS upgrade](#).
4. Requires TFS 2013.3 to be installed on the application-tier server and existing team projects to be updated to support Test Plan and Test Suite.

Additional fields

The following fields do not appear on work item forms, but these fields are tracked for test cases or test suites. You can use some of these fields to filter queries and create reports.

Field name	Description	Reference name
Automated Test Storage	The assembly that contains the test that automates the test case.	Microsoft.VSTS.TCM.AutomatedTestI
Automated Test Type	The type of test that automates the test case.	Microsoft.VSTS.TCM.AutomatedTestI
AutomatedTestId	The ID of the test that automates the test case. (Test Case only)	Microsoft.VSTS.TCM.AutomatedTestI
AutomatedTestName	The name of the test that is used to automate the test case. (Test Case only)	Microsoft.VSTS.TCM.AutomatedTestI

LocalDataSource	The local data source that supports the test. (Test Case only)	Microsoft.VSTS.TCM.LocalDataSource
Query Text	Field used to capture the query defined for a Query-based suite type.	Microsoft.VSTS.TCM.QueryText
Test Suite Audit (See Note 1)	Tracks additional operations performed when modifying a test suite, for example: adding tests to a test suite or changing configurations. This field can be viewed through the History tab or through a separate query. There will be a consolidated history view, including changes performed to work items field and changes resulting from related artifacts such as test points and configurations.	Microsoft.VSTS.TCM.TestSuiteAudit

<p>Test Suite Type ID (Note 1, 2)</p>	<p>A system assigned value that corresponds to the test suite category and only applicable to test suites. Assigned values are:</p> <ul style="list-style-type: none"> • 1 (Static) • 2 (Query-based) • 3 (Requirement-based) 	<p>Microsoft.VSTS.TCM.TestSuiteTypeIc</p>
---	--	---

Notes

1. Requires TFS 2013.3 to be installed on the application-tier server and existing team projects to be updated to support Test Plan and Test Suite.
2. Do not customize the pick list for these fields. The system accepts only those values listed.

See Also

Concepts

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Code review and feedback field reference

[See Also](#) [Send Feedback](#)

You can use the code review and feedback fields to create queries and reports that track the status of these processes. The fields appear in the following work item types, which are included with the default process templates for Visual Studio Team Foundation Server 2013: Code Review Request, Code Review Response, Feedback Request, and Feedback Response.

 **Note**

If your Team Foundation Server has been upgraded from an earlier version you might need to update your team project to get access to these work item types. See [Configure features after a TFS upgrade](#)

Fields used to track code reviews

The following fields are used to track code review requests and responses. A code review response is created for each person who's been requested to provide review comments. See *Day in the life of an ALM Developer: Suspend work, fix a bug, and conduct a code review.*

Field name	Description	Reference name	Int
Accepted By	The name of the code reviewer.	Microsoft.VSTS.CodeReview.AcceptedBy	Stri
Accepted Date	The date and time when the code-reviewer responded.	Microsoft.VSTS.CodeReview.AcceptedDate	Dat
Associated Context	The name assigned to the code work requested for review.	Microsoft.VSTS.CodeReview.Context	Stri
Associated Context Code	An integer value that captures whether the code review is for 1 (shelveset) or 2	Microsoft.VSTS.CodeReview.ContextCode	Inte

(changeset).

Associated Context Owner	The GUID assigned to the shelve set owner who requested the code review.	Microsoft.VSTS.CodeReview.ContextOwner	Stri
--------------------------	--	--	------

Associated Context Type	The type of code work requested for review: Shelve set or Changeset.	Microsoft.VSTS.CodeReview.ContextType	Stri
-------------------------	--	---------------------------------------	------

Closed Status	<p>The status selected by the reviewer when closing the code review request. The number is stored in the system and written to the data warehouse as follows:</p> <ul style="list-style-type: none">• 0 - Not Reviewed• 1 - Looks Good• 2 - With	Microsoft.VSTS.CodeReview.ClosedStatus	Stri
---------------	--	--	------

Comments

- 3- Needs Work
- 4 - Declined
- 5 - Removed

Closed Status Code

A value ranging from 0 to 5 that corresponds to the status selected by the reviewer when closing the code review request.

Microsoft.VSTS.CodeReview.ClosedStatusCode Int

Closing Comments

The comment entered by the reviewer when closing the review request.

Microsoft.VSTS.CodeReview.ClosingComment Stri

Reviewed By

The name of the team member who reviewed the code. The State transitions to Reviewed when the code

Microsoft.VSTS.Common.ReviewedBy

Stri

reviewer
responds.
(Code Review
Response)

Reviewed Date	The date-time stamp when the reviewer closed the request. (Code Review Response)	Microsoft.VSTS.Common.ReviewedDate	Dat Tin
------------------	---	------------------------------------	------------

State Code	Mirror field used to track the current state in code.	Microsoft.VSTS.Common.StateCode	Inte
------------	--	---------------------------------	------

Fields used to track feedback

The following fields track feedback requests and responses. You complete the first three fields in the feedback request form. A feedback response is created for each person and for each item for which feedback is requested. See [Get feedback](#).

Field name	Description	Reference name
Application Launch Instructions	Instructions to stakeholders on how to start the application.	Microsoft.VSTS.Feedback.ApplicationLaunc
Application Start Information	Guidelines to direct stakeholder feedback.	Microsoft.VSTS.Feedback.ApplicationStartIr
Application Type	The type of application that stakeholders will provide feedback on. The valid types are specified in the process configuration file, ProcessConfiguration. The default values are Web Application, Remote Machine, and Client Application.	Microsoft.VSTS.Feedback.ApplicationType

The number of stars that an item receives from a reviewer in a star-based ranking system. (Feedback Response)

The number is stored in the system and written to the data warehouse as follows:

Rating

- 0 - Not Rated
- 1 - Poor
- 2 - Fair
- 3- Good
- 4- Very Good
- 5 - Excellent

Microsoft.VSTS.Common.Rating

See Also

Concepts

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Requirements field reference (CMMI)

[See Also](#) [Send Feedback](#)

When you create a team project using the process template for Microsoft Solutions Framework (MSF) for CMMI Process Improvement, you can define fields to track requirements to be developed and their importance to the overall product.

For more information about data types and field attributes, see [Define and modify work item fields](#)

Field name	Description	Reference name
-------------------	--------------------	-----------------------

	The kind of requirement to implement. You can specify one of the following values (see note 1):	
--	---	--

- Business Objective
- Feature

Requirement Type	(default)	Microsoft.VSTS.CMMI.RequirementType	9
	<ul style="list-style-type: none"> • Functional • Interface • Operational • Quality of Service • Safety • Scenario • Security 		
Impact Assessment	<p>The customer impact of not implementing this requirement. You might include details from the Kano model about whether this requirement is in the surprise, required, or obvious categories.</p>	Microsoft.VSTS.CMMI.ImpactAssessmentHtml I	
	<p>The status of the user acceptance test for a requirement. You can specify</p>		

one of the following values (1):

- Pass (default)
- Fail
- Not Ready
- Ready
- Skipped
- Info Received

User Acceptance Test

Microsoft.VSTS.CMMI.UserAcceptanceTest §

You specify Not Ready when the requirement is in the Active state, and you specify Ready when the requirement is in the Resolved state.

The names of one to three team members who are familiar with the customer area that this requirement represents.

Subject matter experts

Microsoft.VSTS.CMMI.SubjectMatterExpert1

...

Microsoft.VSTS.CMMI.SubjectMatterExpert3 §

You can specify
the names of
valid team
members only.

Notes:

1. To change the menu selection, see [Customize a pick list](#).

See Also

Concepts

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Change request field reference (CMMI)

[See Also](#) [Send Feedback](#)

You can track change requests for CMMI work items by using these six fields: Justification, Impact on Architecture, Impact on User Experience, Impact on Test, Impact on Development, and Impact on Technical Publications. A description and a reference name for each of the change request fields are provided in the following table. When you open a work item form for a change request, the Justification field appears on the Justification tab, and all other fields appear on the Analysis tab.

The Change Request work item type is provided only with the process template for Microsoft Solutions Framework (MSF) CMMI Process Improvement.

None of these fields are reportable or indexed. They all have a data type of HTML.

Field name	Description	Reference name
Justification	Why the change has been proposed and what value it would bring to the product and the customer.	Microsoft.VSTS.CMMI.Justification
	The impact	

Impact on
Architecture

that the change would have on architecture. You can use this field to describe in detail which sections of the architecture would be affected and how much the change would cost to implement.

Microsoft.VSTS.CMMI.ImpactOnArchitecture

Impact on
User
Experience

The impact that the change would have on the user experience. You can use this field to describe in detail which sections of the user interface would be affected and how much the change would cost to implement.

Microsoft.VSTS.CMMI.ImpactOnUserExperience

The impact that the change would have on

Impact on
Test

testing. You can use this field to describe in detail which tests would be affected and how much the change would cost to implement.

Microsoft.VSTS.CMMI.ImpactOnTest

Impact on
Development

The impact that the change would have on development and product designs. You can use this field to describe in detail which development areas and designs would be affected and how much the change would cost to implement.

Microsoft.VSTS.CMMI.ImpactOnDevelopment

The impact that the change would have on product documentation. You can use

Impact on Technical Publications	this field to describe in detail which sections of documentation would be affected and how much the change would cost to implement.	Microsoft.VSTS.CMMI.ImpactOnTechnicalPublic
--	---	---

See Also

Concepts

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Review meeting field reference (CMMI)

[See Also](#) [Send Feedback](#)

The following fields track information and changes for review meetings. Your team can specify this kind of information by using the Review type of work item that is provided with the process template for Microsoft Solutions Framework (MSF) CMMI Process Improvement.

None of these fields are reportable or indexed. For more information about data types, see [Define and modify work item fields](#).

Field name	Description	Reference name	Data type
Purpose	The purpose and focus of the meeting.	Microsoft.VSTS.CMMI.Purpose	HTML
Comments	Additional information that you want to record.	Microsoft.VSTS.CMMI.Comments	HTML
	The details of what the team discussed		

Minutes	<p>and agreed upon during the meeting. You can use this field to record what the team reviewed, what criteria the team applied, and what problems the team identified.</p>	Microsoft.VSTS.CMMI.Minutes	HTML
Meeting Type	<p>The meeting venue. You can specify one of the following values:</p> <ul style="list-style-type: none"> • Meeting • Offline 	Microsoft.VSTS.CMMI.MeetingType	String
Called Date	<p>The date and time when the meeting is scheduled.</p>	Microsoft.VSTS.CMMI.CalledDate	DateTime
Called By	<p>The name of the team member who scheduled</p>	Microsoft.VSTS.CMMI.CalledBy	String

the meeting.

Required Attendee 1	The names of each team member who is required to attend the meeting.	Microsoft.VSTS.CMMI.RequiredAttendee1	
...	String
Required Attendee 8		Microsoft.VSTS.CMMI.RequiredAttendee8	
Optional Attendee 1	The name of each team member who is invited but not required to attend the meeting.	Microsoft.VSTS.CMMI.OptionalAttendee1	
...	String
Optional Attendee 8		Microsoft.VSTS.CMMI.OptionalAttendee8	
Actual Attendee 1	The name of each team member who attended the meeting.	Microsoft.VSTS.CMMI.ActualAttendee1	
...	String
Actual Attendee 8		Microsoft.VSTS.CMMI.ActualAttendee8	

See Also

Concepts

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Bugs, issues, and risks field reference (CMMI)

[See Also](#) [Send Feedback](#)

The following fields track information about bugs, issues, and risks. These work item types are defined within the process template for Microsoft Solutions Framework (MSF) CMMI Process Improvement.

Bug tracking fields

These fields are neither reported nor indexed:

Field name	Description	Reference name
Symptom	The unexpected behavior.	Microsoft.VSTS.CMMI.Symptom
Proposed Fix	The proposed change to fix the reported problem.	Microsoft.VSTS.CMMI.ProposedFix
Found in Environment	The software setup and configuration where the bug was found.	Microsoft.VSTS.CMMI.FoundInEnvironment
Root Cause	<p>The cause of the error. You can specify one of the following values:</p> <ul style="list-style-type: none">• Coding Error• Design Error• Specification Error• Communication	Microsoft.VSTS.CMMI.RootCause

Error

- Unknown

To change the menu selection, see [Customize a pick list](#).

How Found How the bug was found. For example, a bug might have been found during a customer review or through ad hoc testing. Microsoft.VSTS.CMMI.HowFound

Issue tracking fields

These fields are not indexed:

Field name	Description	Reference name
Analysis	The root cause of the issue and one or more solutions that might resolve it.	Microsoft.VSTS.CMMI.Analysis
Corrective Action Actual Resolution	What the team actually did to correct the issue.	Microsoft.VSTS.CMMI.CorrectiveActionActualResolut
Corrective Action Plan	The proposed corrective action on which the team has agreed.	Microsoft.VSTS.CMMI.CorrectiveActionPlan

Target
Resolve
Date

The date
when the
issue
becomes
critical and
starts to
affect the
critical path
of the
project
plan.

Microsoft.VSTS.CMMI.TargetResolveDate

Risk tracking fields

These fields are neither reported nor indexed:

Field name	Description	Reference name	Data type
	The actions to take if the risk occurs.		
	You can create and link tasks to the Risk work item to track the work that the team		
Contingency Plan	must complete to implement the contingency plan. Also, you can create an Issue work item to track one or more issues on which the risk has an impact.	Microsoft.VSTS.CMMI.ContingencyPlan	HTML

The actions to take to reduce the probability or impact of the risk.

Mitigation Plan

You can create and link tasks to the Risk work item to track the work that the team must complete to implement the mitigation plan.

Microsoft.VSTS.CMMI.MitigationPlan

HTML

Mitigation Triggers

The conditions or events that determine how the team might mitigate a risk. For example, the triage team might authorize

and obtain a Microsoft.VSTS.CMMI.MitigationTriggers HTML

reserve generator if the weather forecast is predicting an ice storm or hurricane to hit within 50 miles of the office within the next four days.

A number that indicates the chance that the risk will occur.

Probability	A valid probability number is between 1 and 99, where 99 indicates that the risk is almost certain to occur.	Microsoft.VSTS.CMMI.Probability	Integer
-------------	--	---------------------------------	---------

See Also

Concepts

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Link controls, restrictions, and field reference

[See Also](#) [Send Feedback](#)

Work item forms contain these tabs for linking: Implementation, Links, All Links and Test Cases. This topic describes the controls and data fields that are associated with creating links and hyperlinks.

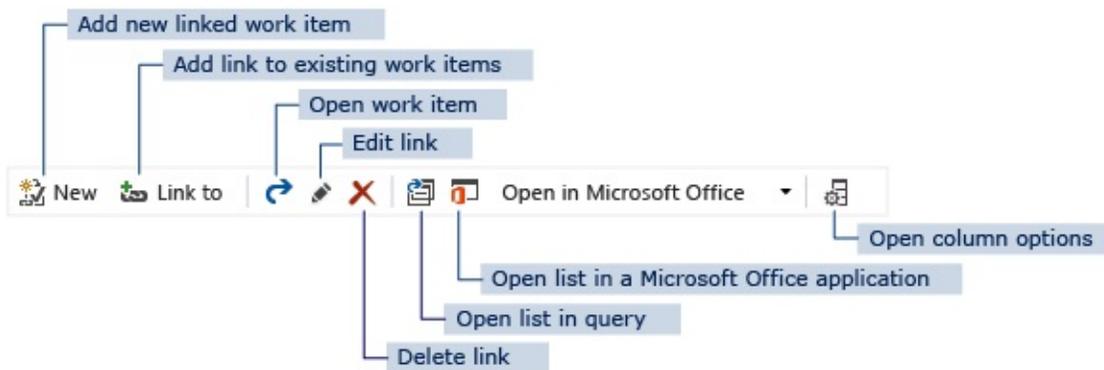
For more information about how to use links, see Link work items. To add link types, see

[Manage link types \[witadmin\]](#). To customize the link controls and restrictions, you modify the definition of the LinksControlOptions for a work item type, see Define link controls to restrict link relationships.

Link toolbar buttons

Each tab has a toolbar with buttons. The links control toolbar for Team Web Access, shown here, has a subset of these controls:

Links control toolbar (Team Explorer)



These buttons become available only after you perform a specific action:

- The button to create a work item that is linked to the open work item (💡) becomes available only after you save the open work item.
- The buttons to open the list of work items in a query (📄) and 📄 Open in Microsoft Office become available only when at least one work item is listed in the links control tab.
- The buttons to open a work item (↻), edit a link (✎), and delete a link (✖) become available only after you click one or more work items listed in the links control tab.

The Storyboards links control restricts users to add links only to storyboards or network shared files. With this control, you can add a new link, open a linked item, and delete a link. Also, only the Team Web Access version displays the Start Storyboarding link within the toolbar.

Storyboards links control (Team Web Access)

DESCRIPTION	STORYBOARDS	TEST CASES	TASKS
Start storyboarding	  		
Title			

Link controls and restrictions

All tabs that support creating links between work items are implemented by using the **LinksControl** element on the work item form. This element controls filtering and restricting the types of work items to which you can link, the types of links that you can create, and whether you can link to work items in another team project. For more information about how to restrict links, see

[LinksControlOptions Elements](#).

Scrum process template

The Scrum process template defines the link control restrictions shown here. For descriptions of the work item types, see Visual Studio Scrum process template for Visual Studio ALM.

Tab name	Work item type	Link restrictions
All Links	Feedback Request	• No restrictions.
	Feedback Response	
	Product Backlog Item	
Links	Bug	• No restrictions.
	Impediment	
	Shared	

	steps	
	Task	
	Test Case	
Links	Code Review Request	<ul style="list-style-type: none"> • Allows only Parent and Child links to Code Review Response work items. • Excludes links to work items in other team projects. • Allows only Related links. • Allows links to Bug and Product Backlog Items. • Excludes links to
Stories	Feedback Response	

		work items in other team projects.
Storyboards	Product Backlog Item	<ul style="list-style-type: none"> • Allows only Storyboard links.
Tasks	Product Backlog Item	<ul style="list-style-type: none"> • Allows only Child links to Tasks. • Excludes links to work items in other team projects.
Test Cases	Product Backlog Item	<ul style="list-style-type: none"> • Allows only Tested By links. • Allows links only to test cases.
	Bug	<ul style="list-style-type: none"> • Excludes links to work items in other team

		projects.
Tested Backlog Items	Test case	<ul style="list-style-type: none"> • Allows only Tests links. • Allows links to Bug and Product Backlog Items. • Excludes links to work items in other team projects.

Agile process template

The Agile process template defines the link control restrictions shown here. For descriptions of the work item types, see Agile process template work item types and workflow.

Tab name	Work item type	Link restrictions
	User story	
	Bug	<ul style="list-style-type: none"> • No

All Links	Feedback Request	restrictions.
	Task	
	Test Case	
Implementation	User story	<ul style="list-style-type: none"> • Allows only Parent and Child links between user stories and tasks.
	Task	<ul style="list-style-type: none"> • Excludes links to work items in other team projects.
Links	Issue	<ul style="list-style-type: none"> • No restrictions.
	Shared steps	
Links	Code Review Request	<ul style="list-style-type: none"> • Allows only Parent and Child links to Code Review Response work items.

Stories	Feedback Response	<ul style="list-style-type: none">• Excludes links to work items in other team projects.• Allows only Related links to user stories.• Excludes links to work items in other team projects.
Storyboards	User Story	<ul style="list-style-type: none">• Allows only Storyboard links.
Test Cases	User story Bug	<ul style="list-style-type: none">• Allows only Tested By links.• Allows links only to test cases.

Tested User Stories	Test case	<ul style="list-style-type: none"> • Excludes links to work items in other team projects. • Allows only Tests links. • Allows links only to user stories. • Excludes links to work items in other team projects.
---------------------	-----------	--

≡ CMMI process template

The CMMI process template defines the link control restrictions shown her . For descriptions of the work item types, see CMMI process template work item types and workflow.

Tab name	Work item type	Link restrictions
	Requirement	
	Bug	

	Change Request	
	Feedback Request	
All Links	Issue	<ul style="list-style-type: none"> • No restrictions.
	Review	
	Risk	
	Shared Steps	
	Task	
	Test Case	
Implementation Task		<ul style="list-style-type: none"> • Allows only Parent and Child links between requirements and tasks. • Excludes links to work items in other team projects.
Links	Code Review	<ul style="list-style-type: none"> • Allows only Parent and Child links to Code Review Response work items.

	Request	<ul style="list-style-type: none"> Excludes links to work items in other team projects.
Requirements	Change Request	<ul style="list-style-type: none"> Allows only Affects link type to link change requests to requirements. Excludes links to work items in other team projects.
Stories	Feedback Response	<ul style="list-style-type: none"> Allows only Related links to requirements. Excludes links to work items in other team projects.
Storyboards	Requirement	<ul style="list-style-type: none"> Allows only Storyboard links. Allows only

Test Cases	Requirement Bug	<p data-bbox="729 197 886 275">Tested By links.</p> <ul data-bbox="691 323 943 705" style="list-style-type: none"> <li data-bbox="691 323 943 443">• Allows links only to test cases. <li data-bbox="691 491 943 705">• Excludes links to work items in other team projects.
Tested Requirements	Test case	<ul data-bbox="691 821 943 1331" style="list-style-type: none"> <li data-bbox="691 821 943 898">• Allows only Tests links. <li data-bbox="691 947 943 1066">• Allows links only to requirements. <li data-bbox="691 1115 943 1331">• Excludes links to work items in other team projects.

Default data fields in lists of links

All lists of links display these data fields:

- Work item ID
- Work Item Type
- Title
- Assigned to
- State
- [Link Comment]

You can add or remove columns from the list of links, and you can customize the default columns and the column order. For more information, see

[LinksControlOptions Elements](#).

For more information about these fields, see [Titles, IDs, Descriptions, and History field reference](#) and [Assignments and workflow field reference](#).

The following table describes the [Link Comment] data field. For information about data types and default field values, see [Define and modify work item fields](#).

Field Name	Description	Reference Name	Data Type	Default Value for the Reportable Type Attribute	Default Value for the Index Attribute
	Contains				

	comments from the team member who created the link.			
Link Comment	You can configure this attribute field to appear as a column in a list of links on a work item form.	System.Links.Comment PlainText	None	False

Additional fields that support query and reporting

Some fields do not appear on the work item forms but are tracked for each type of work item when you link them. You can use these fields to filter queries and create reports.

Field Name	Description	Reference Data Name	Type	Default Value for the Reportable Type Attribute	Default Value for the Index Attribute
External Link Count	The number of links from the work item to artifacts that are not work items and that are registered through the Team Foundation Linking Service. For more information, see this topic on the Microsoft				

Web site:

[Linking Service](#).

System.ExternalLinkCount

Integer

None

False

Hyperlink Count

The number of hyperlinks that are defined for the work item.

System.HyperLinkCount

Integer

None

False

Related Link Count

The number of links from the work item to other work items.

System.RelatedLinkCount

Integer

None

False

Link Description

Contains the work item type, ID, and title of the work item that is the target of the link. You can configure this link attribute field to appear as a column in a list of links on a work item form.

System.Links.Description

PlainText

None

False

■ Reports that require links between work items

The default process templates in TFS provide reports that require you to create links between specific work items.

Process template	Report	Link requirements
Scrum	Backlog Overview (Scrum)	Link PBIs and tasks (Parent-Child) and PBIs and test cases (Tested by-Tests).
Agile	Stories Overview Report (Agile)	Link user stories and tasks (Parent-Child) and user stories and test cases (Tested by-Tests). Link each bug to the test case that identified the code defect (Tested By) or link it to the user story (Related).
	Stories Progress Report (Agile)	

CMMI

Requirements and
Overview Report
(CMMI)

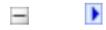
Requirements
Progress Report
(CMMI)

Link requirements and tasks (Parent-Child) requirements and test cases(Tested by-Tests). Link each bug to the test case that identified the code defect (Tested By) or link it to the requirement (Related).

See Also

Concepts

[Work item field reference for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Attachment control and field reference

[See Also](#) [Send Feedback](#)

You add files to work items from the Attachments tab on the work item form. For example, you can attach an email thread, a document, an image, a log file, or another type of file to any work item. This topic describes the controls and the data field that are associated with file attachments.

For more information about how to add attachments, see [Create or delete a file attachment in a work item](#).

Attachment Toolbar Buttons

The Attachments tab is located in the lower half of most work item forms provided with the default process templates for Team Foundation Server, as the following illustration shows:



You must choose or attach a file before the buttons to open an attachment (↻), save a copy (💾 Save Copy), and delete an attachment (✖) become available.

Attachment File Count

The field in the following table does not appear on any work item form, but this information is tracked for each work item when you attach a file to it. You can use this field to filter queries and create reports. For information about data types and default field values, see

[Define and modify work item fields.](#)

Field name	Description	Reference name	Data type	Default value of the reportable type attribute	Default value of the index attribute
Attached File Count	The number of files attached to the work item and stored in the work item tracking database.	System.AttachedFileCount	Integer	None	False

See Also

Concepts

[Work item field reference for Visual Studio ALM](#)

Other Resources

[Create or delete a file attachment in a work item](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Fields that support integration with test, build, and version control

[See Also](#) [Send Feedback](#)

You can customize work item types (WITs) to contain information that is generated by automated processes by adding fields that integrate with Team Foundation Build, Microsoft Test Manager, and Team Foundation version control.

Fields that integrate with Team Foundation Build

Team Foundation Build is the automated build system of Team Foundation Server. You can configure your build process by using Team Foundation Build, and Team Foundation Build can generate work items when a build fails. It can also add build information to work items that were resolved in a particular build. For this to work, Team Foundation Build requires that the following two fields be added to the work item type definition: Found In and Integration Build.

In the default process templates that Team Foundation Server provides, Found In and Integrated in Build fields appear in the type definitions for bugs. These fields associate bugs with the builds where they were found or fixed. You can use the following code snippet to add these fields to a WIT definition.

 **Copy Code**

```
<FIELD name="Found In" refname="Microsoft.VSTS.Build.FoundIn" type='
  <HELPTEXT>Product build number (revision) in which this item was
    <SUGGESTEDVALUES>
      <LISTITEM value="&lt;None&gt;" />
    </SUGGESTEDVALUES>
</FIELD>
<FIELD name="Integration Build" refname="Microsoft.VSTS.Build.Integr
  <HELPTEXT>Product build number this bug was fixed in</HELPTEXT>
    <SUGGESTEDVALUES>
      <LISTITEM value="&lt;None&gt;" />
    </SUGGESTEDVALUES>
</FIELD>
```

When the Found In field is present in a WIT definition, Team Foundation Build creates a work item when a build fails, and sets the Found In field to the build number of the build that just failed. If the Found In field is missing, Team Foundation Build does not create a work item for the failed build, and everything else works as expected.

When the Integration Build field is present in the WIT definition, Team

Foundation Build identifies work items that were resolved with each build and then updates those work items to set the build number in which they were resolved in the Integration Build field. If the Integration Build field is missing, Team Foundation Build does not store the build number in the work items, and everything else works as expected.

▣ **Build associations with changesets and work items**

A standard build that's based on the default build template will associate changesets and work items to builds. It does this by first retrieving the label for the previous successful build for the given build's build definition and then determining which changesets are included in the current build that were not included in the previous build. Some or all changesets might have work items associated with them, and those work items get associated with the build. This is done as part of the AssociateChangesetsAndWorkItems activity.

▣ **Builds and global list auto-population**

The first time you queue a build for a team project using Team Foundation Build, TFS automatically adds a global list labeled "Build - <team project name>." Each time a build is run, a LISTITEM is added to this global list with the name of the build.

By adding a GLOBALLIST element to the FIELD definition, you can provide a drop-down menu of builds that users can choose from. For example:

Copy Code

```
<FIELD name="Found In" refname="Microsoft.VSTS.Build.FoundIn" type='
  <HELPTEXT>Product build number (revision) in which this item was
    <SUGGESTEDVALUES>
      <LISTITEM value="&lt;None&gt;" />
    </SUGGESTEDVALUES>
    <SUGGESTEDVALUES expanditems="true" filteritems="excludegrou
      <GLOBALLIST name="Builds - TeamProjectName" />
    </SUGGESTEDVALUES>
  </FIELD>
```

Fields that Integrate with Microsoft Test Manager

With Test Manager, you can automate the creation of a bug or other type of work item when a test fails. For more information, see [Submitting Bugs in Microsoft Test Manager](#).

When a work item has been created in this manner, information about the system and the steps to reproduce the bug are captured in the System Info and Repro Steps fields.

You can add these fields to work item types that you create for tracking defects using the following code snippet.

Copy Code

```
<FIELD name="System Info" refname="Microsoft.VSTS.TCM.SystemInfo" ty  
<FIELD name="Repro Steps" refname="Microsoft.VSTS.TCM.ReproSteps" ty
```

For more information about additional fields used by Test Manager, see

[Build and test integration field reference](#).

Fields that integrate with Team Foundation Version Control

One of the features available in Team Foundation version control enables you to associate or resolve work items when you check in code. You might have worked on a particular work item when you make a code change and you can set that association from within the source-control check-in window when you are finished working on the code.

The ability of Team Foundation version control to resolve a work item requires that work items contain a particular action. The source control system then queries work item tracking to determine whether the work item supports that action, and if it does support that action, it also queries for the source and destination states of the transition. If the action is found, the source control system can transition the work item according to the set transition when it checks in the code.

Note

When you use the Checkin action, you must set appropriate 'from' and 'to' states to reflect the state transition that you want.

For more information about Actions, see

[Automate field assignments based on State, Transition, or Reason.](#)

Example of the Checkin Action

Copy Code

```
<TRANSITION from="Active" to="Resolved">
....
  <ACTIONS>
    <ACTION value="Microsoft.VSTS.Actions.Checkin"/>
  </ACTIONS>
```

....
</TRANSITION>

Q & A

Q: What other fields are associated with builds and Test Manager?

A: See [Build and test integration field reference](#) for additional fields,

See Also

Tasks

What development has been done since a previous build?

Other Resources

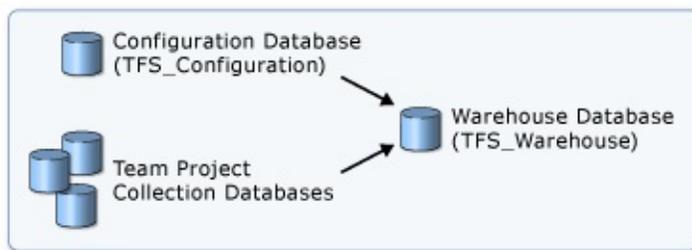
[Modify or add a field to support queries, reports, and workflow](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Table reference for the relational warehouse database for Visual Studio ALM
[See Also](#) [Send Feedback](#)

You can create reports and query for data from Visual Studio Application Lifecycle Management (ALM) by using the warehouse database. The data in the warehouse is collected from the operational stores and organized in a set of tables, views, and table-valued functions that are designed for creating reports.



The warehouse contains data about builds, source code, test results and code coverage, and work items such as tasks and bugs. The relationships between these sets of data are created in Visual Studio ALM and retained in the warehouse. Therefore, you can create queries to explore relationships between these integrated sets of data and better understand what is happening on your projects.

In this topic:

- [Builds](#)
- [Source Code](#)
- [Test Results](#)
- [Code Coverage](#)
- [Tasks, Bugs, and Other Types of Work Items](#)

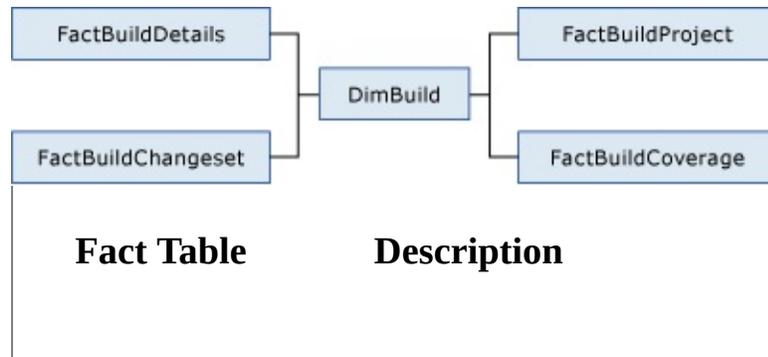
The data in the warehouse is stored in fact and dimension tables. This pattern for storing data in star and snowflake schemas is described in the following topic on the Microsoft Web site: [Introduction to Dimensions](#).

Some other common patterns occur in the warehouse:

- All fact tables reference DimTeamProject, because all data in Visual Studio ALM is organized by team projects and team project collections. Some dimensions also reference the DimTeamProject so that you can more easily use them as project-filter parameters in your reports.
- All fact tables reference DimDate to indicate the date on which the fact occurred.
- Many fact tables reference DimPerson, sometimes more than once. For example, the work item facts reference DimPerson to indicate to whom a work item is assigned to and who changed it most recently.

Builds

You can query for data about builds by using four fact tables, as the following illustration shows. Each fact table uses the build dimension table DimBuild to store many details about the builds.



[Build Details tables](#)

Basic information about each build, such as the build status and quality.

[Build Changeset tables](#)

Changesets that were included in each build.

[Build Project tables](#)

Files that were built and the platforms and flavors of the builds.

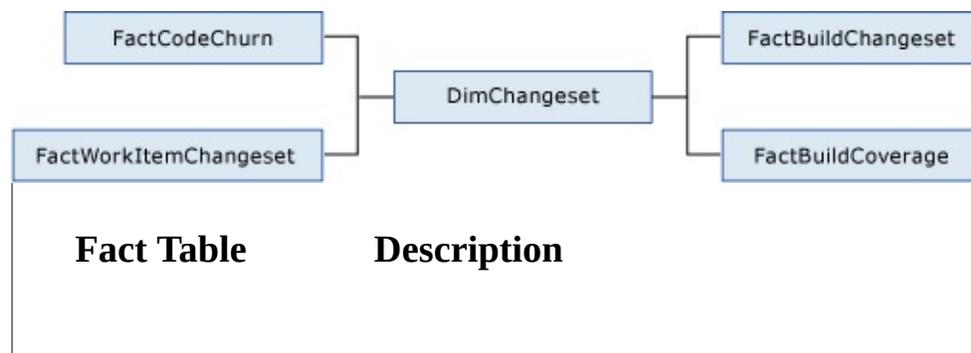
[Build Coverage tables](#)

The extent to which the code was covered by tests that were performed against the build.

For more information about builds, see [Build the application](#).

Source Code

You can query for data about code and other files that are under version control by using the fact tables in the following illustration. DimFile and DimChangeset are the two primary dimensions that are associated with these fact tables. DimFile provides information about files and folders, without regard to specific versions of those files. DimChangeset provides information about the changesets.



[Code Churn tables](#)

Basic information about each build.

[Build Changeset tables](#)

Changesets that were included in the build.

[Build Project tables](#)

Files that were built and the platforms and flavors that were used.

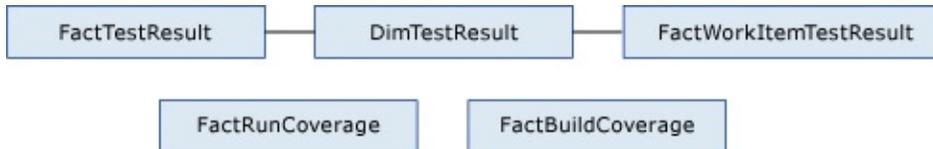
[Work Item Changeset tables](#)

Links between work items and changesets.

For more information about version control, see [Use version control](#).

Tests

You can query for information about test results and analyze how well the tests cover the code by using the tables in the following illustration.



Fact Table	Description
-------------------	--------------------

[Test Result tables](#)

Tests and their results.

[Run Coverage tables](#)

The extent to which the code was covered by tests in a test run.

[Build Coverage tables](#)

The extent to which the code was covered by tests that were performed against the build.

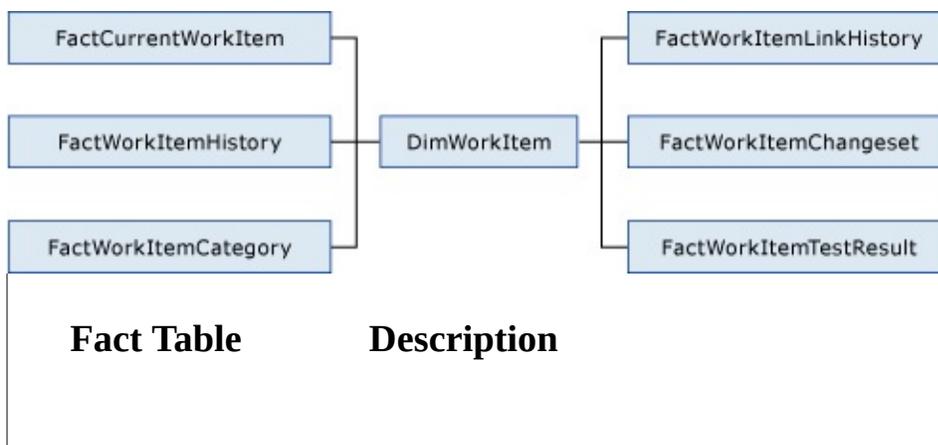
[Work Item Test Result tables](#)

Links between work items and test results.

For more information, see [Testing the application and Using Code Coverage to Determine How Much Code is being Tested](#).

Tasks, Bugs and Other Types of Work Items

You can query for information about tasks, bugs, and other types of work items by using the work item tables in the following illustration. Each work item fact table uses the work item dimension table to store many details about the work items.



[Current Work Item tables](#)

The current state of each work item.

[Work Item History tables](#)

The full history of each work item.

[Work Item Link History tables](#)

The full history of links between work items.

[Work Item Category tables](#)

The categories that are used to associate similar types of work items.

[Work Item Changeset tables](#)

Links between work items and changesets.

[Work Item Test Result tables](#)

Links between work items and test results.

For more information, see [Track work with Visual Studio ALM and TFS](#).

See Also

Concepts

[Create, customize, and manage reports for Visual Studio ALM](#)



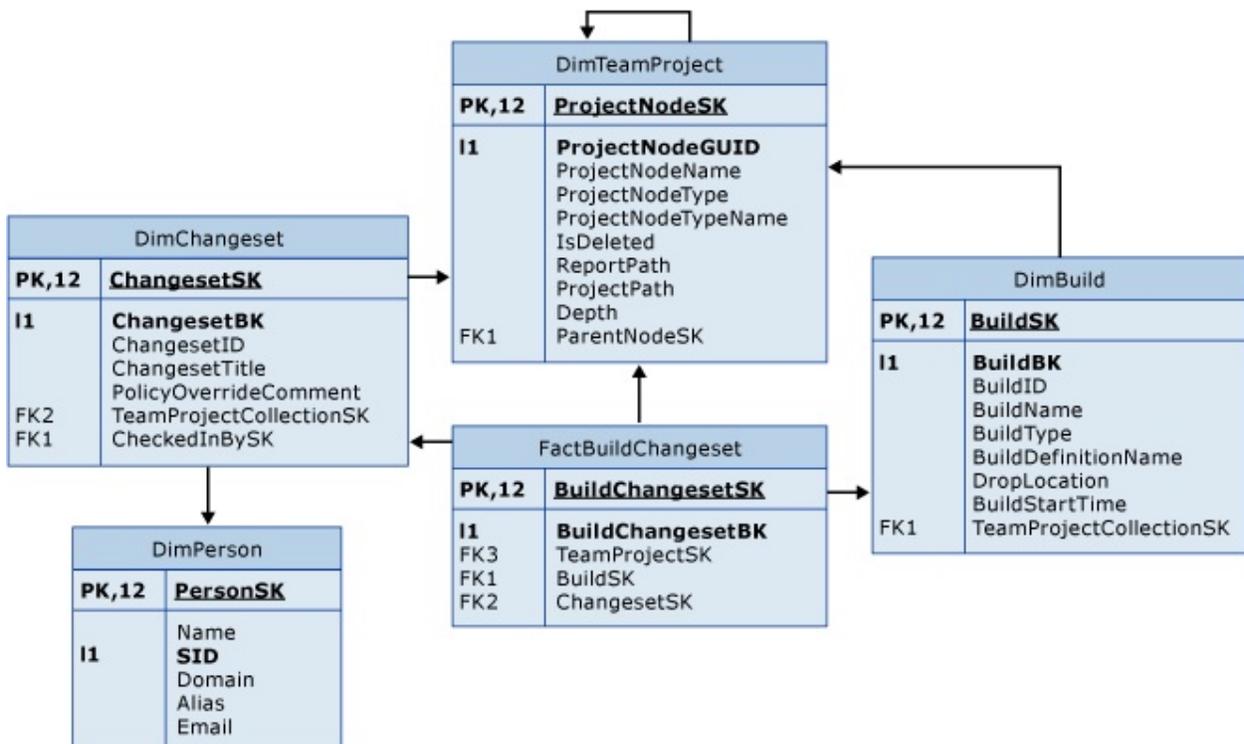
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Build Changeset tables

[See Also](#) [Send Feedback](#)

You can query for data about the changesets that were incorporated in each build by using FactBuildChangeset and the associated dimension tables.

For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on build details and build coverage using the Build perspective](#).



FactBuildChangeset is associated with the following dimension tables:

- DimBuild

- DimChangeset
- DimPerson
- DimTeamProject

See Also

Concepts

[Analyze and report on build details and build coverage using the Build perspective](#)

Find and view changesets

Build the application

[Table reference for the relational warehouse database for Visual Studio ALM](#)



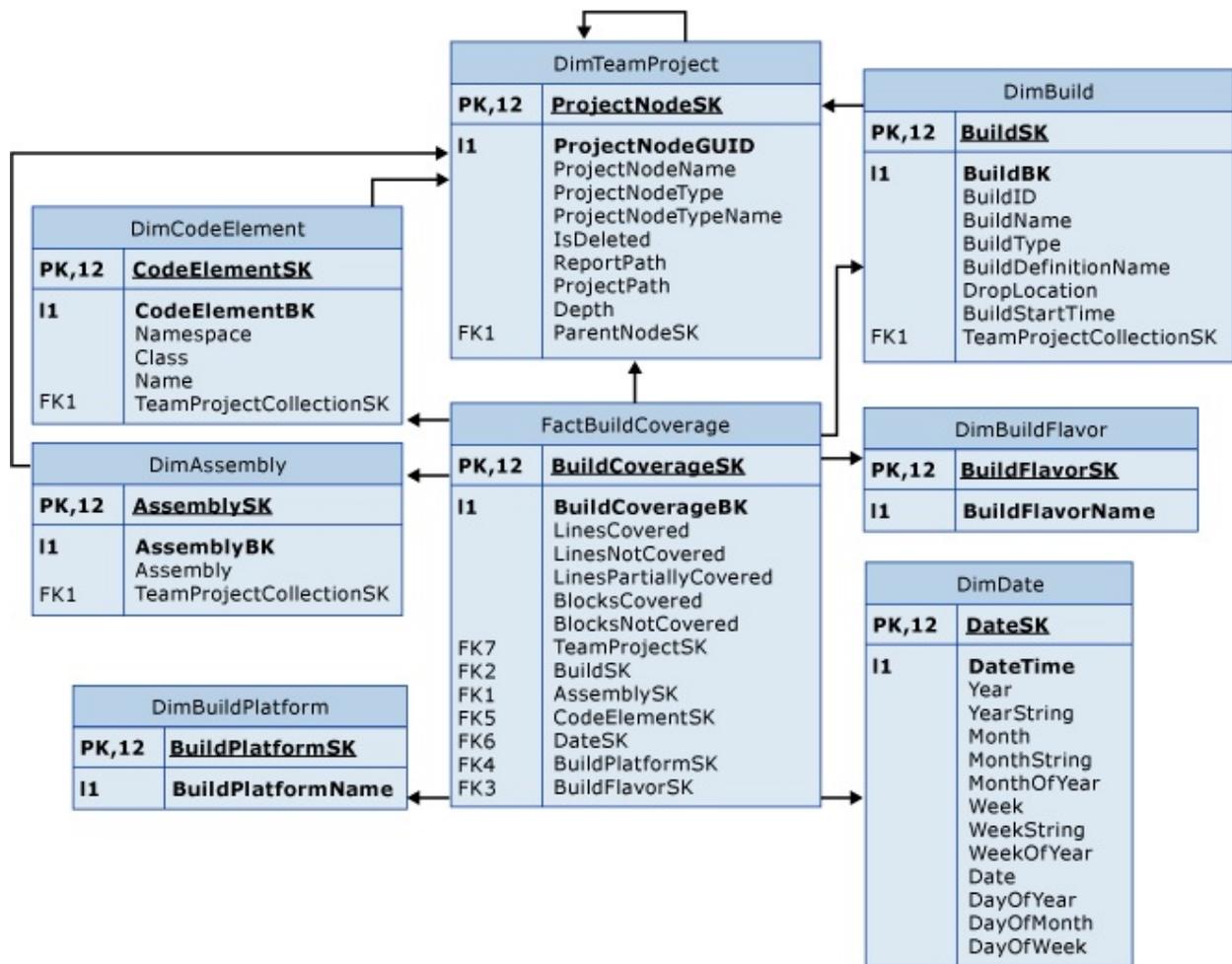
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Build Coverage tables

[See Also](#) [Send Feedback](#)

You can query for data about how thoroughly source code was covered by the tests that use specific builds by using FactBuildCoverage and the associated dimension tables.

For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on build details and build coverage using the Build perspective](#).



FactBuildCoverage is associated with the following dimension tables:

- DimAssembly
- DimBuild
- DimBuildFlavor
- DimBuildPlatform
- DimCodeElement
- DimDate
- DimTeamProject

See Also

Concepts

[Analyze and report on build details and build coverage using the Build perspective](#)

Code Coverage Excel Report

[Run Coverage tables](#)

Build the application

[Table reference for the relational warehouse database for Visual Studio ALM](#)

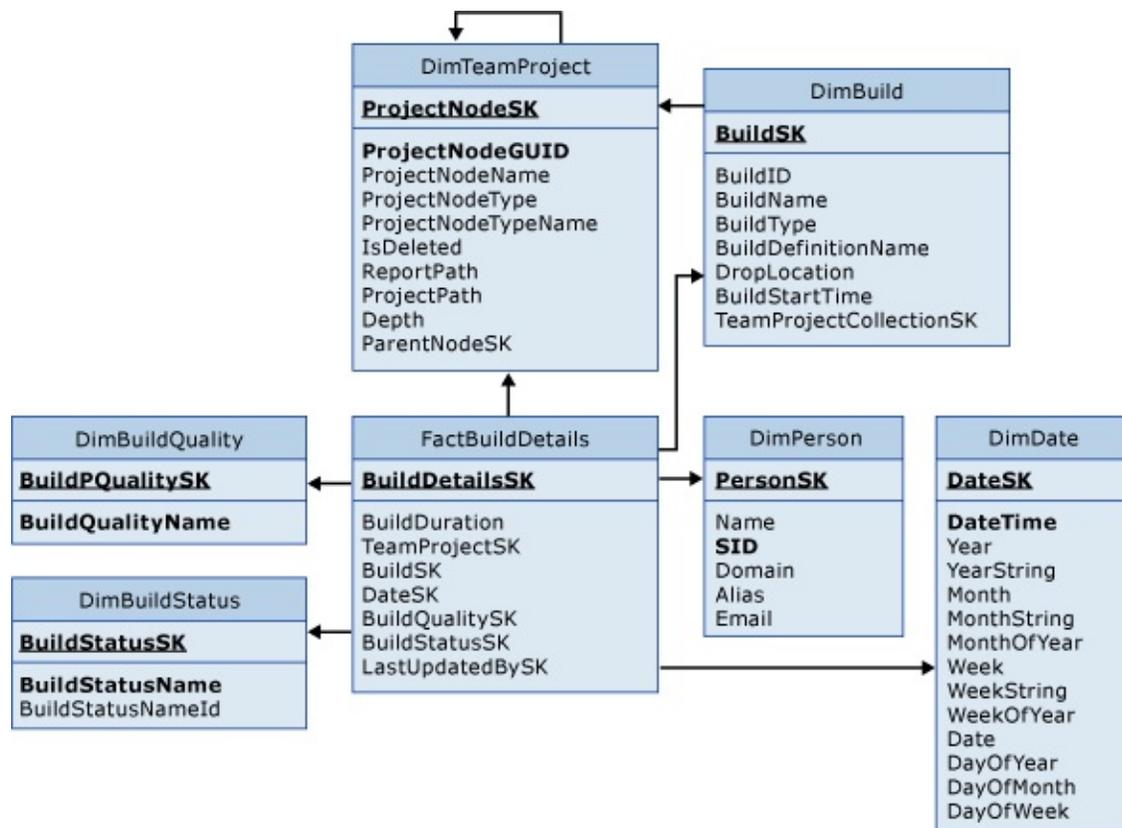
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Build Details tables

[See Also](#) [Send Feedback](#)

You can query for data about builds, such as the status and quality, by using FactBuildDetails and the associated dimension tables.

For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on build details and build coverage using the Build perspective](#).



FactBuildDetails is associated with the following dimension tables:

- DimBuild
- DimBuildQuality
- DimBuildStatus
- DimDate
- DimPerson
- DimTeamProject

See Also

Concepts

[Analyze and report on build details and build coverage using the Build perspective](#)

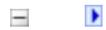
Build Quality Indicators Report

Build Success Over Time Report

Build Summary Report

Build the application

[Table reference for the relational warehouse database for Visual Studio ALM](#)



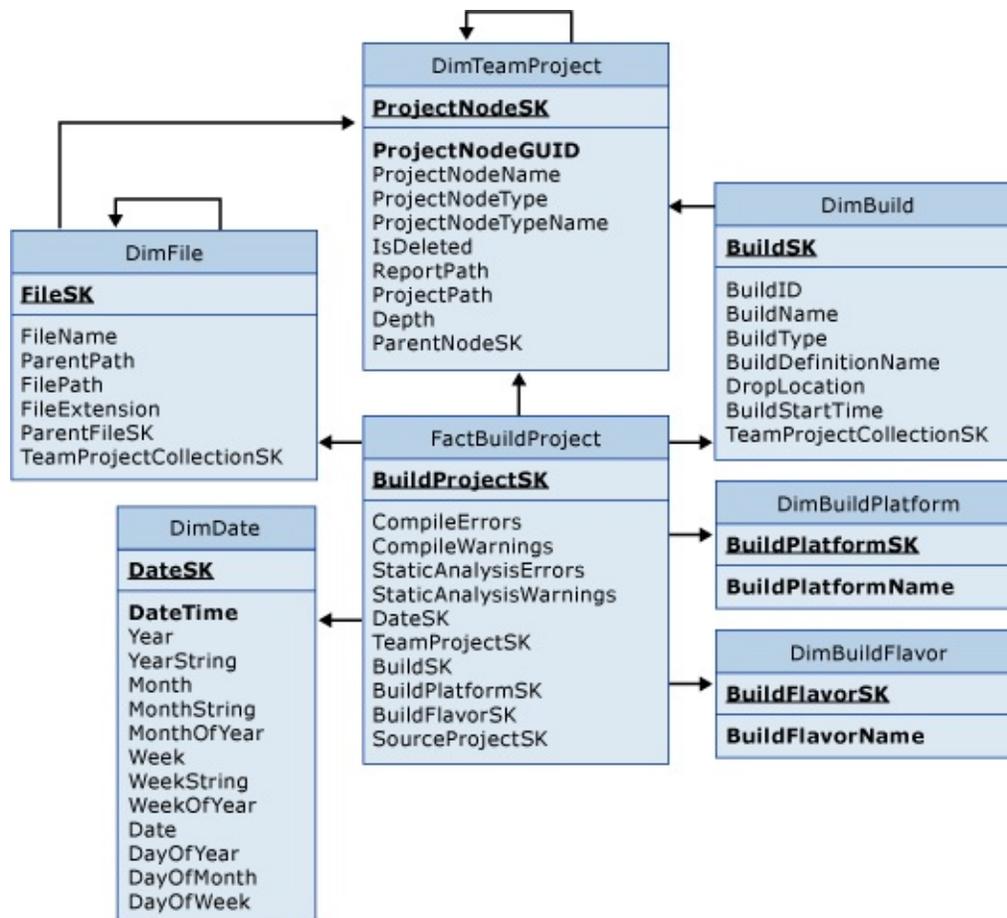
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Build Project tables

[See Also](#) [Send Feedback](#)

You can query for data about the files and projects that were built and information about those files by using FactBuildProjects and the associated dimensions. Records contain the static analysis and compiler warnings and errors for each file that was built. If it was built for more than one platform or flavor, you can find records for each platform and flavor.

For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on build details and build coverage using the Build perspective](#).



FactBuildProject is associated with the following dimension tables:

- DimBuild
- DimBuildFlavor
- DimBuildPlatform
- DimDate
- DimFile
- DimTeamProject

See Also

Concepts

[Analyze and report on build details and build coverage using the Build perspective](#)

Build Quality Indicators Report

Build Success Over Time Report

Build Summary Report

[Build Details tables](#)

Build the application

[Table reference for the relational warehouse database for Visual Studio ALM](#)



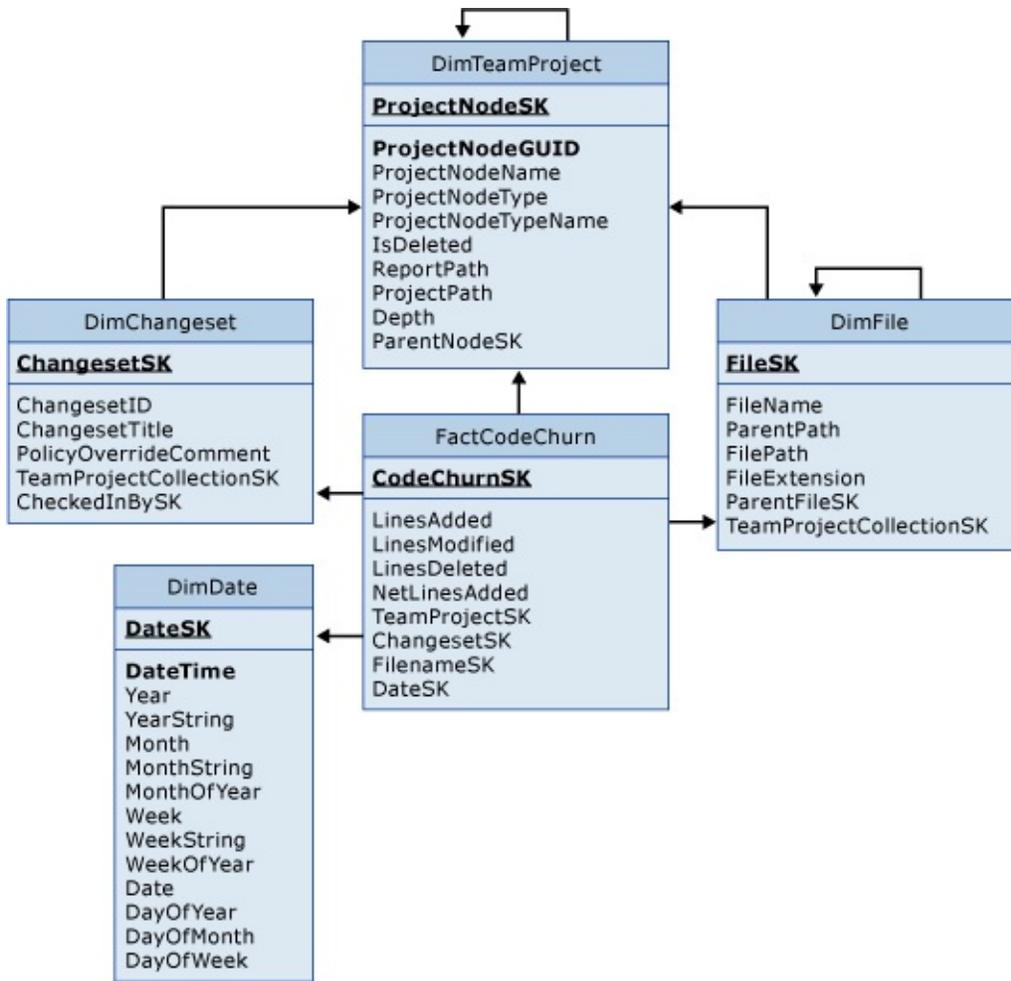
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Code Churn tables

[See Also](#) [Send Feedback](#)

You can query for data about the changes in the code that is under version control by using FactCodeChurn and the associated dimension tables. The fact table contains one record for each revision of a file in each changeset.

For a description of the measures and dimensions that are associated with these tables in the Analysis Services cube, see [Analyze and report on code churn and code coverage using the code churn and run coverage perspectives](#).



FactCodeChurn is associated with the following dimension tables:

- DimChangeset
- DimDate
- DimFile
- DimTeamProject

See Also

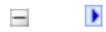
Concepts

[Analyze and report on code churn and code coverage using the code churn and run coverage perspectives](#)

Code Churn Excel Report

[Run Coverage tables](#)

[Table reference for the relational warehouse database for Visual Studio ALM](#)



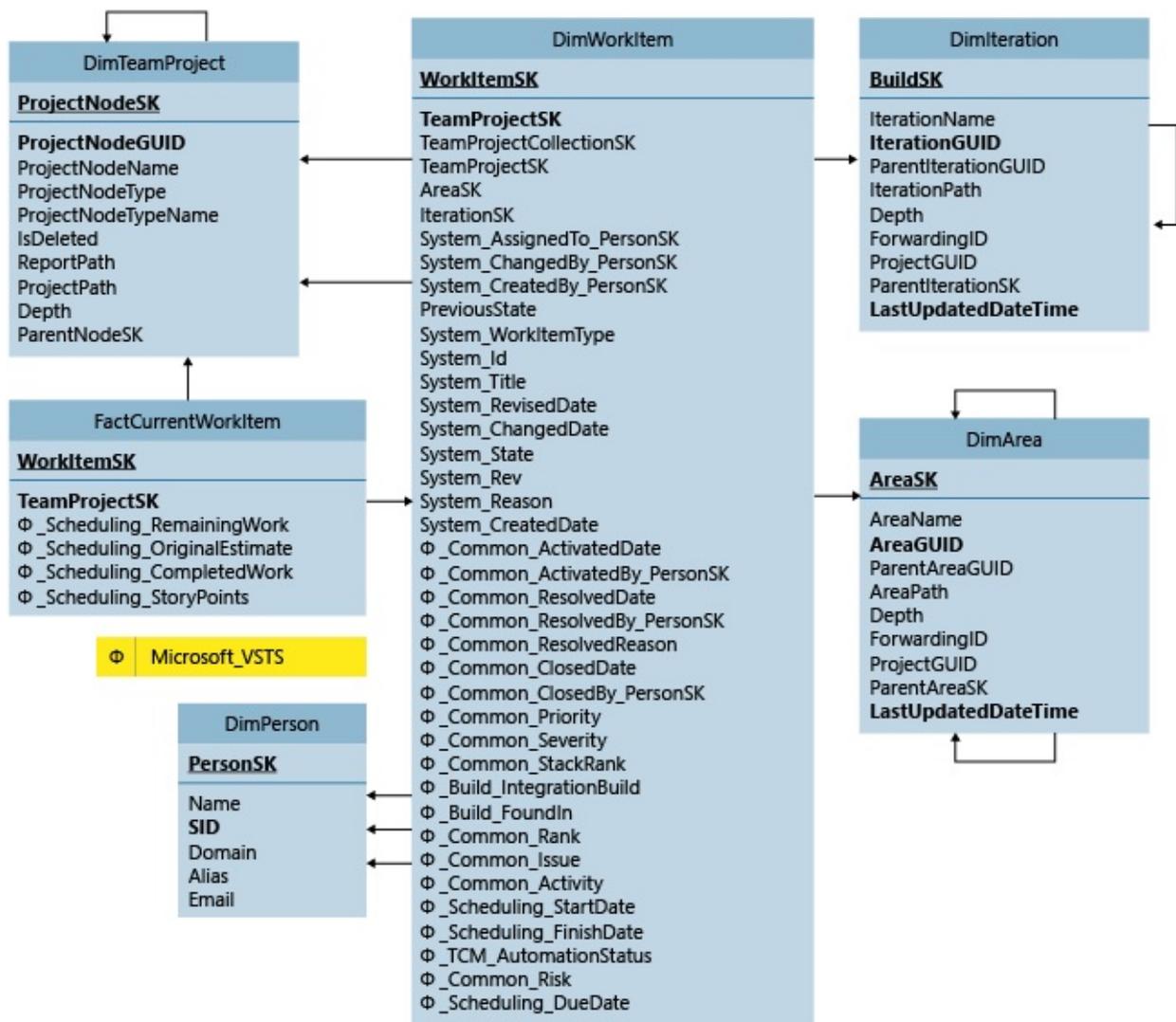
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Current Work Item tables

[See Also](#) [Send Feedback](#)

You can query for data about the current state of bugs, tasks, and other types of work items by using the FactCurrentWorkItem table and the associated dimension tables.

For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on work items and test case data using the Work Item perspective](#).



FactCurrentWorkItem is associated with the following dimension tables:

- DimArea
- DimIteration
- DimPerson
- DimTeamProject
- DimWorkItem

See Also

Concepts

[Analyze and report on work items and test case data using the Work Item perspective](#)

[Work Item History tables](#)

[Table reference for the relational warehouse database for Visual Studio ALM](#)



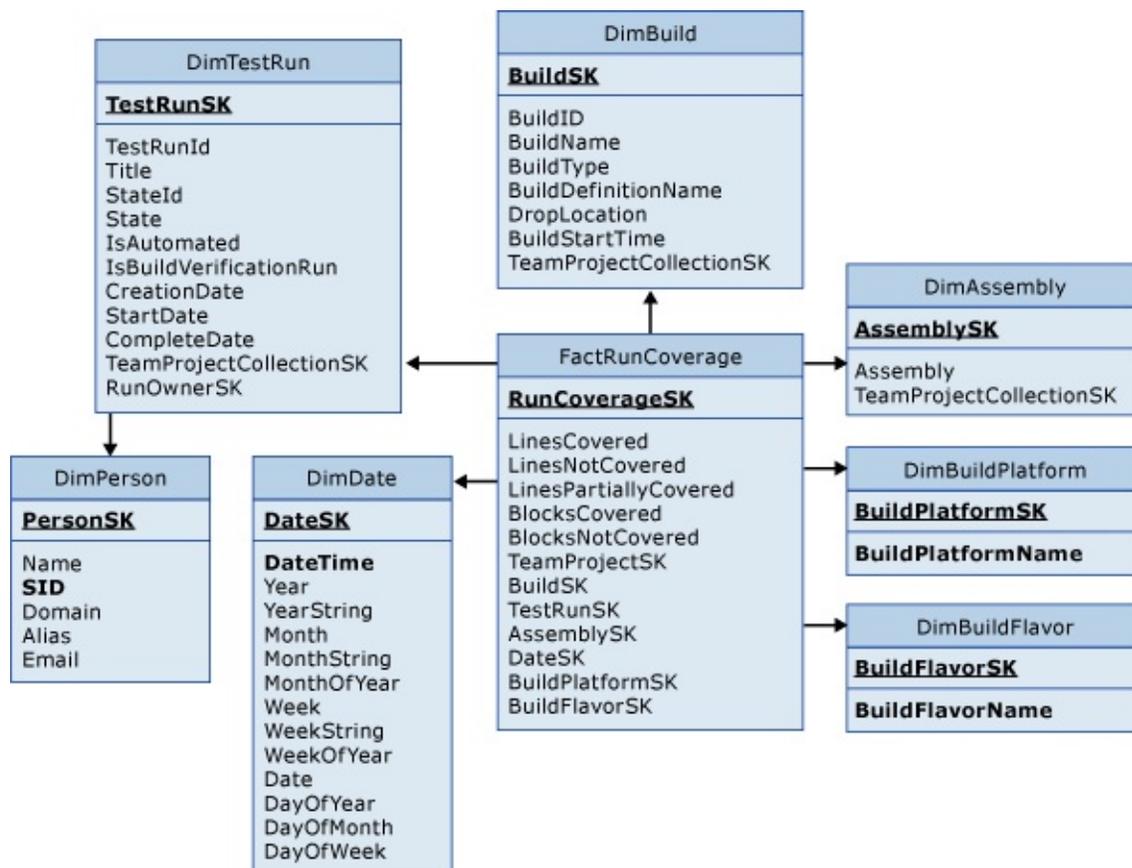
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Run Coverage tables

[See Also](#) [Send Feedback](#)

You can query for data about how thoroughly a particular test run covered the code that it was intended to test by using FactRunCoverage and the associated dimensions.

For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on code churn and code coverage using the code churn and run coverage perspectives](#).



☑Note

You cannot aggregate these coverage values to determine code coverage for a build. To determine the code coverage in a build, you must use FactBuildCoverage. For more information, see [Build Coverage tables](#).

FactRunCoverage is associated with the following dimension tables:

- DimAssembly
- DimBuild
- DimBuildFlavor
- DimBuildPlatform
- DimDate
- DimPerson
- DimTestRun

See Also

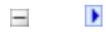
Concepts

[Analyze and report on code churn and code coverage using the code churn and run coverage perspectives](#)

Code Coverage Excel Report

Build the application

[Table reference for the relational warehouse database for Visual Studio ALM](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Test Result tables

[See Also](#) [Send Feedback](#)

You can query for data about test results by using the FactTestResult table and the associated dimension tables. For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see

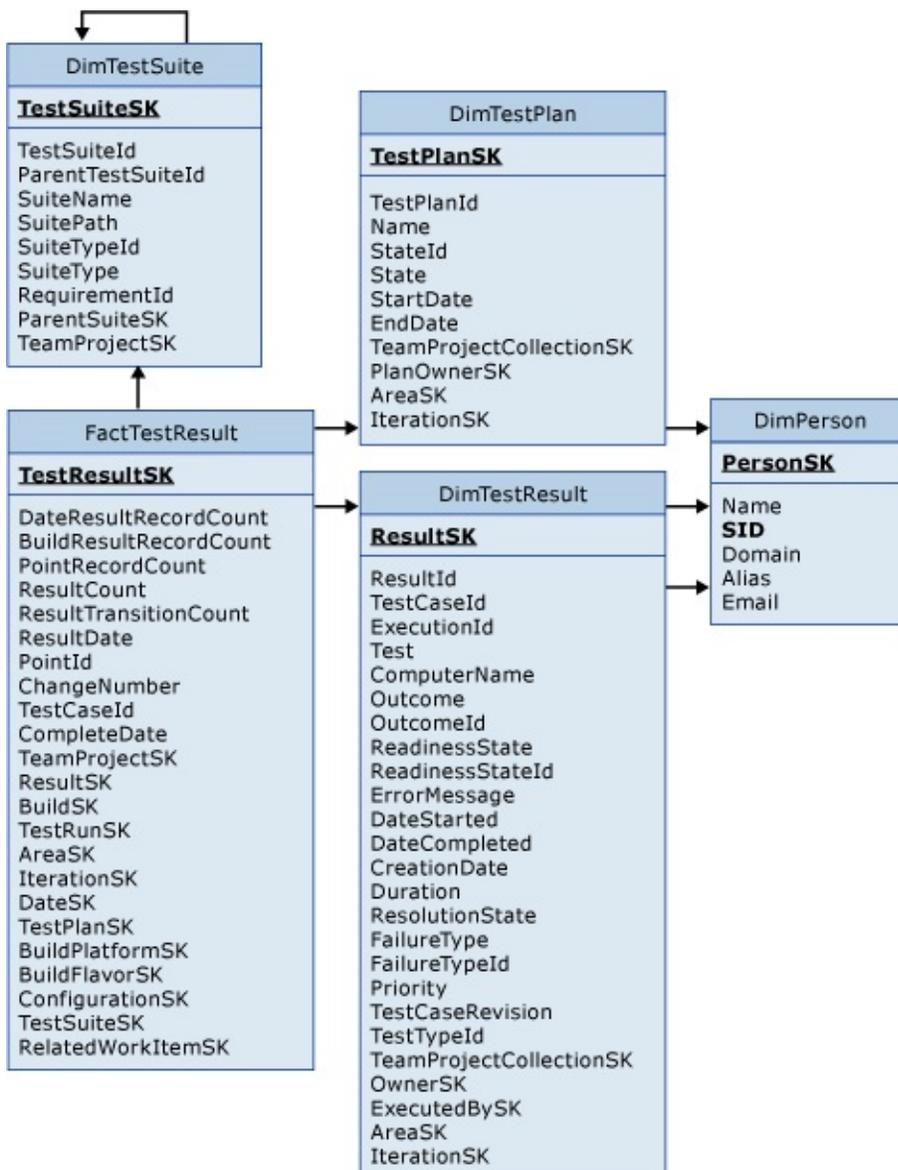
[Analyze and report on test results using the test perspective in the Analysis Services database for Visual Studio ALM.](#)

In this topic

- [Test Results, Test Suites, and Test Plans](#)
- [Test Results and Test Runs](#)
- [Test Results and Builds](#)
- [Test Results and Team Project Data](#)

Test Results, Test Suites, and Test Plans

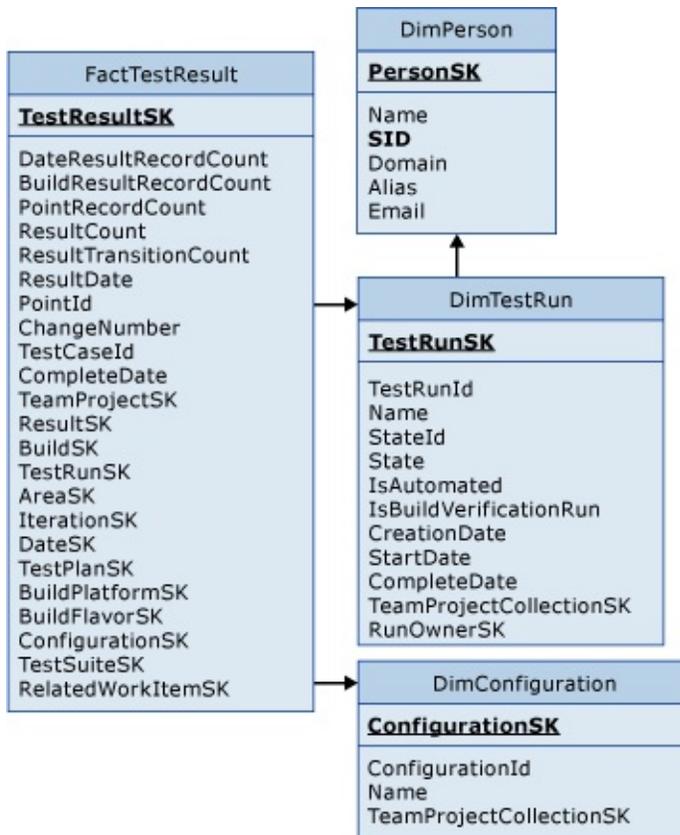
The DimTestResult table provides details about the test results themselves, and you can use the DimTestPlan and DimTestSuite tables to organize the results by test plan and test suite.



[Back to top](#)

Test Results and Test Runs

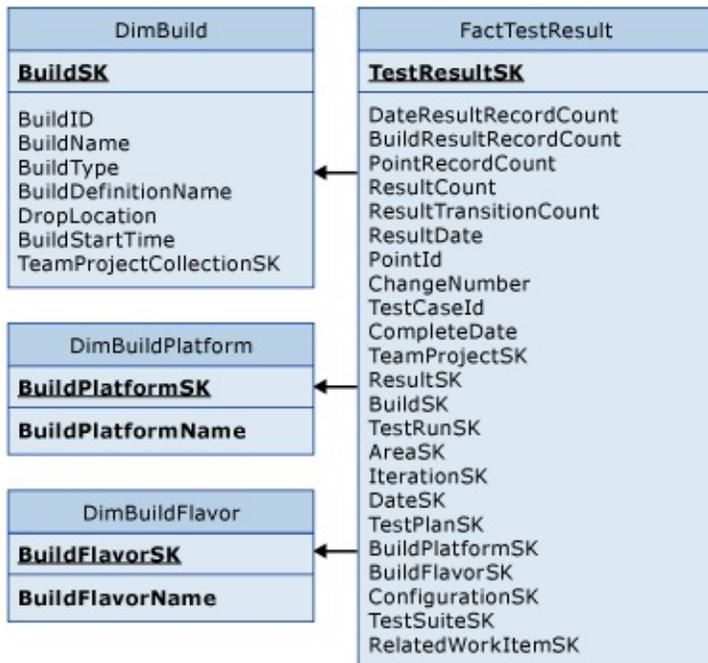
You can use the DimTestRun, DimConfiguration, and DimPerson dimension tables to include data about how the tests were run.



[Back to top](#)

Test Results and Builds

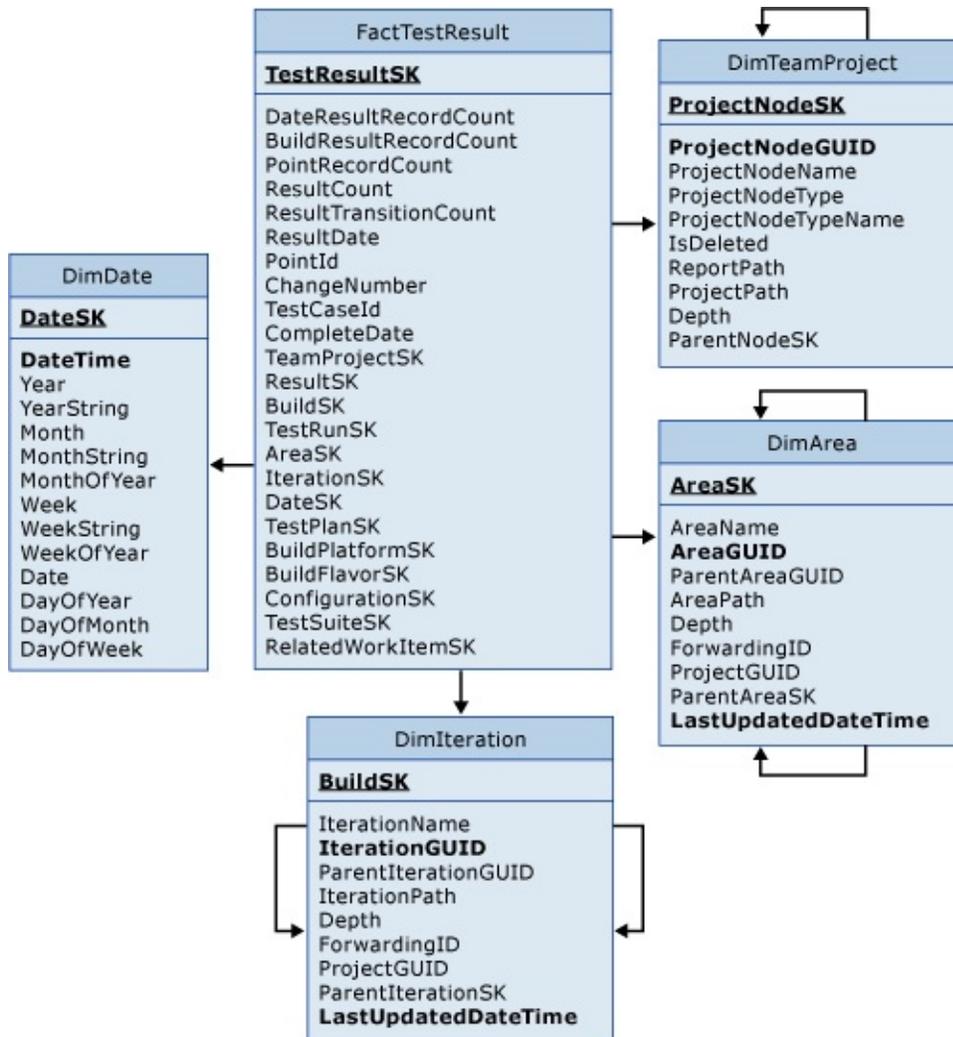
You can use the DimBuild, DimBuildFlavor, and DimBuildPlatform dimension tables, to include data about the build that was used to create the test results.



[Back to top](#)

Test Results and Team Project Data

You can use the DimTeamProject, DimArea, and DimIteration tables to obtain details about how the tests are organized in the team project, and you can use DimDate to show when the test results were created.



[Back to top](#)

See Also

Concepts

[Analyze and report on test results using the test perspective in the Analysis Services database for Visual Studio ALM](#)

Testing the application

[Work Item Test Result tables](#)

[Table reference for the relational warehouse database for Visual Studio ALM](#)

Other Resources

[Test Management Reports](#)



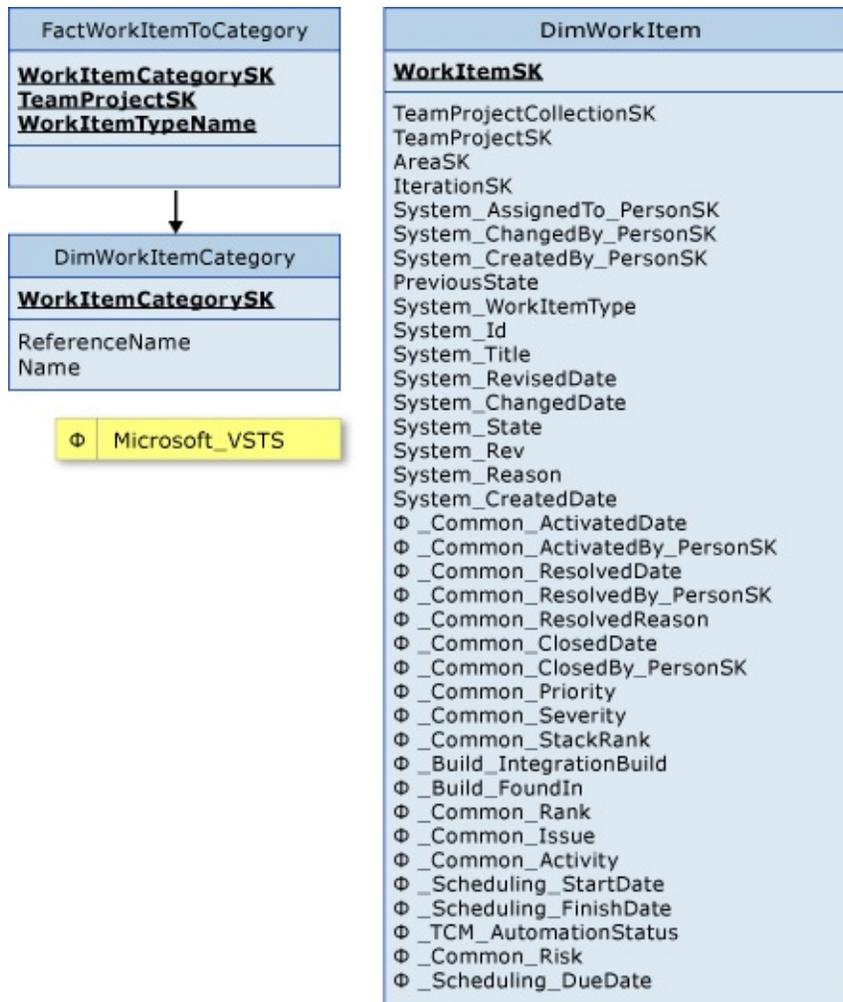
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Work Item Category tables

[See Also](#) [Send Feedback](#)

You can query for categories of work items by using the FactWorkItemToCategory and DimWorkItemCategory tables. You can organize the results of other work item queries by category if you join these tables with the DimWorkItem by using FactWorkItemToCategory.WorkItemTypeName = DimWorkItem.System_WorkItemType.

For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on work items and test case data using the Work Item perspective](#).



FactWorkItemToCategory is associated with the following dimension tables:

- DimWorkItem
- DimWorkItemCategory

See Also

Concepts

[Analyze and report on work items and test case data using the Work Item perspective](#)

[Table reference for the relational warehouse database for Visual Studio ALM](#)

Other Resources

[Use categories to group work item types](#)

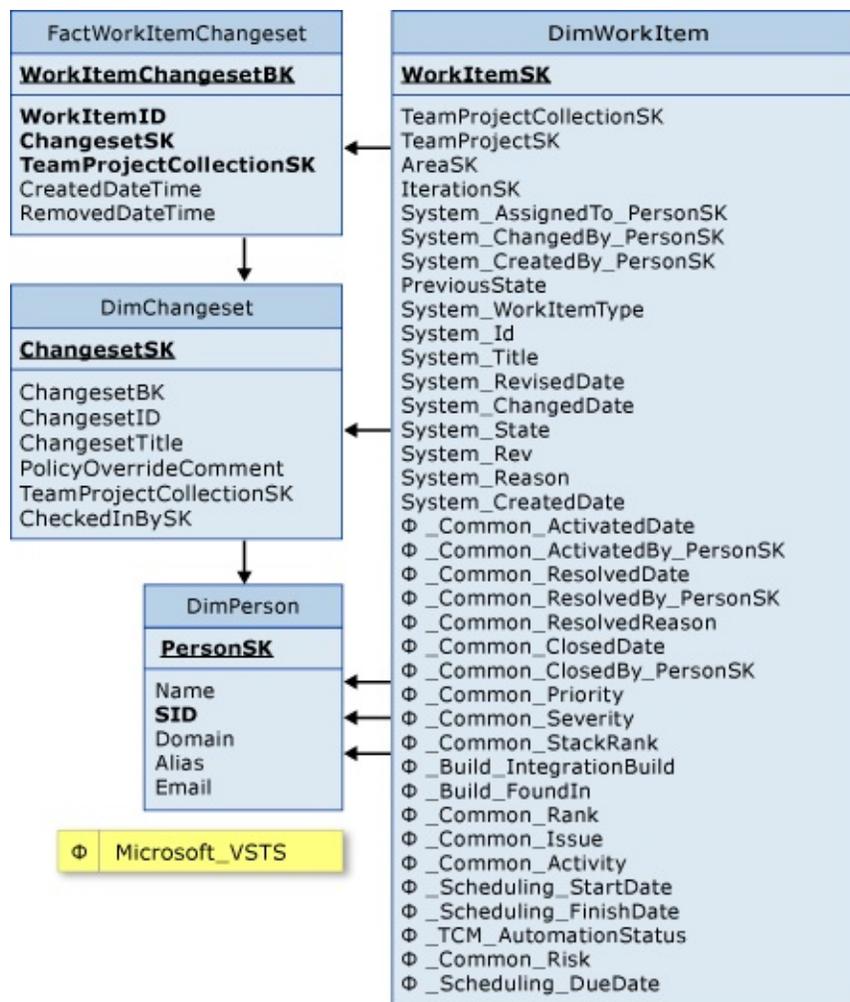


- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Work Item Changeset tables

[See Also](#) [Send Feedback](#)

You can query for data about work items that are linked to changesets by using FactWorkItemChangeset and the associated dimensions. For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on work items and test case data using the Work Item perspective](#).



FactWorkItemChangeset is associated with the following dimension tables:

- DimChangeset
- DimPerson
- DimWorkItem

See Also

Concepts

[Analyze and report on work items and test case data using the Work Item perspective](#)

Find and view changesets

[Table reference for the relational warehouse database for Visual Studio ALM](#)



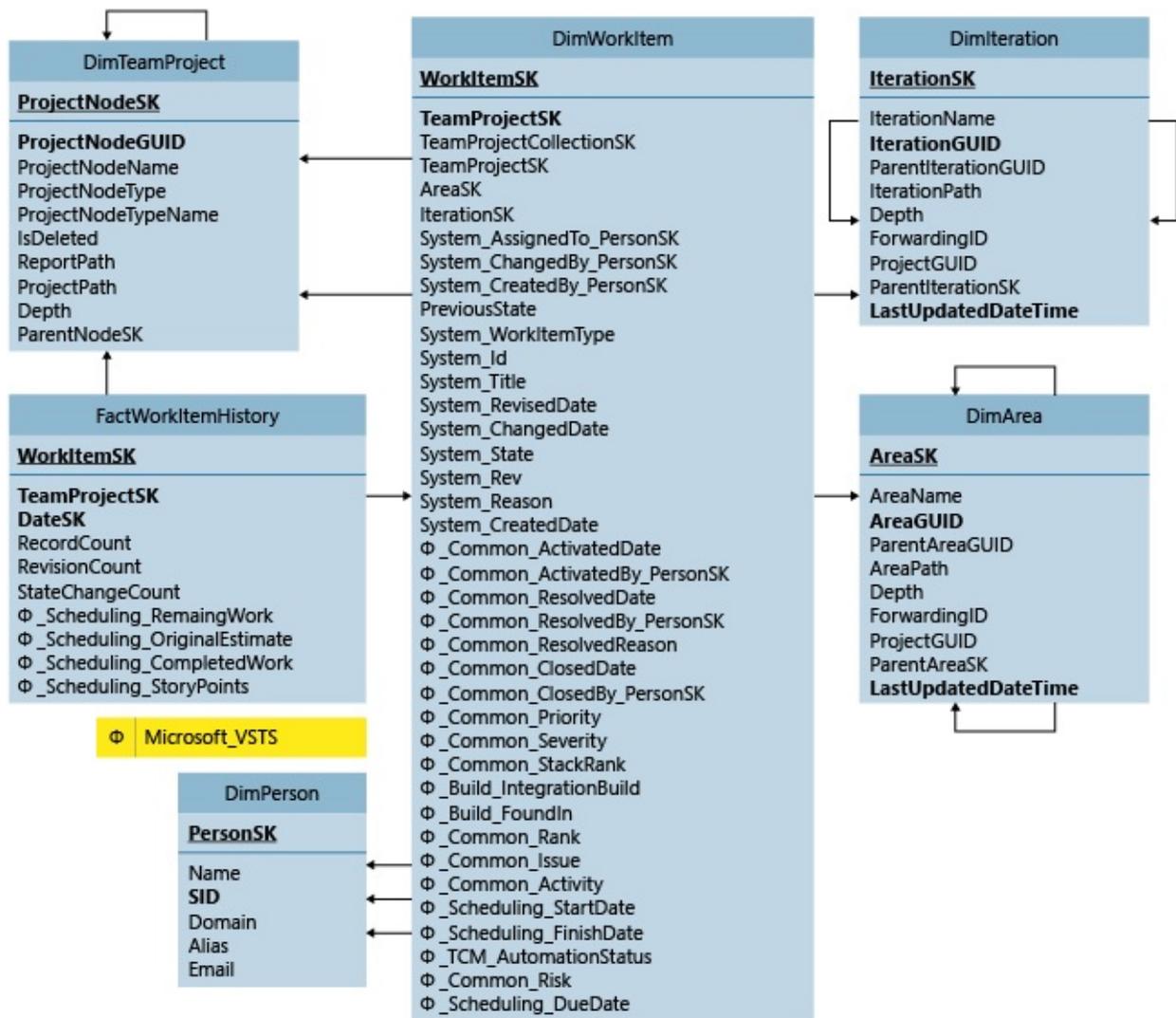
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Work Item History tables

[See Also](#) [Send Feedback](#)

You can query for historical data about bugs, tasks, and other types of work items by using FactWorkItemHistory and the associated dimension tables as the following illustration shows. Historical data provides information about the status of a work item or a value of a field for a work item as it changed over time. Progress and burndown charts are examples of reports that are built from work item history tables. The data is stored by using compensating records.

For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on work items and test case data using the Work Item perspective](#).



FactWorkItemHistory is associated with the following dimension tables:

- DimArea
- DimIteration
- DimPerson
- DimTeamProject
- DimWorkItem

You can use the following sample query to find the historical workload trend for the period between 2009-09-21 and 2009-09-30 for certain user stories. For each

user story in the team project, this query returns information about the total completed work, the original estimated work, the remaining work, and the total story points for every day during that period.

📌Note

This query assumes that a user story is linked to other work items through child links.

```
declare @TeamProjectNodeSK int
select @TeamProjectNodeSK = ProjectNodeSK from GetProjectNodeInfoFromTable
-- This table value function returns the ProjectNodeSK: the Surrogate Key

declare @TeamProjectCollectionGuid nvarchar(36)
select @TeamProjectCollectionGuid = pc.ProjectNodeGUID from DimTeamProjectCollection pc
-- This query finds the team project collection GUID by joining TeamProjectCollection

select
    d.DateSK
    ,wi.System_Title
    ,wi.System_Id
    ,coalesce(sum(wih_child.Microsoft_VSTS_Scheduling_CompletedWork),0)
    ,coalesce(sum(wih_child.Microsoft_VSTS_Scheduling_OriginalEstimate),0)
    ,coalesce(sum(wih_child.Microsoft_VSTS_Scheduling_RemainingWork),0)
    ,coalesce(sum(wih_child.Microsoft_VSTS_Scheduling_StoryPoints),0)
from
    DimDate d
cross apply
    DimWorkItem wi
cross apply
    GetWorkItemsTree(@TeamProjectCollectionGuid, wi.System_Id,
N'Child', d.DateSK) wit
left join
    FactWorkItemHistory wih_child
        on wih_child.WorkItemSK = wit.ChildWorkItemSK
where
    d.DateSK >= N'2009-09-21 00:00:00.000'
    and d.DateSK <= N'2009-9-30 00:00:00.000'
    and wi.TeamProjectSK = @TeamProjectNodeSK
    and wi.System_WorkItemType = N'User Story'
    and wi.System_ChangedDate <= d.DateSK
```

```
and wi.System_RevisedDate > d.DateSK
and wi.System_State = N'Active'
and (wih_child.RecordCount != -1 or wih_child.RecordCount is nul
group by d.DateSK, wi.System_Id, wi.System_Title
```

Additional resources

For more information, see the following page on the Microsoft Web site:

[COALESCE \(Transact-SQL\)](#)

For more information about compensating records, see the following page on the Microsoft Web site: [NEricson's Weblog](#).

See Also

Concepts

[Analyze and report on work items and test case data using the Work Item perspective](#)

Burndown Excel Report

Test Team Progress Excel Report

[Table reference for the relational warehouse database for Visual Studio ALM](#)



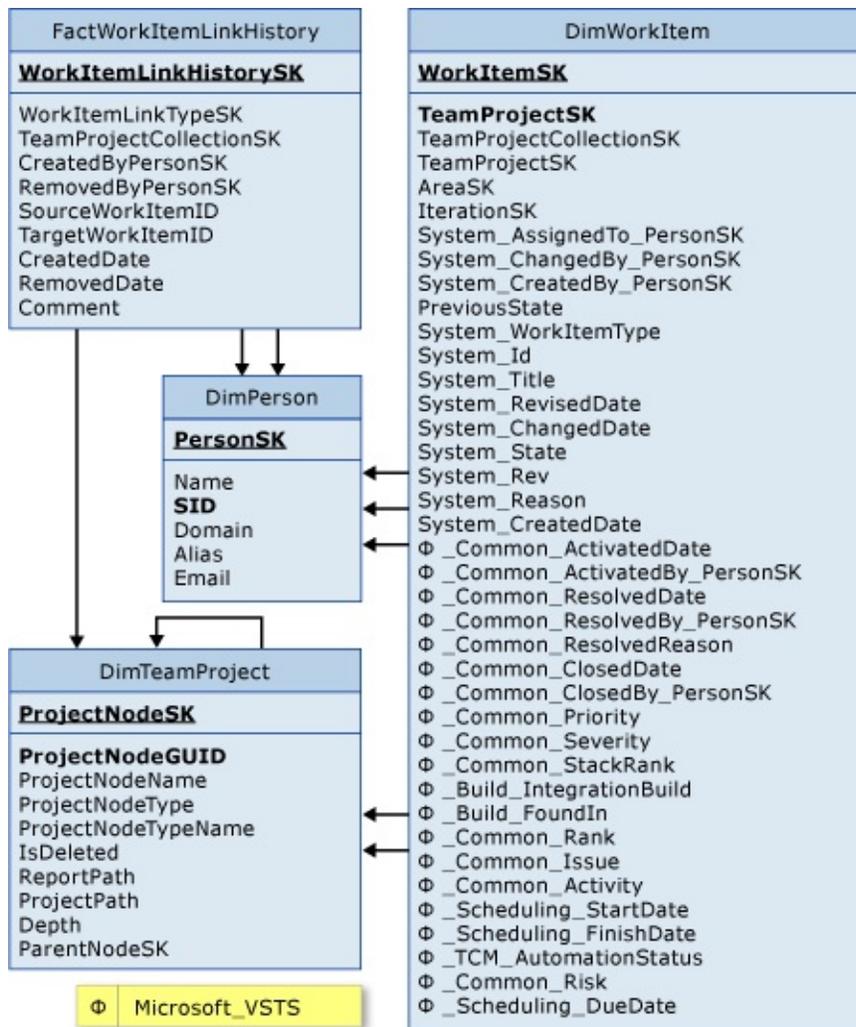
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Work Item Link History tables

[See Also](#) [Send Feedback](#)

You can query for links between bugs, tasks, and other types of work items by using FactWorkItemLinkHistory and the associated dimension tables. To include details about the linked work items, you join SourceWorkItemID and TargetWorkItemID to Dim.System_ID.

For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on work items and test case data using the Work Item perspective](#).



FactWorkItemLinkHistory is associated with the following dimension tables:

- DimTeamProject
- DimPerson
- DimWorkItem

📌 Note

This table contains links that have been removed. Links that have not been removed have the RemovedDate set to Jan 1, 9999. When a link is removed, the removed date is set to the date and time when it was removed. You can use `RemovedDate > GetDate()` to filter out links that have been removed.

You can use the following sample query to find the following types of information:

- total number of hours for completed work
- original estimated work
- remaining work
- total story points for each user story in a team project under a specified area path

For information about the Coalesce function that is used in the sample query, see the following page on the Microsoft Web site: [COALESCE \(Transact-SQL\)](#).

Note

This query assumes that a user story is linked to other work items through Child links.

```
declare @TeamProjectNodeSK int
select @TeamProjectNodeSK = ProjectNodeSK from GetProjectNodeInfoFromTeamProject
-- This table-valued function returns the ProjectNodeSK: the Surrogate Key for the Team Project

declare @TeamProjectCollectionGuid nvarchar(36)
select @TeamProjectCollectionGuid = pc.ProjectNodeGUID from DimTeamProjectCollection pc
-- This query finds the team project collection GUID by joining TeamProjectCollection to TeamProject

select
    wi.System_Title
    ,wi.System_Id
    ,coalesce(sum(cwi_child.Microsoft_VSTS_Scheduling_CompletedWorkHours),0)
    ,coalesce(sum(cwi_child.Microsoft_VSTS_Scheduling_OriginalEstimateHours),0)
    ,coalesce(sum(cwi_child.Microsoft_VSTS_Scheduling_RemainingWorkHours),0)
    ,coalesce(sum(cwi_child.Microsoft_VSTS_Scheduling_StoryPoints),0)
from
    DimWorkItem wi
cross apply
    GetWorkItemsTree(@TeamProjectCollectionGuid, wi.System_Id, N'Child') as cwi_child
left join
    FactCurrentWorkItem cwi_child
    on cwi_child.WorkItemSK = wi.ChildWorkItemSK
```

```
where
    wi.TeamProjectSK = @TeamProjectNodeSK
    and wi.System_WorkItemType = N'User Story'
    and wi.System_RevisedDate = CONVERT(datetime, '9999', 126)--The
    and wi.System_State = N'Active'
group by wi.System_Id, wi.System_Title
order by wi.System_Id
```

See Also

Concepts

[Analyze and report on work items and test case data using the Work Item perspective](#)

[Table reference for the relational warehouse database for Visual Studio ALM](#)

Other Resources

[Define a custom link type](#)

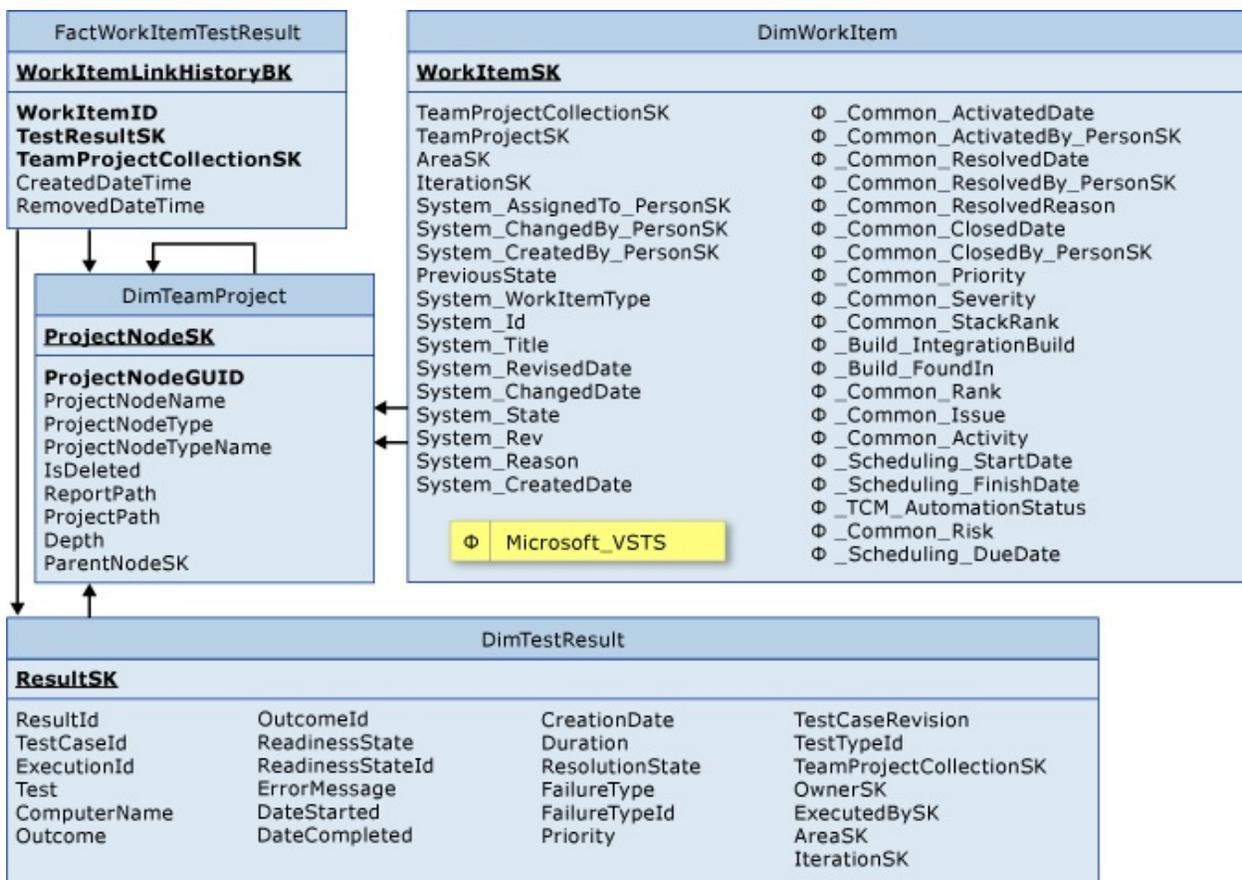


- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Work Item Test Result tables

[See Also](#) [Send Feedback](#)

You can query for data about work items that are linked to test results by using FactWorkItemTestResult and the associated dimension tables. For information about the measures and dimensions that are associated with these tables in the SQL Server Analysis Services cube, see [Analyze and report on test results using the test perspective in the Analysis Services database for Visual Studio ALM](#).



FactWorkItemTestResult is associated with the following dimension tables:

- DimTeamProject
- DimTestResult
- DimWorkItem

See Also

Concepts

[Analyze and report on test results using the test perspective in the Analysis Services database for Visual Studio ALM](#)

[Test Result tables](#)

[Current Work Item tables](#)

Testing the application

[Table reference for the relational warehouse database for Visual Studio ALM](#)

Other Resources

[Test Management Reports](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Perspectives and measure groups provided in the Analysis Services cube for Visual Studio

[See Also](#) [Send Feedback](#)

By using the SQL Server Analysis Services cube for Visual Studio Team Foundation Server, you can generate reports of aggregated information about the data that is stored in team project collections. You can easily use this data to create PivotTable and PivotChart reports in Office Excel. You can drag the cube elements onto PivotTable or PivotChart reports to formulate questions and retrieve answers quickly. The cube is optimized to answer questions such as "How many bugs were active, resolved, and closed on each day of the project?"

Note

If your data warehouse for Visual Studio ALM is using SQL Server Enterprise Edition, the list of cubes will include Team System and a set of perspectives. The perspectives provide a focused view of the data so that you do not have to scroll through all of the dimensions and measure groups that are defined for the whole Team System cube.

In this topic

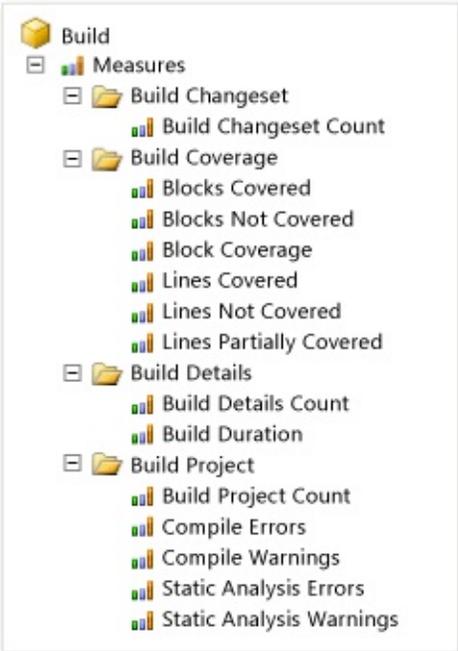
- [Perspectives and measure groups](#)
- [Combining dimensions and measures](#)
- [Cube terms and definitions](#)

▣ Perspectives and measure groups

The following table describes the measure groups, which are grouped according to their cube perspective. The Team System cube contains all measure groups in addition to the ones that are listed as belonging just to the Team System perspective. Most measure groups contain one or more cumulative counts. You can use cumulative counts to generate trend reports. In some cases, the name for a cumulative count has a trend label (for example, Build Result Count Trend and Point Count Trend).

📌 Note

All measure groups contain cumulative counts based on the start, end, creation, and completion dates for test results.

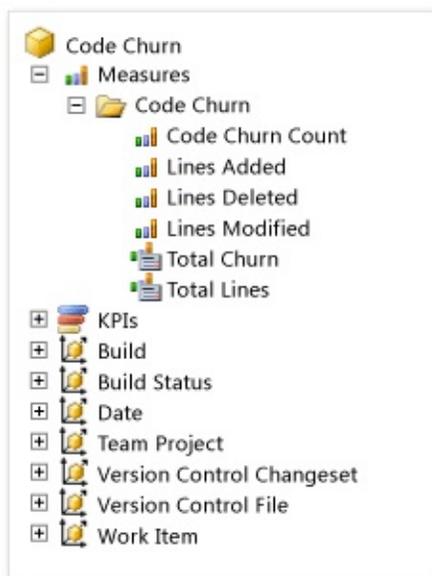
Perspectives and Measure Groups	Related topic and description
	

[Analyze and report on build details and build coverage using the Build](#)

[perspective](#)

Provides metrics that describe builds, such as build time and build frequency, and that can be analyzed by various dimensions, such as who performed the build, the build type, the build flavor, and the build outcome.

- Which builds failed, and which builds succeeded?
- Which builds have a significant number of changes to the code?
- How much of the code in a build or a group of builds did the tests cover?

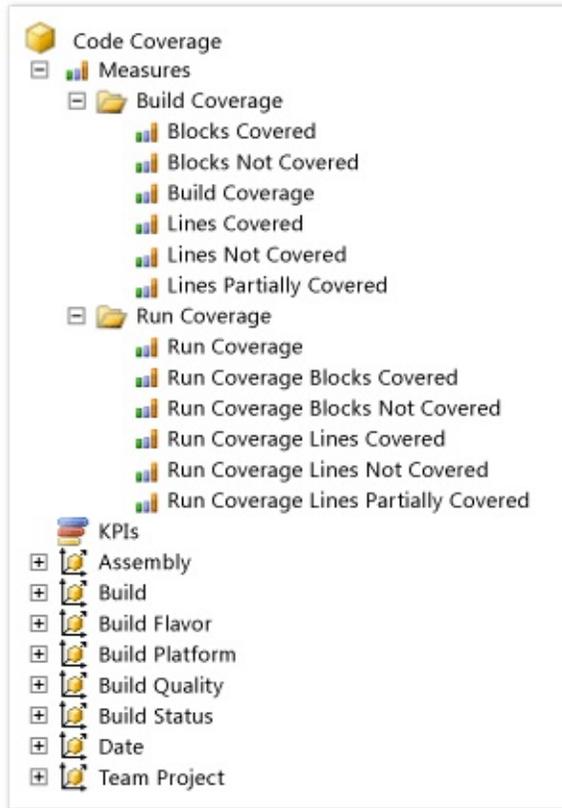


[Analyze and report on code churn and code coverage using the code churn and run coverage perspectives](#)

Supports reports that are focused on the number of file versions that are stored in Team Foundation version control and the extent to which the code has changed. Metrics can be analyzed by file directory, build, or team member who is checked in code. All totals can be analyzed over time so that you can answer the following types of questions:

- How many files of a specific file name extension changed in a particular build?
- How many lines of code are in the source base for a particular build?

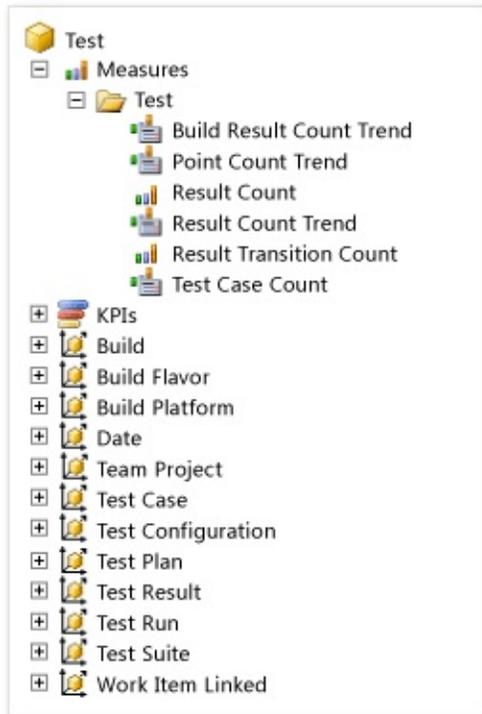
- Which changesets have been submitted, and what were the details of each change (for example, who performed the change, what files were modified, and on what date was the change made).



[Analyze and report on code churn and code coverage using the code churn and run coverage perspectives](#)

Provides metrics about how many lines and blocks of code were tested in various build and run configurations. You can use the Run Coverage perspective to create reports that answer the following questions:

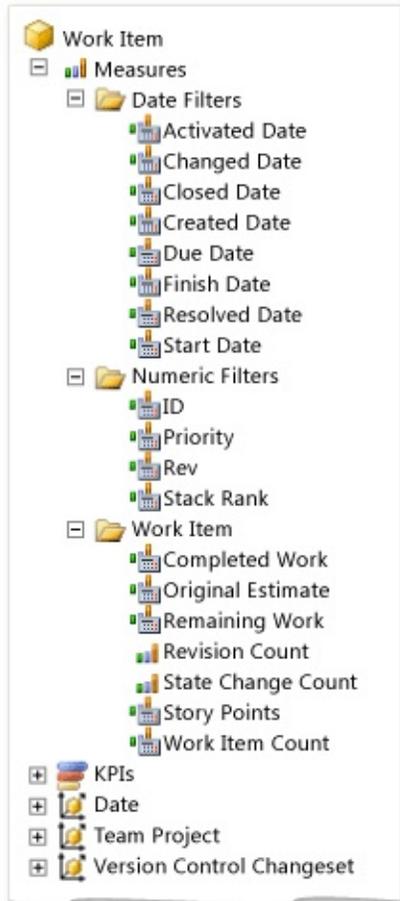
- Which assemblies and projects have the least code coverage?
- Which test runs give you the most code coverage?
- Which architectures or build types have the most code coverage?



[Analyze and report on test results using the test perspective in the Analysis Services database for Visual Studio ALM](#)

Provides metrics about test runs and test results. Test results are tracked over time and can be analyzed by their outcome, the build they were testing, the type of test, and other dimensions. By using the Test perspective, you can create reports that answer the following questions:

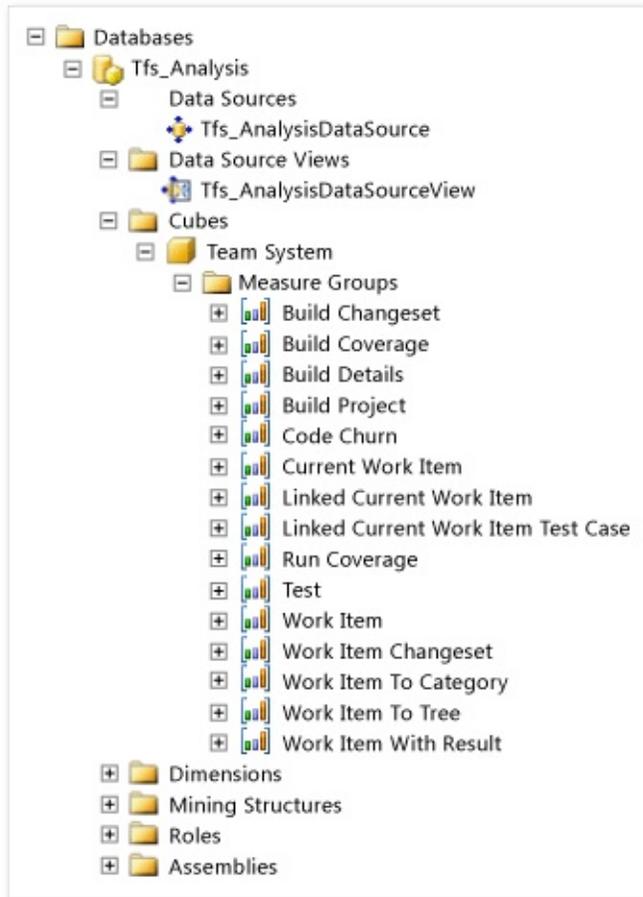
- What is the status of testing of specific user stories or product areas?
- What is the quality of builds based on the number of failed and passed tests?
- How many test cases have never been run?
- Which test cases have never been run?



[Analyze and report on work items and test case data using the Work Item perspective](#)

Provides metrics and detailed information about work items, including historical information that enables total work item counts to be analyzed over time or as of a current date. You can use this measure group to answer the following types of questions:

- What was the total count of active bugs each day in the last iteration?
- How many scenarios were active each month during the last year?
- How many bugs of each priority have been active each day in the last month?



Provides all metrics for all measure groups that appear previously in this table.

■ **Combining dimensions and measures**

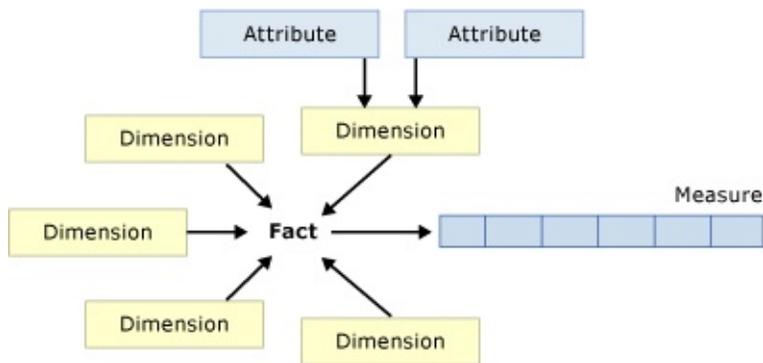
Not all combinations of dimensions and measures will produce useful results. For you to get meaningful results, the measures must relate to the dimensions in your reports. To determine which dimensions relate to a specific measure, you can check the fact tables in the data warehouse by using SQL Server Management Studio.

If you are using Excel, you can use the Show fields related to box to specify a group of measures and dimensions that cover a specific section of the cube. By working within a single set of related fields, you can find data more easily because not all fields in the cube appear, and you are much more likely to get useful numbers.

You can combine measures from different groups of fields as long as the measures have a common set of dimensions that are used in the report. The Date dimension, for example, appears in multiple groups of fields. Therefore, you could combine the Remaining Work or Completed Work data with test result counts for a day or days that you specify because both these sets of measures have Date as a related dimension.

▣ Cube terms and definitions

A cube represents a set of measures that are grouped into measure groups and hierarchically organized by dimensions. The cube is constructed from data retrieved from relational database. The following illustration shows the relationship that exists between different aspects of the cube.



The following table describes the terms and definitions that are used when referring to different aspects of the cube.

Term	Description
------	-------------

	Each attribute is connected to a column in a corresponding dimension table in the data warehouse. Each dimension is associated with a set of attributes and potentially a set of hierarchies. Area and iteration paths are examples of hierarchies.
--	---

	Some work item dimension attributes are also stored as numeric and date filter values. When you use one of
--	--

Attribute

these dimension attributes in the rows or columns section, you can use these values to filter the report. For example, you can filter a report to show work items that were created after Oct 1, 2009, by using the value filter

"System_CreatedDate is greater than Oct 1, 2009."

You can also use the measure values to filter a report. For example, you can filter the report to show only work items that have more than two hours of work remaining by using the value filter

"Remaining Work is greater than 2." For more information about value filters, see the following page on the Microsoft Web site:

[Filter Numbers in the Values Area.](#)

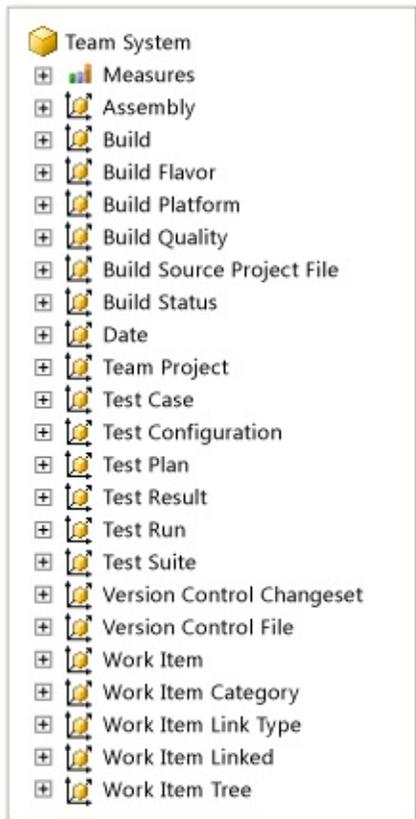
Dimension

Dimensions enable you to extract different views of data. Data values are associated with a set of dimensions that allow you to show aggregate results that are filtered using a specific set of dimension values.

You can use dimensions to disaggregate the data and show more detail. For example, you can use the Date dimension in the rows or columns section of a PivotTable or PivotChart report to show a trend over time. You can also use dimensions to filter the report. Place a dimension or dimension attribute in the filter area, and then specify the values that you want to include in the report.

Some dimensions are used in more than one measure group. For example, all measure groups share the Date, Team Project, Person, Area, and Iteration

dimensions. The following illustration shows the dimensions in the cube.



Dimensions are groups of attributes that are based on columns from tables or views in a data source view. Dimensions exist outside of a cube, can be used in multiple cubes, can be used multiple times in a single cube, and can be linked between Analysis Services instances. A dimension that exists outside of a cube is referred to as a database dimension, and an instance of a database dimension within a cube is referred to as a cube dimension.

Fact and fact table

A fact represents data that can be associated with multiple dimensions. This data may also be aggregated. Fact tables hold these values.

Each data warehouse includes one or more fact tables. Central to a "star" or "snowflake" schema, a fact table captures the data that measures the team's operations. Fact tables usually contain large numbers of rows, especially when they contain one or more years of history for a large team project.

A key characteristic of a fact table is that it contains numerical data (facts) that can be summarized to provide information about the history of the operation of the organization. Each fact table also includes a multipart index that contains, as foreign keys, the primary keys of related dimension tables. The related dimensions contain attributes of the fact records. Fact tables should not contain descriptive information or any data other than the numerical measurement fields and the index fields that relate the facts to corresponding entries in the dimension tables.

For a list of the fact tables that are defined for the data warehouse, see [Table reference for the relational warehouse database for Visual Studio ALM](#).

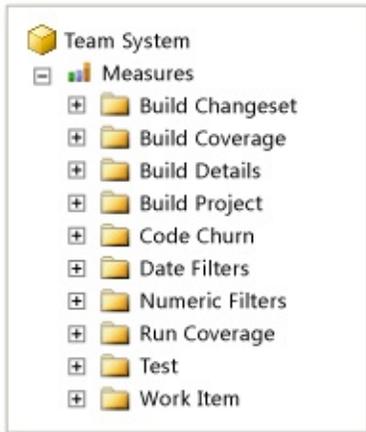
KPIs

In business terminology, a key performance indicator (KPI) is a quantifiable measurement for gauging business success.

In Analysis Services, a KPI is a collection of calculations that are associated with a measure group in a cube and that are used to evaluate business success. Typically, these calculations are a combination of Multidimensional Expressions (MDX) expressions or calculated members. KPIs also have additional metadata that provides information about how client applications should display the results of the KPI's calculations.

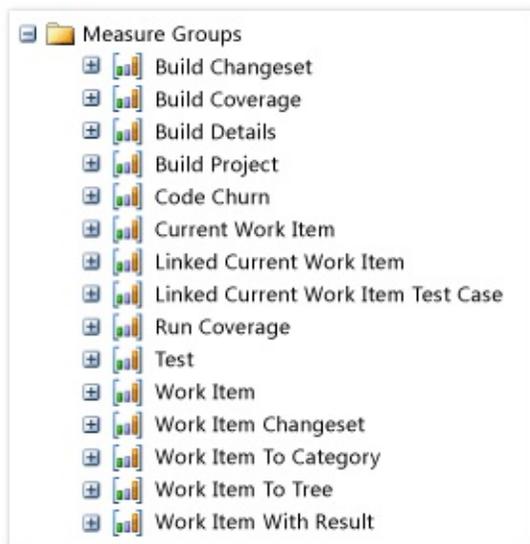
Measure

Measures are values that correspond to columns in the corresponding fact table. Also, fields whose reportable attribute is set to Measure appear as measures in the cube. The following illustration shows the measures in the cube.



Measure group

Each measure group contains measures, such as Work Item Count, and dimensions, such as Date and Team Project. The measures are the numeric values that provide summaries at different levels of aggregation. You can use them in the Values section of a PivotTable or PivotChart report. The following illustration indicates the measure groups for Team Foundation.



Perspective

By using perspectives, you can view portions of a cube, making it easier to focus on just the set of information that is of interest for creating a report.

Note

Perspectives are available only when your data warehouse for Visual Studio ALM is using SQL Server Enterprise Edition. Otherwise, you will see only a single perspective, the Team System cube.

Each perspective provides a focused view of the data so that you do not have to scroll through all of the dimensions and measure groups that are defined for the whole cube. A perspective is a subset of the features and objects of a cube.

See Also

Concepts

Resolve schema conflicts that are occurring in the data warehouse

Create Excel reports from a work item query

Create, customize, and manage reports for Visual Studio ALM

[Manage TFS reports, data warehouse, and analysis services cube](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Shared dimensions in the Analysis Services Cube

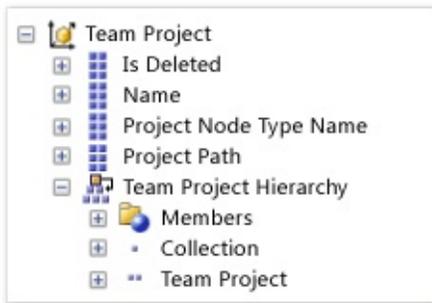
[See Also](#) [Send Feedback](#)

You can report data from across team project collections because all reportable data in a deployment of Team Foundation is written to the SQL Server Analysis Services cube for Visual Studio Team Foundation Server. You can use the Team Project dimension to show the data for only those team projects that you specify. You can use the Date dimension to create trend reports that show changes over a day, a week, a month, or a year. All perspectives in the cube share these dimensions.

You can use dimension attributes to summarize the measures in the cube according to various categories. For example, you can use the Priority, Work Item Type, State, Closed By, and Closed Date attributes to determine how many priority one bugs were closed by a specific team member in November.

Filter Reports or Mine Data Based on Project Collections and Team Projects

You can use the Team Project dimension to filter a report or highlight specific details of a project collection or team project. All measures and dimensions in the Team System cube record data that is specific to a team project and project collection. The following illustration shows the attributes within the Team Project dimension as it appears in the Report Builder Query Designer. The Team Project Hierarchy node contains two hierarchical attributes, the first at the collection level and the second at the team-project level.



You can filter the report to the team projects of interest by specifying one or more of the attributes in the following table.

Attribute	Description
Team Project Hierarchy	The hierarchy of project collections and team projects. Use this field when you must filter a report from many project collections and team projects.

Is Deleted	A Boolean value that specifies whether a project is deleted.
Name	The name of a team project or project collection.
Project Node Type Name	The name of the node. Valid values are Team Project and Team Project Collection.
Project Path	A flat list of team project paths that start with the project collection.

Create and Filter Trend Reports

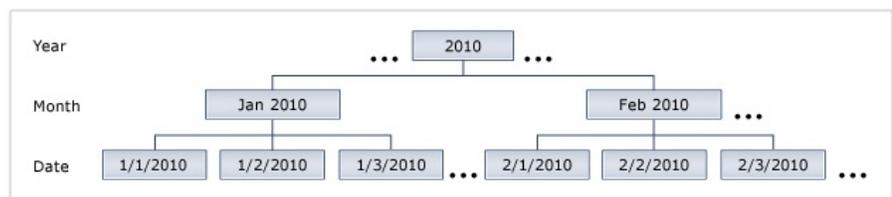
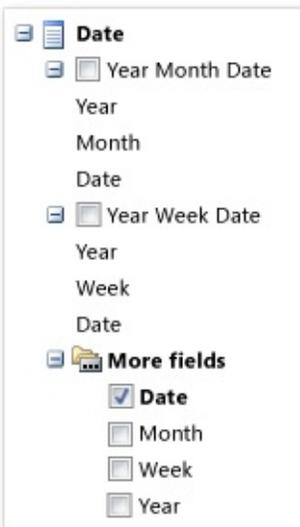
By using the Date dimension, you can create trend reports that summarize changes that occur over time to work items, builds, tests, and files in version control. Reports that include the Date dimension show information for the specific day on which an action occurred. All information in the data warehouse is tracked as it appeared at the end of each day.

Note

The dates in the Date dimension are stored according to the date that is set for the application-tier server that runs the warehouse adapters that move data from the data stores to the relational warehouse database. Dates are not adjusted based on time zone.

You can use date dimension hierarchies to aggregate values by day, week, month, or year. For example, you can compare the percentage of opened versus closed bugs from one month to another.

The first two attributes, Year Month Date and Year Week Date, correspond to hierarchical tree filters. The first hierarchy, as the following illustration shows, supports filtering by month. The second hierarchy supports filtering by week.



Under More fields, you can specify the granularity of the time axis for displaying trend reports as Date, Month, Week, or Year.

You can use the Date hierarchy with the Year Month Day attribute to aggregate measures by year, month, and the day of the month.

You can use the Date hierarchy with the Year Week Day attribute to aggregate measures by year, week of year, and the day of the week.

▣ **See Also**

Concepts

[Perspectives and measure groups provided in the Analysis Services cube for Visual Studio](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Analyze and report on build details and build coverage using the Build perspective

[See Also](#) [Send Feedback](#)

By using the Build perspective, you can view just the measures, dimensions, and attributes in the SQL Server Analysis Services cube for Visual Studio Team Foundation Server that pertain to the build process. For example, you can use these measures to determine how many builds are failing and how much of the code changed within a build.

The Build perspective is based on the relational tables that enable reporting on builds as either a property of the build, code coverage, or a changeset in version control. For more information, see

[Build Details tables](#), [Build Project tables](#), [Build Coverage tables](#), and [Build Changeset tables](#).

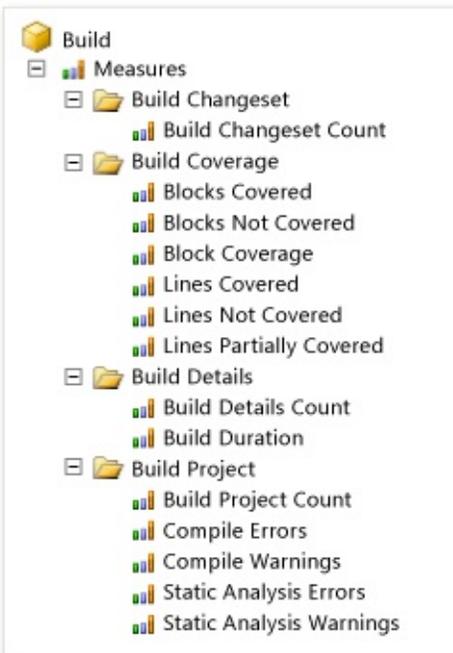
By using the Build perspective, you can create reports that answer the following questions:

Status reports:

- Which builds failed and which

builds
succeeded?

- Which builds reflect a significant number of changes to the code?
- Which builds are ready to install?



Trend reports:

- What is the status of all builds over time?
- How much of the code was executed by the tests over time?

Note

If your data warehouse for Visual Studio Application Lifecycle Management (ALM) is using

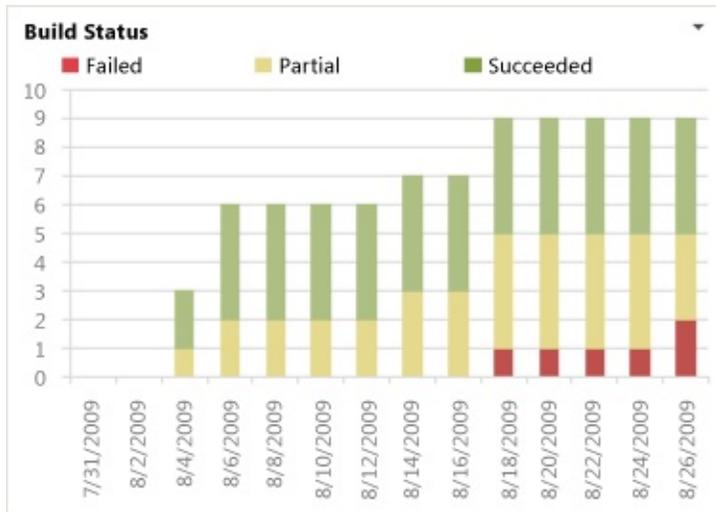
SQL Server Enterprise Edition, the list of cubes will include Team System and a set of perspectives. The perspectives provide a focused view of the data so that you do not have to scroll through all of the dimensions and measure groups that are defined for the whole Team System cube.

In this topic

- [Example: Build status report](#)
- [Build measures](#)
- [Dimensions and attributes in the build perspective that support filtering and categorization](#)
- [Required activities](#)

Example: Build status report

By using PivotChart reports in Excel, you can display the build status over time, similar to the data in the following illustration.



The process templates for Microsoft Solutions Framework (MSF) Agile and CMMI include the Build Status report in Excel. For more information, see [Build Status Excel Report](#).

Pivot field selection and filters

Drag fields between areas below:

Report Filter	Legend Fields (Series)
Project Path	Build Status Name
Year Week Date	
Build Flavor	
Axis Fields (Categories)	Values
Date	Build Details Count

You can create the summary report for build status by performing the following steps:

1. In Excel, connect to the Analysis Services cube for Team Foundation Server, and insert a PivotChart report.

For more information, see [Create Excel reports from a work item query](#).

2. Open the context menu for the chart and then choose Change Chart Type, Area, Stacked Column.
3. For each report filter, open the context menu for each of the following fields, specify the hierarchies, weeks, or other elements of interest, and then drag the field to the Report Filter area.
 - Team Project Hierarchy from the Team Project dimension
 - Year Week Date from the Date dimension
 - Build Definition Name from the Build dimension
4. In the Date dimension, expand More fields, and drag the Date, Week, or Month fields to the Axis Fields (Categories) area to specify how granular a report you want to generate.
5. Drag the Build Details Count field from the Build Details measure group to the Values area.
6. Drag the Build Status Name field from the Build Status dimension to the Legend Fields (Series) area.
7. (Optional) Filter the Build Status Name field to display only those builds that Failed, Partially Succeeded, or Succeeded.

Build measures

The following table describes the measures that are associated with builds. The Build Coverage measure group requires that the test team instrument tests to gather code coverage data. For more information, see

[Required activities](#) later in this topic. For an example of a report that uses several of these measures, see Build Quality Indicators Report.

Measure Group	Measure	Description
Build Details	Build Details Count	Number of times that a specific build has been run.
	Build Duration	Number of minutes that the build took to finish.
Build Changeset	Build Changeset Count	Number of changesets in the selected set of builds.
		Number of blocks that the selected build covers. If multiple test runs are performed

Build Coverage

Blocks Covered

against a build, the build coverage reflects the combined coverage of the runs. However, the runs may cover blocks that overlap.

Blocks Not Covered

Number of blocks that the selected build does not cover. If multiple test runs are performed against a build, the build coverage reflects the combined coverage of the runs. However, the runs may cover blocks that overlap.

Build Coverage

Number of builds that are associated with code coverage statistics.

Number of lines that the selected build covers. If

Lines
Covered

multiple test runs are performed against a build, the build coverage reflects the combined coverage of the runs. However, the runs may cover lines that overlap.

Lines Not
Covered

Number of lines that the selected build does not cover. If multiple test runs are performed against a build, the build coverage reflects the combined coverage of the runs. However, the runs might cover lines that overlap.

Lines
Partially
Covered

Number of lines that the selected build partially covers. If multiple test runs are performed against a build, the build coverage reflects

		the combined coverage of the runs. However, the runs might cover lines that overlap.
Build Project	Build Project Count	Number of .csproj files, .vbproj files, and other project files in the selected set of builds.
	Compile Errors	Number of compile errors that occurred for the selected builds.
	Compile Warnings	Number of compile warnings that occurred for the selected builds.
	Static Analysis Errors	Number of static analysis errors that occurred for the selected builds.
		Number of static

Static analysis
Analysis warnings that
Warnings occurred for the
selected builds.

Dimensions and attributes in the Build perspective that support filtering and categorization

You can use the attributes in the following table to aggregate a measure, filter a report, or specify a report axis. These attributes supplement the Team Project and Date shared dimensions that Working with Shared Dimensions describes.

Note

To use the Assembly, Build Flavor, or Build Platform dimension attributes, the test team must publish the test results to the TFS data store. For more information, see

[Required activities](#) later in this topic.

Dimension	Attribute	Description
Assembly	Assembly	(Published test results only) The name of the code of the application that is tested as part of the build. For more information, see Run tests in your build process. Name that is

Build	Build Definition Name	assigned to the build definition for which a build was executed.
	Build ID	The number that is assigned to the build. Each time that a particular build definition is run, the Build ID is incremented by 1.
	Build Name	The name or expression that uniquely identifies a build. For more information, see Use build numbers to give meaningful names to completed builds.
	Build Start Time	The date and time when the build started.

The reason why the build was run. Build types are associated with the trigger that was defined for the build.

Team

Foundation

Server

supports the following

types of builds:

Build Type manual, continuous (triggered by every check-in), rolling (accumulate check-ins until the previous build finishes), gated check-in, and scheduled. For more information, see Specify build triggers and reasons.

The Uniform Resource Locator (URL) for the completed

Drop
Location

build. A URL specifies the protocol with which web browsers will to locate Internet resources. Each URL includes the name of the server on which the details of the build resides. You can also include the path to a resource.

Build
Flavor

Build
Flavor

(Published test results only) A name that designates the category of builds that was assigned to a set of completed builds that were published as part of a test run. For example, a build flavor can designate a beta release or final release.

For more information, see Command-Line options for publishing test results.

The name of the platform for which an end-to-end (not desktop) build was made (for example, x86 or Any CPU). For an example of a report that uses this attribute, see Build Summary Report.

For more information, see Use the Default Template for your build process.

The quality of the build. For example, you can rate a

Build
Platform

Build
Platform

Build Quality	Build Quality	completed build's quality as Ready for Deployment, Rejected, or Under Investigation. For more information, see Add or remove build quality values.
Build Status	Build Status Name	The current state of the build. Valid values are Failed, Partially Succeeded, Stopped, Succeeded, and Unknown. For more information, see Manage your builds in Build Explorer.
Build Source Project File	File Hierarchy	The full network path of the source file. The extension

	File Extension	of the name of the source file.
Version Control Changeset	Changeset ID	The number that is assigned to the changeset.
	Checked In By	The user name of the team member who checked in the changeset.
	Description	The check-in comment that is associated with the changeset.
	Policy Override Comment	The comment that is provided when a policy is overridden. If a policy was not overridden with a changeset, the field is null.

Required activities

To create reports that contain useful data about builds, team members should review the information in the following topics:

- Run tests in your build process
- Using Code Coverage to Determine How Much Code is being Tested

See Also

Concepts

Build Quality Excel Report

Build Status Excel Report

Build Quality Indicators Report

Build Success Over Time Report

Build Summary Report

[Perspectives and measure groups provided in the Analysis Services cube for Visual Studio](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Analyze and report on code churn and code coverage using the code churn and run coverage perspectives

[See Also](#) [Send Feedback](#)

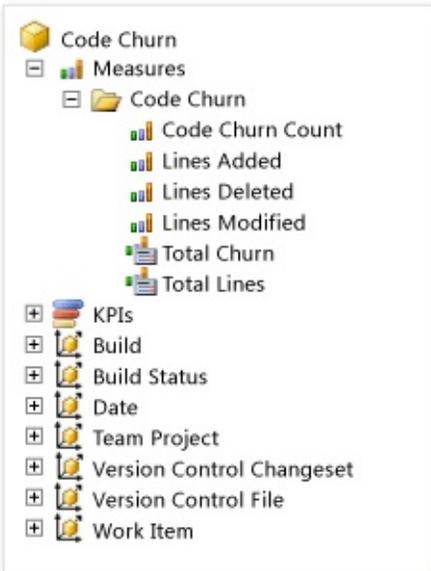
You can report on the software quality by using the Code Churn and Run Coverage perspectives from the SQL Server Analysis Services cube for Visual Studio Team Foundation Server. By using these perspectives, you can view just the measures, dimensions, and attributes that are associated with the changes in lines of codes and the extent to which code is covered in builds and test runs.

These perspectives are based on the relational tables that you can use to report on code changes and coverage as a property of the build, the build assembly or platform, the test run, or the changeset. For more information, see

[Code Churn tables](#) and [Run Coverage tables](#).

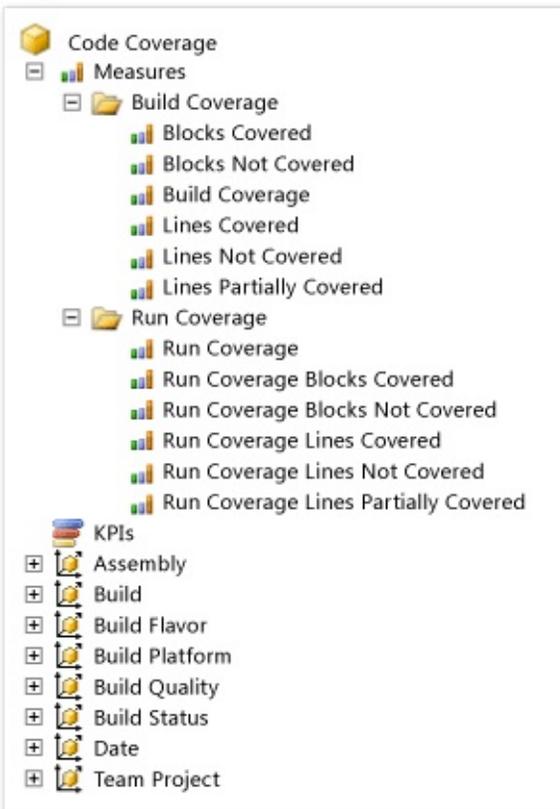
By using the Code Churn perspective, you can create reports that answer the following questions:

- How many files with a specific file name extension changed in a particular build?



- How many lines of code are in the source base for a particular build?
- Which changesets have been submitted, and what are the details of each change? (For example, who made the change, which files were changed, and on what date was the change made)?

By using the Run Coverage perspective, you can create reports that answer the following questions:



- Which assemblies have the least test coverage?
- Which test runs cover the most code?
- Which architectures or build types have the most test coverage?

Note

If your data warehouse for Visual Studio Application Lifecycle Management (ALM) is using SQL Server Enterprise Edition, the list of cubes will include Team System and a set of perspectives. The perspectives provide a focused view of the data so that you do not have

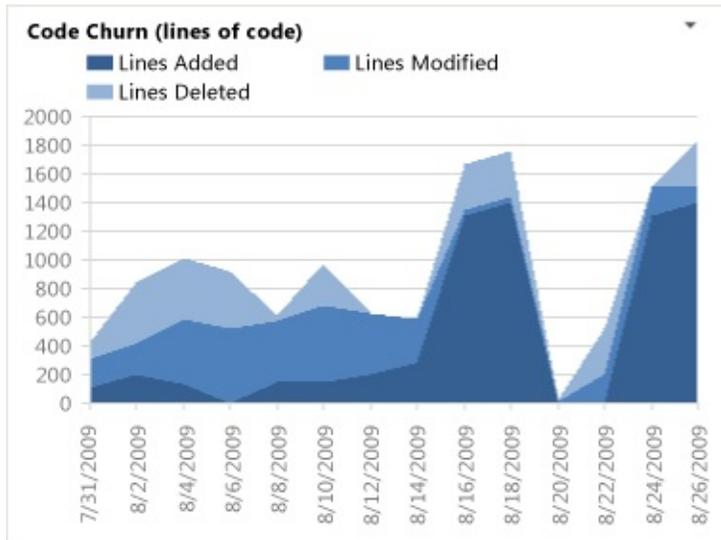
to scroll through
all of the
dimensions and
measure groups
in the whole
Team System
cube.

In this topic

- [Example: Code Churn Report](#)
- [Code Churn Measures](#)
- [Run Coverage Measures](#)
- [Dimensions and Attributes in the Code Churn Perspective That Support Filtering, and Categorization](#)
- [Dimensions and Attributes in the Run Coverage Perspective That Support Filtering and Categorization](#)
- [Required activities](#)

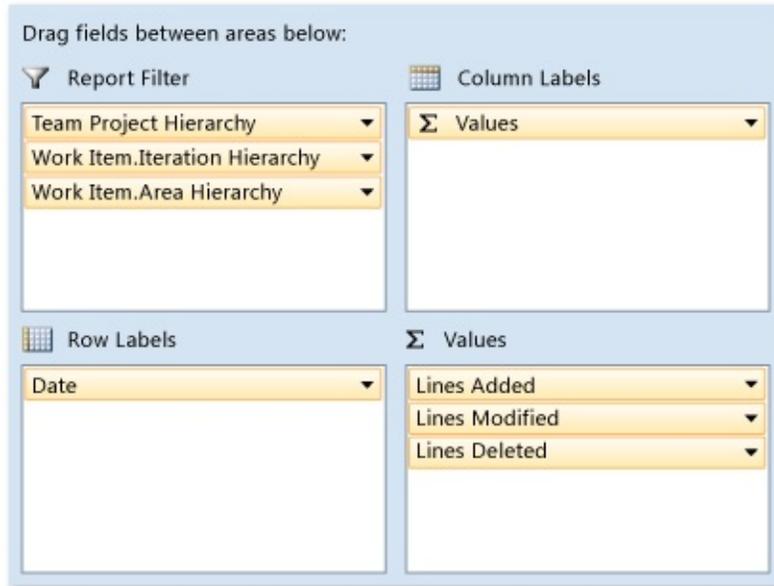
Example: Code Churn Report

By using a PivotChart report in Excel, you can create a trend report that displays the code churn over time, similar to the report that the following illustration shows.



The process templates for Microsoft Solutions Framework (MSF) Agile and CMMI provide the Code Churn report in Excel. For more information, see Code Churn Excel Report.

Selecting and filtering pivot fields



You can create a code churn report by performing the following steps:

1. In Excel, connect to the SQL Server Analysis Services cube for Visual Studio Team Foundation Server, and insert a PivotChart report.

For more information, see [Create Excel reports from a work item query](#).
2. Right-click the chart and then choose Change Chart Type, Area, Stacked Area.
3. For each report filter, open the shortcut menu for each of the following fields, specify the hierarchies, weeks, or other elements of interest, and then drag the field to the Report Filter area.
 - Team Project Hierarchy from the Team Project dimension
 - Work Item.Iteration Hierarchy from the Work Item dimension
 - Work Item.Area Hierarchy from the Work Item dimension
 - Year Week Date from the Date dimension
4. In the Date dimension, expand More fields, and drag the Date, Week, or Month fields to the Axis Fields (Categories) area based on how granular a report you want to generate.

5. Drag the Lines Added, Lines Modified, and Lines Deleted fields from the Code Churn measure group to the Values area. You must drag each field separately.

Code churn measures

Code churn measures quantify how much change is occurring in your project. In general, high levels of churn indicate project instability. You should expect high rates of churn at the start of a product cycle or after the team has implemented many changes. Toward the end of an iteration or before a release, you should expect the level of churn to decrease, which indicates that your project is more stable.

The following table describes the measures in the Code Churn measure group. By using these measures, you can create reports that show how many file versions are stored in Team Foundation version control and how much the code has changed. You can analyze metrics by file directory, build, or team member who checked in changes, and you can determine how those metrics change over time.

For information about similar metrics that you can collect for builds, see

[Analyze and report on build details and build coverage using the Build perspective.](#)

Measure	Description
Code Churn Count	The number of times that the team changed files in version control.
Lines Added	The number of lines of code that the team added to files for the dimensions that you specify.
Lines	The number of lines of code that the team deleted from

Deleted files for the dimensions that you specify.

Lines Modified The number of lines of code that the team modified during the time period that you specify.

Total Churn Churn in the code, computed as: [Lines Added] + [Lines Deleted] + [Lines Modified].

Total Lines The number of lines in the part of the file path hierarchy that you specify. You must also specify one or more builds to indicate the point or points at which to perform this calculation. If you do not specify one or more builds, NULL is returned. The number of lines is calculated by aggregating the lines added and lines deleted that have contributed to a specific combination of build type and operating system.

 **Tip**

The Total Lines measure can cause the OLAP query to timeout. If your report takes too long to render, consider shortening the changeset, build, test run, or date range.

|

Run coverage measures

The following table describes the measures in the Run Coverage measure group. By using these measures, you can create reports that show the extent to which the code was covered by tests in a test run. For information about similar metrics that you can collect for builds, see

[Analyze and report on build details and build coverage using the Build perspective.](#)

Measure	Description
Run Coverage	The number of test runs that have code coverage statistics associated with them.
Run Coverage Blocks Covered	The number of blocks that all tests in a run cover. However, coverage across the tests might overlap.
Run Coverage Blocks Not Covered	The number of blocks that are not covered by any tests in a run. However, coverage across the tests might overlap.
Run Coverage Lines Covered	The number of lines that all tests in a run cover. However, coverage across the tests might overlap.

Run Coverage Lines Not Covered The number of lines that are not covered by any tests in a run. However, coverage across the tests might overlap.

Run Coverage Lines Partially Covered The number of lines that tests in a run partially cover. However, coverage across the tests might overlap.

Dimension and attributes in the Code Churn Perspective that support filtering and categorization

The following table describes the dimensions and attributes in the Code Churn perspective. These attributes supplement the Team Project and Date shared dimensions, which Working with Shared Dimensions describes. You can aggregate the measures along each of these attributes.

Dimension	Attribute	Description
Build	Build Definition Name	The name that is assigned to the build definition for which a build was run.
	Build ID	The number that is assigned to the build. Each time that a particular build definition is run, this attribute is incremented by 1.
		The name or

Build Name expression that uniquely identifies a build. For more information, see Use build numbers to give meaningful names to completed builds.

Build Start Time The date and time when the build started.

Build Type The reason why the build was run. Build types are associated with the trigger that was defined for the build. Team Foundation Server supports the following types of builds: manual, continuous (triggered by every check-in), rolling

(accumulate check-ins until the previous build finishes), gated check-in, and scheduled. For more information, see Specify build triggers and reasons.

Drop
Location

The Uniform Resource Locator (URL) for the completed build. A URL specifies the protocol with which web browsers will locate Internet resources. Each URL includes the name of the server on which the details of the build reside. You can also include the path to a resource.

Version Control Changeset	Changeset ID	The number that is assigned to the changeset that included the file changes.
	Checked In By	The user name of the team member who checked in the changeset.
	Description	The check-in comment that is associated with the changeset.
	Policy Override Comment	The comment that is provided when a policy is overridden. If a policy was not overridden with this changeset, this field is null.
Version Control File	Version Control File.File Hierarchy	The full network path of the source file.

Version
Control File.File Extension The extension of the name of the source file.

Work Item Type and more For more information, see

[Analyze and report on work items and test case data using the Work Item perspective.](#)

■ Dimensions and Attributes in the Run Coverage Perspective That Support Filtering and Categorization

The following table describes the dimensions and attributes in the Run Coverage perspective. These attributes supplement the Team Project and Date shared dimensions that Working with Shared Dimensions describes later in this topic. You can aggregate the measures along each of these attributes.

■ Note

Before you can use the Assembly or Build Flavor attributes, the test team must specify them and publish the test results to the data store for Team Foundation Server. For more information, see

[Required activities](#) later in this topic.

Dimension	Attribute	Description
Assembly	Assembly	(Published test results only) The name of the code of the application that is tested as part of the build. For more information, see Run tests in your build process.

Build

Build Definition Name The name that is assigned to the build definition for which a build was run.

Build ID The number that is assigned to the build. Each time that a particular build definition is run, the Build ID is incremented by 1.

Build Name The name or expression that uniquely identifies a build. For more information, see Use build numbers to give meaningful names to completed builds.

Build Start Time Date and time when the build started.

The reason why

Build
Type

the build was run. Build types are associated with the trigger that was defined for the build.

Team

Foundation

Server supports the following types of builds:

manual,

continuous

(triggered by every check-in),

rolling

(accumulate

check-ins until

the previous

build finishes),

gated check-in,

and scheduled.

For more

information, see

Specify build

triggers and

reasons.

The Uniform

Resource

Locator (URL)

for the

completed build.

A URL specifies

the protocol

with which web

browsers will

locate Internet

Drop

Location resources. The URL also includes the name of the server on which the resource resides. You can also specify the path to a resource.

Build Flavor

Build Flavor

(Published test results only) A name that designates the category that is assigned to a set of completed builds that were published as part of a test run. For example, you can use a build flavor to designate a beta release or final release.

(Published test results only) The name of the machine platform for which an end-to-end (not desktop) build

Build Platform	Build Platform	was made and that was published as part of a test run (for example, x86 or Any CPU). For an example of a report that uses this attribute, see Build Summary Report.
----------------	----------------	---

Test Run	Complete Date Hierarchy by Month or by Week	Date dimensions that are based on the date when the test run was created and finished. For more information, see Shared dimensions in the Analysis Services Cube .
	Creation Date Hierarchy by Month or by Week	

Required activities

To create reports that contain code churn and code coverage data, team members should review the information in the following topics:

- Run tests in your build process Using Code Coverage to Determine How Much Code is being Tested
- Configuring Unit Tests by using a .runsettings File

See Also

Concepts

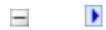
Code Churn Excel Report

Code Coverage Excel Report

[Code Churn tables](#)

[Run Coverage tables](#)

[Perspectives and measure groups provided in the Analysis Services cube for Visual Studio](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Analyze and report on test results using the test perspective in the Analysis Services database for Visual Studio ALM

[See Also](#) [Send Feedback](#)

By using the Test perspective in the SQL Server Analysis Services cube for Visual Studio Team Foundation Server, you can view just the measures, dimensions, and attributes that pertain to reporting on tests results and test runs. For example, you can use these measures to determine the overall quality of each build, the tests that a particular build affected, and the number of test cases that were run. You can also answer questions about changes to the result outcomes.

The Test measure group is based on the Test Results relational table, which enables reporting on test results as either a property of the tests or an independent outcome. For more information, see

[Test Result tables.](#)

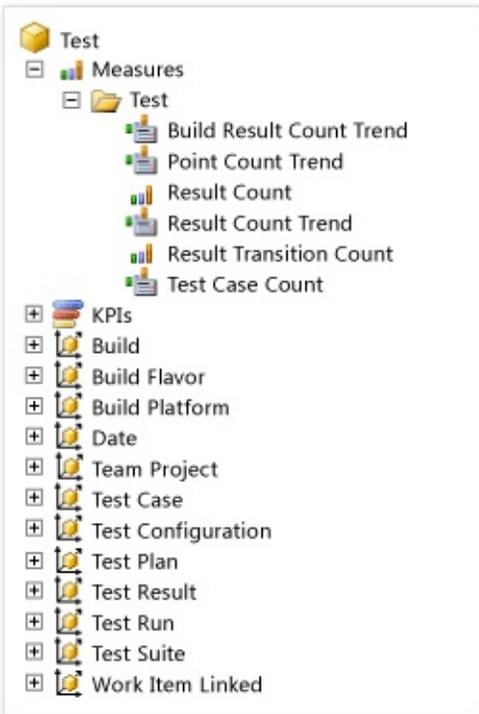
By using the Test perspective, you can create reports that answer the following questions:

Status reports:

- What is the status of testing of specific

user stories
or product
areas?

- What is the quality of builds based on the number of failed and passed tests?
- How many test cases have never been run?
- Which test cases have never been run?



Trend reports:

- How many tests are blocked, passing, or failing over time?
- How many tests are regressing?
- How consistent is the manual test

activity
over time?

 **Note**

If your data warehouse for Visual Studio Application Lifecycle Management (ALM) is using SQL Server Enterprise Edition, the list of cubes will include Team System and a set of perspectives. The perspectives provide a focused view of the data so that you do not have to scroll through all of the dimensions and measure groups in the whole Team System cube.

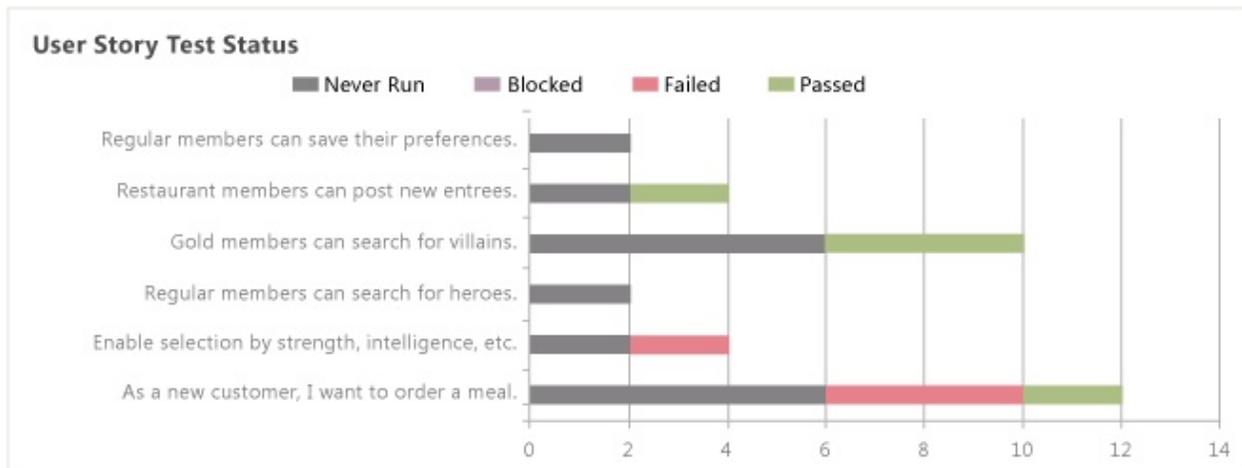
To use many Test measures and dimension attributes, the test team must publish the test results to the data store for Team Foundation Server. For more information, see [Required Activities for Managing Tests and Builds](#) later in this topic.

In this topic

- [Example: Progress report for testing user stories](#)
- [Test measures](#)
- [Dimensions and attributes in the Test Perspective that support filtering and categorization](#)
 - [Build, build flavor, and build platform dimensions](#)
 - [Test case, test configuration, test plan, and test suite dimensions](#)
 - [Test result dimension](#)
 - [Test run dimension](#)
 - [Work item and work item linked dimensions](#)
- [Required activities](#)

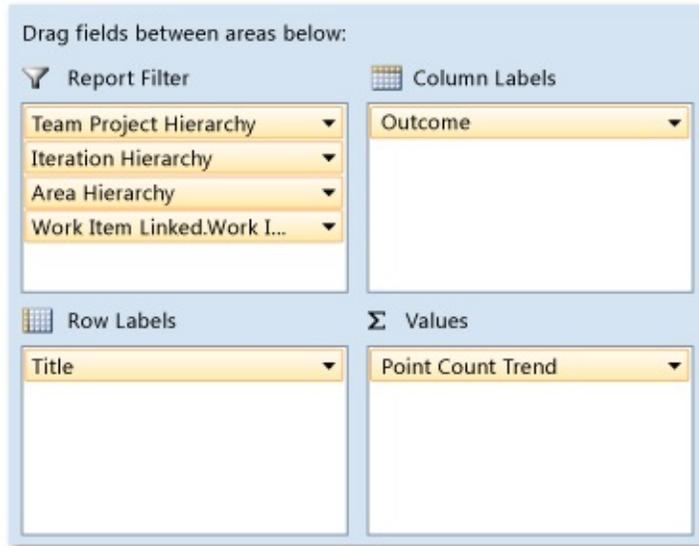
Example: Progress report for testing user stories

By using PivotTable and PivotChart reports in Excel, you can create a status report that shows the test progress on user stories, similar to the report in the following illustration.



The process templates for Microsoft Solutions Framework (MSF) Agile and CMMI include the User Story Test Status Excel Report (Agile) and Requirement Test Status Excel Report (CMMI) in Excel, respectively.

Specify and filter pivot fields



By performing the following steps, you can create a progress report for testing user stories:

1. In Excel, connect to the Analysis Services cube for Team Foundation Server, and then insert a PivotChart report.

For more information, see [Create Excel reports from a work item query](#).

2. Right-click the chart, then choose Change Chart Type, Area, Stacked Bar.
3. For each report filter, right-click each of the following fields, specify the hierarchies or elements of interest, and then drag the field to the Report Filter area.

- Team Project Hierarchy from the Team Project dimension
- Area Path from the Team Project dimension
- Iteration Path from the Test Case dimension
- Work Item Type from the Work Item Linked dimension

Specify the type as user story, requirement, or another type of work item that has test cases linked to it that you want to report.

4. Drag the Point Count Trend field from under the Test measure group to the

Values area.

5. Drag the Outcome field from under the Test Result dimension to the Column Labels area.

Test measures

The following table describes the measures that the Test measure group includes. You can analyze test results by the aggregate of tests results and their outcome for a particular build or by the changed outcome for a test result.

Measure	Description
Build Result Count Trend	Counts the most recent version of each result in a particular build. For an example of a report that uses this measure, see Build Quality Excel Report.
Point Count Trend	Count of the most recent version of each test result in a particular build. If a test is run multiple times against a build, the Point Count Trend counts the most recent result for that test using that build. If a test case is not included in the build, the test case is counted as "Never Run."
	Use this measure to determine which tests or how many tests are failing in the current build.
	Counts the most recent

Result
Count

version of each test result.
Use this measure when you
want to determine the
overall volume of testing.

For an example of a report
that uses this measure, see
Build Quality Indicators
Report.

Result
Transition
Count

Counts all the results whose
outcome changed in a
particular build. Use this
measure when you want to
determine which tests were
affected by a particular
build.

Test Case
Count

Number of test cases. Use
this measure when you want
to determine how many test
cases were run for a
particular test run or build.

▣ Dimensions and attributes in the Test Perspective that support filtering and categorization

By using the attributes that this section describes, you can aggregate a measure, filter a report, or specify a report axis. These attributes are in addition to the Team Project and Date shared dimensions that Working with Shared Dimensions describes.

▣ Build, build flavor, and build platform dimensions

You can filter test reports based on build definition, build flavor, or build platform by using the attributes that the following table describes.

Dimension	Attribute	Description
Build	Build Definition Name	The name that is assigned to the build definition for which a build was executed. For an example of a report that uses this attribute, see Build Quality Excel Report.
		The number that

Build ID is assigned to the build. Each time that a particular build definition is run, the Build ID is incremented by 1.

Build Name The name or expression that uniquely identifies a build. For more information, see Use build numbers to give meaningful names to completed builds.

Build Start Time The date and time when the build started.

The reason why the build was run. Build types are associated with the trigger that was defined for the build.

Team
Foundation
Server supports

Build
Type

the following types of build: manual, continuous (triggered by every check-in), rolling (accumulate check-ins until the previous build finishes), gated check-in, and scheduled. For more information, see Specify build triggers and reasons.

Drop
Location

The drop folder that is defined for the build and that is specified as a Uniform Resource Locator (URL). A URL specifies the protocol with which web browsers will locate Internet resources. The URL also includes the name of the server on which the resource resides. You can

also include the path to a resource.

For more information, see

[Select a staging location and set up a drop folder.](#)

Build Flavor

Build Flavor

(Published test results only) A name that designates the category of builds that are assigned to a set of completed builds that were published as part of a test run. For example, a build flavor can be used to designate a beta release or a final release.

Build Platform

Build Platform

The name of the machine platform for which an end-to-end (not desktop) build was made (for example, x86 or Any CPU). For more information, see Use the Default Template for your build process.

■ **Test case, test configuration, test plan, and test suite dimensions**

The Test Case, Test Configuration, Test Plan, and Test Suite dimensions correspond to how you can organize, configure, automate, and run tests by using Microsoft Test Manager from Visual Studio 2010 Ultimate or Visual Studio Test Professional.

The test case corresponds to a type of work item that the test team uses to define both manual and automated tests that your team can run and manage by using Microsoft Test Manager. A test plan consists of test configurations and test suites. A test configuration defines the software or hardware on which you want to run your tests. A test suite defines a hierarchy within the plan so that you can group test cases together.

For more information, see [Testing the application](#).

Dimension Attribute	Description
Test Case Area Hierarchy and more	The Work Item and Test Case dimensions contain all attributes that relate to work items, such as State, Work Item Type, and Work Item ID. For information about the structure of the Test Case dimension, see

[Analyze and report on work items and test case data using the Work Item perspective](#).

For a description of each attribute, see [Work item field reference for Visual Studio ALM](#).

For information about how to work with date, area, and iteration hierarchies, see [Shared dimensions in the Analysis Services Cube](#).

This measure group contains additional attributes when custom fields in the definition for a type of work item specify Dimension as the reportable attribute. For more information about how to use the optional reportable attribute and its values, see [Add or modify work item fields to support reporting](#).

Test Configuration

Configuration ID and Configuration Name

The number that the system assigns and the name of a test configuration.

Test Plan

Area Hierarchy, Area Path, Iteration Hierarchy, and Iteration Path

The product area and milestone that is assigned to the test plan.

For more information, see [Analyze and report on work items and test case data using the Work Item perspective](#).

End Date Hierarchy By Month or By Week

Start Date Hierarchy By Month or By Week

Optional values that a test plan owner can assign to the test plan. They represent the date on which the test plan should start and the date on which the test plan should finish.

For more information about how to work with date hierarchies, see [Shared dimensions in the Analysis Services Cube](#).

Test Plan Id and Test Plan Name

The number that the system assigns and the name that the test plan owner assigns.

Test Plan Owner

The user name of the test team member who created or currently is assigned as the owner of the test plan.

Test Plan ID and State

The system-assigned number and name of the state of the test plan. For example, Inactive indicates that the test plan is being defined, and Active indicates that the test plan is ready to be reviewed and run.

Test Suite

Test Suite Hierarchy

Provides a mechanism to specify multiple filters based on project collection, team project, and test suite.

Suite Path

Corresponds to the hierarchy of test suites that are configured for all team projects in all team project collections.

■ **Test result dimension**

The following table lists all dimensions and attributes that are specific to the test measures in the cube. Before you can report on Failure Type or Resolution, the test team must populate this information as part of their test activities.

Attribute	Description
Failure Type and Failure Type Id	Corresponds to one of the following reasons why a test failed: None, Known Issue, New Issue, or Regression. Microsoft Test Manager automatically assigns a number or an ID to each reason. The test team can, but is not required to, assign a failure type to each failed test. ■ Note You cannot add to or change the set of failure types.

For an example of a trend report that shows the outcome of test results based on failure type, see Failure Analysis Excel Report.

Outcome and Outcome Id

The outcome of the test (for example, Passed, Failed, or Inconclusive).

For an example of a trend report that shows the outcome of test plans and test configurations, see Test Plan Progress Report.

Readiness State and Readiness State Id

The state of a particular test within a test run. Valid values are Completed, InProgress, None, NotReady, and Ready.

Resolution State

(Optional) The name of the Resolution with which a tester identified the cause of a failed test. By default, all MSF process templates have the following resolution states: Needs investigation, Test issue, Product issue, and Configuration issue. The test team can, but is not required to, assign a resolution state to each

failed test.

Note

You can change these states or add states using the tcm command line tool. See

[Customize and manage the test experience \[tcm and Microsoft Test Manager\]](#).

Test Result Executed By

The name of the user or other account under which the test was run.

For an example of a report that uses this attribute, see Test Team Productivity Excel Report.

Test Result Owner

The name of the user or other account that is assigned as the owner of the test result. The assignment corresponds to the value that is set by using the tcm /resultowner switch.

Test Result Priority

The priority of a particular test within a test run.

Test run dimension

The following table describes the attributes that are defined for the Test Run dimension. Many of these attributes correspond to parameters that the test team specifies when it runs tests.

Attribute	Description
Complete	

Date, Creation Date, Start Date, Hierarchy By Month or By Week	Dates when the test run was created, completed, or started. You can use these attributes to filter or structure a report. For more information, see
--	---

[Shared dimensions in the Analysis Services Cube.](#)

Is Automated

Flag that indicates that the test run contains one or more automated tests.

For an example of a report that uses this attribute, see Build Quality Excel Report.

Is Build Verification Run

Flag that indicates whether the test run contains build verification tests that check the basic functionality of the build. This flag corresponds to the tcm /buildverification switch.

For an example of a report that uses this attribute, see Build Quality Excel Report.

Test Run Id

The number that the system assigned to the test run.

Test Run Owner

Corresponds to the owner who is assigned to the test run that the test team created or published. Corresponds to the tcm /owner switch.

Test Run State and Id

Name or number that is assigned to the state of a test run (for example, Aborted, Completed, In Progress, Not Started, or Unknown).

Test Run Title

Corresponds to the title that is assigned to the test run that the test team created or published. Corresponds to the tcm /title switch.

▣ **Work item and work item linked dimensions**

You can link test cases to other work items such as user stories, requirements, and bugs. By using the Work Item Linked dimension, you can create a report that provides test results that relate to the linked work items. The progress report for testing user stories, described earlier in this topic, provides an example of using the linked work item.

For a description of each attribute, see

[Work item field reference for Visual Studio ALM.](#)

Required activities

To create reports that contain useful data about test efforts and test results, team members should review the information in the following topics:

- Run tests in your build process
- Using Code Coverage to Determine How Much Code is being Tested

See Also

Concepts

Test Plan Progress Report

[Perspectives and measure groups provided in the Analysis Services cube for Visual Studio](#)

Other Resources

[Test Management Reports](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Analyze and report on work items and test case data using the Work Item perspective

[See Also](#) [Send Feedback](#)

You can analyze current or historical data for work items and test cases by using the measures, dimensions, and attributes that are defined for the Work Item perspective in the SQL Server Analysis Services cube for Visual Studio Team Foundation Server. A test case is a type of work item that is associated within its own dimension and used specifically to support Microsoft Test Manager. For more information, see [Create Manual Tests using Team Web Access](#).

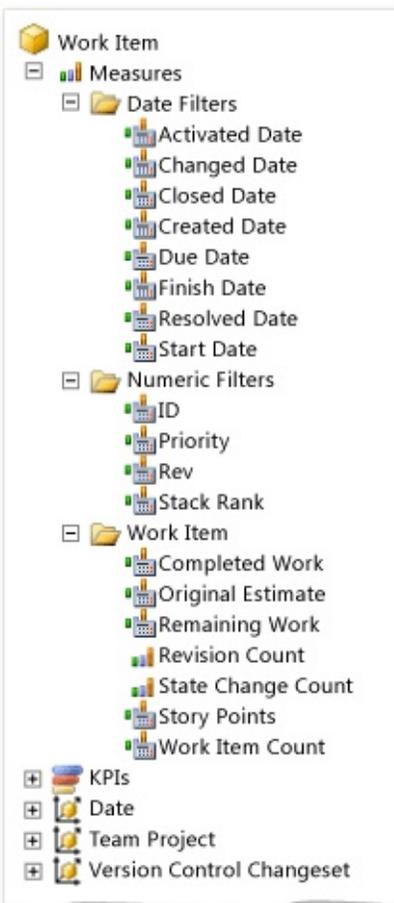
The Work Item perspective is based on the relational tables that enable reporting on work items as either a property of the work item or a linked work item. For more information, see

[Work Item History tables](#).

Note

You can use Create Report in Microsoft Excel to create status and trend reports based on a work item query. For more information, see [Create Excel reports from a work item query](#).

By using the Work Item perspective, you can create reports that answer the following questions:



- What was the total count of active bugs each day in the last iteration?
- How many scenarios were active each month during the last year?
- How many bugs of each priority have been active each day in the last month?
- How much outstanding and remaining work has a set of work items had over the last month?
- How much work did a particular group of developers

finish?

- How much additional work was created after a particular date?

 **Note**

If your data warehouse for Visual Studio Application Lifecycle Management (ALM) is using SQL Server Enterprise Edition, the list of cubes will include Team System and a set of perspectives. The perspectives provide a focused view of the data so that you do not have to scroll through all of the dimensions and measure groups in the whole Team

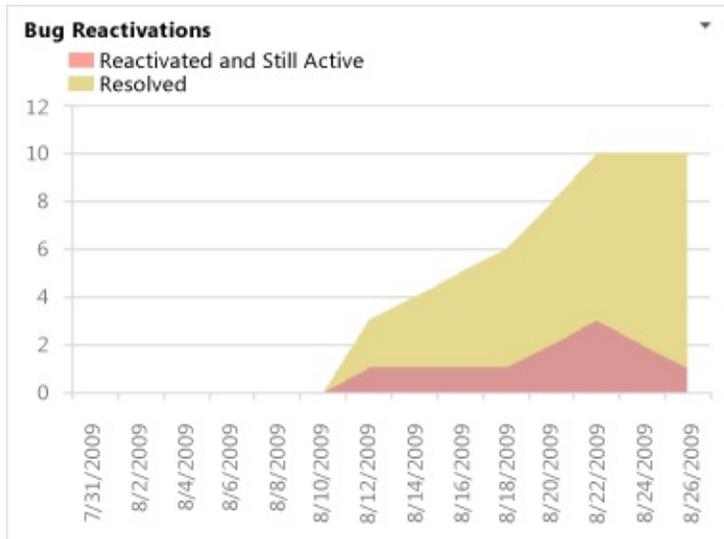
System cube.

In this topic

- [Example: Bug Reactivations Report](#)
- [Work Item Measures](#)
- [Date Filters and Numeric Filters](#)
- [Dimensions and Attributes in the Work Item Perspective That Support Filtering and Categorization](#)
 - [Filter by Work Item Fields](#)
 - [Filter by Link Type or Fields That Are Defined for a Linked Work Item](#)
 - [Filter By Changesets](#)
 - [Filter By Category](#)
 - [Filter By Hierarchical Relationships](#)
 - [Filter By Test Result](#)
- [Required Activities to Monitor and Track Work](#)

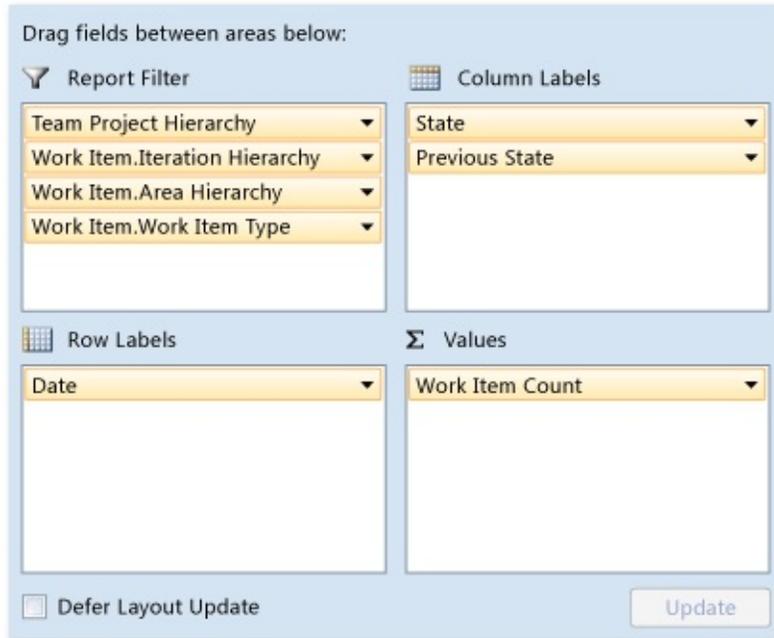
Example: Bug Reactivations Report

By using PivotChart reports in Excel, you can display the number of bugs that were closed and then reactivated over time, similar to the data in the following illustration.



The process templates for Microsoft Solutions Framework (MSF) Agile and CMMI include the Bug Reactivations report in Excel. For more information, see [Bug Reactivations Excel Report](#).

Select and filter pivot fields



You can create a bug reactivations report by performing the following steps:

1. In Excel, connect to the Analysis Services cube for Team Foundation Server, and insert a PivotTable report.

For more information, see [Create Excel reports from a work item query](#).

2. Open the shortcut menu for the chart, choose Change Chart Type, choose Area, and then choose Stacked Area.
3. For each report filter, open the shortcut menu for each of the following fields, specify the hierarchies, weeks, or other elements of interest, and then drag the field to the Report Filter area.
 - Team Project Hierarchy from the Team Project dimension.
 - Work Item.Iteration Hierarchy from the Work Item dimension.
 - Work Item.Area Hierarchy from the Work Item dimension.
 - Work Item.Work Item Type from the Work Item dimension. Select the Bug check box as the only type to display.
 - Year Week Date from the Date dimension.

4. In the Date dimension, expand More fields, and drag the Date, Week, or Month fields to the Axis Fields (Categories) area to specify how granular a report you want to generate.
5. Drag Work Item Count from the Work Item measure group to the Values area.
6. Drag State from the Work Item dimension to the Column Labels area, and then select the Active and Resolved check boxes.
7. In the Work Item dimension, expand More Fields, drag Previous State to the Column Labels area, and then select the Active and Resolved check boxes.
8. In the PivotTable report, collapse the Active column to show only a single column, and then rename the cell as Reactivated and Still Active.
9. In the PivotTable report, collapse the Resolved column to show only a single column.
10. (Optional) Right-click any filter that appears on the chart, and then choose Hide All Field Buttons on Chart.

Work item measures

The following table describes the measures that you can use to filter or report on the status or progress of work. With the Work Item Count measure, you can report on the total number of work items in a particular state at a particular point in time. To report activity that occurred on a particular day, you can use the State Change Count or Revision Count measures. The Work Item measure group contains additional measures when custom fields in the definitions of work item types specify Measure as the reportable attribute. For more information about how to use the reportable attribute, which is optional, and its values, see Add or modify work item fields to support reporting.

Note

The process templates from Microsoft Solutions Framework (MSF) v5.0 include calculated measures that are associated with the scheduling work fields. When a measure in the cube is based on a field in a process template, the label for the measure is based on the reference name of the originating field. However, a localized translation appears for the measure labels when you browse the cube with Microsoft Excel or other reporting tools.

Measure	Description
Business Value	A subjective unit of measure that captures the relative business value of a product backlog item. The reference name of this measure is Microsoft.VSTS.Common.BusinessValue.
Completed Work	The number of hours of work that were completed for work items that meet the criteria in the query or report. The reference name of this measure is Microsoft.VSTS.Scheduling.CompletedWork.

Effort

The aggregate of the units of effort for product backlog items that meet the criteria in the query or report. Effort is a subjective unit of measure that captures the level of effort to implement a product backlog item. A larger number indicates more work. This field is defined only in the Visual Studio Scrum process templates. The reference name of this measure is Microsoft.VSTS.SchedulingEffort.

Original Work

The number of hours of work from the baseline plan for work items that meet the criteria in the query or report. The reference name of this measure is Microsoft.VSTS.Scheduling.OriginalWork.

Remaining Work

The number of hours that are recorded as estimates of the work remaining to complete work items that meet the criteria in the query or report. The reference name of this measure is Microsoft.VSTS.Scheduling.RemainingWork.

Revision Count

The number of times that work items have been revised. You can use this measure to view detailed history about a set of work items or a particular work item. For example, you can use this measure to display the number of times that each member of the team has modified a work item during a span of time that you specify. To display this information, you create a query in which you slice the Revision Count measure by the Changed By dimension and also filter the measure by a date range.

State
Change
Count

The number of times that the team changed the states of a filtered set of work items. You can use this measure to count bug activations in a particular product area during a span of time that you specify. Note that this measure returns the number of state transition events but not the number of work items in state transition. For example, this measure would return the resolved rate of 2 if the team resolved, reactivated, and then re-resolved the same bug.

Size

A subjective unit of measure that captures the size of a requirement. This field is defined only in the MSF process templates for CMMI. The reference name of this measure is Microsoft.VSTS.Scheduling.Size.

Story
Points

A subjective unit of measure that captures the size of a user story. If you assign more points to a user story, you indicate that it requires more work to implement. This field is defined only in the MSF process templates for Agile software development. The reference name of this measure is Microsoft.VSTS.Scheduling.StoryPoints.

Work Item
Count

The count of work items that is based on the dimensions or filters that you specify. If your filter includes a date dimension, the measure returns historical information. Otherwise, the measure returns current information.

Hidden measures

To build the calculations that provide point-in-time totals, several hidden measures are used. These measures are not exposed to client tools such as Microsoft Excel, Report Builder, or Report Designer, but the measures are present in definitions in the deployed cube. Hidden measures perform a calculation by using the Multidimensional Expressions (MDX) LastChild function, which aggregates the total for the measure as of a particular date.

Measure	Description
LastChild Record Count	A hidden measure that is used to calculate the Work Item Count measure.
LastChild Microsoft_VSTS_Scheduling_RemainingWork	A hidden measure that is used to calculate the Remaining Work measure.
LastChild Microsoft_VSTS_Scheduling_CompletedWork	A hidden measure that is used to calculate the Completed Work

LastChild
Microsoft_VSTS_Scheduling_BaselineWork

measure.

A hidden
measure
that is used
to calculate
the Baseline
Work
measure.

Date Filters and Numeric Filters

You can use Date Filters and Numeric Filters to filter a report based on one or more values in this set. These filters appear under the measure group that is labeled Values. You can use the selections within each group to filter the set of work items. You should not add them to the PivotTable report directly. Instead, you filter the rows or columns within the PivotTable report by performing the following steps:

1. Choose the filter icon for either Row Labels or Column Labels within the PivotTable report, point to Value Filters, and then specify the criteria that you want (for example, Greater Than).
2. In the first field, choose the date or numeric filter that you want (for example, Created Date).
3. In the third field, type the value that meets your filter criteria, and then choose OK.

For example, to display all work items that were created after June 1, 2010, type 6/1/2010.

The following table describes the date filters that you can use to refine the set of work items that a report returns. You can even combine filters.

Field name	Description
Activated Date	The date and time when the work item was activated or reactivated. This filter corresponds to the <code>Microsoft.VSTS.Common.ActivatedDate</code> field.

Closed Date	The date and time when a work item was closed. This filter corresponds to the Microsoft.VSTS.Common.ClosedDate field.
Created Date	The date and time when a work item was created. This filter corresponds to the Microsoft.VSTS.Common.CreatedDate field.
Due Date	The date and time by which the team forecasted that a task or an issue will be completed. This field applies only to task and issue work items. This filter corresponds to the Microsoft.VSTS.Scheduling.DueDate field.
Finish Date	The date and time when the schedule indicates that the task will be completed. This filter corresponds to the Microsoft.VSTS.Scheduling.FinishDate field.
Resolved Date	The date and time when a bug or other type of work item was resolved. This filter corresponds to the Microsoft.VSTS.Common.ResolvedDate field.
Start Date	The date and time when the schedule indicates that the task will start. This filter corresponds to the Microsoft.VSTS.Scheduling.StartDate field.

field.

The following table describes the numeric filters that you can use to distill the set of work items that appear in a report.

Measure	Description
ID	The unique ID of each work item. Work item IDs are unique across all team projects in a team project collection. In the data warehouse, you should expect duplicate IDs to exist across collections. This filter corresponds to the System.Id field.
Priority	A subjective rating of the bug, issue, task, or test case as it relates to the business, usually between the values of 1 and 3. This filter corresponds to the Microsoft.VSTS.Common.Priority field.
Rev	A number that is assigned to a revision of a work item. This filter corresponds to the System.Rev field.
Stack Rank	A subjective rating of the user story, task, issue, or bug compared to other work items of the same type. An item that is assigned a lower number should be fixed before an item that is assigned a higher number. This filter

corresponds to the
Microsoft.VSTS.Common.StackRank
field.

▣ Dimensions, Attributes, Filters, and Categories in the Work Item Perspective

You can combine attributes across several dimensions to filter the set of work items that appear in a report or to highlight specific details about a group of work items. These attributes correspond to all the fields for any type of work item that have the reportable attribute set to Dimension. You can filter based on field attributes by work item, test case, linked work item, or linked test case. You can also filter the set of work items based on the type or types of links between them.

When you choose Work Item in the Show fields related to combo box in Excel, you have access not only to the Test Case, Work Item, and Work Item Category dimensions but also the Date and Team Project shared dimensions. Choose Work Item when you want to analyze work items or test cases and filter by date, project collection, project, or work item category. For information about the Date and Team Project dimensions, see

[Shared dimensions in the Analysis Services Cube.](#)

▣ Tip

Always start your filter process by specifying the criteria for the set of project collections and the team project on which you want to report.

To create reports about the status of current work items, you choose Current Work Item in the combo box. This combination of dimensions does not contain the Date dimension. To analyze the historical data of work items in a trend report, choose Work Item in the combo box.

▣ Filter by Work Item Fields Under the Work Item Dimension

The Work Item and Test Case dimensions contain all attributes that are specific

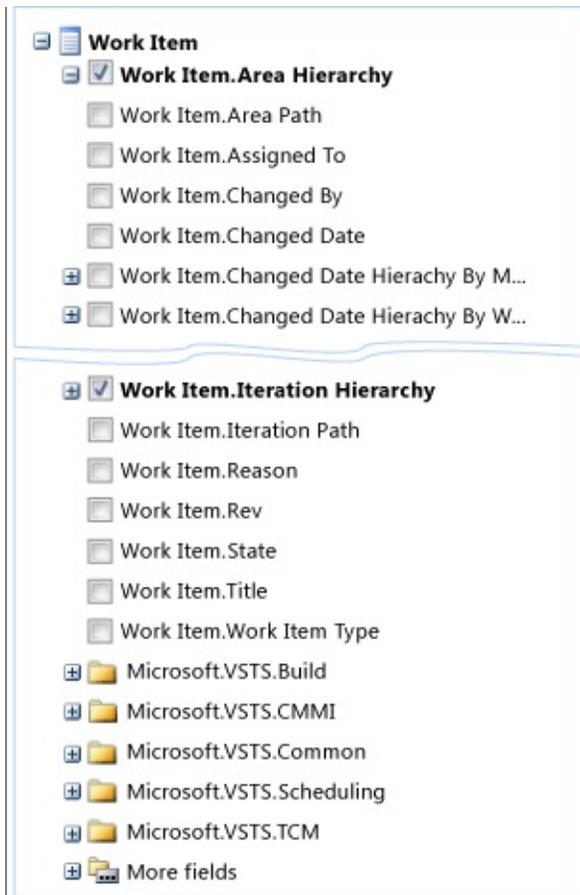
to work items, such as State, Work Item Type, and Work Item ID. Additionally, work item fields in process templates that have the reportable attribute set to "Dimension" are reflected as attributes in the Work Item dimension.

Attributes appear under display folders in the Test Case and Work Item dimensions. Dimension attributes are organized into folders that are based on the reporting reference name that is assigned in the definition of each type of work item. The following types of mapping occur:

- Intrinsic fields, which have the "System" prefix, appear directly under the Test Case or Work Item dimension.
- Other fields appear under folders whose names correspond to the prefixes in the reference names of the fields. For example, fields that have the "Microsoft.VSTS.Common" prefix appear under the folder that is labeled "Microsoft.VSTS.Common."

For example, work item fields appear under the Work Item dimension, as the following illustration shows. The Linked Work Item and Test Case dimensions have similar structures.

For
information
about each
field, see



[Work item field reference for Visual Studio ALM](#). Only fields that have reportable="Dimension" appear under the associated dimensions. Additional attributes appear when custom fields in the definitions of types of work items specify Dimension as the reportable attribute. For more information about how to use the reportable attribute, which is optional, and its values, see [Add or modify work item fields to support reporting](#).

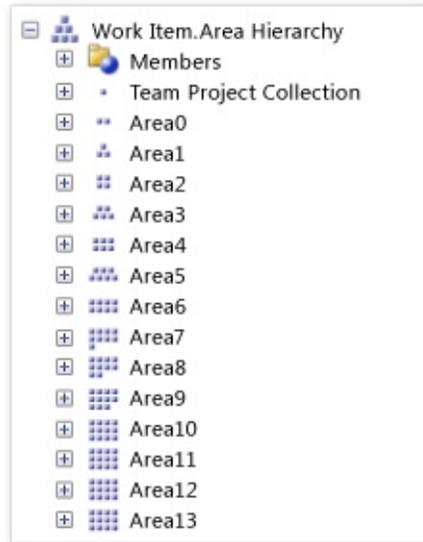
For information about how to work with date hierarchies, see [Shared dimensions in the Analysis Services Cube](#).

The following table describes the attributes that are not associated with a specific work item field and the hierarchical attributes that the Work Item dimension provides.

Attribute	Description
	Contains a 14-level hierarchy, as the

following illustration shows, which correlates the measures in the cube according to the area classifications with which they are associated. You can use this parent-child hierarchy to summarize or filter measures according to their level within the area hierarchy.

Area



For more information, see [Add and modify area and iteration paths](#).

Area Path

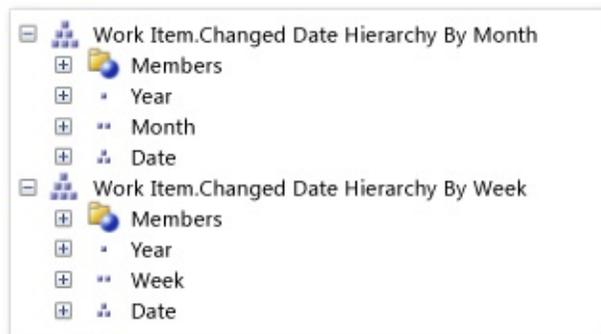
Flat list of the area paths for all team projects.

Changed Date

Flat list of dates that you can use to filter the set of work items based on the dates on which the team modified the items.

Hierarchical tree that supports filtering and reporting on the set of work items based on the months in which the team

modified the items. For example, you can use the Changed Date Hierarchy by Month or by Week, as the following illustration shows, to filter or create a trend report based on the changed date. For more information about how to work with date hierarchies, see [Shared dimensions in the Analysis Services Cube](#).



Hierarchical tree that supports filtering and reporting on the set of work items based on the week in which the team modified the items.

Flat list of dates that you can use to filter the set of work items based on the date on which the team created the items.

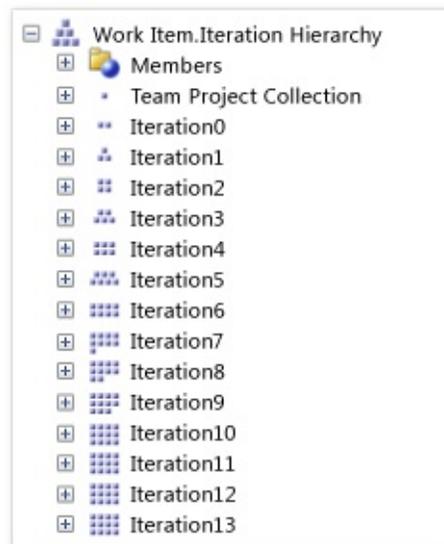
Hierarchical tree that supports filtering and reporting on the set of work items based on the month in which the team created the items.

Hierarchical tree that supports filtering and reporting on the set of work items

Hierarchy based on the week in which the team by Week created the items.

As the following illustration shows, contains a 14-level hierarchy that correlates the measures in the cube according to the iteration classifications with which they are associated. You can use this parent-child hierarchy to summarize or filter measures according to their levels within the iteration hierarchy.

Iteration



For more information, see Add and modify area and iteration paths.

Iteration Path Flat list of the set of iteration paths that are defined for all team projects.

The only field under More fields is Previous State, which you can use to filter a report based on the State to

Previous State which a work item was assigned before it was assigned to its current state. State is an attribute of the workflow for a type of work item. For more information, see [Change the workflow for a work item type](#).

Filter by Link Type or Fields That Are Defined for a Linked Work Item

When you choose the Linked Current Work Item entry in the Show fields related to combo box, you gain access to the Work Item, Work Item Link Type, and Work Item Linked dimensions. Choose this entry when you want to filter the set of work items in a report based on the link type or values of fields in work items that are linked to other work items.

Filter Based on an Attribute of a Link Type

By specifying one or more of the attributes in the following table, you can filter work items that have at least one link relationship whose link type meets the criteria that you specify. For more information, see

[Link type element reference](#) and [Manage link types \[witadmin\]](#).

Note

You can use any of the attributes in the following table to filter the report, but you will find the Link Name and Link Reference Name attributes most useful. All Boolean attributes have a value of either True or False.

Attribute	Description
Is Deleted	A Boolean value that specifies whether a link type is deleted.

Is Deny
Delete

A Boolean value that specifies whether the link type can be deleted. For example, you can delete custom link types but not built-in link types.

Is Deny
Edit

A Boolean value that specifies whether the link type can be modified. For example, you can modify custom link types but not built-in link types.

Is
Directional

A Boolean value that specifies whether the link type is directional. Directional links are used to form dependent relationships and prohibit circular relationships.

Is
Disabled

A Boolean value that specifies whether the link type has been disabled. You can use this attribute to find or filter work items that contain link relationships whose link type has been deactivated or disabled.

Is Non-
Circular

A Boolean value that specifies whether the link type prohibits circular link relationships.

Is Single Target A Boolean value that specifies whether the link type allows users to specify only one target for each link of this type. For example, a child work item can have only one parent, as defined by a Parent link type.

Is Tree A Boolean value that specifies whether the link type is based on the Tree topology.

Link ID The integer that the system assigns to built-in and custom link types.

Link Name The friendly name of the link type. The friendly name corresponds to either the forward or reverse name that is assigned to that link type.

Reference Name The name that is assigned to the link type. For example, System.LinkTypes.Related is the reference name of the Related link type.

Rules

The system assigns values to the rules that the link type uses. Do not use this attribute.

Filter Based on an Attribute of the Linked Work Item

The Work Item Linked dimension contains the same set of attributes as the Test Case and Work Item dimensions. You can use these attributes to find work items or filter measures based on attributes of work items that are linked to the work items that you are analyzing. By running this type of query, you can answer questions such as "How many active bugs are linked to priority 1 user stories?"

Note

You can use the Work Item Link Type and Work Item Linked dimensions to filter reports about current work items only. You cannot use those dimensions to analyze historical data.

The Work Item Link Type and Work Item Linked dimensions are many-to-many dimensions that are linked to the Current Work Item fact table. Team members can link any work item to more than one other work item.

If a dimension corresponds to the attributes of links between work items, the name of the dimension is prefixed with the phrase "Work Item Linked." For example, the "Assigned To" attribute corresponds to the "Work Item Linked.Assigned To" attribute and so on for all other work item fields in the OLAP cube. For more information, see

[Filter by Work Item Fields Under the Work Item Dimension](#) earlier in this topic.

Filter By Changesets

You choose the Work Item Changeset entry in the combo box to access the Version Control Changeset and Work Item dimensions. Choose this entry when

you want to filter the set of work items by information in the changesets that are linked to the work items. When you make and check in changes to the code by using Team Foundation version control, you can associate the changeset with the work item that the changes address. A changeset lists the source files that you checked in for the changeset and a list of all work items that you linked to the changeset. For more information, see [Check in your work to the team's codebase](#).

The Version Control Changeset dimension is a parent-child dimension type and linked to the Current Work Item fact table through an intermediate, many-to-many fact table. Team members can link any work item to more than one changeset. The following table describes the attributes in the Version Control Changeset dimension.

Attribute	Description
Changeset ID	The number that is assigned to the changeset.
Checked In By	The user name of the team member who checked in the changeset.
Description	The check-in comment that is associated with the changeset.
Policy Override Comment	The comment that is provided when a policy is overridden. If a policy was not overridden with a changeset, the field is null.

Filter By category

The Work Item to Category entry in the combo box contains the Work Item and Work Item Category dimensions. Choose this entry when you want to filter the set of work items by their associated categories. You use categories when your team projects contain types of work item that are similar but named differently. For more information, see Use categories to group work item types.

The following table describes the attributes in the Work Item Category dimension.

Attribute	Description
Name	The friendly name that is assigned to the category of a work item.
Reference Name	The reference name that is assigned to the category of a work item.

Filter By Hierarchical Relationships

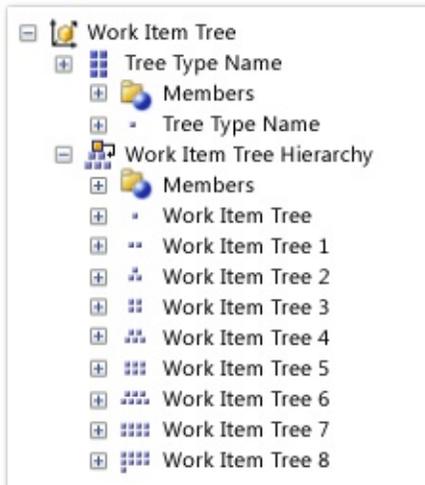
The Work Item to Tree entry in the combo box contains the Work Item and Work Item Tree dimensions. Choose this entry when you want to filter by hierarchical nesting of work items that are linked by using the Parent and Child types of links or other custom-defined types of tree-topology links.

Note

You can use the Work Item Tree dimension to filter reports about current work items only. You cannot use this dimension to analyze historical data about work items.

The Work Item Tree dimension is a parent-child dimension type and linked to the Current Work Item fact table through an intermediate, many-to-many fact table. Any work item can be linked to more than one work item and, therefore, can appear in multiple places in the tree.

As the following illustration shows, The Work Item Tree Hierarchy contains eight hierarchical filters. The first hierarchy supports filtering work items at the top of the tree, Work Item Tree 1 supports filtering work items that are nested one level deep, and so on.



The following table describes the attributes in the Work Item Tree dimension. You use link types that are based on the tree topology to create multi-level, hierarchical relationships among work items. In addition to supporting multi-level, hierarchical views, hierarchical link types support directionality and restrict circular relationships. For more information, see

[Link type element reference.](#)

Attribute	Description
Work Item Tree Hierarchy	Work items that are linked through hierarchical link types.
	The reference name of the

Tree
Type
Name

hierarchical link type (for example System.LinkTypes.Hierarchy). You can filter the report based on a hierarchical type of link that is used within a team project collection. To filter the report based on the forward or reverse name of a type of link, use the Link Name attribute that the Work Item Link Type dimension provides.

Filter By Test Result

The Work Item with Result entry in the combo box contains the Test Result, Version Control Changeset, Work Item, Work Item Category, and Work Item Tree dimensions. You can use the attributes in the Test Result dimension to filter reports based on the test results that are associated with test cases and their linked work items.

You can use the Work Item Count measure not only to determine how many work items are linked to a test result but also to report on work items and their associated test results. For a description of each Test Result attribute, see

[Analyze and report on test results using the test perspective in the Analysis Services database for Visual Studio ALM.](#)

Required Activities to Monitor and Track Work

To create reports that contain useful data about the status, progress, and trends about work items, team members must perform the following activities:

- Create work items. Team members must create work items to manage the backlog of user stories or bugs, to track work by using tasks, and to manage test cases. Update the state of work items based on the process guidance for each type of work item.

For more information, see Agile process template work item types and workflow, CMMI process template work item types and workflow, or Scrum process template work item types and workflow.

- Link work items to other work items or changesets. To track relationships between work items, team members must create links between, for example, user stories and tasks. Team members can create other useful links such as those between user stories and test cases, bugs and test cases, and changesets and user stories, tasks, and bugs. Team members can show relatedness, dependency, or hierarchy by specifying the appropriate type of link.

For more information, see Link work items to support traceability.

- Specify area and iteration paths. To monitor status or trends on product areas or milestones by filtering reports, team members must set the Area and Iteration fields for each work item.
- Assign work to owners and update the work item State. As work progresses, team members must change the State of the work items that are assigned to them as the work items move from a new or proposed state, to active or in progress, to closed or done
- Update the work effort. To monitor progress that a team has made in completing work for an iteration or other interval of time, team members

must update the amount of completed and remaining work for the work items that are assigned to them.

See Also

Concepts

[Perspectives and measure groups provided in the Analysis Services cube for Visual Studio](#)

Other Resources

[Table reference for the relational warehouse database for Visual Studio ALM](#)



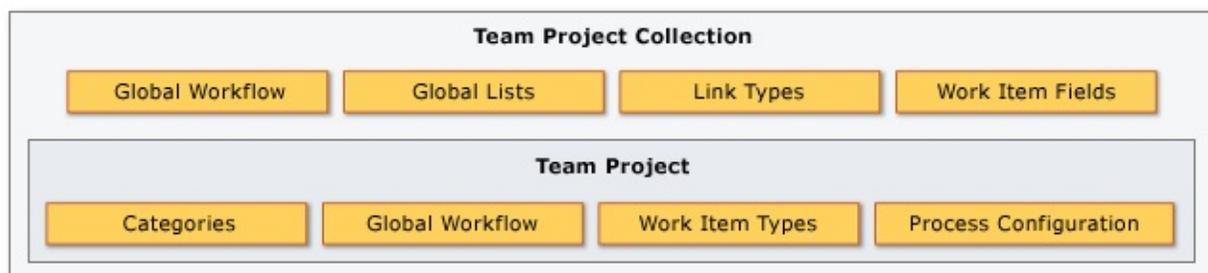
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

XML element reference

[Send Feedback](#)

You can customize your team project to support specific processes and practices that your team uses, and to design your workflow, work item forms, and data fields. Customization requires modifying one or more XML definition files. Each file corresponds to a work item tracking object. You can look up the syntax structure of each XML element from the topics provided in this section.

This illustration shows that you can create or customize eight types of objects. For team projects, you can customize categories, work item types, and process configuration. For team project collections, you can customize global lists, link types, and work item fields. You can customize global workflow for a team project or for a team project collection. For a description of these objects and other customization options, go [here](#).



Index to XML
Elements for work item
tracking

Look up the XML
syntax for an element
used in defining work
item types:

- All WITD elements
- FIELD (Definition)
- WORKFLOW elements
- All FORM elements

Customize an existing team project by modifying the syntax associated with one of the following objects or XML definition files:

- Process configuration
- Global lists
- Categories
- Link types

Index to XML elements for process template plug-ins



Before you create a team project, you can customize a process template to design your workflow, work item forms, and data fields to support your specific processes and practices.

- [Overview of process template files](#)
- Define the root tasks for the process template
- Customize functional areas
- Configure initial groups, teams, members, and permissions
- Define objects for tracking work items

Customize TFS to
Project field mapping

If you use Microsoft Project, you can customize how data is published and refreshed. When publishing or refreshing tasks in Microsoft Project, the project mapping file determines the publishing behavior and how the fields in each task are mapped to fields in Team Foundation.

Objects used to track work

Object	Description
	Use categories to group work item types
Category	A category defines a group of work item types that track similar items of work but are referred to by different names. You can group one or more work item types in the same team project into a category. You define categories to support running queries, generating reports, and setting default work item types in specific instances. You use the In Group operator to find work items that belong to a category. For more information, see Query fields, operators, values, and variables.
	Modify or add a field to support queries, reports, and workflow
	A field defines a type of data that is used to track work. You use work item fields to track data for a work item

type, to define the filter criteria for queries, and to generate reports. You must define each data element that is not built in, that the process template does not provide, and that you want to track, use to define the workflow, or appear on the form for a work item type. You define a data element using the **FIELD** element.

Field

Each field is defined by one or more attributes, which include what type of data it can contain, whether it is used in reporting, and whether it is indexed. You can also specify optional elements that restrict, auto-populate, or specify conditions for the values to which users can set the field by using a work item form.

You can add a field, remove it, or customize how you use it to track data.

Global
list

[Define global lists](#)

A global list defines a list of values, when is known as a pick list, that you can use across work item types to control the value or values to which users can set a

field in a work item. You use global lists to quickly update the contents of pick lists that are used for many types of work items.

You can define global lists within a type of work item type, but this practice is not recommended because the definition of the work item type will overwrite changes that are defined elsewhere if that definition is imported. A best practice is to define and import global lists through a definition file for global lists or global workflow.

Global workflow

[Customize global workflow](#)

A global workflow defines fields and global lists that are available to all types of work items for either a team project or a collection.

Link type

Define a custom link type

A link type defines the rules and restrictions that control the relationships that users can make between work items. In addition to the built-in types of links, you can create link types to support your project-tracking requirements. Before you start to create links between work items, you should analyze how you might use links to plan your project and track the status of work items.

Process configuration

Configure and customize Agile planning tools for a team project

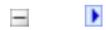
Process configuration elements control the layout and functions of the backlog and task board Agile tools provided with Team Web Access.

If your team project was created using a process template other than those provided by Visual Studio ALM or you have customized the type definitions for work items, you may need to customize the definition files that support process configuration. Also, if you have customized or added types of work items and want to use those fields, then you will want to determine how to modify the process configuration elements to match other objects that you have customized.

Work item type

Modify or add a custom work item type (WIT)

A type of work item defines an object, such as a bug, a requirement, or a risk, that is used to track work for a team project. A work item type defines the fields, workflow, and form for tracking work.



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Work item tracking: Index to XML element definitions

[See Also](#) [Send Feedback](#)

By using the index of elements in this topic, you can look up the syntax structure and find examples of how to use each element of the schema type definition for work item tracking. Elements are organized alphabetically and by the following four major groups:

- **WITD:** The root element of the definition schema for types of work items.
- **FIELD (Definition):** You use this element and its child elements to define a work item field and to specify the rules and conditions that apply to it. See [Modify or add a field to support queries, reports, and workflow](#). To access descriptions for all fields defined within the default process templates, including system fields, see

[Work item field reference for Visual Studio ALM.](#)

- **WORKFLOW:** You use this element and its child elements to define the states, transitions, and reasons that compose the workflow of a type of work item. In addition to the following elements, you can use most of the elements listed under [FIELD Definition Elements](#) later in this topic to reference a field and apply conditions to it when a user changes its state. For more information, see [Change the workflow for a work item type](#).

Note

Changes you make to the workflow can affect the operation of the Agile planning tools, the backlog and task board pages, and other tools. You might need to customize the process configuration. See [Configure and customize Agile planning tools for a team project](#).

- **FORM:** You use this element and its child elements to specify the fields that appear on the work item form and the layout or design of the form. For more information, see [Design the work item form](#).

Alphabetical Index

Note

The schema definition for work item tracking defines all child elements of the FORM element as camel-case and all other elements as all capitalized. If you encounter errors when validating your type definition files, check the case structure of your elements. Also, the case structure of opening and closing tags must match according to the rules for XML syntax.

Many of the XML tags used to define the process configuration for the backlog and task board pages in Team Web Access share the same names but have different syntax structure. For more information, see [Customizing Backlog and Task Board Pages](#) and

[Process configuration XML element reference.](#)

- ACTION
- ACTIONS
- ALLOWEDVALUES
- ALLOWEXISTINGVALUE
- CANNOTLOSEVALUE
- Column
- Content
- Control
- COPY
- FIELDS (Definition)
- FIELDS (Workflow)
- Filter
- FORM
- FROZEN
- GLOBALIST (Field Definition)
- GLOBALIST (Global Lists)
- GLOBALLISTS (Global Lists)
- Group
- LISTITEM (G Lists)
- LISTITEM (Pi
- MATCH
- NOTSAMEAS
- Param
- PROHIBITED
- READONLY
- REASON
- REASONS

- DEFAULT
- DEFAULTREASON
- DESCRIPTION
- EMPTY
- ExternalLinkFilters
- FIELD (Definition)
- FIELD (Workflow)
- LabelText
- Layout
- Link (Label Text)
- Link (Hyperlink)
- Link (Web page)
- LinkColumn
- LinkColumns
- LinksControlOptions
- REQUIRED
- SERVERDEF/
- Splitter
- STATE
- STATES
- SUGGESTED
- Tab
- TabGroup

WITD, FIELD, WORKFLOW, and FORM Index of Elements

The following table provides an index to the reference topics for all child elements of the parent container element. Also, you can access the topic that provides detailed information and examples for each child element.

WITD FIELD WORKFLOW FORM

[All WITD XML elements reference](#)

[All FIELD XML elements reference](#)

[All WORKFLOW XML elements reference](#)

[All FORM XML elements reference](#)

- DESCRIPTION
- FIELDS (Definition)
- FORM
- GLOBALISTS
- WITD
- WORKFLOW
- WORKITEMTYPE
- ALLOWEDVALUES
- ALLOWEXISTINGVALUE

- CANNOTLOSEVALUE
- COPY
- DEFAULT
- EMPTY
- FIELD (Definition)
- FROZEN
- HELPTTEXT
- GLOBALISTS
- LISTITEM (Global Lists)
- LISTITEM (Pick Lists)
- MATCH
- NOTSAMEAS
- PROHIBITEDVALUES
- READONLY
- REQUIRED
- SERVERDEFAULT
- SUGGESTEDVALUES
- VALIDUSER
- WHEN
- WHENNOT
- WHENCHANGED

- WHENNOTCHANGED
- ACTION
- ACTIONS
- DEFAULTREASON
- FIELD (Workflow)
- FIELDS (Workflow)
- REASON
- REASONS
- STATE
- STATES
- TRANSITION
- TRANSITIONS
- WORKFLOW
- Column
- Content
- Control
- ExternalLinkFilters
- Filter
- FORM
- Group
- LabelText

- Layout
- Link (Label Text)
- Link (Hyperlink)
- Link (Web page)
- LinkColumn
- LinkColumns
- LinksControlOptions
- Param
- Splitter
- Tab
- TabGroup
- Text
- WebpageControlOptions
- WorkItemLinkFilters
- WorkItemTypeFilters

See Also

Concepts

[Localization and globalization of WITD child elements](#)

[All WITD XML elements reference](#)

[Customize work tracking objects to support your team's processes](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

All WITD XML elements reference

[See Also](#) [Send Feedback](#)

You can customize an existing work item type (WIT) or create a WIT to meet your project tracking requirements. A WIT defines the rules, fields, states, and transitions for an item of work that will be tracked for a team project, such as a bug, requirement, or risk.

A WIT cannot be empty. The root element in each definition of a WIT is the **WITD** element, which must have only one **WORKITEMTYPE** element defined. The name of each WIT must be unique in a team project, and each type name must be no more than 254 Unicode characters long. For more information, see [Naming conventions for work item tracking objects](#).

To customize or create a WIT definition, you modify the type definition XML file. You maintain and manage definitions of types of work items for each team project.

WITD syntax structure

The following example shows the high-level structure of a WIT definition.

 **Copy Code**

```
<WITD application="work item type editor" version="1.0">
  <WORKITEMTYPE name="bug">
    <DESCRIPTION>Bug work item types are used to track defects :
    <GLOBALLISTS> . . . </GLOBALLISTS>
    <FIELDS> . . . </FIELDS>
    <WORKFLOW> . . . </WORKFLOW>
    <FORM> . . . </FORM>
  </WORKITEMTYPE>
</WITD>
```

A small number of XML elements appear between WORKITEMTYPE tags at the highest levels of the WIT definition. You start defining a WIT by providing a name and a description. You also use certain standard XML tags to wrap the WIT definition.

WITD child elements

The structural elements used in the previous example are described in the following table:

Element	Description
WITD	<p>The complete WIT definition is wrapped by the tag WITD. It takes any name for the application name. The version identifies the work item type that changes from one release to the next.</p> <p> Copy Code</p> <pre><WITD application="Work item type editor" version=" <WORKITEMTYPE> . . .</WORKITEMTYPE> </WITD></pre> <p>Names of WITs must be unique in a specific team project. As you use the name specified by this element. For example, they appear as a menu option. In this case, a user could choose Bi New Work Item menu.</p> <p> Copy Code</p>
WORKITEMTYPE	<pre><WORKITEMTYPE name="workItemTypeName"> <DESCRIPTION>Text</DESCRIPTION> <GLOBALLISTS> . . .</GLOBALLISTS> <FIELDS> . . . </FIELDS> <WORKFLOW> . . . </WORKFLOW> <FORM> . . .</FORM> </WORKITEMTYPE></pre> <p>Specifies a string that describes the type of work item that you are defining. The description should help any user who is customizing the WIT.</p>

DESCRIPTION

Note

You can view the description only in the XML definition. You can view the description anywhere in the user interface, and it has a relationship to the field **System.Definition**.

Copy Code

```
<DESCRIPTION> DescriptionOfWorkItemType</DESCRIPTION>
```

You specify a string of text that describes the type of work items you are defining.

Contains the global list definitions that are used by the work item type. You use global lists to share list items among multiple work items for a team project collection. Using global lists provides support for maintenance and cross-group collaboration.

GLOBALLISTS

Copy Code

```
<GLOBALLIST name="globalListName">  
  <LISTITEM> . . . </LISTITEM>  
</GLOBALLIST>
```

For more information, see

[Define global lists.](#)

FIELDS

Contains the set of field definitions that are used by the WIT. Within the FIELDS element, you define all the fields that you want to use to track data for the type of work item. This includes fields that you will use to run queries and generate reports.

Copy Code

```
<FIELDS>
```

```
<FIELD> . . . </FIELD>
</FIELDS>
```

For more information, see [Define and modify work item fields](#).

WORKFLOW

Contains the set of STATE and TRANSITION elements that define the workflow. The workflow is a set of valid transitions from one state to another and the specific conditions associated with each transition.

 **Copy Code**

```
<WORKFLOW>
  <STATES> . . . </STATES>
  <TRANSITIONS> . . . </TRANSITIONS>
</WORKFLOW>
```

For more information, see [Change the workflow for a work item type](#).

FORM

Contains LAYOUT, CONTROL, GROUP, TAB, TABGROUP, SPLITTER, and other elements that define which fields and controls appear on the form and the display of these elements on the form.

 **Copy Code**

```
<FORM>
  <Layout> . . . </Layout>
</FORM>
```

For more information, see [Design the work item form](#).

■ Process template work item types

TFS provides three default process templates. These files are located in the following directory:

```
%programfiles%/Microsoft Team Foundation Server  
12.0/Tools/Deploy/ProcessTemplateManagerFiles/1033
```

You can find the work item type definitions in the WorkItem Tracking\TypeDefinitions folder. For more information, see [Work with team project artifacts](#), choose a process template.

You can customize or create type definition files in the TypeDefinitions folder. Then you can import your new work item type to a single project, or you can add it to your process template and upload the modified template to your Team Foundation Server. For more information, see

[Add type definitions for work items to a process template.](#)

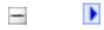
See Also

Concepts

Customize work tracking objects to support your team's processes

Other Resources

[witAdmin: Customize and manage objects for tracking work](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Localization and globalization of WITD child elements

[See Also](#) [Send Feedback](#)

You can change some parts of the work item type definition so that they appear in your native language.

When you design a work item, keep in mind that work item settings reside on the server for the team project collection and do not recognize the language preferences of team members.

The following table summarizes what elements can and cannot be localized. Any element that is not explicitly listed in the following table cannot be localized.

Examples of elements that can be localized	Element attributes that you can localize
<code><WORKITEMTYPE name="bug"></code>	The name of the work item type. Must be 1 to 255 Unicode characters.
<code><DESCRIPTION>Work item descriptive text here</DESCRIPTION></code>	The work item type description. Supports entry of a text string that can contain more than 255 Unicode

characters.

The name of the work item field. Must be 1 to 128 Unicode characters long.

```
<FIELD  
refname="System.Title"  
name="Title"  
type="String">
```

The name that you specify for a field is used in all Team Foundation clients. Field names are also used by team members to create work item type queries.

The field reference name and type are not localizable.

```
<HELPTEXT>This is a  
work item for  
bugs</HELPTEXT>
```

The Help text for work item fields. Supports entry of a text string that can contain more than 255 Unicode characters.

You can define custom text strings for each work item type within each team project.

`<LISTITEM
value="My Value">` The string text for
the value attribute
that appears in a
pick list or global
list.

`<STATE
value="Active" />` The value attribute
that specifies the
name for the

`<STATE
value="Complete" />` STATE and
REASON
elements. Must be
1 to 255 Unicode
characters.

`<TRANSITION
from="Active"
to="Complete">`

`<REASON value="No
Plans to Fix"/>`

`<GROUP Label="label
text" Padding="(left,
top, right, bottom)"
Margin="(left, top,
right, bottom)">` The Label attribute
specified by a
GROUP, TAB, or
CONTROL
element. Label text
appears on work
item forms. Must
be 1 to 80 Unicode
characters when
specified.

`<Tab Label="label text"
Padding="(left, top,
right, bottom)"
Margin="(left, top,
right, bottom)">` The Label attribute
is optional except
for the TAB
element.

`<CONTROL
FieldName="Found In` You can customize
the name of a field

```
Build" Label="Found  
In"  
LabelPosition="Top">
```

by specifying the Label for a field that appears on a work item form by using the CONTROL element.

See Also

Concepts

[Define and modify work item fields](#)

Customize work tracking objects to support your team's processes



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

All FIELD XML elements reference

[See Also](#) [Send Feedback](#)

Use this topic to look up the syntax of the FIELD element or one of its child elements.

You specify these elements in the **FIELD** (Definition) element container. You add a field for a work item type (WIT) by specifying a **FIELD** (Definition) element within the **FIELDS** (Definition) element. You can specify these elements within the definition of a WIT or as part of a global workflow.

You can add child elements to specify the behavior of a field, define default values, or define a pick list of values. You can use field rule elements in combination with each other. You can scope most rules to apply to one or more users or groups or to be ignored for one or more users or groups.

FIELD (Definition) container element

You use the following syntax to define the data fields for a type of work item. This example shows the format of the **FIELD** (Definition) element and all optional child elements. For more information, see

[FIELD \(Definition\) element reference.](#)

```
<FIELD name="fieldDisplayName" refname="fieldReferenceName" type="St
syncnamechanges="true | false" reportingname="reportingDisplayName"
reportable="Dimension | Detail | Measure" formula="avg" >
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>
  <ALLOWEXISTINGVALUE />
  <CANNOTLOSEVALUE />
  <COPY />
  <DEFAULT />
  <EMPTY />
  <FROZEN />
  <HELPTEXT> . . . </HELPTEXT>
  <MATCH />
  <NOTSAMEAS />
  <PROHIBITEDVALUES /> . . . </PROHIBITEDVALUES>
  <READONLY />
  <REQUIRED />
  <SERVERDEFAULT />
  <SUGGESTEDVALUES /> . . . </SUGGESTEDVALUES>
  <VALIDUSER />
  <WHEN>> . . . </WHEN>
  <WHENNOT> . . . </WHENNOT>
  <WHENCHANGED> . . . </WHENCHANGED>
  <WHENNOTCHANGED> . . . </WHENNOTCHANGED>
</FIELD>
```

FIELD child elements

Use child elements to set various restrictions on what data can be entered into a field. You can specify values for a pick list (drop-down menu), set default values, clear entries, or restrict changes. The following table provides the syntax structure for each child element.

To learn how to use these elements, see

[Apply a rule to a work item field](#). Restrictions exist on applying most rules to system fields. All child elements are optional.

Element	Description and Syntax
ALLOWEDVALUES	<p>Defines a list of values that users can specify in a editor. Users must specify one of the values that y</p> <p></p> <pre><ALLOWEDVALUES for="userGroupName" not="us expanditems="true false" filteritems="e <GLOBALLIST name="globalListName"> <LISTITEM value="Name" /> . . . </GLOBALLIST> </ALLOWEDVALUES></pre>
ALLOWEXISTINGVALUE	<p>Specifies that a field can retain an existing value, (field values must be in the list.</p> <p></p> <pre><ALLOWEXISTINGVALUE /></pre>

For more information, see [Define pick lists](#).

Specifies that users cannot clear a field of all values. If a field contains a value, that field must always contain a value.

CANNOTLOSEVALUE



```
<CANNOTLOSEVALUE for="userGroupName" not="userGroupNa
```

For more information, see [Apply a field rule](#).

Copies a specified value to a field when a user creates a new record.

COPY



```
<COPY for="userGroupName" not="userGroupName"
from="value | field | clock | currentuser"
value="valueToCopy" field="fieldReference">
```

For more information, see [Define a default value](#).

Specifies a value for a field that is empty when a user creates a new record. If the user already has a value, the default rule is ignored.

DEFAULT



```
<DEFAULT for="userGroupName" not="userGroupNa
from="value | field | clock | currentuser"
value="value to copy" field="field referer
```

For more information, see [Define a default value](#).

Clears the field of any value that it contains. The **IF** rule you should not use it with the **READONLY** rule.

EMPTY

The field value is cleared when a user saves the work item. This rule is primarily used during state transition when the item is transitioning.



```
<EMPTY for="userGroupName" not="userGroup"
```

For more information, see [Apply a field rule](#).

Specifies that you cannot change the field to a non-empty value soon as a user saves the work item with a value in the field modified.

FROZEN



```
<FROZEN for="userGroupName" not="userGroup"
```

For more information, see [Apply a field rule](#).

Defines the text to appear when a user points to the field. tooltipText: A string of text that contains between 1 and 255 characters.

HELPTEXT



```
<HELPTEXT>tooltipText </HELPTEXT>
```

For more information, see [Apply a field rule](#).

Defines a pattern that values of String type fields must match.

MATCH



```
<MATCH pattern="patternValue" for="userGroup"
```

For more information, see [Apply pattern matching](#)

Specifies that a field is not assigned the same value as the field to which it is assigned. The value of the field attribute must be a

NOTSAMEAS



```
<NOTSAMEAS field="fieldReferenceName" for="fieldReferenceName" />
```

For more information, see [Apply a field rule](#).

Defines a list of values that a field cannot contain. The field cannot contain a value that is in the list. The field cannot contain a prohibited value.

PROHIBITEDVALUES



```
<PROHIBITEDVALUES for="userGroupName" not="userGroupName"
  expanditems="true | false" filteritems="expanditems" />
  <GLOBALLIST name="globalListName">
    <LISTITEM value="Name" />
    . . .
  </GLOBALLIST>
</PROHIBITEDVALUES>
```

For more information, see [Define pick lists](#).

Specifies that you cannot modify the value to which the field is assigned.

READONLY



```
<READONLY for="userGroupName" not="userGroupName" />
```

Note

Do not use this element together with the **EMPTY** element.

field read-only. If you combine these elements, re

For more information, see [Apply a field rule.](#)

Specifies that users must specify a value for the fi
cannot save a work item until they have assigned v



REQUIRED

```
<REQUIRED for="userGroupName" not="userGro
```

For more information, see [Apply a field rule.](#)

Copies a specified server value to a field when a u
appear as read-only on the form.



SERVERDEFAULT

```
<SERVERDEFAULT for="userGroupName" not="us  
from="clock | currentuser" />
```

For more information, see [Define a default value c](#)

Defines a suggested list of values that users can sp
the query editor. Users can specify values other th



SUGGESTEDVALUES

```
<SUGGESTEDVALUES for="userGroupName" not='  
expanditems="true | false" filteritems="e)  
  <GLOBALLIST name="globalListName">  
    <LISTITEM value="Name" />  
  . . .  
  </GLOBALLIST>  
</SUGGESTEDVALUES>
```

For more information, see [Define pick lists](#).

Restricts work items from being modified by user:
The default group is the Team Foundation Valid U

All attributes are optional. All attributes must con:
and 255 characters. You can use tokens to specify

VALIDUSER



```
< VALIDUSER group="groupName" for="userNar
```

For more information, see [Apply a field rule](#).

Specifies one or more rules to apply to the current
The parent **FIELD** element defines the current fie



```
<WHEN field="fieldReferenceName" value="va  
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>  
  <ALLOWEXISTINGVALUE> . . . <ALLOWEXIS  
  <CANNOTLOSEVALUE> . . . </CANNOTLOSEVA  
  <COPY> . . . </COPY>  
  <DEFAULT> . . . </DEFAULT>  
  <EMPTY> . . . </EMPTY>  
  <FROZEN> . . . </FROZEN>  
  <MATCH> . . . </MATCH>  
  <NOTSAMEAS> . . . </NOTSAMEAS>  
  <PROHIBITEDVALUES> . . . </PROHIBITEDV  
  <READONLY> . . . </READONLY>  
  <REQUIRED> . . . </REQUIRED>  
  <SERVERDEFAULT> . . . </SERVERDEFAULT>  
  <SUGGESTEDVALUES> . . . </SUGGESTEDVAI  
  <VALIDUSER> . . . </VALIDUSER>  
</WHEN>
```

WHEN

For more information, see [Assign conditional-bas](#)

Specifies one or more rules to apply to the current specific value. The parent **FIELD** element defines



WHENNOT

```
<WHENNOT field="fieldReferenceName" value=  
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>  
  <ALLOWEXISTINGVALUE> . . . <ALLOWEXISTINGVALUE>  
  <CANNOTLOSEVALUE> . . . </CANNOTLOSEVALUE>  
  <COPY> . . . </COPY>  
  <DEFAULT> . . . </DEFAULT>  
  <EMPTY> . . . </EMPTY>  
  <FROZEN> . . . </FROZEN>  
  <MATCH> . . . </MATCH>  
  <NOTSAMEAS> . . . </NOTSAMEAS>  
  <PROHIBITEDVALUES> . . . </PROHIBITEDVALUES>  
  <READONLY> . . . </READONLY>  
  <REQUIRED> . . . </REQUIRED>  
  <SERVERDEFAULT> . . . </SERVERDEFAULT>  
  <SUGGESTEDVALUES> . . . </SUGGESTEDVALUES>  
  <VALIDUSER> . . . </VALIDUSER>  
</WHENNOT>
```

For more information, see [Assign conditional-based](#)

Specifies one or more rules to apply to the current revision of the work item. The parent **FIELD** element



WHENCHANGED

```
<WHENCHANGED field="fieldReferenceName" >  
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>  
  <ALLOWEXISTINGVALUE> . . . <ALLOWEXISTINGVALUE>  
  <CANNOTLOSEVALUE> . . . </CANNOTLOSEVALUE>  
  <COPY> . . . </COPY>  
  <DEFAULT> . . . </DEFAULT>  
  <EMPTY> . . . </EMPTY>  
  <FROZEN> . . . </FROZEN>  
  <MATCH> . . . </MATCH>  
  <NOTSAMEAS> . . . </NOTSAMEAS>  
  <PROHIBITEDVALUES> . . . </PROHIBITEDVALUES>  
  <READONLY> . . . </READONLY>  
  <REQUIRED> . . . </REQUIRED>  
  <SERVERDEFAULT> . . . </SERVERDEFAULT>
```

```
<SUGGESTEDVALUES> . . . </SUGGESTEDVAI
<VALIDUSER> . . . </VALIDUSER>
</WHENCHANGED>
```

For more information, see [Assign conditional-bas](#)

Specifies one or more rules to apply to the current during the revision of the work item. The parent e



WHENNOTCHANGED

```
<WHENNOTCHANGED field="fieldReferenceName'
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES:
  <ALLOWEXISTINGVALUE> . . . <ALLOWEXIS
  <CANNOTLOSEVALUE> . . . </CANNOTLOSEV/
  <COPY> . . . </COPY>
  <DEFAULT> . . . </DEFAULT>
  <EMPTY> . . . </EMPTY>
  <FROZEN> . . . </FROZEN>
  <MATCH> . . . </MATCH>
  <NOTSAMEAS> . . . </NOTSAMEAS>
  <PROHIBITEDVALUES> . . . </PROHIBITED
  <READONLY> . . . </READONLY>
  <REQUIRED> . . . </REQUIRED>
  <SERVERDEFAULT> . . . </SERVERDEFAULT>
  <SUGGESTEDVALUES> . . . </SUGGESTEDVAI
  <VALIDUSER> . . . </VALIDUSER>
</WHENNOTCHANGED>
```

For more information, see [Assign conditional-bas](#)

GLOBALLIST and LISTITEM child elements

You specify the **GLOBALLIST** and **LISTITEM** elements as child elements of the **ALLOWEDVALUES**, **SUGGESTEDVALUES**, and **PROHIBITEDVALUES** elements. You can use these elements to enumerate a list of values that appears. Users select values from a pick list or a drop-down menu. For more information, see

[Define global lists.](#)

Element	Description
	<p>Defines a set of LISTITEM elements that is stored for a team project collection and that all team projects in a collection can use.</p> <p> Copy Code</p> <pre><GLOBALLIST name="globalListName"> <LISTITEM> . . . </LISTITEM> </GLOBALLIST></pre>
GLOBALIST	<p>globalListName: A string of text that contains between 1 and 255 characters.</p> <p>GLOBALLIST is a required child element of the GLOBALLISTS element and an optional child element of the ALLOWEDVALUES, SUGGESTEDVALUES, and PROHIBITEDVALUES elements. You can define a global list within a work item definition, a global list definition, or</p>

a global workflow.

Defines a valid list value.

 **Copy Code**

```
<LISTITEM value="listName" />
```

LISTITEM

LISTITEM is a required child element of **GLOBALLIST** and an optional child element of the **ALLOWEDVALUES**, **SUGGESTEDVALUES**, and **PROHIBITEDVALUES** elements.

Attributes specified by FIELD child elements

You can qualify most **FIELD** rules to apply or not apply to a set of groups or users by including the **for** or **not** attributes. For more information, see [Apply a field rule](#).

Attribute	Syntax	Description
expanditems	expanditems="true false"	Optional. Specifies whether a group that the LISTITEM element identifies should be expanded to include subordinate groups in the list. The default value of this attribute is true.
filteritems	filteritems="excludegroups"	Optional. Specifies that only the members of groups, and not group names, are included in the list. The only allowed value for this attribute is <code>excludegroups</code> .
for	for="userGroupName"	Optional. Specifies the name of a user or group in Team Foundation to whom the rule applies. Valid values consist of a string of text that contains

not

not="userGroupName"

between 1 and 255 characters.

Optional. Specifies the name of a user or group in Team Foundation to whom the rule does not apply. Valid values consist of a string of text that contains between 1 and 255 characters.

Required. Specifies the source of the value from which to copy a value or specify a default value. The following values are valid:

- clock: Copies the current date and time from the system clock to DateTime fields. No additional attributes are required. For **COPY** and **DEFAULT** rules, this value comes from the local computer clock time. For **SERVERDEFAULT**, the value comes from the server clock when a user saves the work item.
- currentuser: Copies

from

from="value | field | clock |
currentuser"

the name of the user who is currently logged on. Use the short username of the current user as the value. No additional attributes are required. Used for string fields.

- field: Copies the value of the field attribute that you specify. Requires a field="abc" attribute. By default, if the specified "from" field is empty, nothing is performed. The field attribute is used only for <COPY> and <DEFAULT> rules.
- value: Copies the value of the value attribute that you specify. Use the value of a string constant that you specify. Requires a value="abc" attribute. value is used only for <COPY> and <DEFAULT> rules.

If you specify "value" or "field," you must also include the value or field attribute, respectively.

field

field="fieldReferenceName"

Optional. Specifies the name of the field whose value is to be copied into the field when field is specified for the from attribute.

pattern

pattern="patternValue"

Required. Enforces basic pattern matching for strings only. patternValue is a string that consists of between 1 and 255 characters, inclusive. That string must not contain a backslash character (\). Each character in the string is interpreted as a literal, unless it is one of the following six metacharacters:

- "A" or "a" represent a single alphabetical character.
- "N" or "n" represent a single numeric character.
- "X" or "x" represent a single alphanumeric character.

Pattern value: `^[^\\]*$`

For example,
pattern="xxxxx.nn.nn"

value

value="valueToCopy"

matches any five alphanumeric characters, then a period, then two numeric characters, then a period, then two more numeric characters.

Optional. Specifies the value to be copied into the field when value is specified for the from attribute.

See Also

Concepts

[Define and modify work item fields](#)

[Change the workflow for a work item type](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

FIELD (Definition) element reference

[Example](#) [See Also](#) [Send Feedback](#)

You use the **FIELD** (Definition) element to define a work item field and specify the rules and conditions that apply to it. The attributes that you assign to a field determines its data type and whether it is available for inclusion in reports.

Note

For information about the **FIELD** (Workflow) element, which you use to specify rules and conditions to fields during a state change or workflow transition, see [FIELD \(Workflow\) element](#).

Schema Hierarchy

WITD

WORKITEMTYPE

FIELDS

FIELD

```
<FIELD name="fieldDisplayName"
refname="fieldReferenceName"
type="String | Integer | Double | DateTime | PlainText | HTML | Hist
syncnamechanges ="true | false"
reportable="Dimension | Detail | Measure"
formula="sum"
reportingname="ReportingDisplayName"
reportingrefname="ReportingReferenceName" >
```

```
<ALLOWEDVALUES> . . . </ALLOWEDVALUES>
<ALLOWEXISTINGVALUE />
<CANNOTLOSEVALUE />
<COPY />
<DEFAULT />
<EMPTY />
<FROZEN />
<HELPTEXT> . . . </HELPTEXT>
<MATCH />
<NOTSAMEAS />
<PROHIBITEDVALUES /> . . . </PROHIBITEDVALUES>
<READONLY />
<REQUIRED />
<SERVERDEFAULT />
<SUGGESTEDVALUES /> . . . </SUGGESTEDVALUES>
<VALIDUSER />
<WHEN> . . . </WHEN>
<WHENNOT> . . . </WHENNOT>
<WHENCHANGED> . . . </WHENCHANGED>
<WHENNOTCHANGED> . . . </WHENNOTCHANGED>
</FIELD>
```

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Attributes

Attribute	Description
name	<p>Required. The friendly name of the field that appears in the drop-down menus of work item queries. The friendly name must be unique across all fields that are defined within a team project. Also, the friendly name may differ from the label that appears next to the field on the work item form. For more information, see Control XML element reference.</p> <p>The attribute type is <code>typelib:FieldName</code>. Minimum length: 1; maximum length: 128.</p> <p>Pattern value: <code>^[^\.\[\]]+\$</code></p> <p>Pattern value example: Assigned To</p> <p>Required. The unique label that distinguishes a field from all other fields that are defined in</p>

refname

the team project collection.

For additional requirements and restrictions on friendly names and reference names, see [Naming conventions for work item tracking objects](#).

The attribute type is `typelib:ReferenceFieldName`.
Minimum length: 1; maximum length: 70.

Pattern value: `^[a-zA-Z_][a-zA-Z0-9_]*(\.[a-zA-Z0-9_]+)$`

Pattern value example:
`Company.IssueType`

Required. Specifies the type of data that the field accepts.

 **Note**

Fields in different project collections that have the same `reportingrefname` must be assigned the same value for type. If the values differ, data conflicts can occur when the data warehouse is processed. For more information, see [Resolve schema conflicts that are occurring in the data warehouse](#).

The following table lists valid

values for this attribute:

Value	Usage
DateTime	Specifies a date according to Coordinated Universal Time (UTC) moment in time.
Double	Specifies a floating-point value.
GUID	Specifies a field that will contain a unique identifier.
History	Supports discussion threads and keeps track of other historical information. Note The System.History field is the only field that uses this data type. You cannot define a custom field using this data type.

type

HTML	Supports capture of rich-text data and is used for longer text descriptions such as a work item description.
Integer	Specifies a 32-bit signed integer value.
PlainText	Supports entry of a text string that can contain more than 255 Unicode characters.
String	Supports entry of a text string that can contain up to 255 Unicode characters. Use a String field for a label or other short text string up to one line long.
	Specifies a field that displays entries in a hierarchical or tree structure, such as what is required to display the area and iteration paths for a product. To define child nodes, see Add and modify area and

iteration paths.

TreePath  **Note**

The System.AreaPath and System.IterationPath fields are the only fields that use this data type. You cannot define a custom field using this data type.

syncnamechanges

Optional. Specifies whether the work item field is used to store names that you want to be updated as changes are made in Active Directory or a Workgroup. This option is only valid when type="String". The attribute type is xs:boolean.

Specify true to enable synchronization for the data field, specify false to disable synchronization for the data field.

Optional. Specifies whether data from the field is available for inclusion in reports. Fields with the default value of None are neither exported to the relational data warehouse nor processed for the SQL Server Analysis

Services cube. For more information about reportable fields, see *Add or modify work item fields to support reporting*.

 **Note**

Fields in different project collections that have the same value for the `reportingrefname` must be assigned the same value for `reportable`. If the values differ, data conflicts can occur when the data warehouse is processed. For more information, see *Resolve schema conflicts that are occurring in the data warehouse*.

The following table lists valid values for this attribute:

Value	Usage
	Can be specified for fields of type <code>DateTime</code> , <code>Double</code> , <code>Integer</code> , or <code>String</code> .
	The data in this field is moved into

reportable

Detail

the relational warehouse database in the Work Item and Current Work Item tables but not into the Analysis Services cube. This type is a good choice for unrestricted text fields because you can use them in reports without making the cube significantly larger.

Can be specified for fields of type DateTime, Integer, String, or TreePath.

The data in this field enters the relational

Dimension

warehouse database and the Analysis Services cube as an attribute of the Work Item dimension so that the data can be used to filter reports. Use this type for fields that have lists of valid values. Work Item Type and State are good examples of a dimension.

Use the measure type only for Integer and Double fields. Measures are the numeric values in your reports.

When the Analysis Services

Measure

cube is processed, data is precalculated on fields whose reportable attributes are set to measure. For example, the Work Item and Current Work Item measure groups contain cumulative data for the following fields: Original Estimate, Remaining Hours, and Completed Hours.

When you specify measure, you must specify the formula attribute.

Specify None when you do not

	<p>None want to use the field for inclusion in reports. This is the default assignment.</p>
formula	<p>Optional. The aggregation type for the measure reportable type. The only valid value is sum, which returns the sum of all values over the set.</p> <p>Optional. Specifies the name that appears in reports. If you do not specify a value, the value that is assigned to the name attribute is used.</p> <p> Note</p> <p>Fields in different project collections that have the same reportingrefname must be assigned the same value for the reportingname. If the values differ, data conflicts can occur when the data warehouse is processed. For more information, see Resolve schema conflicts that are occurring in the data warehouse.</p>
reportingname	<p>The attribute type is typelib:FieldName. Minimum</p>

length: 1; maximum length: 128.

Pattern value: `^[^\.\[\]]+$`

Pattern value example: Assigned
To

Optional. Specifies the reference name that is used when a reportable field is processed. If you do not specify a value, the value that is assigned to the `refname` attribute is used.

You can use this attribute to either merge or diverge fields that are processed to the data warehouse. To merge two fields that have distinct reference names and that are defined in different project collections, you assign the same `reportingrefname` to both of them. To diverge two fields that have the same reference name but that are defined in different project collections, you assign a different `reportingrefname` to each field.

`reportingrefname`

You should merge fields whenever possible to minimize the number of fields in the warehouse and to keep under the maximum limit of 1024 reportable fields. You can generate cross-group reports with merged fields.

The attribute type is
typelib:ReferenceFieldName.
Minimum length: 1; maximum
length: 70.

Pattern value: `^[a-zA-Z_][a-zA-Z0-9_]*(\.[a-zA-Z0-9_]+)$`

Pattern value example:
Company.IssueType

Child Elements

Element	Description
ALLOWEDVALUES	Optional. Defines a list of allowed values for the field. Allowed values are values that are available for selection in a field list on work item forms and in the query builder. You must select from one of

these
values.

ALLOWEXISTINGVALUE

Optional.
Defines the
field to
allow
existing
values. This
element
allows the
field values
that already
exist to be
used, even
if they are
not valid.
All new
field values
must be
valid.

CANNOTLOSEVALUE

Optional.
Defines the
field as
cannot lose
value. This
element
keeps the
current field
value and it
cannot be
cleared or
made
empty.

COPY

Optional.
Specifies another field that contains a value to be copied into the current field.

DEFAULT

Optional.
Defines a default value for the field.

EMPTY

Optional.
Defines the field as empty.

FROZEN

Optional.
Defines the field as frozen. A frozen field cannot be changed to any non-empty value after changes are committed. However, you can manually

clear the field, save the work item, and then specify a different value.

HELPTEXT

Optional.
Defines the text displayed in the ToolTip for the field.

MATCH

Optional.
Defines a pattern for the field that the field value must match.

NOTSAMEAS

Optional.
Specifies another field, the value of which cannot be identical to the value of the current field.

PROHIBITEDVALUES

Optional.
Defines a list of prohibited values for the field.

READONLY

Optional.
Defines the field as read-only.

REQUIRED

Optional.
Defines the field as required.

SERVERDEFAULT

Optional.
Specifies a server component that will provide the value for the field.

Optional.
Defines a list of suggested values for the field.
Suggested values are

SUGGESTEDVALUES

values that are available for selection in a field list on work item forms and in the query builder. You can enter other values additionally to the ones in the list.

VALIDUSER

Optional. Specifies that the list of allowed values must consist only of valid users of the system.

WHEN

Optional. Specifies one or more rules to apply to the current field when another field has a specific

value.

WHENCHANGED

Optional.
Applies one or more rules to the current field when a specific field's value is changed.

WHENNOT

Optional.
Applies one or more rules to the current field when another field does not have a specific value.

WHENNOTCHANGED

Optional.
Applies one or more rules to the current field when a specific field's value is not changed.

Parent Elements

Element	Description
FIELDS	Required. Contains the work item type field definitions.

Remarks

1. FIELD (Definition) is a required child element of FIELDS (Definition).
2. For an overview of all system and predefined fields that are defined for the TFS process templates, see [Work item field reference for Visual Studio ALM](#).
3. You cannot define a custom field that starts with the System. prefix. You can define a field by using the Microsoft. prefix, however, this practice is strongly discouraged because it might impede Team Foundation Server functionality.

For additional requirements and restrictions on field-friendly names and reference names, see [Naming conventions for work item tracking objects](#).

4. Although you can rename a field's friendly name, you can't rename the field's reference names. You can list fields and change several field attributes using the **witadmin** command-line tool. See [Manage work item fields \[witadmin\]](#).
5. You can define no more than 1,024 work item fields in the same team project collection, and you can set no more than 1,024 fields to reportable in all team project collections.

All fields defined within all work item types (WITs) for all team projects defined for a project collection are for a team project collection. Therefore, attributes that you assign to fields that are defined in one WIT must match across all WITs for all team projects in a collection. In addition, all reportable fields from all collections are exported to the data warehouse databases.

6. For information about how to label fields for reporting purposes, see [Add or modify work item fields to support reporting](#).
7. When you add an existing field to a different WIT, either explicitly set the reporting attributes to be the same as the current field definition, or let them

default to these values.

Schema conflicts can occur if different reporting attributes are assigned to the same field in different WITs or the same WITs in different team projects. To fix these conflicts, see [Resolve schema conflicts that are occurring in the data warehouse](#).

8. To understand how fields are used to support queries, reports, and work item tracking, see [Modify or add a field to support queries, reports, and workflow](#).
9. For an overview of how to apply constraints or conditions on a FIELD by using child elements, see [Apply a rule to a work item field](#).

Example

 **Copy Code**

```
<FIELD name="Activity" refname="Microsoft.VSTS.Common.Activity" type="Text"
  <HELPTEXT>Type of work involved</HELPTEXT>
  <SUGGESTEDVALUES>
    <LISTITEM value="Development"/>
    <LISTITEM value="Testing"/>
    <LISTITEM value="Requirements"/>
    <LISTITEM value="Design"/>
    <LISTITEM value="Deployment"/>
    <LISTITEM value="Documentation"/>
  </SUGGESTEDVALUES>
</FIELD>
```

See Also

Reference

[Manage work item fields \[witadmin\]](#)

Concepts

[Naming conventions for work item tracking objects](#)

[Work item field reference for Visual Studio ALM](#)

Customize work tracking objects to support your team's processes

[Define and modify work item fields](#)

Other Resources

[Work item tracking: Index to XML element definitions](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define and modify work item fields

[See Also](#) [Send Feedback](#)

You use work item fields to track data for a work item type and to define the filter criteria for queries as well as to generate reports. Any data element, except for core system data fields in Team Foundation Server (TFS), that you want to track, use to define the workflow, or appear on the form for a work item type must be defined as a work item field. You can define work item fields within the definition of a work item type or global workflow.

Work item fields are maintained for a team project collection. You add fields when you perform one of the following tasks:

- Create a team project. All fields that are defined within the definitions for work item types or global workflow and that are defined for the selected process template are created. The core system fields are automatically defined for every work item type that is defined for a team project. For a list of these fields, see

[Work item field reference for Visual Studio ALM.](#)

- Import a definition for a type of work item. All new fields that are defined within the definition for a type of work item are added to the collection. For more information, see [All WITD XML elements reference](#).
- Import a global workflow definition. All new fields that are defined within the global workflow are added to the collection. You define a global workflow when you want to maintain a set of work item fields that several types of work items share. For more information, see [Customize global workflow](#).
- Map a project collection to an instance of Project Web App (PWA). After you install the Team Foundation Server Extensions for Project Server, configure the

integration by mapping various components of Team Foundation. When you map a collection, a global workflow definition that supports several fields in Project Server fields is imported. For more information, see [Project Server fields added to TFS to support data synchronization](#).

All fields that are defined in all work item types and all global workflows for all team projects make up the complete set of fields that are defined for the collection. You can change the attribute of, rename, and delete existing fields. However, you incur certain costs when you make these kinds of changes. For more information, see [Customize work tracking objects to support your team's processes](#).

To add or customize a field for a collection, modify the XML content for the definition of a type of work item or global workflow. Define each field through a FIELD element within the FIELDS section of the definition for either a type of work item or global workflow. For information about the structure and location of these files, see [All WITD XML elements reference](#) and [Customize global workflow](#).

FIELD (Definition) syntax structure

Each FIELD (Definition) element has an friendly name, which is optional, and a reference name, which is required. The reference name must be unique within the collection. For more information, see

[Naming conventions for work item tracking objects.](#)

Important

You can change the friendly name for a field by using the **witadmin changefield** command-line tool. However, if you rename a field, it affects all work item types in all team projects within the collection.

Define a field by the following XML syntax in the definition file for the type of work item.

 Copy C

```
<FIELD name="fieldDisplayName"
  refname="fieldReferenceName" type="String | Integer | Double | Da
  syncnamechanges="true | false"
  reportingname="reportingDisplayName"
  reportingrefname="reportingReferenceName"
  reportable="Dimension | Detail | Measure"
  formula="sum" >
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>
  <ALLOWEXISTINGVALUE />
  <CANNOTLOSEVALUE />
  <COPY />
  <DEFAULT />
  <EMPTY />
  <FROZEN />
  <HELPTTEXT> . . . </HELPTTEXT>
  <MATCH />
  <NOTSAMEAS />
  <PROHIBITEDVALUES /> . . . </PROHIBITEDVALUES>
  <READONLY />
  <REQUIRED />
  <SERVERDEFAULT />
  <SUGGESTEDVALUES /> . . . </SUGGESTEDVALUES>
```

```
<VALIDUSER />  
<WHEN>> . . . </WHEN>  
<WHENNOT> . . . </WHENNOT>  
<WHENCHANGED> . . . </WHENCHANGED>  
<WHENNOTCHANGED> . . . </WHENNOTCHANGED>  
</FIELD>
```

For more information, see [FIELD \(Definition\) element reference](#).

Data types

The type of a field defines the kind and size of data that you can store in the field. A field can have only one type defined within a team project collection. This restriction encourages organizations to use common fields across projects and work item types.

The following list shows the supported field types:

Data type	Description
DateTime	Specifies a date according to Coordinated Universal Time (UTC) moment in time. Add a DateTime field to a work item form by using either the FieldControl or DateTimeControl type attributes.
Double	Specifies a floating-point value. Double fields are frequently used in query filters and results lists. Add a Double field to a work item form by using the FieldControl type attribute.
	Specifies a field that will contain a unique identifier.

GUID You cannot add a GUID field to a work item form.

Supports viewing discussion threads and tracking of other historical information.

 **Note**

History The System.History field is the only field that uses this data type. You cannot define a custom field using this data type.

Add the System.History field to a work item form by using the WorkItemLogControl type attribute which supports rich text format controls.

HTML Supports the ability to capture rich-text data and to use longer text descriptions such as a work item description. An HTML field differs from a PlainText field in that an HTML field is strongly typed to HTML for richer displays of information.

Add an HTML field to a work item form by using the HTMLControl type attribute, which supports rich text format

controls.

Integer

Specifies a 32-bit signed integer value. Integer fields are frequently used in query filters and results lists.

Add an Integer field to a work item form by using the FieldControl type attribute.

Supports entry of a text string that can contain more than 255 Unicode characters.

PlainText

Add a PlainText field to a work item form by using either the FieldControl type attribute and limited to plain text, or the HTMLControl type attribute which supports rich text format controls.

String

Supports entry of a text string that can contain up to 255 Unicode characters. Use a String field for a label or other short text string up to one line long. String fields are frequently used in query filters and results lists.

Add a String field to a work item form by using the FieldControl type attribute.

Specifies a field that displays entries in a hierarchical or tree structure, such as the requirements to display area and iteration paths for a product.

 **Note**

TreePath

The System.AreaPath and System.IterationPath fields are the only fields that use this data type. You cannot define a custom field using this data type.

Add the System.AreaPath and System.IterationPath fields to a work item form by using the WorkItemClassificationControl type attribute.

Reportable attributes

Some field values are especially useful for reporting. By using the work item type definition language, you can specify the following optional attributes:

- **reportable.** Set the reportable attribute to None, Detail, Dimension, or Measure, depending on whether and how you want the field to be included in reports. Data from fields that have a value other than None for this attribute are exported to the data warehouse and can be included in reports.

When you add an existing field to a work item type, the current value for the reportable attribute is used. When you add a field to a work item type, reporting is disabled unless you explicitly specify it by using the reportable attribute.

- **reportingname.** Assign a different label to a field that is used when data appears in reports. If you do not specify a value, the friendly name that is assigned for the name attribute is used.
- **reportingrefname.** Assign a different reference name to a field that is used when data is exported to the relational data warehouse. If you do not specify a value, the value that is assigned to the refname attribute is used.

Use this attribute to either merge or diverge fields that are included in reports. To merge two fields that have distinct reference names and that are defined in different project collections, you assign the same **reportingrefname** to both fields. To diverge two fields that have the same reference name and that are defined in different project collections, you assign a different **reportingrefname** to each field.

After you define a field, you can use the **witadmin changefield** command at a command prompt to change the value of all attributes except for the **refname** attribute.

For information about best practices in labeling fields for reporting purposes, see [Add or modify work item fields to support reporting](#).

Reportable attribute values

As the following table describes, you can assign one of the following values to the reportable attribute: none, dimension, detail, and measure.

Note

You can make a field reportable after it has been used for a work item. After you set the reportable value, new revisions of the work item that are copied to the warehouse will contain the field value. However, the revisions that are already in the warehouse will not be backfilled with the existing values.

Attribute value	Description
Detail	Use the Detail type only for Integer, Double, String, or DateTime fields.
	The data in this field is moved into the relational warehouse database in the Work Item and Current Work Item tables but not into the SQL Server Analysis Services cube. By using this type for unrestricted text fields, you can use them in reports without making the cube significantly larger.
	Use the Dimension type only for Integer, String, or DateTime fields.

Dimension

The data in this field enters the relational warehouse database and the Analysis Services cube as an attribute of the Work Item dimension so that the data can be used to filter reports. Use this type for fields that have lists of valid values. Work Item Type and State are good examples of a dimension.

Use the Measure type only for Integer and Double fields. Measures are the numeric values in your reports.

Measure

During processing of the Analysis Services cube, data is precalculated on fields that are set to Measure. For example, the Work Item and Current Work Item measure groups contain cumulative data for the following fields: Original Estimate, Remaining Hours, and Completed Hours. For more information about measure groups, see

[Perspectives and measure groups provided in the Analysis Services cube for Visual Studio.](#)

When you specify Measure, you must specify sum for the formula attribute,

which returns the sum of each measure referenced in the query.

None

Specify None when you do not want to include the field in reports. This value is the default assignment.

▣ **Examples Showing How to Specify Reporting**

Detail Example

```
<FIELD refname="MyCorp.Summary" name="Summary" type="String"
reportable="detail">
```

Dimension Example

```
<FIELD refname="MyCorp.Category" name="Category" type="String"
reportable="dimension">
```

Measure Example

```
<FIELD refname="MyCorp.Cost" name="Cost" type="Integer"
reportable="measure" formula="sum">
```

Indexed Fields

You can enable or disable indexing for a work item field by using the **witadmin indexfield** command. When you enable indexing for a field, you may increase the performance of finding work items whose queries specify that field. If you add a custom field that you use in many of your work item queries, you may want to enable indexing for that field. For more information, see

[Manage work item fields \[witadmin\]](#).

FIELD (Definition) child elements

There are a number of optional child elements that you can specify for a work item field, for example:

- Help text. Defines the tool tip or help information that displays for a field on the work item form (**HELPTEXT**).
- Field restrictions. Specifies a field as required, read-only, empty, frozen, or matching a specific pattern (**REQUIRED, READONLY, EMPTY, FROZEN, MATCH**).
- Field value auto-population. Specifies a value for the field (**DEFAULT, COPY, SERVERDEFAULT**).
- Field lists. Specifies allowed, suggested, or prohibited values. (**ALLOWEDVALUES, SUGGESTEDVALUES, PROHIBITEDVALUES**)
- Field conditions. Applies one or more rules to a field when another field is changed or not changed, or another field has or does not have a specific value, or other condition (**WHEN, WHENNOT, WHENCHANGED, WHENNOTCHANGED, CANNOTLOSEVALUE, NOTSAMEAS**).

For more information, see

[All FIELD XML elements reference.](#)

System and predefined fields

All system defined fields have reference names that begin with System, for example, System.AreaPath, System.AssignedTo, and continue in that pattern.

Predefined fields defined by the default process templates that TFS provides begin with Microsoft.VSTS and then further differ based on their usage. Examples of predefined fields that are used in common, for scheduling purposes and integration with Microsoft Project, for integration with Team Foundation Build, and integration with Team Foundation are as follows:

- Microsoft.VSTS.Common.Priority
- Microsoft.VSTS.Scheduling.DueDate
- Microsoft.VSTS.Build.FoundIn
- Microsoft.VSTS.TCM.Steps

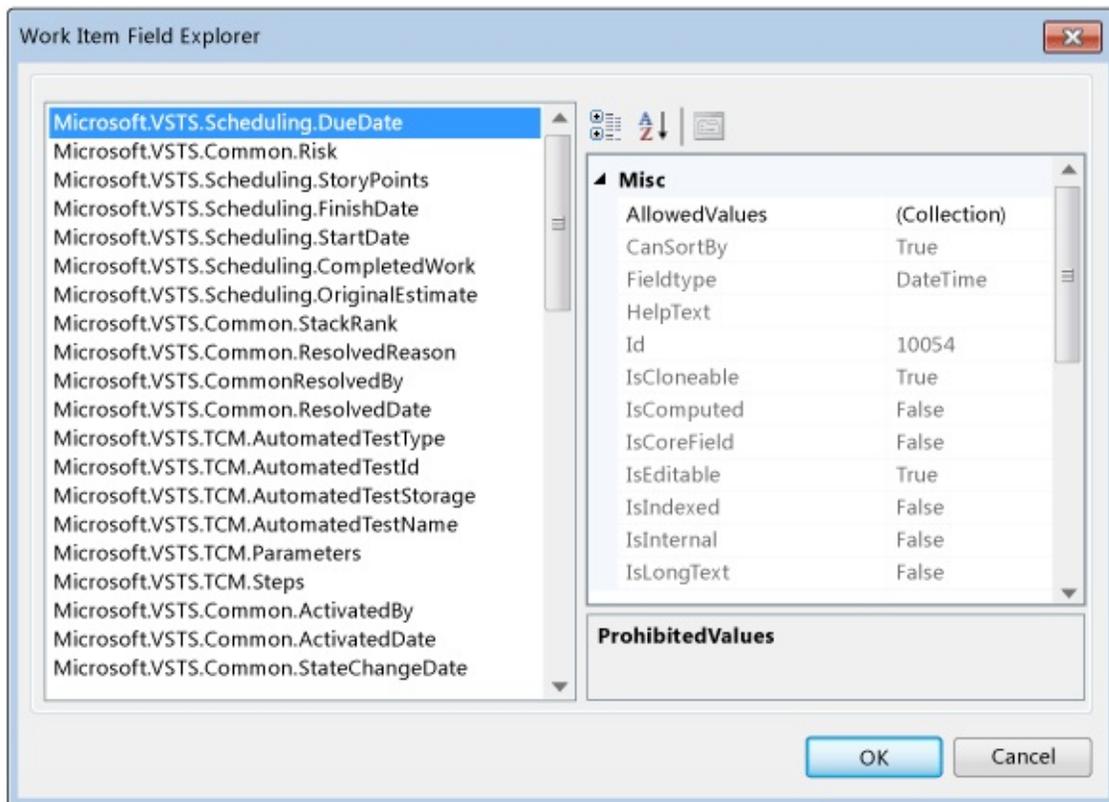
For an overview of all system and predefined fields that are defined for the default process templates that TFS provides, see

[Work item field reference for Visual Studio ALM](#). For more information about specifying field names, see [Naming conventions for work item tracking objects](#).

Hidden and non-changeable attributes and the Work Item Field Explorer

In addition to the attributes that you can change for a work item field, there are a number of non-changeable and virtually hidden attributes for each field. You can look up the assignments of these fields using the Work Item Field Explorer tool. Access the Work Item Field Explorer tool from the Process Editor power tool.

Work Item Field Explorer



Note

The Process Editor power tool provides a graphical user interface for

customizing Team Foundation Server process templates. You can use this tool to import and export work item types, modify the contents of the process template, and explore the set of fields defined for a team project collection. For more information, see the following page on the Microsoft website: [Team Foundation Server Power Tools](#).

See Also

Reference

[Manage work item fields \[witadmin\]](#)

Concepts

Add or modify work item fields to support reporting

[All FIELD XML elements reference](#)

Other Resources

[Modify or add a field to support queries, reports, and workflow](#)

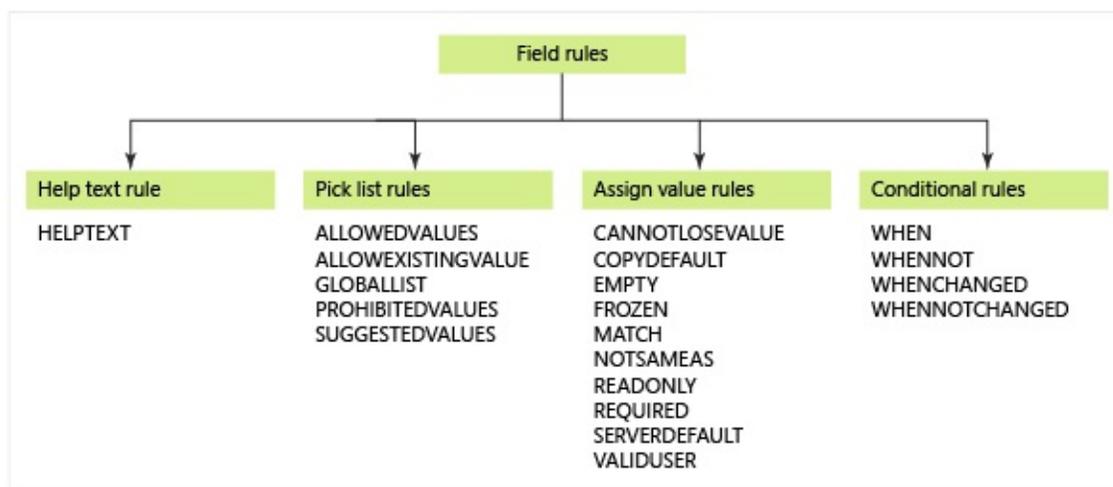
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Apply a rule to a work item field

[See Also](#) [Send Feedback](#)

Depending on a field's data type, you can set various restrictions on what data can be entered into that field. You can specify values for a pick list (drop-down menu), set default values, clear entries, or restrict changes. With conditional rules, you can apply rules to a field based on dependencies between different fields' values. You can also restrict who can modify a field or scope a rule to only apply to a group.

All of these rule elements can be defined within the **FIELD** definition of a work item type (WIT) definition, subject to some [restrictions for System fields](#). And, with the exception of **HELPTTEXT**, you can specify these rules to take affect during a workflow transition or as child elements within a **FIELD** (Global workflow) element.



You can define any combination of rules to a field, subject to the constraints as described in this topic.

[Assign value rules:](#) Define runtime behavior and constraints:

- [Clear, set a default, copy a value, or force values to match a pattern](#) [Conditional rules:](#) Specify when a set of rules will be applied to a parent field.

[Help text:](#) Specify tooltip text to appear in a work item form for a field.

[Pick list:](#) Specify a drop-down menu or pick list of allowed, suggested, or prohibited values.

- [Require, read-only, and restrict values assigned to a field](#) [Set conditions based on user role:](#) Apply rules based on who is creating or modifying the work item.
- [Restrict who can create or modify a work item](#) [Use tokens to specify a group:](#) Specify the domain or scope of the group using the right token.

[What rules can be](#)

[Where](#)

[How do I modify the State and](#)

applied to System fields?	should I apply a field rule?	Reason fields?
How can I avoid validation errors on person-name fields?	How are rules evaluated? What order is applied?	How do I make a field hold a value that is the sum of two other fields?
Is there a way to define a multi-select pick list?	How do keystroke entries in a form affect rule evaluation?	When would I define field rules using global workflow?

Field rules are one component you have to customize work item tracking. To learn more, see [Customize work tracking objects to support your team's processes](#).

For information on modifying fields or adding field rules to a WIT definition file, see [Modify or add a field to support queries, reports, and workflow](#).

Help text

You can customize the help text or tooltip text that appears when a user points to a field that appears on a work item form. You can customize and localize the Help text for the same field that appears in different WITs and different team projects. Help text is restricted to 255 Unicode characters.

The following example shows the assignment of Help text to a custom Business Justification field:

Copy Code

```
<FIELD name="Business Justification" refname="Fabrikam.BusinessJusti:  
<HELPTEXT>Only required when you set the Urgencyfield to Need Immedi:  
</FIELD>
```

To provide users guidance that exceeds the 255 characters limit, see [Provide help text, hyperlinks, or web content on a work item form](#).

Note

The presence of HELPTEXT adds to the size of data stored and can impact scalability. If you support several hundreds of team projects within a single TFS instance, be conservative in your use of HELPTEXT rules.

Pick list rules

Pick list rules define the values that a user can or can't choose for a String field. Values defined in a pick list appear on a work item form and the query editor. You can combine lists, and expand or contract lists. You can also use the **for** and **not** attributes to

[apply or ignore these rules](#), based on who is modifying the work item.

Rule	Usage
ALLOWEDVALUES	Limits the values a user can choose based on the specified values.
ALLOWEXISTINGVALUE	Allows a field to retain an existing value, even if it is no longer in a pick list. Including this rule is recommended when you change the field values in a pick list or for pick lists that contain person

names.

GLOBALLIST

Specifies the name of a global list that contains values maintained for a team project or project collection.

PROHIBITEDVALUES

Prevents specified values from being assigned. The work item can't be saved if the field contains a prohibited value.

SUGGESTEDVALUES

Defines a list of values that users can choose from, but are not restricted to selecting. Users can specify values other than those in this

list.

For examples of using pick lists, see [Define pick lists](#).

Assign value rules

Assign value rules define runtime behavior and constraints, such as specifying default values, clearing fields, requiring fields to be defined, and more. You can

[apply or ignore these rules](#) based on who is modifying the work item using the **for** and **not** attributes.

Clear, set a default, copy a value, or force values to match a pattern

These rules support setting defaults, copying values from one field to another, or enforcing a field value to match a prescribed pattern.

Rule	Usage
COPY	Copies a specified value to a field when a user creates or modifies a work item.
DEFAULT	Specifies a value for a field that is empty when a user creates or modifies a work item. If a field already has a value, the DEFAULT rule is ignored.

EMPTY

Clears the field of any value that it contains and then makes the field read-only when a user saves the work item. You shouldn't use **EMPTY** with **READONLY**.

EMPTY is primarily used during state transition to clear fields that apply to the state to which the item is transitioning.

MATCH

Forces entries made to a String field to conform to a specified pattern of characters or numbers.

SERVERDEFAULT

Copies either the current user name or the server clock value to a field when a user saves a work

item. These fields usually appear as read-only on the form.

For the syntax structure and examples, see

[Define a default value or copy a value to a field.](#)

▣ **Require, read-only, and restrict values assigned to a field**

These rules specify restrictions on specifying or changing the value of a field.

Rule	Usage
CANNOTLOSEVALUE	Prevents users from clearing a field of a value once a value has been specified.
FROZEN	Prevents users from changing the value of a field once it contains a value. As soon as a user saves the work item with a value

NOTSAMEAS

in that field,
the value can
no longer be
modified.

Prevents a
field from
being
assigned the
same value as
that which
was assigned
to another
field.

Prevents a
field from
being
modified at
all. You
might want to
apply this
rule under
certain
conditions.
For example,
after a work
item is
closed, you
want to make
a field read-
only to
preserve the
data for
reporting
purposes.

READONLY

Do not use **READONLY** with the **EMPTY** element because **EMPTY** also makes a field read-only. If you combine these elements, results will be inconsistent.

In addition, you can make a field appear as read-only from the work item form using the Control element `ReadOnly` attribute. The field can be written to by other clients, but not through the work item form.

Requires a user to specify a value for the

REQUIRED

field. Users cannot save a work item until they have assigned values to all required fields.

For the syntax structure, see

[All FIELD XML elements reference.](#)

Restrict who can create or modify a work item

You can control who can create or modify a work item by applying the **VALIDUSER** element to person-name fields. When you specify this element, you indicate which user or group of users can be assigned as a value for the field. You can set this element to support the optional **group** attribute, which requires that the person who is assigned to the field must be a direct or indirect member of the group that you specify. By default, all members of the Team Foundation Valid Users group can be specified in the field.

The **VALIDUSER** element is valid only for String field types. You can

[allow or restrict whether the rule applies to the user who is modifying the work item](#) by specifying a user or group for the `for` or `not` attributes, respectively.

Copy Code

```
<VALIDUSER group="groupName" for="userName" not="userName" />
```

You can use the **VALIDUSER** rule only when you refer to person-name fields. The following system fields are examples of person-named fields:

- Activated By (System.ActivatedBy)
- Assigned To (System.AssignedTo)

- Authorized As (System.AuthorizedAs)
- Changed By (System.ChangedBy)
- Closed By (System.ClosedBy)
- Created By (System.CreatedBy)

In addition to the system fields, you can create a custom string field and use it as a person-named field. Also, you can synchronize custom person-named fields with Active Directory (specify `syncnamechanges="true"`).

Work item fields do not distinguish between user identities in different domains. Therefore, "Fabrikam\ctsoapo" and "Contoso\ctsoapo" are treated as the same user when they are entered into a field that uses the VALIDUSER rule.

Conditional rules

Conditional rules let you specify when a set of rules will be applied to a parent field. You can set conditions based on whether another field is assigned (or not assigned) a specified value or when another field changes (or doesn't change). You can include pick list and assign value rules within a conditional rule element.

	Rule	Usage
WHEN		Specifies the rules to apply to the parent field when another field is assigned a specified value.
WHENNOT		Specifies the rules to apply to the parent field when another field isn't assigned a specified value.
		Specifies the rules to

WHENCHANGED

apply to the parent field when a specified field's value changes.

WHENNOTCHANGED

Specifies the rules to apply to the parent field when a specified field's value doesn't change.

You can specify multiple conditional rules per field. However, you can only specify a single driving field per conditional rule. You can't nest conditional rules. For the syntax structure and examples, see

[Assign conditional-based values and rules.](#)

Apply or ignore rules based on who is creating or modifying the work item

You can make a pick list or assign value rule to apply or not apply to a group of users by using the for or not attributes. Scope the rule to a group. To have the rule scoped to multiple groups, you must create a parent TFS group that includes the set of groups that you want to use.

- Make a field required only for a specified group:

Use for to apply a rule to a group. This example requires any user in the Junior Analysts group to complete the Second Approver field.

Copy Code

```
<FIELD name="Second Approver">  
<REQUIRED for="Example1\Junior Analysts"/>  
</FIELD>
```

- Restrict modification of a field to a group of users:

Use not to exclude a group from a rule. This example defines the Triage Description field as read-only for everyone except those users in the Triage Committee group.

Copy Code

```
<FIELD name="Triage Description">  
<READONLY not="[Project]\Triage Committee" />  
</FIELD>
```

- Make a field required for some users and not for others:

Use a combination of for and not to simultaneously apply a rule to some

and not for others. This example defines Severity as a required field for users in the Project Members group, but not for those in the Project Admins group.

Copy Code

```
<FIELD name="Severity">  
<REQUIRED for="[Project]\Project Members" not="[Global]\Project  
</FIELD>
```

Because Deny takes precedence over Allow, if a user is in both groups, the "not" statement would be enforced, and the field would not be required.

■ Use tokens to reference groups

When you restrict a rule to a group, you must indicate the domain or scope of the group. For some values, you can use tokens.

Person-name fields can accept values that reference both users and groups. Field attributes, for and not, apply to groups. You can use the following tokens when specifying values for these items.

- Scope to a team project - [Project] :

The [Project] token is used to specify a group that is defined for a team project. This could correspond to a team, built-in TFS group, such as the [Project]\Contributors group, a custom TFS group you create at the project level, or a Windows group that you added to a TFS group. For example:

- Team: [Project]\Fabrikam Team

When you create a team, a TFS group is created that contains the members assigned to the team.

- Team project group: [Project]\Contributors
- Windows group added to a team project: [Project]\ Triage Committee

Tip: You can view a list of valid groups by opening the Security page in the Team Web Access (TWA) administration context.

- Scope to a project collection - [CollectionName]:

Use [CollectionName] to reference a collection-scoped TFS group, such as the Project Collection Administrators group or a Windows group you add to a collection. For example:

Copy Code

```
<FIELD name="Title">  
<READONLY for="[DefaultCollection]\Project Collection Valid Use  
</FIELD>
```

- Scope to a server instance - [GLOBAL]:

Use the [GLOBAL] token to reference a server-scoped TFS group, such as a built-in group or a Windows group you add to a server-level group. For example:

Copy Code

```
<FIELD name="Title">  
<READONLY for="[Global]\Team Foundation Valid Users"/>  
</FIELD>
```

- Specify a domain qualified account or group:

Domain-qualified account name, such as the one shown in the following example, can be used to reference a domain user or group. Note that some rules only support groups and do not support referencing domain users.

Copy Code

```
<LISTITEM value="FABRIKAM\Christie Church's Direct Reports"/>
```

All users and groups must be qualified by one of these tokens. For example, the following XML isn't valid because it doesn't qualify the specified group with a valid token.

Copy Code

```
<FIELD name="Title">  
<READONLY for="Dev Team"/>  
</FIELD>
```

Q & A

Q: What rules can be applied to System fields?

A: System fields have System.Name reference names, for example System.Title and System.State. TFS restricts customization of these fields, except for these instances:

- HELPTEXT rule can be assigned to all fields.
- READONLY rule can be assigned to the State and Reason fields.
- Most rules can be assigned to the Title, Assigned To, Description or Changed By System fields.

Q: How can I avoid validation errors on person-name fields?

A: To avoid validation errors that would otherwise occur when members leave the team and are no longer registered as project contributors, include the **ALLOWEXISTINGVALUE** element for the Assigned To field.

 **Copy Code**

```
<FIELD name="Assigned To" refname="System.AssignedTo" type="String"
  <HELPTEXT>The user who is working on this work item</HELPTEXT>
  <ALLOWEXISTINGVALUE />
  <VALIDUSER />
  <ALLOWEDVALUES expanditems="true" filteritems="excludegroups">
    <LISTITEM value="Active" />
    <LISTITEM value="[project]\Contributors" />
  </ALLOWEDVALUES>
  <DEFAULT from="field" field="System.CreatedBy" />
</FIELD>
```

Q: Is there a way to define a multi-select pick list?

A: This feature isn't natively supported, However, you might be able to adapt source code provided in this CodePlex project:

[Custom Controls for TFS Work Item Tracking.](#)

Q: How do I modify the State and Reason fields?

A: The State and Reason fields are defined within the WORKFLOW section of the WIT definition. You can specify most field rules to apply to a field during a change of state, selection of a reason, or for a specific transition. To learn more, see [Change the workflow for a work item type](#).

Q: Where should I apply a field rule?

A: When you want a rule to apply to a field throughout the life of the work item, specify it within the **FIELD** definition. For example, a field that is required for a bug that is new and active remains required until the bug is closed.

Otherwise, specify a rule to be evaluated only during a change in state. These rules are defined within the **WORKFLOW** section under the **STATE**, **REASON**, or **TRANSITION** elements. All rules, except for **HELPTTEXT**, can be applied within a **FIELD (Workflow)** element.

Field rules are additive. That is, you can specify four sets of rules for the same field which will all be evaluated by the work item rule-engine.

- Work item type-specific rules apply regardless of the location of a work item in its state model. For example, a `<REQUIRED />` rule performs the following check:

```
"MyField Value" != NULL
```

- State-specific rules are scoped to a work item instance when it is in a certain state. A state-specific rule is enforced when the following condition is true:

```
State field value == "MyState" && "MyField Value" != NULL
```

- Transition-specific rules that you specify for a specific transition are scoped to a work item that is undergoing a certain transition. These rules are enforced when the following conditions are true:

State field value == "ToState" &&

"Previous State Before Edit/New" == "FromState"

&& "MyField Value" != NULL

- Reason-specific rules that you specify for a specific reason are scoped to a particular reason for a particular transition. They are processed when the following conditions are true:

Reason field == "MyReason" &&

State field value == "ToState" &&

"Previous State Before Edit/New" == "FromState" && "MyField Value" != NULL

The following example restricts modification of the customer severity field when the work item is in the Active state.

Copy Code

```
<STATE name="Active">
  <FIELDS>
    <FIELD refname="MyCorp.Severity" >
      <READONLY />
    </FIELD>
  </FIELDS>
</STATE>
```

≡Q: How are rules evaluated? What order is applied?

A: Rules are typically processed in the sequence in which they are listed. However, when you use the **WHEN***, **DEFAULT**, and **COPY** elements, additional behaviors may apply.

You can gain some idea of how rules are evaluated when you apply multiple rules to a field. How rules are evaluated is not completely deterministic. This section describes the expected behavior and interactions when you are using the **WHEN***, **DEFAULT**, and **COPY** rules.

The following steps show, in the correct sequence, the interactions that TFS performs and by the user of a work-item form. Only steps 1, 8, and 13 are performed by the user.

1. From a Team Foundation client—such as Visual Studio, Team Explorer, Team Web Access, or Team Explorer Everywhere—, a user creates a new work item or edits an existing work item.
2. Fill in field defaults. For all fields, use any **DEFAULT** rules that are outside **WHEN*** rules.
3. Copy field values. For all fields, use any **COPY** rules that are outside **WHEN*** clauses.
4. For all fields with a **WHEN** rule that matches, first do **DEFAULT** and then **COPY** rules inside.
5. For all fields with a **WHENNOT** rule that matches, first do **DEFAULT** and then **COPY** rules inside.

TFS always processes **WHEN** rules before **WHENNOT** rules.

6. For all fields that have had their values changed since step 1 and that contain **WHENCHANGED** rules, first do **DEFAULT** and then **COPY** rules inside.
7. Allow the user to start editing.
8. The user changes a field value and then moves focus from the field.
9. Raise any **WHEN** rules for that field that match the new value.
10. Raise any **WHENNOT** rules for that field that match the new value.
11. Raise any **WHENCHANGED** rules for that field that match the new value.
12. Return editing ability to the user.
13. The user saves the changes to the database.
14. For all fields, perform **SERVERDEFAULT** operations that are defined for

the field either directly or indirectly under a **WHEN** or a **WHENNOT** rule.

≡Q: How do keystroke entries in a form affect rule evaluation?

A: The system sets a new value for a field every time a user enters a keystroke within a field through the UI work item form. This means that a conditional rule can occur unexpectedly whenever the rule's prerequisite conditions are met.

In the following XML example, SubStatus will be emptied as you type "Approved Again" into the Status field because the **WHEN*** rule occurs as soon as the user types the letter "e" in Approved, even if the intended final value is not "Approve". For this reason, think carefully when you are using conditional rules.

Copy Code

```
<FIELD refname="MyCorp.SubStatus" />
<WHEN field="MyCorp.Status" value="Approve" >
<EMPTY />
</WHEN>
</FIELD>
```

≡Q: How do I make a field hold a value that is the sum of two other fields?

A: This feature is not natively supported at this time.

≡Q: When would I define field rules using global workflow?

A: Use global workflow only when you are tasked with maintaining many fields with the same definitions and rules across multiple team projects. Similar to global lists, using global workflow can minimize the work required when you have to update field definitions. For more information, see

[Customize global workflow.](#)

See Also

Concepts

[All WITD XML elements reference](#)

Other Resources

[Modify or add a field to support queries, reports, and workflow](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define a default value or copy a value to a field

[See Also](#) [Send Feedback](#)

You can specify a default value for a field, or you can copy the value from another field or system-defined value. Field defaults are rules that control how field values are automatically assigned. You can assign a field default by using one of the following elements: COPY, DEFAULT, and SERVERDEFAULT. You can specify these elements as child elements of the FIELD (Definition) element or the FIELD (Workflow) element.

You add these elements to the definition for a work item type (WIT). To learn more, see [Modify or add a custom work item type \(WIT\)](#).

The COPY and DEFAULT elements fill in values at the start of editing, but the SERVERDEFAULT rule fills in a value when the work item is committed to the database. This action occurs when a user saves changes to a work item, and the user cannot override the value. Such fields usually appear as read-only on the work item form. The SERVERDEFAULT rule is used for fields such as "Last Changed By" and "Last Changed On" to support secure audit trails.

Syntax

You can use the **COPY**, **DEFAULT**, and **SERVERDEFAULT** elements to copy a value from one field to another, copy a server value into a field, or specify a default value to be defined for a field.

Note

If the work item is being changed, the **COPY** or **DEFAULT** elements rule may select either the current or previous value from the source field.

- You can specify a value to copy from another field, the value of the clock, or the name of the current user. If you specify value or field for the from attribute, you must specify the value or field attribute. When a user changes or creates a work item, the **COPY** rule fills in a field value regardless of any value that is already in the field.

```
<COPY for="userGroupName" not="userGroupName" from="value | fi
```

- You can specify a default value for a field by using the **DEFAULT** rule. When a user creates or edits a work item, the **DEFAULT** rule fills in a value if that field is empty. You can specify a value to copy from another field, the date-time stamp that the server clock records, or the name of the current user. If a field already has a value, this rule is ignored.

```
<DEFAULT for="userGroupName" not="userGroupName" from="value |
```

- You can specify a value to copy from the server into a field when the work item is saved. When a user changes the state of a work item, the **SERVERDEFAULT** rule specifies a value to be copied into the current field from a value that is derived from a server component. The **DEFAULT** and

COPY elements fill in values when a user opens a work item to modify it, but the SERVERDEFAULT rule fills in a value when the work item is committed to the database. This action occurs when the user saves the work item. The user cannot override the value. These fields usually appear as read-only on the form. The SERVERDEFAULT rule is used for fields such as "Last Changed By" and "Last Changed On" to support secure audit trails.

Copy Code

```
<SERVERDEFAULT for="userGroupName" not="userGroupName" from="cli
```

Each of these rule elements specifies a from="fromType" attribute that identifies the source of the value. Depending on the fromType value, additional attributes may be required. The following table describes all of the attributes that the COPY, DEFAULT, and SERVERDEFAULT rule elements reference.

Attribute	Description
for	<p>Optional. Specifies the name of a user or group in Team Foundation to whom the rule applies. Valid names consist of a string of text that contains between 1 and 255 characters.</p> <p>Pattern value: <code>^[^\\]+\\</code> <code>[^\\]+\$</code></p> <p>Pattern value example: Domain\UserID</p>
	<p>Optional. Specifies the name of a user or group in Team Foundation to whom the rule does not apply. Valid names</p>

not

consist of a string of text that contains between 1 and 255 characters.

Pattern value: `^[^\\]+\\
[^\\]+$`

Pattern value example:
Domain\UserID

Required. Specifies whether to copy the default value from the value attribute, the field attribute, the system clock, or the current user. If you specify value or field for the from attribute, you must specify the value or field attribute, respectively. You can specify the following values:

- clock: Copies the time from the system clock. Uses the current date and time as the value. No additional attributes are required. For the **COPY** and **DEFAULT** rules, the value is taken from the local computer clock. For the **SERVERDEFAULT** rule, the value is taken from the server clock at commit time. Valid only for DateTime

from

fields.

- **currentuser**: Copies name of the user who is logged on. Use the short user name of the current user as the value. No additional attributes are required. Valid only for string fields.
- **field**: Copies the value that is defined for the field attribute that you specify. Requires a `field="abc"` attribute. By default, if the specified "from" field is empty, nothing is performed. The field attribute is used only for the **COPY** and **DEFAULT** rules.
- **value**: Copies the value of the specified value attribute.

value

Optional. Specifies the value to be copied into the field when value is specified for the from attribute. Valid values consist of a string of text that contains between 1 and 255 characters.

The value to copy can be empty.

Optional. Specifies the name of the field whose value is to be copied into the field when field is specified for the from attribute. You must define this attribute if the from attribute has "field" as its value.

field

The reference name of the field to copy. The reference name must match the reference name as defined in the FIELD (Definition) element. For more information, see

[FIELD \(Definition\) element reference.](#)

Pattern value: `^[a-zA-Z_][a-zA-Z0-9_]*(\.[a-zA-Z0-9_]+)+$`

Pattern value example: `Company.Division.IssueType`

[Back to top](#)

Define a default value

The following example sets P3 as the default for the Priority field.

Copy Code

```
<FIELD refname="MyCorp.Priority" name="Priority" type="String">
<HELPTEXT>Specify the severity of the problem</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="P1"/>
    <LISTITEM value="P2"/>
    <LISTITEM value="P3"/>
  </ALLOWEDVALUES>
  <DEFAULT from="value" value="P3"/>
</FIELD>
```

Clear a field automatically

In the following example, the status field is cleared.

 **Copy Code**

```
<FIELD refname="MyCorp.Status" name="Status" type="String">  
  <COPY from="value" value="" />  
</FIELD>
```

Save a field value

In the following example, the name of the user who changed a work item most recently is saved.

 **Copy Code**

```
<FIELD refname="System.Last Changed By" name="Last Changed By" type="Text"
  <HELPTEXT>The name of the user who most recently modified this item />
  <VALIDUSER group="[Project]\MyProjectMembers" />
  <SERVERDEFAULT from="currentuser" />
</FIELD>
```

Specify the clock as a default

In the following example, the value of a field uses the current date but users can change that value.

Copy Code

```
<FIELD refname="MyCorp.FoundOn" name="Found On" type="DateTime">  
  <HELPTEXT>Defines when a bug was found.</HELPTEXT>  
  <DEFAULT from="clock" />  
</FIELD>
```

Note

For values that contain an apostrophe, such as "Won't Fix", you must use double quotation marks in the XML, as the following example shows:

```
<LISTITEM value="Won't Fix"/>
```

See Also

Concepts

[All FIELD XML elements reference](#)

Other Resources

[Apply a rule to a work item field](#)

Modify or add a field to support queries, reports, and workflow



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define pick lists

[See Also](#) [Send Feedback](#)

You can enumerate a set of values for a field by defining a pick list as part of its definition or at some point during the workflow. You can specify that the list can contain only allowed values, cannot contain prohibited values, or can suggest values. If you suggest values, users can specify a value other than those in the pick list.

You can also define dependent pick lists, in which you define two or more lists for a field but only one list appears at run time based on the evaluation of a conditional rule.

In this topic

-

[Syntax structure](#)

- [Syntax structure for GLOBALLIST and LISTITEM Elements](#)
- [Allow an existing value](#)
- [Specify a set of allowed values](#)
- [Define dependent pick lists](#)

Note

To add or modify a pick list for a FIELD definition, use the **witadmin** command-line tool to import and export the definition for the work item type. See [Import, export, and manage work item types \[witadmin\]](#).

Syntax structure

You can use the ALLOWEDVALUES, SUGGESTEDVALUES, and PROHIBITEDVALUES elements to specify a list of values that a user must specify, may specify, or must not specify as a value for a field. If you use each of these elements, you can enumerate a list of items or specify a global list. You can use the ALLOWEXISTINGVALUE to allow a field to store an existing value if you remove an item from the pick list.

You can specify these elements as child elements of the FIELD (Definition) or FIELD (Workflow) element.

- You use ALLOWEDVALUES to define a list of values that users can specify in a work item form or the query editor. Users must specify one of the values in the GLOBALLIST or the set of LISTITEM entries.

 Copy Code

```
<ALLOWEDVALUES for="userGroupName" not="userGroupName" expandit
  <GLOBALLIST name="globalListName" />
  <LISTITEM value="Name1" />
  <LISTITEM value="Name2" />
  <LISTITEM value="Name3" />
  . . .
</ALLOWEDVALUES>
```

- You use PROHIBITEDVALUES to define a list of values that a field cannot contain. Users cannot save a work item if the field contains a prohibited value. You use this element if you want to restrict the use of a value that was previously allowed but is no longer valid.

 Copy Co

```
<PROHIBITEDVALUES for="userGroupName" not="userGroupName" expansi
  <GLOBALLIST name="globalListName" />
  <LISTITEM value="Name1" />
  <LISTITEM value="Name2" />
  <LISTITEM value="Name3" />
  . . .
```

</PROHIBITEDVALUES>

- You use SUGGESTEDVALUES to define a list of values that a field can contain. Users can specify other values in addition to those that you suggest.

 Copy Co

```
<SUGGESTEDVALUES for="userGroupName" not="userGroupName" expand:
  <GLOBALLIST name="globalListName" />
  <LISTITEM value="Name1" />
  <LISTITEM value="Name2" />
  <LISTITEM value="Name3" />
  .
  .
  .
</SUGGESTEDVALUES>
```

For each of these elements, you can specify one or more of the attributes that the following table describes:

Attribute	Description
for	Optional. Specifies the name of a user or group in Team Foundation to whom the rule applies. Valid names consist of a string of text that contains between 1 and 255 characters. Pattern value: $^{\wedge}[\wedge\backslash]^+\backslash[\wedge\backslash]^+{\$}$ Pattern value example: Domain\UserID

not

Optional. Specifies the name of a user or group in Team Foundation to whom the rule does not apply. Valid names consist of a string of text that contains between 1 and 255 characters.

Pattern value: `^[^\\]+\\
[^\\]+$`

Pattern value example:
Domain\UserID

expanditems

Optional. Specifies whether a group identified in the **LISTITEM** element should be expanded to include subordinate groups in the list. The default value is true.

filteritems

Optional. Specifies that the list includes only the members of groups, not group names. The only valid value of this attribute is `excludegroups`.

For more information, see

[Expand list items and exclude groups from lists.](#)

▣ Syntax structure for **GLOBALLIST** and **LISTITEM** elements

You can use the **GLOBALLIST** and **LISTITEM** elements to enumerate a list of items that can be updated globally or that are specific to a single field.

- You use **GLOBALLIST** to define a set of **LISTITEM** elements that is stored for a team project collection and that all team projects in that collection can use. **GLOBALLIST** is a required child element of the **GLOBALLISTS** element and an optional child element of the **ALLOWEDVALUES**, **SUGGESTEDVALUES**, and **PROHIBITEDVALUES** elements. You can define a global list within a work item definition, a global list definition, or a global workflow.

Copy Code

```
<GLOBALLIST name="globalListName">
  <LISTITEM value="Name1" />
  <LISTITEM value="Name2" />
  <LISTITEM value="Name3" />
  .
  .
  .
</GLOBALLIST>
```

globalListName: A string of text that contains between 1 and 255 characters.

Important

If you define a global list in an XML file that defines a type of work item, you must give the list a unique name. Otherwise, you might overwrite an existing list that was defined for the collection.

- You use **LISTITEM** to enumerate a set of values. **LISTITEM** is a required child element of **GLOBALLIST** and an optional child element of the **ALLOWEDVALUES**, **SUGGESTEDVALUES**, and

PROHIBITEDVALUES elements.

 **Copy Code**

```
<LISTITEM value="listName" />
```

listName: A string of text that contains between 1 and 255 characters.

■ Allow an existing value

You can use the `ALLOWEXISTINGVALUE` element to allow a field to maintain existing values, after you specify a pick list of items by using the `ALLOWEDVALUES` element. If you do not specify the `ALLOWEXISTINGVALUE` element, you force the user, at edit time, to specify one of the current valid values for that field. The `ALLOWEXISTINGVALUE` element modifies only those elements in the same block.

Copy Code

```
<ALLOWEXISTINGVALUE />
```

You specify this element under the **FIELD** (Workflow) element to apply it to the rules that are defined for the field. This application includes the rules that the **STATE**, **TRANSITION**, **DEFAULTREASON**, and **REASON** elements of the field specify. You specify this element under the **STATE** element to apply it not only to the rules that are defined in the state for the field but also to the rules that are defined for the field for all transitions into the state.

You specify this element under the **TRANSITION** element to apply it not only to the rules that are defined in the transition for the field but also to the rules that are defined for the field in the **REASON** and **DEFAULTREASON** elements.

Specify a list

Field lists are composed of individual list items. Each field list must contain at least one item.

Note

Global lists must not include project-scoped groups because they are not scoped to a project.

To specify items in a field list, use the `<LISTITEM value="">` element. You can specify a string, a user name, or a group name.

Copy Code

```
<LISTITEM value="Emergency"/>
<LISTITEM value="Major"/>
<LISTITEM value="Minor"/>
<LISTITEM value="Domain\joe"/>
<LISTITEM value="[Global]\GlobalGroup" />
<LISTITEM value="[Project]\ProjectGroup" />
```

Note

At run time, items within a list appear alphabetically based on the language of the server that is running Visual Studio Team Foundation Server.

■ Specify a set of allowed values

In this example, the Customer Severity field can have any one of three values: Emergency, Major, and Minor. The field is defined as required with a default value of Minor. At run time, users can specify one of the values in a drop-down list.

Copy Code

```
<FIELD refname="System.Title" name="Title" type="String">
<HELPTEXT>Provide a brief description of the work item</HELPTEXT>
<REQUIRED/>
</FIELD>
<FIELD refname="MyCorp.CusSeverity" name="Customer Severity" type="S
<HELPTEXT>Indicate the severity of the problem</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Emergency">
    <LISTITEM value="Major">
    <LISTITEM value="Minor">
  </ALLOWEDVALUES>
<DEFAULT from="value" value="Minor"/>
</FIELD>
```

Define dependent pick lists

You can define a pick list that is active only when its parent conditional clause is true. In the following example, two sets of pick lists are defined for My Field. At run time, only one list appears based on whether Requirements is assigned to the MyCompany.MyTeam.Discipline field.

Note

The **WHEN** and **WHENNOT** rules in this example can also apply to other rules to specify when those rules should be evaluated. For more information, see

[Assign conditional-based values and rules.](#)

Copy Code

```
<FIELD name="My Field" refname="MyCompany.MyProcess.MyField" type="S">
  <WHEN field="MyCompany.MyTeam.Discipline" value="Requirements">
    <ALLOWEDVALUES>
      <LISTITEM value="Planning" />
      <LISTITEM value="Review" />
    </ALLOWEDVALUES>
  </WHEN>
  <WHENNOT field=" MyCompany.MyTeam.Discipline" value="Requirements">
    <ALLOWEDVALUES>
      <LISTITEM value="Process Management" />
      <LISTITEM value="Planning" />
      <LISTITEM value="Review" />
    </ALLOWEDVALUES>
  </WHENNOT>
</FIELD>
```

See Also

Other Resources

[Work item tracking: Index to XML element definitions](#)

Modify or add a field to support queries, reports, and workflow



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Expand list items and exclude groups from lists

[See Also](#) [Send Feedback](#)

You can expand and filter lists by using the `expanditems` and `filteritems` attributes. You can apply these attributes to these list type elements: **ALLOWEDVALUES**, **SUGGESTEDVALUES**, and **PROHIBITEDVALUES**.

To better understand how these attributes are used to populate a field's drop-down menu, review the examples provided below.

Expand lists and groups

You can assign the values true and false to expanditems; its value is true by default. When expanditems has the value of true, list items that represent groups or global lists are expanded recursively. A group's subgroups are expanded; the subgroups of those subgroups are also expanded, and continues in this pattern. After expansion, list items that represented groups include both groups and users as list item values. If expanditems is set to false, no group or global list expansion is performed.

Exclude groups

You can assign only the value `excludegroups` to the `filteritems` attribute. When this attribute appears, all the list items are evaluated and any groups are removed. Use the `filteritems` attribute to show only users, not groups.

Contents of lists and groups used in the examples

The examples provided in this topic use the following values:

List or group name	List or group contents	Description
	JayHamlin	A team project group that contains the names of three business analyst team members.
[Project]\Business Analysts	PilarAckerman	 Note
	ReshmaPatel	Use the literal prefix [Project] instead of using the actual name of the team project.

Example1\MyTeam	Development	A team project group that contains one team member, juser, and three subgroups, where each subgroup contains the name of one team member.
	devuser	
	Test	
	Test user	
	Program Management	
	pmuser	
Example1\MyReports	juser	A team project group that contains the names of three team members and one subgroup which contains the names of two team members.
	Userone	
	Usertwo	
	Userthree	
	MyRemotes	
	userfour	
BoolValues	userfive	A global list with two entries.
	true	
	false	

Example: Expand lists and exclude groups

In this example, the field contains a string value, a group, and a global list. At the time it is run, the list is expanded and groups are excluded.

Example	Doc list
	string
 Copy Code	true
<pre><ALLOWEDVALUES expanditems="true" filteritems="excludegroups"> <LISTITEM value="string" /> <LISTITEM value="[Project]\Business Analysts" /> <GLOBALLIST name="BoolValues" /> </ALLOWEDVALUES></pre>	false
	JayH
	Pilar
	Resh

Example: Expand lists and groups and do not filter

In this example, the field contains a string value, two groups, and a global list. At the time it is run the list is expanded and groups are not excluded.

Example

**Drop-down
list values**

string

true

false

juser

juser2

devuser

testuser

pmuser

Development

Test

Program

Management

 **Copy Code**

```
<ALLOWEDVALUES expanditems="true">  
  <LISTITEM value="string" />  
  <LISTITEM value="Example1\MyReports"/>  
  <LISTITEM value="Example1\MyTeam" />  
  <GLOBALLIST name="BoolValues" />  
</ALLOWEDVALUES>
```

Example: Do not expand lists or groups, and do not filter

In this example, the field contains a string value, two groups, and a global list. At run time, the list is not expanded and groups are not filtered out. This means that group names are displayed, but not the users within those groups.

Note

The global list name and contents are not displayed.

Example

Drop-down list values

Copy Code

```
<ALLOWEDVALUES expanditems="false">
  <LISTITEM value="string" />
  <LISTITEM value="Example1\MyReports"/>
  <LISTITEM value="Example1\MyTeam" />
  <GLOBALLIST name="BoolValues" />
</ALLOWEDVALUES>
```

string

MyTeam

MyReports

Example: Expand lists and exclude groups and global lists

In this example, the field contains a string value, one group, and a global list. At run time, the list is expanded and groups are filtered out.

Note

MyTeam is a group that is excluded and not expanded, and BoolValues is a global list, so neither one is expanded or shown.

Example

Drop
down
list
value

Copy Code

```
<ALLOWEDVALUES expanditems="true" filteritems="excludegroups">  
  <LISTITEM value="string" />  
  <LISTITEM value="Example\MyTeam" />  
  <GLOBALLIST name="BoolValues" />  
</ALLOWEDVALUES>
```

String

See Also

Concepts

[Define pick lists](#)

[Define global lists](#)

Other Resources

[Apply a rule to a work item field](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Combine list types

[See Also](#) [Send Feedback](#)

You can specify multiple types of lists for a single field. This topic defines how the resulting list of items is determined.

The following descriptions use these conventions:

- All values for an ALLOWEDVALUES list are identified as {set A}.
- All values for a SUGGESTEDVALUES list are identified as {set S}.
- All values for a PROHIBITEDVALUES list are identified as {set P}.

Valid value determination

The valid values allowed for a field are obtained by subtracting {set P} from {set A}. If {set A} has no entries, {set A} is considered to be all possible values. This is because no allowed values have been defined; everything is allowed except those values specifically identified in {set P}.

{Set S} plays no role in determining valid values for a field, but it does help determine the values that display in the drop-down list.

Populating a drop-down list with list values

The following rules use the content of the ALLOWEDVALUES, SUGGESTEDVALUES, and PROHIBITEDVALUES sets to determine the values that populate a drop-down list.

Copy Code

```
If {set S} AND {set A} have no entries
    Result: Empty list
If {set S} has entries and {set A} has no entries
    Result: The values are obtained by subtracting {set P} from
If {set S} AND {set A} have entries
    Result: The list of values are obtained by:
        a. Intersecting {set A} with {set S} to get {interme
        b. Subtracting {set P} from {intermediate set I}
If {set S} has no entries and {set A} has entries
    Result: The list of values are obtained by subtracting {set
```

■ Specify multiple lists

If you specify multiple <ALLOWEDVALUE> sets at a particular point in time (for example, a work item type-wide <ALLOWEDVALUE> set plus a state-scoped <ALLOWEDVALUE> set), the intersection of these multiple sets is used as the final set, {set A}.

If you specify multiple <SUGGESTEDVALUES> sets or <PROHIBITEDVALUES> sets, the union of each of these multiple sets is taken as the final set, {set S} or {set P}, respectively.

▣ See Also

Concepts

[Define pick lists](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define global lists

[See Also](#) [Send Feedback](#)

By using global lists in Team Foundation Server (TFS), you can minimize the work that is required to update a list that appears in the definitions of several work item types (WITs). Global lists are pick lists that you can include within one or more fields and WIT definitions. You can define a global list within a WIT that you add to a team project, as a global list for a team project collection, or within a global workflow. You can share list items among several WITs for a collection by including the list items in one or more **GLOBALLIST** elements.

As you define WITs, you might find that some fields share the same values. Frequently, you can share across several WITs and even across several team projects. Some of these values, such as the build number of nightly builds, change frequently, which requires an administrator to frequently update these lists in many locations. Global lists can be especially useful when a list must be derived from an external system. For example, suppose a company maintains a separate customer database. When you file a bug that a customer discovered, the customer's name is entered into a custom **Found By Customer** field.

You can manage global lists for a collection as an XML file that you can list, import, export, and delete. The name of each global list can have up to 254 Unicode characters and must be unique within a collection.

Note

There are no system-defined global lists, nor predefined global lists in the default process templates that TFS provides.

Requirements

- To export or list global lists, you must be a member of the Project Collection Valid Users group or have your View collection-level information permission set to Allow.
- To define or edit global lists, you must be a member of the Project Administrators group or have the Edit project-level information permission set to Allow.
- To add or modify a global list, use the **witadmin** command-line tool to import and export the definition for global lists. See

[Manage global lists for work item types \[witadmin\]](#). To use a global list, add it to the FIELD definition within a work item type. See [All FIELD XML elements reference](#).

■ Add and manage global lists

A global list is a set of **LISTITEM** elements that is stored and used globally by all team projects in a collection. Global lists are useful for fields that are defined within several types of work items, such as Operating System, Found in Build, and Fixed in Build.

You can define global lists and their items by using one of the following methods:

- **Team project collection:** You can export, modify, delete, and import the global lists that are defined for a team project collection. These global lists are available to all team projects in the collection.
- **Work item type definition:** After a team project is created, you can add the global lists that you want to have available for a type of work item to its definition.
- **Global workflow definition:** After a team project is created, you can add the global lists that you want to have available for all types of work items to the global workflow definition for a team project or collection. For more information, see

[Customize global workflow.](#)

Syntax structure

The following table describes the **GLOBALLIST** and **LISTITEM** elements. You can use these elements to enumerate a list of values that is presented to the user as a pick list or drop-down menu of items.

Element	Syntax	Description
		Defines a set of LISTITEM elements that is stored for a collection and that all team projects in a collection can use.
		globalListName: A string of text that contains between 1 and 255 characters.
	 Copy Code	
GLOBALLIST	<pre><GLOBALLIST name="globalListName"> <LISTITEM> . . . </LISTITEM> </GLOBALLIST></pre>	GLOBALLIST is a required child element of the GLOBALLIST element and an optional child element of the ALLOWEDVALUES , SUGGESTEDVALUES , and PROHIBITEDVALUES elements. For more information, see

[Define pick lists.](#)

LISTITEM

Copy Code

```
<LISTITEM value="Name" />
```

Defines a valid list value.

Note

Global lists must not include project-scoped groups because they are not scoped to a project.

LISTITEM is a required child element of **GLOBALLIST** and an optional child element of the **ALLOWEDVALUES**, **SUGGESTEDVALUES**, and **PROHIBITEDVALUES** elements.

Sample global list

By adding the following syntax, you can define a global list within an XML definition file for a type of work item or a global workflow:

Copy Code

```
<GLOBALLISTS>
  <GLOBALLIST name="name of global list">
    <LISTITEM value="List item 1" />
    <LISTITEM value="List item 2" />
    <LISTITEM value="List item 3" />
    <LISTITEM value="List item 4" />
    . . .
    <LISTITEM value="List item n" />
  </GLOBALLIST>
</GLOBALLISTS>
```

By using the following syntax, you can reference a global list within an XML definition file for a type of work item:

Copy Code

```
<GLOBALLISTS>
  <GLOBALLIST name=" name of global list 1" />
  <GLOBALLIST name=" name of global list 2" />
  . . .
  <GLOBALLIST name=" name of global list n" />
</GLOBALLISTS>
```

For information about the structure and location of definition files for types of work items or global workflow, see

[All WITD XML elements reference](#) or [Global Workflow XML Element Reference](#), respectively.

Sample global List maintained for a project collection

To add a global list to a project collection, you can import the following syntax by using the **witadmin importgloballist** command:

 **Copy Code**

```
<gl:GLOBALLISTS xmlns:gl="http://schemas.microsoft.com/VisualStudio,
  <GLOBALLIST name="NameOfGlobalList">
    <LISTITEM value="ListItem1" />
    <LISTITEM value="ListItem2" />
    <LISTITEM value="ListItem3" />
    <LISTITEM value="ListItem4" />
    .
    .
    .
    <LISTITEM value="ListItemN" />
  </GLOBALLIST>
</gl:GLOBALLISTS>
```

A global list cannot be empty. Each **GLOBALLIST** element must have at least one **LISTITEM** element defined.

Q & A

Q: Are any global lists auto-populated with data?

A: Yes. The global list named Builds - TeamProjectName gets appended each time a build is run. Over time, the list can become very long. Best practice is to routinely remove unused items from the list.

To learn more about using this list, see

[Fields that support integration with test, build, and version control.](#)

See Also

Reference

[Manage global lists for work item types \[witadmin\]](#)

Concepts

[Customize work tracking objects to support your team's processes](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Apply pattern matching to a string field

[See Also](#) [Send Feedback](#)

You can use the **MATCH** element to force values in a field of String type to follow a pattern that you specify. If you define multiple **MATCH** elements, the value will be valid if it matches any of the patterns that you specify for the field. If at least one element succeeds, the field has a valid value.

Note

To add a rule to a FIELD definition, use the **witadmin** command-line tool to import and export the definition for the work item type. See

[Import, export, and manage work item types \[witadmin\]](#).

MATCH Element Syntax Structure

You use the **MATCH** element to enforce basic pattern matching by defining a pattern that values in String type fields must match.

 **Copy Code**

```
<MATCH pattern="patternValue" for="userGroupName" not="userGroupName
```

Attributes

You can specify to which users the **MATCH** rule applies. If you do not define any optional attributes, all valid users and groups in Team Foundation Server must specify a value that matches the pattern.

Attribute	Description
pattern	<p>Required. Enforces basic pattern matching for strings only, and only for strings whose syncnamechanges="false".</p> <p>Valid values for the patternValue are "A", "N", and "X", which denote the following types of characters:</p> <ul style="list-style-type: none">• "A" represents an alphabetical character.• "N" represents a numeric character.• "X" represents any alphanumeric character. <p>All other values are taken as literals. Minimum length: 1; maximum length: 255.</p> <p>Pattern value: <code>^[^\\]*\$</code></p>

Pattern value example:
xxxxx.nn.nn

Optional. Specifies the name of a user or group in Team Foundation to whom the rule applies. Valid names consist of a string of text that contains between 1 and 255 characters.

for

Pattern value: `^[^\\]+\\
[^\\]+$`

Pattern value example:
Domain\UserID

Optional. Specifies the name of a user or group in Team Foundation to whom the rule does not apply. Valid names consist of a string of text that contains between 1 and 255 characters.

not

Pattern value: `^[^\\]+\\
[^\\]+$`

Pattern value example:
Domain\UserID

[Back to top](#)

▣ Pattern Matching Examples

The following examples illustrate successful and unsuccessful pattern matches for a variety of field uses.

▣ Release Number

Pattern: ANN.NN.NN

Validates	R01.03.04 or V05.08.99
Fails validation	1.3.4 or V5.8.99 or v1.3

▣ A Flexible Identifier

Pattern: XXX-XXX

Validates	001-abc or a00-b02
Fails validation	1-abc or 001.abc

▣ Priority

Pattern: PN

Validates	P1 or P5 or P9
-----------	----------------

Fails validation 1 or P10

Match tags are case-insensitive. Therefore, "PN" matches both P1 and p1.

[Back to top](#)

See Also

Other Resources

[Apply a rule to a work item field](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Assign conditional-based values and rules

[See Also](#) [Send Feedback](#)

You can define rules that are run conditionally by using the **WHEN**, **WHENNOT**, **WHENCHANGED**, and **WHENNOTCHANGED** elements. You use these rules to define which elements are run when the defined clause is True. You can define conditions that are based on what value is assigned to a specific field or whether a user modifies a specific field. For example, you can create a dependent pick list to provide detailed security or custom behavior.

Field conditions are additional elements that you list inside a FIELD (Definition) element or the FIELD (Workflow) element. For more information about these elements, see [FIELD \(Definition\) element reference](#) and [FIELD \(Workflow\) element](#).

The following code is a simple example of the **WHEN** clause:

```
<FIELD . . . >  
  
    <WHEN field="referenceName" value="yyy">  
  
</FIELD>
```

This clause means that anything within this FIELD element is applicable as long as the field refname has the value "yyy". The field must be a valid field reference name. For more information, see [Naming conventions for work item tracking objects](#).

Note

The value attribute is case-insensitive. Therefore, if the field reference name holds "YYY", matches include the values "yyy" and "YYY".

In this topic

- [Syntax Structure for Conditional Elements](#)
- [Defining a Dependent Required Field](#)
- [Defining a Conditional Pick List](#)
- [Defining a Field When the User Changes Another Field \(WHENCHANGED\)](#)
- [Defining a Field Value Based on a User Not Modifying a Field \(WHENNOTCHANGED\)](#)

Syntax Structure for Conditional Elements

The following table describes conditional rules that you can specify as child elements of the FIELD (Definition) element or FIELD (Workflow) element. These elements accept one or more of the following attributes:

- **field:** A string that describes the field. Must contain 1 to 255 characters.
- **value:** When the specified field has this value, the rules in the WHEN and WHENNOT elements are applied to the current field.

Element	Syntax
WHEN	<div style="text-align: right; margin-bottom: 10px;">  Copy Code </div> <pre> <WHEN field="fieldReferenceName" value="value <ALLOWEDVALUES> . . . </ALLOWEDVALUES> <ALLOWEXISTINGVALUE> . . . <ALLOWEXISTING <CANNOTLOSEVALUE> . . . </CANNOTLOSEVALUE <COPY> . . . </COPY> <DEFAULT> . . . </DEFAULT> <EMPTY> . . . </EMPTY> <FROZEN> . . . </FROZEN> <MATCH> . . . </MATCH> <NOTSAMEAS> . . . </NOTSAMEAS> <PROHIBITEDVALUES> . . . </PROHIBITEDVALU <READONLY> . . . </READONLY> <REQUIRED> . . . </REQUIRED> <SERVERDEFAULT> . . . </SERVERDEFAULT> <SUGGESTEDVALUES> . . . </SUGGESTEDVALUES <VALIDUSER> . . . </VALIDUSER> </WHEN> </pre>

WHENNOT

Copy Code

```
<WHENNOT field="fieldReferenceName" value="va
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>
  <ALLOWEXISTINGVALUE> . . . <ALLOWEXISTING
  <CANNOTLOSEVALUE> . . . </CANNOTLOSEVALUE
  <COPY> . . . </COPY>
  <DEFAULT> . . . </DEFAULT>
  <EMPTY> . . . </EMPTY>
  <FROZEN> . . . </FROZEN>
  <MATCH> . . . </MATCH>
  <NOTSAMEAS> . . . </NOTSAMEAS>
  <PROHIBITEDVALUES> . . . </PROHIBITEDVALU
  <READONLY> . . . </READONLY>
  <REQUIRED> . . . </REQUIRED>
  <SERVERDEFAULT> . . . </SERVERDEFAULT>
  <SUGGESTEDVALUES> . . . </SUGGESTEDVALUES
  <VALIDUSER> . . . </VALIDUSER>
</WHENNOT>
```

WHENCHANGED

 **Copy Code**

```
<WHENCHANGED field="fieldReferenceName" >  
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>  
  <ALLOWEXISTINGVALUE> . . . <ALLOWEXISTING  
  <CANNOTLOSEVALUE> . . . </CANNOTLOSEVALUE  
  <COPY> . . . </COPY>  
  <DEFAULT> . . . </DEFAULT>  
  <EMPTY> . . . </EMPTY>  
  <FROZEN> . . . </FROZEN>  
  <MATCH> . . . </MATCH>  
  <NOTSAMEAS> . . . </NOTSAMEAS>  
  <PROHIBITEDVALUES> . . . </PROHIBITEDVALU  
  <READONLY> . . . </READONLY>  
  <REQUIRED> . . . </REQUIRED>  
  <SERVERDEFAULT> . . . </SERVERDEFAULT>  
  <SUGGESTEDVALUES> . . . </SUGGESTEDVALUES  
  <VALIDUSER> . . . </VALIDUSER>  
</WHENCHANGED>
```

Copy Code

WHENNOTCHANGED

```
<WHENNOTCHANGED field="fieldReferenceName">
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>
  <ALLOWEXISTINGVALUE> . . . <ALLOWEXISTING
  <CANNOTLOSEVALUE> . . . </CANNOTLOSEVALUE
  <COPY> . . . </COPY>
  <DEFAULT> . . . </DEFAULT>
  <EMPTY> . . . </EMPTY>
  <FROZEN> . . . </FROZEN>
  <MATCH> . . . </MATCH>
  <NOTSAMEAS> . . . </NOTSAMEAS>
  <PROHIBITEDVALUES> . . . </PROHIBITEDVALU
  <READONLY> . . . </READONLY>
  <REQUIRED> . . . </REQUIRED>
  <SERVERDEFAULT> . . . </SERVERDEFAULT>
  <SUGGESTEDVALUES> . . . </SUGGESTEDVALUES
  <VALIDUSER> . . . </VALIDUSER>
</WHENNOTCHANGED>
```

The following table describes how each optional, conditional-based rule is applied to the parent field when the conditional clause that you specify by using a **WHEN**, **WHENNOT**, **WHENCHANGED**, or **WHENNOTCHANGED** element is true. For more information, see

[Apply a rule to a work item field.](#)

Element	Description
ALLOWEDVALUES	The parent field must have a value that comes from the specified list of values.

ALLOWEXISTINGVALUE

The value of the parent field that already exists will be allowed, even if it violates other rules. This element is not applicable if the value of the parent field is changed.

CANNOTLOSEVALUE

Users can change the value of the parent field to NULL, but they cannot change it to any other value.

COPY

The value of a third field is automatically copied into the parent field. You specify the third field in the **COPY** element.

This element

DEFAULT	specifies the default value of the parent field.
EMPTY	The parent field must not contain a value.
FROZEN	The parent field is frozen. When a field is frozen, you can change its value to NULL, but you cannot change it to any other value.
MATCH	The value of the parent field must match the pattern that you specify.
NOTSAMEAS	The value of the parent field cannot match the value of a third field. You specify the third field in the NOTSAMEAS element.

PROHIBITEDVALUES

The parent field cannot contain any values in the enumerated list.

READONLY

The parent field is read-only.

REQUIRED

The parent field must contain a value that is not NULL.

SERVERDEFAULT

The parent field takes its value from the specified server component. The valid server components are clock, which is the time when the work item is updated, and currentuser, which is the identity of the user who updated the

work item.

SUGGESTEDVALUES

The enumerated list contains suggested values for the parent field.

VALIDUSER

Only the users whom you specify can modify the parent field.

[Back to top](#)

Defining a Dependent Required Field

You can specify that a field is required only when another field contains a specific value. In the following example, when a customer reports a bug, a customer severity must be specified. If the bug was not reported by a customer, a customer severity is not required.

Copy Code

```
<FIELD refname="MyCorp.Severity" name="Customer Severity" type="Stri
  <ALLOWEDVALUES>
    <LISTITEM value="Blocking" />
    <LISTITEM value="Major" />
    <LISTITEM value="Minor" />
  </ALLOWEDVALUES>
  <WHEN field="MyCorp.CustomerReported" value="true">
    <REQUIRED />
  </WHEN>
</FIELD>
```

Defining a Conditional Pick List

The following example demonstrates a conditional pick list in which the allowed values for the Problem Type field are limited, based on whether the value of the ProblemCharacteristic field is set to Documentation.

Copy Code

```
<FIELD refname="MyCorp.ProblemType" name="Problem Type" type="String"
  <WHEN field="MyCorp.ProblemCharacteristic" value="Documentation"
    <ALLOWEDVALUES>
      <LISTITEM value="Spelling Error" />
      <LISTITEM value="Bad Format" />
      <LISTITEM value="Missing Info" />
    </ALLOWEDVALUES>
  </WHEN>
</FIELD>
```

Defining a Field When the User Changes Another Field (WHENCHANGED)

In the following example, when a user changes the value of the MyCorp.State field, the MyCorp.StateDate field is set to the current date and time, as the server clock shows.

 **Copy Code**

```
<FIELD refname="MyCorp.StateDate" name="Date Of Last State Change" type="Date" />
  <WHENCHANGED field="MyCorp.State">
    <COPY from="clock" />
  </WHENCHANGED>
</FIELD>
```

In the following example, when a user changes the value of the MyCorp.State field, the value of the MyCorp.Status field is cleared.

 **Copy Code**

```
<!-- Clear the status field whenever someone changes the state -->
<FIELD refname="MyCorp.Status" name="Status" type="String">
  <WHENCHANGED field="MyCorp.State">
    <COPY from="value" value="">
  </WHENCHANGED>
</FIELD>
```

Defining a Field Value Based on a User Not Modifying a Field (WHENNOTCHANGED)

In the following example, when a user does not change the value of the MyCorp.State field, the MyCorp.StateDate field becomes read-only.

 **Copy Code**

```
<FIELD refname="MyCorp.StateDate" name="Date Of Last State Change" t
<!-- Make the StateDate field read-only when the State field is not
  <WHENNOTCHANGED field="MyCorp.State">
    <READONLY />
  </WHENNOTCHANGED>
</FIELD>
```

See Also

Other Resources

[Apply a rule to a work item field](#)

[Apply a rule to a work item field](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

All WORKFLOW XML elements reference

[See Also](#) [Send Feedback](#)

You can use the information in this topic as a quick reference to all the elements and main attributes that control the workflow for a type of work item. You specify these elements in the WORKFLOW element container, which is the second major section of the definition of a work item type. You use the **STATES** and **TRANSITIONS** child elements to define the different states that a user can specify for a type of work item and which transitions are allowed between states. For more information about how to define these elements, see [Change the workflow for a work item type](#)

To modify the workflow, you modify the definition for a work item type. See [Modify or add a custom work item type \(WIT\)](#).

WORKFLOW example

The following example shows the overall structure of the **WORKFLOW** element. You specify all the states to which a user can first assign a work item type within a **STATES** container. Next, you specify all the transitions that are allowed from one state to another. Each state corresponds to a **STATE** element, and each transition corresponds to a **TRANSITION** element. Within each **STATES** and **TRANSITIONS** container element, you can define the states and transitions in any sequence that you want.

For each transition, you specify a set of reasons for changing the state of the work item, including a default reason. You can assign values for each state, transition, or reason, and you can place conditions on the values of one or more fields by using the **FIELD** (Workflow) element. In addition, you can trigger an action to occur during a transition by specifying the **ACTION** element.

 **Copy Code**

```
<WORKFLOW>
<STATES>
  <STATE value="Active">
    <FIELDS> . . . </FIELDS>
  </STATE>
  <STATE value="Resolved">
    <FIELDS> . . . </FIELDS>
  </STATE>
  <STATE value="Closed" />
</STATES>
<TRANSITIONS>
  <TRANSITION from="" to="Active">
    <REASONS>
      <DEFAULTREASON value="New" />
    </REASONS>
    <FIELDS> . . . </FIELDS>
  </TRANSITION>
  <TRANSITION from="Active" to="Resolved">
    <REASONS> . . . </REASONS>
    <FIELDS> . . . </FIELDS>
    <ACTIONS > . . . </ACTIONS >
  </TRANSITION>
  <TRANSITION from="Resolved" to="Closed">
    <REASONS> . . . </REASONS>
```

```
<FIELDS> . . . </FIELDS>
  <ACTIONS > . . . </ACTIONS >
</TRANSITION>
<TRANSITION from="Resolved" to="Active">
  <REASONS> . . . </REASONS>
  <FIELDS> . . . </FIELDS>
</TRANSITION>
<TRANSITION from="Active" to="Closed ">
  <REASONS> . . . </REASONS>
  <FIELDS> . . . </FIELDS>
</TRANSITION>
<TRANSITION from="Closed" to="Active">
  <REASONS> . . . </REASONS>
  <FIELDS> . . . </FIELDS>
</TRANSITION>
</TRANSITIONS>
</WORKFLOW>
```

Syntax structure

By using the elements that the following table describes, you can specify to which states a team member can set a work item of a particular type. In the WORKFLOW section of the definition, you define states first, and then you define transitions. For more information, see [Change the workflow for a work item type](#).

Element	Description and syntax	Required?
ACTION	<p>Defines a text string that corresponds to an action to be performed when the system for tracking work items calls the <code>WorkItem.GetNextState</code> method to get the post-action state of the work item.</p> <p> Copy Code</p> <pre><ACTION value="NameOfAction" /></pre> <p>For more information, see Automate field assignments based on State, Transition, or Reason</p>	

Optional

ACTIONS

Defines a collection of ACTION elements.

 **Copy Code**

```
<ACTIONS>  
  <ACTION>. . . </ACTION>
```

</ACTIONS>

Optional

DEFAULTREASON

Defines the most common cause of a team member changing a work item from one particular state to another particular state.

 **Copy Code**

```
<DEFAULTREASON value="ValueOfDefaultReason">  
  <FIELDS>. . . </FIELDS>  
</DEFAULTREASON>
```

Required

FIELD

Specifies a field for a type of work item and the rules and conditions that will be applied to that field when a team member changes the state of a work item of that type and specifies a reason for the change.

 **Copy Code**

```
<FIELD refname="FieldReferenceName">  
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>  
  <ALLOWEXISTINGVALUE />  
  <CANNOTLOSEVALUE />  
  <COPY />  
  <DEFAULT />  
  <EMPTY />  
  <FROZEN />  
  <MATCH />  
  <NOTSAMEAS />  
  <PROHIBITEDVALUES /> . . . </PROHIBITEDVALUES>  
  <READONLY />  
  <SERVERDEFAULT />  
  <SUGGESTEDVALUES /> . . . </SUGGESTEDVALUES>  
  <VALIDUSER />  
  <WHEN>> . . . </WHEN>  
  <WHENNOT> . . . </WHENNOT>
```

```
<WHENCHANGED> . . . </WHENCHANGED>  
<WHENNOTCHANGED> . . . </WHENNOTCHANGED>  
<FIELD>
```

For more information, see [FIELD \(Workflow\) element](#).

Optional

FIELDS

Specifies a collection of FIELD elements.

 **Copy Code**

```
<FIELDS>  
  <FIELD . . . </FIELD>  
</FIELDS>
```

Optional

REASON

Defines an additional explanation for why a team member changed the state of a work item.

 **Copy Code**

```
<REASON value="NameOfReason">  
  <FIELDS>. . . </FIELDS>  
</REASON>
```

Optional

REASONS

A collection of one DEFAULTREASON and optional REASON elements that explain why a team member changed the state of a work item.

 **Copy Code**

```
<REASONS>
  <DEFAULTREASON>. . . </DEFAULTREASON>
  <REASON>. . . </REASON>
</REASONS>
```

STATE

Defines a valid state for the work item type. This element can contain a FIELDS element, which references rules and conditions that will be applied to a specific field when a team member changes the state of a work item of that type.

 **Copy Code**

```
<STATE value="NameOfState">
  <FIELDS>. . . </FIELDS>
</STATE>
```

Required

STATES

Specifies a collection of STATE elements that define the valid states to which a user can assign a work item of that type.

 **Copy Code**

```
<STATES>
  <STATE>. . . </STATE>
</STATES>
```

Required

TRANSITION

Specifies a valid progression or regression from one state to another for work items of a particular type.

 **Copy Code**

```
<TRANSITION from="NameOfStartingState"
```

```
to="NameOfEndingState"  
for="UserOrGroupName"  
not="UserOrGroupName">  
  <ACTIONS>. . . </ACTIONS>  
  <REASONS>. . . </REASONS>  
  <FIELDS>. . . </FIELDS>  
</TRANSITION>
```

For more information, see TRANSITION Element.

Required

TRANSITIONS

Specifies a collection of TRANSITION elements.

 **Copy Code**

```
<TRANSITIONS>  
  <TRANSITION>. . . </TRANSITION>  
</TRANSITIONS>
```

Required

WORKFLOW

Specifies the collection of STATES and TRANSITIONS container elements that together define the workflow for the type of work item.

 **Copy Code**

```
<WORKFLOW>  
  <STATES>. . . </STATES>  
  <TRANSITIONS>. . . </TRANSITIONS>  
</WORKFLOW>
```

Required

See Also

Reference

[FIELD \(Workflow\) element](#)

Concepts

[Change the workflow for a work item type](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

FIELD (Workflow) element

[See Also](#) [Send Feedback](#)

You use the **FIELD** (Workflow) element to specify the rules and conditions that apply to a field during a state change or workflow transition. The rule is applied based on where the **FIELD** (Workflow) element appears under the STATE, TRANSITION, DEFAULTREASON, or REASON element of which its parent FIELDS element is a child. To learn more, see Q: Where should I apply a field rule?

Note

For information about the **FIELD** (Definition) element, which you use to define fields for a type of work item, see [FIELD \(Definition\) element reference](#).

```
<FIELD refname="fieldReferenceName">
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>
  <ALLOWEXISTINGVALUE />
  <CANNOTLOSEVALUE />
  <COPY />
  <DEFAULT />
  <EMPTY />
  <FROZEN />
  <MATCH />
  <NOTSAMEAS />
  <PROHIBITEDVALUES /> . . . </PROHIBITEDVALUES>
  <READONLY />
  <REQUIRED />
  <SERVERDEFAULT />
  <SUGGESTEDVALUES /> . . . </SUGGESTEDVALUES>
  <VALIDUSER />
  <WHEN> . . . </WHEN>
  <WHENNOT> . . . </WHENNOT>
  <WHENCHANGED> . . . </WHENCHANGED>
```

<WHENNOTCHANGED> . . . </WHENNOTCHANGED>
<FIELD>

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Attributes

Attribute	Description
refname	Required. The reference name of the field where the rules and conditions are applied. The reference name must match the reference name defined in the field definition's FIELD (Definition) element. For more information, see FIELD (Definition) element reference .

Child Elements

Element	Description
	Optional. Defines a list of allowed values for the field. Allowed

ALLOWEDVALUES

values are values that are available for selection in a field list on work item forms and in the query builder. You must select from one of these values.

ALLOWEXISTINGVALUE

Optional. Defines the field to allow existing values. This element allows the field values that already exist to be used, even if they are not valid. All new field values must be valid.

Optional.

CANNOTLOSEVALUE

Defines the field as cannot lose value. This element keeps the current field value and it cannot be cleared or made empty.

COPY

Optional. Specifies another field that contains a value to be copied into the current field.

DEFAULT

Optional. Defines a default value for the field.

EMPTY

Optional. Defines the field as empty.

Optional.

FROZEN

Defines the field as frozen. A frozen field cannot be changed to any non-empty value after changes are committed. However, you can manually clear the field, save the work item, and then specify a different value.

MATCH

Optional. Defines a pattern for the field that the field value must match.

NOTSAMEAS

Optional. Specifies another field, the value of which cannot be

	identical to the value of the current field.
PROHIBITEDVALUES	Optional. Defines a list of prohibited values for the field.
READONLY	Optional. Defines the field as read-only.
REQUIRED	Optional. Defines the field as required.
SERVERDEFAULT	Optional. Specifies a server component that will provide the value for the field.
	Optional. Defines a list of

SUGGESTEDVALUES

suggested values for the field. Suggested values are values that are available for selection in a field list on work item forms and in the query builder. You can enter other values additionally to the ones in the list.

VALIDUSER

Optional. Specifies that the list of allowed values must consist only of valid users of the system.

Optional. Specifies one or more rules to apply to the

WHEN

current field
when
another
field has a
specific
value.

WHENCHANGED

Optional.
Applies one
or more
rules to the
current field
when a
specific
field's value
is changed.

WHENNOT

Optional.
Applies one
or more
rules to the
current field
when
another
field does
not have a
specific
value.

WHENNOTCHANGED

Optional.
Applies one
or more
rules to the
current field
when a
specific

field's value
is not
changed.

Parent Elements

Element	Description
FIELDS	A collection of FIELD (Workflow) elements that reference a field that is defined for a type of work item and that specify the rules and conditions that apply to the field. The reference is based on the STATE , TRANSITION , DEFAULTREASON , or REASON element of which its parent FIELDS element is a child.

Remarks

You must define at least one child element for the FIELD (Workflow) element.

FIELD (Workflow) is a required child element of FIELDS (Workflow).

`minOccurs="1"`

`maxOccurs="unbounded"`

See Also

Reference

[FIELD \(Definition\) element reference](#)

Concepts

[Define and modify work item fields](#)

Change the workflow for a work item type

Other Resources

[Work item tracking: Index to XML element definitions](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

TRANSITION XML element

[Example](#) [See Also](#) [Send Feedback](#)

You use the **TRANSITION** element to specify a valid progression or regression from one state to another for a type of work item. The **TRANSITION** element is a required child element of the **TRANSITIONS** element.

To modify the workflow, you modify the definition for a work item type. See [Modify or add a custom work item type \(WIT\)](#).

Schema Hierarchy

WITD

WORKITEMTYPE

WORKFLOW

TRANSITIONS

TRANSITION

```
<TRANSITION from="NameOfStartingState" to="NameOfEndingState" for="l
  <ACTIONS> . . . </ACTIONS>
  <REASONS> . . . </REASONS>
  <FIELDS> . . . </FIELDS>
</TRANSITION>
```

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Attributes

Attribute	Description
from	Required. The name of the state from which the work item is transitioning.
to	Required. The name of the state to which the work item is transitioning.
	Transition attribute.
for	Optional. The name of a user or group who is allowed to perform the transition.
	Transition attribute.
not	Optional. The name of a user or group who is restricted from performing the transition.

Child Elements

Element	Description
ACTIONS	Optional. Defines a collection of ACTION elements that each define an action string that can be used when calling the WorkItem.GetNextState method to get the post-action state of the work item.
REASONS	Required. A collection of DEFAULTREASON and REASON elements that define valid reasons the work item can complete the transition.
FIELDS (Workflow)	Optional. A collection of FIELD (Workflow) elements that reference the field definitions that are defined for the type of work item.

Parent Elements

Element	Description
	Required. A collection of TRANSITION

TRANSITIONS elements that define
the valid state
transitions for the
work item type.

Remarks

TRANSITION is a required child element of TRANSITIONS.

You must define exactly one transition to move the work item from nothing (from="") to a named state such as Active. This transition identifies the default state for a new work item.

minOccurs="1"

maxOccurs="unbounded"

All valid transitions between two states must be specified. If no transition is specified, then by default no transition is allowed.

Additionally, you can optionally use the attributes for and not in the transition element of workflow to refine who is and who is not able to perform a transition. When you do this, denies takes precedence over allows. If neither of these attributes is specified, anyone can modify the work item.

Multiple groups are supported only by creating a parent group and specifying that parent group in the TRANSITION element. To learn more about the for and not attributes, see [Apply a rule to a work item field](#).

Example

In the following example, the reasons are defined for the transition from the Active to the Resolved workflow state.

 **Copy Code**

```
<TRANSITION from="Active" to="Resolved">
  . . .
  <REASONS>
    <DEFAULTREASON value="Fixed"/>
    <REASON value="Deferred"/>
    <REASON value="Duplicate"/>
    <REASON value="As Designed"/>
    <REASON value="Unable to Reproduce"/>
    <REASON value="Obsolete"/>
  </REASONS>
  . . .
</TRANSITION>
```

In the following rule, the ability to transition a work item from the Resolved to the Completed state is restricted to all project testers, except for new testers who have just joined the team.

 **Copy Code**

```
<TRANSITION from="Resolved" to="Complete" for="[project]\AllTesters'
</TRANSITION>
```

See Also

Concepts

[Change the workflow for a work item type](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Automate field assignments based on State, Transition, or Reason

[See Also](#) [Send Feedback](#)

You may want to automatically transition work items from one state to another state based on an event that occurs elsewhere in Visual Studio Application Lifecycle Management (ALM) or an event that occurs outside of Visual Studio ALM. For example, you may want to automate the transition of a bug from one state to another based on what occurs in a call tracking tool. The work item type model and the Work Item Tracking API are extended to support automatic transitioning of work items by other systems.

If you have code that changes the state of a work item, you can generalize that code by associating your action with the appropriate state transition by using the **ACTION** element. You can pass the value of your action to the [WorkItem.GetNextState] method to get the post-action state of that work item. The version control check-in dialog box uses this method to resolve bugs and close tasks that are associated with the check-in.

ACTION is an optional child element of ACTIONS.

Note

The Work Item Tracking API is part of the Visual Studio ALM SDK, as described by the following page on the Microsoft website:

[Extending Team Foundation.](#)

For example, a tool is preset to automatically transition a work item to "Resolved" after the user checks in a change. However, as an integration provider, you do not know what state the work item type author has declared as "Resolved". The author might mean Resolved, Closed, Completed, Ready For Test, Include In Build, and so on. One option would be to require all work item

type authors to include a state explicitly named "Resolved".

That solution is too restrictive. It is also poor from an international perspective because it does not enable localization of states. Instead, system integrators can declare an action such as "Check-in" or "Complete" that induces an automatic transition for work items. The work item type author would then declare this action on the appropriate transition.

In this topic

- [Syntax for the ACTION Element](#)
- [Required Steps to Support Automation](#)
- [Associating a State Transition with an Action](#)
- [Transition Action Details](#)
- [Auto-Transition Error Checking](#)

■ Syntax for the ACTION element

The following syntax is used for the **ACTION** element. The value attribute specifies the name of the action and is required. You should follow the same naming conventions for actions as for field reference names. For example, Team Foundation version control uses Microsoft.VSTS.Actions.CheckIn to identify the transition that is appropriate for work items that are associated with the check-in. For more information, see

[Naming conventions for work item tracking objects.](#)

 **Copy Code**

```
<ACTION value="NameOfAction" />
```

```
minOccurs="0"
```

```
maxOccurs="unbounded"
```

Required steps to support automation

To integrate a tool with Work Item Tracking, the tool must perform the following steps:

1. Determine what state the work item should be transitioned to when the action is performed.
2. Set the work item to the "to" state.

The Work Item Tracking API provides methods for performing these steps. The Work Item Tracking API is part of the Visual Studio ALM SDK. For more information, see the following page on the Microsoft website:

[Team Foundation Server SDK.](#)

Note

The transaction action that caused a particular state transition to occur is not recorded. If you must track which action caused a transition, you can specify an additional work item field to track it, or you can define a Reason value.

[Back to top](#)

Associate a State transition with an Action

You can use state transition actions to automate transitions of work items at various points in their workflow. For example, a Team Foundation Server version control system must support automatic transitions of work items at check-in time. To support this, a "microsoft.vsts.actions.checkin" action has been defined.

A work item type author can define a "Defect" work item type that has a state called "Working" and use this work item when a developer is making changes. The work item type author can define another state called "Ready To Build," which means that the developer has declared the code that was affected by the defect to be ready for the nightly build.

The author can automatically transition the work item from the "Working" state to the "Ready To Build" state during a check-in operation by declaring the following:

Copy Code

```
<TRANSITION from="Working" to="Ready To Build">
  <ACTIONS>
    <ACTION value="microsoft.vsts.actions.checkin"/>
  </ACTIONS>
</TRANSITION>
```

[Back to top](#)

Transition action details

Use state transition actions to automate transitions of work items at various points in their workflow. You might consider the following usage details about transition actions:

- Transition actions are optional. If the current state of the work item instance has an action entry for the specified action, it returns the "to" state. If not, the return value is Null. Integrations should handle Null return values gracefully. That is:
 - Do not fail.
 - Leave a trace or log that indicates that the integration did not auto-transition because it required an action that was not found.
- For each work item type, actions must be unique for From State/Action pairs. This means that work item type authors cannot specify multiple "to" states for the same action.
- However, multiple actions on the same transition are supported to allow for multiple auto-transition integrations as shown in the following example:

Copy Code

```
<TRANSITION from="Working" to="Ready To Build">
  <ACTIONS>
    <ACTION value="Microsoft.VSTS.Actions.Checkin"/>
    <ACTION value="ADatum.Actions.Complete"/>
  </ACTIONS>
</TRANSITION>
```

- Action names are programmatic names for which you can use only English characters.
- Action names should follow the same reference namespace convention as field reference names, to avoid action name conflicts between vendors and customers. However, this convention is not enforced by the tool. Visual

Studio ALM uses **Microsoft.VSTS.Actions.<your action>**.

Auto-transition error checking

Integrators can try two types of auto-transitions. The first is an auto-transition that occurs because of a user action. The second is an auto-transition that occurs by unattended automation, such as a nightly build.

- **User action auto-transitions** For this kind of auto-transition, a user is present to react to any rule-related issues that appear. You must make sure that you support the situation that occurs when the author of a work item type adds a required field that the integration does not recognize. To support this situation, perform the auto-transition and then inspect the work item type for rule violations. If you find any, display the form for the user to resolve.
- **Unattended automation auto-transitions** You must assume that no user is present to resolve these issues. In this case, the integration should fail gracefully. An error log should state that the auto-transition was tried, and it should give a reason for the failure.

When defining either type of auto-transition, define the transition so that each work item reaches a valid state at the end of the transition without requiring user intervention. In other words, all the rules that are defined for the state being transitioned to are met by providing defaults or copied values for all fields. If any field becomes invalid after the transition, the state transition will fail.

In order to keep fields from becoming invalid, do the following:

- Define a **DEFAULTREASON** for the state transition.
- For fields that would become required after the state transition, use the **DEFAULT** or **COPY** rule elements to specify a value for the field.

For example, you have created the transition action Check-In, which transitions the state of a work item from "Working" to "Ready to Build". The work item's rules for "Ready to Build" require that the "Resolved By" field be set. You would then define a **DEFAULT** or **COPY** rule element for "ResolvedBy" in the **TRANSITION** section. Additionally, you would define a **DEFAULTREASON**

to make sure that the required field can be set without user intervention.

See Also

Other Resources

[Apply a rule to a work item field](#)

Associating a State Transition with an Action

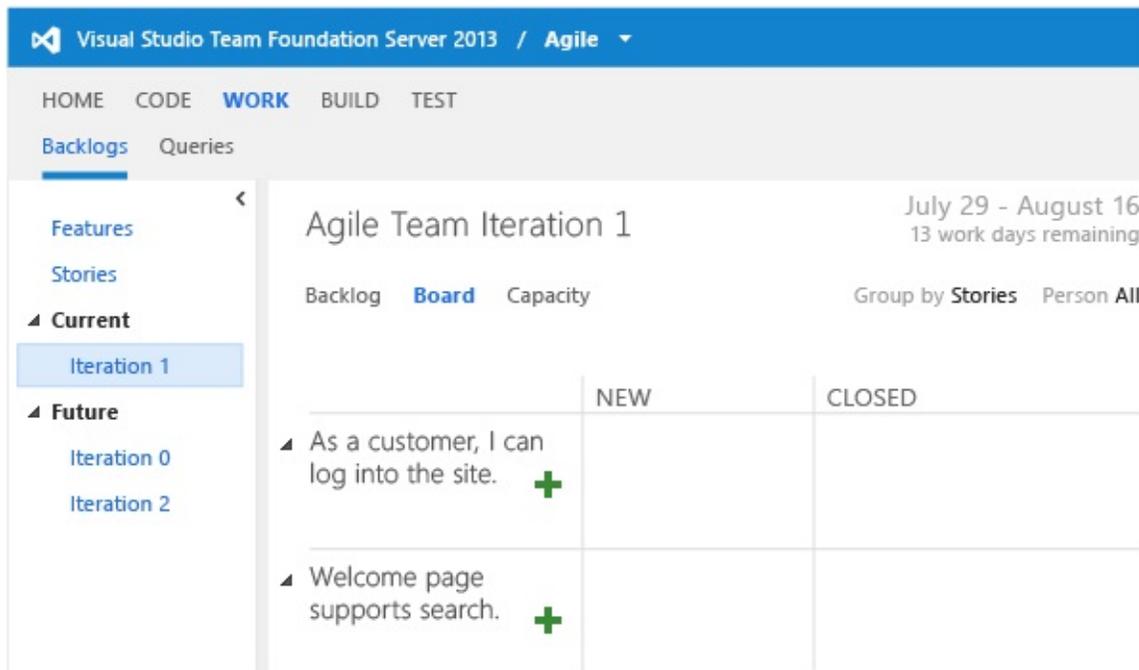
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Update the workflow for agile team projects

[See Also](#) [Send Feedback](#)

After you have configured features for your upgraded team project, which was created with a process template based on Microsoft Solutions Framework (MSF) for Agile Software Development v5.0, you should manually update the user story and task workflow assignments. If you don't update the workflow, then the task board provides only two states, Active and Closed, as shown in the following illustration. This prevents you and your team from distinguishing between tasks that are in progress from those that haven't been started.

Task board for an Agile team project without the recommended manual updates



Updates made by the Configure Features wizard were made to match the

existing workflow that is defined for your team project. For version 5.0 of the MSF Agile process template, this means the New and Removed states introduced with the latest version are not present. These states support transitioning user stories from a New state to an Active state. Without the New state defined, a User Story assigned to an iteration remains on the product backlog until it is resolved, which is not the desired or expected behavior.

In the same way, adding the New state to the Task workflow lets users move tasks from New, to Active, to Closed using the task board. In addition, by adding the Removed state to both the User Story and Task type definitions you enable users to cut stories and tasks using the planning tools.

To support the desired behaviors, you must add the New and Removed states to the user story and task work item types, and update the metastate mappings assigned in the process configuration.

Note

You can perform some of the procedures described below using Process Editor, a power tool add-in for Visual Studio which you can download and install. Located under the Tools menu, Process Editor provides a graphical user interface for customizing Team Foundation Server process templates. You can use this tool to import and export work item types and modify the contents of the process template. For more information, see the following page on the Microsoft website: [Team Foundation Server Power Tools](#).

Requirements

- To run the **witadmin** command-line tool, you must be a member of one of the following groups: Team Foundation Administrators, Project Collection Administrators, or Project Administrators group for the team project. See [Permission reference for Team Foundation Server](#).

Update the workflow and metastates for an Agile team project

⚠️ Important

The procedures in this topic were written based on the MSF for Agile Software Development 6.0. To update your team project to support the latest workflow defined for MSF for Agile Software Development 6.1, you can adapt these procedures to instead copy the `WORKFLOW` sections of the latest Agile process template. Significant updates were made to the workflow for several work item types in the latest quarterly update. These changes support backward transitions so that when you inadvertently drag a work item on the Kanban board or the task board to a resolved or closed state, you can drag it back to an earlier workflow state.

To learn more about the update, see [What's new in planning and tracking](#).

You will need to perform the following steps:

- Export the definitions for the User Story and Task types and for process configuration.
- Insert code snippets to the workflow section for both User Story and Task.
- Update the metastate mappings for the process configuration.
- Import the updated definitions for the work item types and process configuration.
- Verify that the changes appear on the task board.

⚠️ Important

For the following procedures to work, the Active state must be specified in the workflow defined for the User Story and Task types of work items.

To update the workflow states and transitions for User Story

1. To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and type:

Copy Code

```
cd %programfiles%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 64-bit edition of Windows,
replace **%programfiles%** with **%programfiles(x86)%**.

2. To export the type definition for User Story, enter the following command, substituting your data for the arguments that are shown here, where **CollectionURL** specifies the URL of a team project collection, **ProjectName** specifies the name of a team project defined within the collection, and **"DirectoryPath\FileName.xml"** specifies the name and location for the file to export. Then press Enter.

Copy Cod

```
witadmin exportwitd /collection:CollectionURL /p:"ProjectName" .
```

Use this format for the URL:

`http://ServerName:Port/VirtualDirectoryName/CollectionName`, for
example: `http://srvalm:8080/tfs/DefaultCollection`.

3. In a text editor or in Visual Studio, open the file you exported.
4. Add this code snippet between the lines `<STATES>` and `<STATE value="Active">`:

Copy Code

```
<STATE value="New">  
  <FIELDS>  
    <FIELD refname="Microsoft.VSTS.Common.ResolvedDate">  
      <EMPTY />  
    </FIELD>  
  </FIELDS>  
</STATE>
```

```

</FIELD>
<FIELD refname="Microsoft.VSTS.Common.ResolvedBy">
  <EMPTY />
</FIELD>
<FIELD refname="Microsoft.VSTS.Common.ClosedDate">
  <EMPTY />
</FIELD>
<FIELD refname="Microsoft.VSTS.Common.ClosedBy">
  <EMPTY />
</FIELD>
<FIELD refname="Microsoft.VSTS.Common.ActivatedDate">
  <EMPTY />
</FIELD>
<FIELD refname="Microsoft.VSTS.Common.ActivatedBy">
  <EMPTY />
</FIELD>
</FIELDS>
</STATE>
<STATE value="Removed" />

```

5. Replace the section that begins with <TRANSITION from="" to="Active"> and ends with <TRANSITION> with this code snippet:

```

<TRANSITION from="" to="New">
  <REASONS>
    <DEFAULTREASON value="New" />
  </REASONS>
  <FIELDS>
    <FIELD refname="System.Description">
      <DEFAULT from="value" value="As a &lt;type of user" />
    </FIELD>
  </FIELDS>
</TRANSITION>
<TRANSITION from="New" to="Active">
  <REASONS>
    <DEFAULTREASON value="Implementation started" />
  </REASONS>
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ActivatedBy">
      <COPY from="currentuser" />
      <VALIDUSER />
      <REQUIRED />
    </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ActivatedDate">
      <SERVERDEFAULT from="clock" />
    </FIELD>
  </FIELDS>
</TRANSITION>

```

```

        </FIELD>
        <FIELD refname="System.AssignedTo">
            <DEFAULT from="currentuser" />
        </FIELD>
    </FIELDS>
</TRANSITION>
<TRANSITION from="New" to="Removed">
    <REASONS>
        <DEFAULTREASON value="Removed from the backlog" />
    </REASONS>
</TRANSITION>
<TRANSITION from="Active" to="New">
    <REASONS>
        <DEFAULTREASON value="Implementation halted" />
    </REASONS>
</TRANSITION>
<TRANSITION from="Removed" to="New">
    <REASONS>
        <DEFAULTREASON value="Reconsidering the User Story" />
    </REASONS>
</TRANSITION>

```

6. Save and close the file.
7. Import the file, substituting your data for the arguments that are shown.

 **Copy Code**

```
witadmin importwitd /collection:CollectionURL /p:"ProjectName" ,
```

To update the workflow states and transitions for Task

1. Export the type definition for Task, substituting your data for the arguments that are shown.

 **Copy Code**

```
witadmin exportwitd /collection:CollectionURL /p:"ProjectName" ,
```

2. In a text editor or in Visual Studio, open the file you exported.

3. Add this code snippet between the lines `<STATES>` and `<STATE value="Active">`:

Copy Code

```
<STATE value="New">
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
      <EMPTY />
    </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
      <EMPTY />
    </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ActivatedDate">
      <EMPTY />
    </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ActivatedBy">
      <EMPTY />
    </FIELD>
  </FIELDS>
</STATE>
<STATE value="Removed" />
```

4. Replace the first transition block, corresponding to the lines that fall between `<TRANSITION from="" to="Active">` and its closing `</TRANSITION>` tag, with this code snippet:

Copy Code

```
<TRANSITION from="" to="New">
  <REASONS>
    <DEFAULTREASON value="New" />
  </REASONS>
</TRANSITION>
<TRANSITION from="New" to="Active">
  <ACTIONS>
    <ACTION value="Microsoft.VSTS.Actions.StartWork" />
  </ACTIONS>
  <REASONS>
    <DEFAULTREASON value="Work started" />
  </REASONS>
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ActivatedBy">
      <COPY from="currentuser" />
      <VALIDUSER />
      <REQUIRED />
    </FIELD>
  </FIELDS>
</TRANSITION>
```

```

        </FIELD>
        <FIELD refname="Microsoft.VSTS.Common.ActivatedDate">
            <SERVERDEFAULT from="clock" />
        </FIELD>
        <FIELD refname="System.AssignedTo">
            <DEFAULT from="currentuser" />
        </FIELD>
    </FIELDS>
</TRANSITION>
<TRANSITION from="Active" to="New">
    <ACTIONS>
        <ACTION value="Microsoft.VSTS.Actions.StopWork" />
    </ACTIONS>
    <REASONS>
        <DEFAULTREASON value="Work halted" />
    </REASONS>
</TRANSITION>
<TRANSITION from="New" to="Closed">
    <ACTIONS>
        <ACTION value="Microsoft.VSTS.Actions.Checkin" />
    </ACTIONS>
    <REASONS>
        <DEFAULTREASON value="Completed" />
        <REASON value="Cut" />
        <REASON value="Deferred" />
        <REASON value="Obsolete" />
    </REASONS>
    <FIELDS>
        <FIELD refname="Microsoft.VSTS.Common.ActivatedDate">
            <READONLY />
        </FIELD>
        <FIELD refname="Microsoft.VSTS.Common.ActivatedBy">
            <READONLY />
        </FIELD>
        <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
            <SERVERDEFAULT from="clock" />
        </FIELD>
        <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
            <COPY from="currentuser" />
            <VALIDUSER />
            <REQUIRED />
        </FIELD>
    </FIELDS>
</TRANSITION>
<TRANSITION from="Closed" to="New">
    <REASONS>
        <DEFAULTREASON value="Reactivated" />
    </REASONS>
    <FIELDS>
        <FIELD refname="System.AssignedTo">

```

```

        <COPY from="field" field="Microsoft.VSTS.Common.Closed
    </FIELD>
    </FIELDS>
</TRANSITION>
<TRANSITION from="New" to="Removed">
    <REASONS>
        <DEFAULTREASON value="Removed from the backlog" />
    </REASONS>
</TRANSITION>
<TRANSITION from="Active" to="Removed">
    <REASONS>
        <DEFAULTREASON value="Removed from the backlog" />
    </REASONS>
</TRANSITION>
<TRANSITION from="Removed" to="New">
    <REASONS>
        <DEFAULTREASON value="Reconsidering the Task" />
    </REASONS>
</TRANSITION>

```

5. Save and close the file.
6. Import the file, substituting your data for the arguments that are shown.

Copy Code

```
witadmin importwitd /collection:CollectionURL /p:"ProjectName" ,
```

To update the metastate mappings for process configuration

1. Type the following command to export the ProcessConfiguration file, substituting your data for the arguments that are shown.

Copy C

```
witadmin exportprocessconfig /collection:CollectionURL /p:"Proj
```

Where, CollectionURL specifies the URL of the team project collection and ProjectName specifies the name of your team project. Use this format for the URL: http://ServerName:Port/VirtualDirectoryName/CollectionName,

for example: <http://srvalm:8080/tfs/DefaultCollection>.

2. In Notepad or in another text editor, open the ProcessConfiguration.xml file.
3. Locate the tag `<RequirementWorkItems category="Microsoft.RequirementCategory" plural="Stories">`.
4. Replace the line `<State type="Proposed" value="Active" />` with these lines:

Copy Code

```
<State type="Proposed" value="New" />
<State type="InProgress" value="Active" />
```

5. Locate the tag `<TaskWorkItems category="Microsoft.TaskCategory">`, and replace the line `<State type="Proposed" value="Active" />` with these lines:

Copy Code

```
<State type="Proposed" value="New" />
<State type="InProgress" value="Active" />
```

6. Verify that you have the following mappings:

Copy Code

```
<RequirementWorkItems category="Microsoft.RequirementCategory" |
  <States>
    <State type="Proposed" value="New"/>
    <State type="InProgress" value="Active" />
    <State type="InProgress" value="Resolved" />
    <State type="Complete" value="Closed" />
  </States>
</RequirementWorkItems>
<TaskWorkItems category="Microsoft.TaskCategory">
  <States>
    <State type="Proposed" value="New" />
    <State type="InProgress" value="Active" />
    <State type="Complete" value="Closed" />
  </States>
```

```
</TaskWorkItems>
```

7. Save and close the file.
8. Import the configuration file.

 **Copy Code**

```
witadmin importprocessconfig /collection:CollectionURL /p:ProjectName
```

To verify that the changes appear on the task board

1. Open Team Web Access and then connect to your team project using a URL that has the following format:

 **Copy Code**

```
http://MyServer:8080/tfs/TeamProjectCollectionName/MyProject
```

If TWA is already open, then refresh your browser to refresh the cache.

2. Verify that the task board columns now display New, Active, and Closed.

Task board for an Agile team project with the updated workflow states

Visual Studio Team Foundation Server 2013 / Agile

HOME CODE **WORK** BUILD TEST

Backlogs Queries

Features
Stories

Current
Iteration 1

Future
Iteration 0
Iteration 2

Agile Team Iteration 1

July 29 - August 16
13 work days remaining

Backlog **Board** Capacity

Group by Stories Person All

	NEW	ACTIVE	CLOSED
As a customer, I can log into the site. +			
Welcome page supports search. +			

To learn more about using the task board, see [Work in sprints](#).

▣ **See Also**

Concepts

Track work with Visual Studio ALM and TFS
Configure features after a TFS upgrade



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

All FORM XML elements reference

[See Also](#) [Send Feedback](#)

You can use the information in this topic as a quick reference to all the elements and main attributes that control the form for a type of work item. You specify these elements in the FORM element container, the third and final major section of the definition of a type of work item. Many elements are nested within others to form groups, sections, or tabs in a work item form. For more information about how to group these elements, see [Design the work item form](#).

In this topic

-

[FORM Example](#)

- [Syntax for FORM elements](#)
- [Attributes that FORM elements specify](#)
- [Elements that specify standalone Labels, Hyperlinks, Web Pages, or HTML content](#)
- [Elements that filter and display link relationships](#)

FORM example

The following example shows the overall structure of the **FORM** element. You specify the layout of a form by using the **Layout** element. You can specify different layouts that target different clients. For example, you can specify one layout for Windows clients and a different layout for Team Web Access. A layout typically consists of the top of the form and then a group of tabs. The sequence in which you define the elements within the layout determines the sequence in which the elements appear on the form.

You group elements to appear within columns by using the **Group** and **Column** elements. You use a **Control** element to define each field that you want to appear on the form. You use the **Tab** element to support different functional areas of field groups.

```
<FORM>
  <Layout>
    <Group>
      <Column PercentWidth="70">
        <Group>
          <Column PercentWidth="100">
            <Control FieldName="System.Title" Type="FieldControl" />
            <Control FieldName="System.AreaPath" Type="WorkItemControl" />
            <Control FieldName="System.IterationPath" Type="WorkItemControl" />
            <Group>
              <Column PercentWidth="50">
                <Control FieldName="Microsoft.VSTS.Common.Product" Type="FieldControl" />
              </Column>
              <Column PercentWidth="50">
                <Control FieldName="Microsoft.VSTS.Common.Priority" Type="FieldControl" />
              </Column>
            </Group>
          </Column>
        </Group>
      </Column>
      <Column PercentWidth="30">
        <Group Label="Status">
          <Column PercentWidth="100">
            <Control FieldName="System.Id" Type="FieldControl" />
            <Control FieldName="System.State" Type="FieldControl" />
          </Column>
        </Group>
      </Column>
    </Group>
  </Layout>
</FORM>
```

```
        <Control FieldName="System.AssignedTo" Type="FieldC<br>    </Column><br>  </Group><br></Column><br></Group>
```

```
  . . .<br></Layout><br></FORM>
```

Form elements

You can specify how information and work item fields are grouped and appear in a work item form using the elements that are described in the following table.

Note

For best results, you should nest Control elements in a Group, and you should nest Group elements in a Column, even if the column spans the full width of the form. Also, you should nest every Column section in a Group, even if the group has no visible label or boundary.

Element	Description
	Divides a form's regions into columns.
	 Copy Code
Column	<pre><Column PercentWidth="WidthPerceOfContainingElement" FixedW <Group> . . . </Group> <Control> . . . </Control> <TabGroup> . . . </TabGroup> <Splitter> . . . </Splitter> </Column ></pre>
	Defines a field, text, hyperlink, or other control element to appear on th
	 Copy Code
Control	<pre><Control FieldName="FieldName" Type="AttachmentsControl D HtmlFieldControl LabelControl LinksControl WebpageCon workItemLogControl" Label="LabelText" LabelPosition="Top Padding="(top, bottom, left, right)" Margin="(top, bottom, left, right)" ReadOnly="True False Name="InstanceName" /></pre>

For more information, see

[Control XML element reference.](#)

Required

FORM

Defines the top-level form element.

 **Copy Code**

```
<FORM>  
  <Layout> . . . </Layout>  
</FORM>
```

Required

Group

Provides a visual grouping of elements, similar to the Windows GroupBox.

 **Copy Code**

```
<Group Label="LabelText" Padding="(top, bottom, left, right)" Margin="(  
  <Column> . . . </Column>  
</Group >
```

Recommended

Layout

Defines the layout of the work item form.

 **Copy Code**

```
<Layout Target="ClientName" MinimumSize="(width,height)" Padding="(t  
Margin="(left, top, right, bottom)" ControlSpacing="Distance" Labels  
  <Group> . . . </Group>  
  <Control> . . . </Control>  
  <TabGroup> . . . </TabGroup>
```

```
<Splitter> . . . </Splitter>
</Layout>
```

Required

Splitter

Divides a form into two areas to support the layout of two sibling form elements.

 **Copy Code**

```
Splitter />
```

Optional

Tab

Defines the layout of a single tab in a tab group.

 **Copy Code**

```
<Tab Label="LabelText" Padding="(top, bottom, left, right)" Margin='
  <Group> . . . </Group>
  <Control> . . . </Control>
  <TabGroup> . . . </TabGroup>
  <Splitter> . . . </Splitter>
</Tab>
```

Optional

TabGroup

Defines one or more tabs for the form.

 **Copy Code**

```
<TabGroup Padding="( top, bottom, left, right)" Margin="(top, botton
  <Tab> . . . </Tab>
</TabGroup>
```

Optional

Attributes that Are Used to Format FORM Elements

You can control the spacing and size of many elements on a work item form by specifying the attributes that the following table describes. For more information, see Design the work item form.

Attribute	Description	Applicable Elements
ControlSpacing	Optional. Specifies the vertical offset of controls. Integer.	Layout
FixedWidth	Optional. The column width in pixels. PercentWidth and FixedWidth are mutually exclusive.	Column
Label	Optional. Text to appear for a control or group. If a label is specified, it overrides any label that is associated with the field in the metadata for the control.	Control Group
	Required for the Tab element. Text to	Tab

appear for a tab.

LabelText is a text string that contains between 1 and 80 characters.

LabelText is a text string that contains between 1 and 80 characters.

Optional. The label position relative to the field data. Possible values are Top, Bottom, Left, and Right.

Control

Optional. Specifies the number of pixels between the label and the edit region of the control.

Layout

Optional. String of the form (left, top, right, bottom) that specifies, in pixels, the amount of space around the control and between the control and its neighbors. You can vary the amount of space on each side.

Control

Group

Layout

Tab

Pattern value: $\backslash(\d+\backslash,\d+\backslash,\d+\backslash,\d+\backslash)\$$

TabGroup

Pattern value

example: (2,0,2,0)

MinimumSize	Optional. String of the form (width, height). This value specifies the minimum size for the form itself. When the container control smaller than this size, horizontal and vertical scrollbars appear.	Control Layout
Name	Optional. Name for the control. If unspecified, the name is the same as the FieldName.	Control
Padding	Optional. String of the form (top, bottom, left, right) that specifies, in pixels, the amount of room around the outside border of the control and around the inside border. You can vary the amount of space on each side.	Control Group Layout Tab TabGroup
	Optional. The column width as a	

PercentWidth percentage of the containing element's width. **PercentWidth** and **FixedWidth** are mutually exclusive. **Column**

ReadOnly Optional. You can display a read-only field in a control. Different controls respond to this attribute in slightly different ways. **Control**

Target Optional. String that specifies to which client this layout applies. Visual Studio displays this work item type in this layout if the **Target** attribute is omitted or set to **Windows Forms**. External clients can specify additional layouts by using custom **Target** attributes that Visual Studio ignores. **Layout**

The following values are valid:

- **WinForms:**
Applies the form to **Team**

Explorer and
Team Explorer
Everywhere.

- Web: Applies
the form to
Team Web
Access.

Type

Required. The type
of the control. For
more information,
see

[Control XML element reference.](#)

Control

Elements that specify stand-alone Labels, Hyperlinks, Web Pages, or HTML Content

The WebpageControlOptions element and its child elements have the following syntax structure:

 **Copy Code**

```
<WebpageControlOptions AllowScript="true | false" ReloadOnParamChange="true | false"
  <Link UrlRoot="UrlRoot" UrlPath = "UrlPathWithParameters">
    <Param index="IndexValue" value="ParamValue" type = "Original | New" />
  </Link>
  <Content>
    <![CDATA[Contents of HTML]]>
  </Content>
</WebpageControlOptions>
```

You use the elements that are described in the following table to define plain text or hyperlinked labels, add hyperlinks to a field, or display Web page content in a work item form. For more information, see Provide help text, hyperlinks, or web content on a work item form.

Element	Required?	Description
Content	Optional	Specifies the CDATA HTML-based content to appear in a work item form.
WebpageControlOptions element.		 Copy Code <Content> <![CDATA[Contents of HTML]]> </Content>

For more information, see

[WebpageControlOptions XML Elements Reference](#).

LabelText

Optional Control element when type="LabelControl".

Defines either a plain or hyperlinked label.

For more information, see [LabelText and Text XML Elements Reference](#).

Link

Optional Control element when type="FieldControl".

Optional Text element.

Optional WebpageControl element.

Optional WebpageControlOptions element.

Defines the hyperlink for a field, label, or Web browser content to appear in a work item form.

Copy Code

```
<Link URLRoot="URLRoot" URLPath ="URLPathWithParameters">  
  <Param />  
</Link>
```

For more information, see [Link and Param XML Elements Reference](#).

Param

Optional Link element.

Specifies a value to determine the URL of the hyperlink when URLPath is specified for the Link element.

Copy Code

```
<Param index="IndexValue " value="ParamValue " type ="Original | Cur
```

For more information, see [Link and Param XML Elements Reference](#).

Text

Optional LabelText element.

Container element for the information or label to appear on the work item form.

Copy Code

```
<Text>  
  <Link >  
    <Param />  
  </Link>  
  LabelText  
</Text>
```

For more information, see [LabelText and Text XML Elements Reference](#).

WebpageControlOptions

Optional Control element when type="WebpageControl"

Container element that specifies the options for the Web page control.

For more information, see [WebpageControlOptions XML Elements Reference](#).

Elements that filter and display link relationships

You use the `LinksControlOptions` element to define the options for controlling what links can be added to a work item and the default columns that you want to appear for the list of links in a work item. When you add link control to a work item form, you can specify filters that restrict the types of links that users can create and the types of work items between which users can create links. The `LinksControlOptions` element and its child elements have the following structure:

 **Copy Code**

```
<LinksControlOptions>
  <WorkItemLinkFilters FilterType="include | exclude | includeAll |
    <Filter LinkType="linkTypeRefName" FilterOn="reverseName | fo
  </WorkItemLinkFilters>
  <ExternalLinkFilters FilterType ="include | exclude | includeAll
    <Filter LinkType="externalLinkName"/>
  </ExternalLinkFilters>
  <WorkItemTypeFilters Scope=" project | all" FilterType=" include
    <Filter WorkItemType="workItemTypeReferenceName"/>
  </WorkItemTypeFilters>
  <LinkColumns>
    <LinkColumn RefName="referenceName" | LinkAttribute="linkAttr:
  </LinkColumns>
</LinksControlOptions>
```

Specifically, you use the elements that are summarized in the following table. For more information about how to use these elements, see [Define link controls to restrict link relationships and](#)

[LinksControlOptions Elements.](#)

Element	Required?	Description
		Provides a

LinksControlOptions Optional Control element when type="LinksControl" container for elements that define the options for controlling what links can be added to a work item and the default columns that you want to appear for the list of links in a work item.

WorkItemLinkFilters Optional LinksControlOptions element Provides a container for one or more Filter elements that specify the link filter criteria to use for link types that are defined for the team project collection.

Provides a container for

ExternalLinkFilters	Optional LinksControlOptions element	one or more Filter elements that specify the filter criteria to use for link types that are defined in other team projects.
WorkItemTypeFilters	Optional LinksControlOptions element	Provides a container for one or more Filter elements that specify the filter criteria to apply to work item types.
Filter (link types)	Required WorkItemLinkFilters element when the FilterType attribute is exclude or include. Required ExternalLinkFilters element when the FilterType attribute is exclude or include.	Specifies the types of links to include or exclude from the set of links that can be created for the work item.

Filter (work item types)	Optional WorkItemTypeFilters element when the FilterType attribute is exclude or include.	Specifies the types of work items to include or exclude from the set of work item types between which users can link.
LinkColumns	Optional LinksControlOptions element	Provides a container for one or more LinkColumn tags.
LinkColumn	Required LinkColumns element	Specifies the work item fields and link type attributes to appear on the work item form for the list of links.

See Also

Concepts

Design the work item form

Other Resources

[Work item tracking: Index to XML element definitions](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Control XML element reference

[Example](#) [See Also](#) [Send Feedback](#)

You use the **Control** element to define a work item field, text, hyperlink, or other control element to display in a work item form. For examples of how to use this element, see [Specify work item form controls](#) and [Design the work item form](#).

Schema Hierarchy

WITD

WORKITEMTYPE

FORM

Layout

Group

Column

Tab

```
<xs:element name="Control" type="ControlType" minOccurs="0"/>
```

```
<Control FieldName="FieldName" Type="AttachmentsControl | DateTimeControl | HtmlFieldControl | LabelControl | LinksControl | WebpageControl | WorkItemLogControl" ControlFontSize="FontSize" EmptyText="TextString" Label="LabelText" LabelPosition="Top | Bottom | Left | Right" Padding="(top, bottom, left, right)" Margin="(top, bottom, left, right)" ReadOnly="True | False" MinimumName="InstanceName" >
```

```
<LabelText> . . . </LabelText>  
<Link> . . . </Link>  
<LinksControlOptions> . . . </LinksControlOptions>  
<WebpageControlOptions> . . . </WebpageControlOptions>  
</Control>
```

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Attributes

Attribute	Description
	Optional Control attribute.
FieldName	Specifies the work item field with which the control is associated. The attribute type is typelib:ReferenceFieldName . Minimum length: 1; maximum length: 70. Pattern value: <code>^[a-zA-Z_][a-zA-Z0-9_]*(\.[a-zA-Z0-9_]+)\$</code> Pattern value example: Company.Division.IssueType
	Required Control attribute.
	Specifies the data type of the control. The attribute type is ValidControlsType Simple Type. Specify a string from one of the following built-in types: <ul style="list-style-type: none">• AttachmentControl: Use to display work item attachments. This control does not have an

associated field or field type.

- **DateTimeControl:** Use to display formatted date fields with a field type of `DateTime`.
- **FieldControl:** Use to display plain textual or numeric fields and lists of values for fields with a field type of `String`, `Identity`, `Integer`, `Double`, and `PlainText`.
- **HtmlFieldControl:** Use to display multi-line, rich-text format of fields with a field type of `HTML`.
- **LabelControl:** Use to display text that is not associated with a field. The text can be plain or hyperlinked. You can specify additional controls using the `LabelText`, `Link` and `Text` elements. See [LabelText and Text XML Elements Reference](#) and [Link and Param XML Elements Reference](#).
- **LinksControl:** Use to display the links control toolbar and manage link relationships between work items and storyboards. This control does not have an associated field or field type. You specify the types of links to filter and other control options using the `LinksControlOptions` element.

Type

See [LinksControlOptions Elements](#).

- **WebpageControl:** Use to display HTML-based content defined by a URI or embedded within a CDATA tag. This control does not have an associated field or field type. . You specify the content and links to display using the `WebpageControlOptions` element. See [WebpageControlOptions XML Elements Reference](#).
- **WorkItemClassificationControl** Use to display the hierarchical path fields with a field type of `TreePath`.

Note

The `System.AreaPath` and `System.IterationPath` fields are the only fields that use this data type. You cannot define a custom field using this data type.

- **WorkItemLogControl:** Use to display work item history information and fields with a field type of `History`.

Note

The `System.History` field is

the only field that uses this data type. You cannot define a custom field using this data type.

For more information, see Specify work item form controls.

Note

You can use a custom control by setting the Type attribute to a custom value. Any control whose Type value does not match the name of a built-in type is considered a custom control. For more information, see the Microsoft Web site: [Work Item Tracking Custom Controls](#).

Optional Control attribute.

ControlFontSize Specifies the font size of the field name or label on the form. Valid values of FontSize: small, normal, large, xlarge, and xxlarge.

Optional Control attribute.

EmptyText Specifies a text string between 1 and 255 characters in length that appears when a field is empty.

Optional Control attribute.

Label Specifies the visible text on the form that identifies the control. Specify a string of no more than 80 characters.

LabelFontSize Specifies the font size of the label on the form. Valid values for `FontSize`: small, normal, large, xlarge, and xxlarge.

Optional Control attribute.

LabelPosition Specifies the position of the label relative to the control data. Specify a string from one of the following values:

- **Top**: Places the label above the control data.
- **Bottom**: Places the label below the control data.
- **Left**: Places the label to the left of the control data.
- **Right**: Places the label to the right of the control data.

Optional Control attribute.

Padding Specifies the amount of space in pixels around the inside border of the control which corresponds to the top, bottom, left, and right spaces. The attribute type is **SizeType**.

Pattern value: ^\

(\d+\,\d+\,\d+\,\d+)\\$

Pattern value example: (2,0,2,0)

Optional Control attribute.

Specifies the amount of space in pixels around the outside border of the control which corresponds to the top, bottom, left, and right spaces. The attribute type is **SizeType**.

Margin

Pattern value: ^\
(\d+\,\d+\,\d+\,\d+)\\$

Pattern value example: (2,0,2,0)

Optional Control attribute.

Specifies that the field is read-only. The attribute type is **ReadOnlyType**. Specify a string from one of the following values:

ReadOnly

- True: Specifies that the control data is read-only.
- False: Specifies that the control data is not read-only.

Optional Control attribute.

Specifies the minimum size in pixels that the control should occupy in the form. The syntax is specified in (width,height). The attribute type is **SizeType**.

Pattern value: $\backslash(\backslash d+\backslash,\backslash d+\backslash)\$$

Pattern value example: (100,100)

MinimumSize

Note

If you do not have sufficient vertical space, then a scrollbar appears to keep its minimum size. Without this attribute, the controls are drawn with their default sizes, unless controls in other tabs take more space. The overall size of any one form control depends on the size of the largest form area or tab.

Optional Control attribute.

Identifies a control uniquely. The Name is important if more than one control on the form is associated with the same work item field. The attribute type is **xs:string**.

Note

Name

You use the Name attribute when you want to have the same field displayed in more than one location on the form. You specify a unique value for the Name attribute for both control entries so that the system identifies each control uniquely. It is useful to show the same control in various locations based on the context of tabs.

Optional Control attribute that is valid only when it is used with FieldControl.

Specifies the characters that you can enter in the field control. Valid values that you can use are as follows:

NumberFormat

- WholeNumbers: Specifies that whole numbers are allowed.
- SignedWholeNumbers: Specifies that signed whole numbers are allowed.
- DecimalNumbers: Specifies that decimal numbers are allowed.
- SignedDecimalNumbers: Specifies that signed decimal numbers are allowed.

Optional Control attribute that is valid only when it is used with FieldControl.

MaxLength

Specifies the maximum length of allowed characters for a field control. The attribute type is xs:integer.

Optional Control attribute that is valid only when it is used with DateTimeControl.

Specifies the format for the date-

time field corresponding to one of the values of the DateTimePickerFormat enum. Valid values that you can use are as follows:

- Custom: Displays the date/time value in a custom format. The Custom format can be set by specifying 'Custom' as the value for the Format attribute, and setting another attribute named CustomFormat with a custom format string. For example:

```
Format="Custom"
```

```
CustomFormat = "MMM dd,  
'of the year' yyyy "
```

For more information, see the Microsoft Web site:

[DateTimePicker.CustomFormat Property](#).

- Long: Displays the date/time value in the long date format set by the user's operating system.
- Short: Displays the date/time value in the short date format set by the user's operating system.
- Time: Displays the date/time value in the time format set by the user's operating system.

Format

For more information, see the Microsoft Web site:
[DateTimePickerFormat Enumeration](#).

Optional Control attribute that is valid only when it is used with DateTimeControl and the Format attribute.

CustomFormat Specifies the custom format for the date-time field corresponding to the syntax defined for the DateTimePicker.CustomFormat property. For more information, see the Microsoft Web site:
[DateTimePicker.CustomFormat Property](#).

Child Elements

Element	Description
	Optional element. You can specify this element when the Type attribute is specified as FieldControl or LabelControl.
LabelText	Container element for specifying the text, hyperlink, and position of a field or label whose text is partly associated

with a hyperlink.

For more information, see [LabelText and Text XML Elements Reference](#) and Provide help text, hyperlinks, or web content on a work item form.

Optional element. You can specify this element when the Type attribute is specified as FieldControl or LabelControl.

Link

Defines the hyperlink for a field or label.

For more information, see [Link and Param XML Elements Reference](#) and Provide help text, hyperlinks, or web content on a work item form.

Optional element. You can specify this element when the Type attribute is specified as LinksControl.

LinksControlOptions

Defines the link filters and layout of column fields to display the list

of links.

For more information, see [Define link controls to restrict link relationships](#).

Optional element. You can specify this element when the Type attribute is WebpageControl.

Specifies the controls that govern loading the Web page target in the work item form.

WebpageControlOptions

For more information, see [WebpageControlOptions XML Elements Reference](#) and Provide help text, hyperlinks, or web content on a work item form.

Optional element.

anyAttribute

Optional. Control is extensible through use of the anyAttribute element.

Parent Elements

Element	Description
Layout	Required. Defines how to display the elements on the work item form.
Column Element	Required. Defines columns for the form.
Tab Element	Required. Defines a tab for the form.

Remarks

Control is an optional child element of Layout, Column, and Tab.

Layout and Column indicators:

minOccurs = "0"

maxOccurs = "1"

Tab indicators:

minOccurs = "0"

maxOccurs = "unbounded"

To implement a custom control, you must specify the name of the control for the Type attribute. For more information, see the Microsoft Web site: [Work Item Tracking Custom Controls](#).

For an example of each control type, see Specify work item form controls.

Example

 **Copy Code**

```
<Control Type=" WorkItemClassificationControl " FieldName="System.Ar  
  Label="Area" LabelPosition="Left" Padding="(2,0,2,0)" Margin="(2,2,2,2)" />
```

Element Information

Namespace <http://schemas.microsoft.com/VisualStudio/2008/workitemtracking/t>

Schema Name Work Item Type Definition

Validation File WorkItemTypeDefinition.xsd

Element Type ControlType Complex Type

Can Be Empty Not applicable

See Also

Concepts

Design and customize a work item form [redirected]

[All FORM XML elements reference](#)

[Specify work item form controls](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

LabelText and Text XML Elements Reference

[See Also](#) [Send Feedback](#)

You can add a standalone label or informational text to a work item form by using the **LabelText** element. The label is not associated with any work item field. Optionally, you can add a hyperlink to some or all of the text.

To add elements to a form, you modify a work item type. See [Modify](#) or add a custom work item type (WIT).

The **LabelText** element is a child element of the **Control** element.

Schema Hierarchy

WITD

WORKITEMTYPE

FORM

Layout

Group

Column

TabGroup

Tab

Control

<LabelText>

```
<Text>
  <Link OpenInNewWindow="true | false" UrlPath="URLLinkWithParam"
    <Param index="IndexValue " value="ParamValue " type ="Orig:
  </Link>
  LabelText
</Text>
</LabelText>
```

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Child Elements and Attributes

Element	Description
	Required Control element when type="LabelControl".
LabelText	Container element for a standalone label or informational text. The element type is complex type: LabelElementType .
	This element overwrites the value specified by the Label attribute specified by the Control element.
	Required LabelText element.
	Container element for the information or label to appear on the work item form. If the Link element is specified within the label, the portion of the text within the element is hyperlinked. If the Link element is not specified, the text is rendered without a

hyperlink.

Text

Different parts of the label text can be hyperlinked with different URLs. Multiple text element values can be specified in a series.

The element type is complex type: [TextElementType](#).

If the Text element is not specified, then the Control element Label attribute text appears on the work item form.

Optional Text element.

Container element for the hyperlink to be applied to a field or standalone label that appears on the work item form. The element type is `HyperLinkType`.

Link

Note

You must specify the Label attribute in a Control element when you specify a child Link element.

For more information, see [Link and Param XML Elements Reference](#).

Parent Elements

Element	Description
Control	Required. Defines a field, text, hyperlink, or other control element to appear on the work item form.

Remarks

You can combine plain and hyperlinked text by adding multiple Textelements in a LabelText element.

The LabelText element is only recognized by clients that are running the latest version of Team Explorer. Therefore, you must specify the Label attribute in a Control element. For clients that are running the latest version of Team Explorer, the text provided by the LabelText element appears in the work item form. For clients that are running earlier versions of Team Explorer, the text specified by the Control element Label attribute appears in the work item form.

Example: Standalone Label

Description

The following example adds the plain text "Enter details about how to reproduce the bug in the space below:" to a work item form.

Code

 **Copy Code**

```
<Control Type="LabelControl" Label=" Enter details about how to rep  
  <LabelText>  
    <Text>Enter details about how to reproduce the bug in the spac  
    </Text>  
  </LabelText>  
</Control>
```

Example: Standalone Label with Hyperlink

Description

The following example adds a hyperlink which is labeled "How do I use this work item?" to a work item form.

Code

 **Copy Code**

```
<Control Type="LabelControl" Label="How do I use this work item?">
  <LabelText>
    <Text>
      <Link UrlRoot="http://www.live.com"></Link>
        How do I use this work item?
      </Text>
    </LabelText>
  </Control>
```

Example: Hyperlink Applied to Portions of Text

Description

The following example adds a two-part label to a work item form. The first part, "Iteration Path", is associated with a hyperlink. The second part, "(must be 3 levels deep)" appears on the work item form as plain text.

Code

 **Copy Code**

```
<Control Type="FieldControl" FieldName="System.IterationPath" LabelF
  <LabelText>
    <Text>
      <Link UrlRoot="@wssSiteUrl/render.aspx?wit=bug&topic=It
      </Link>
      Iteration Path
    </Text>
    <Text> (must be 3 levels deep)</Text>
  </LabelText>
</Control>
```

Schema Definitions

Element: LabelText

 Copy Code

```
<xs:element name="LabelText" type="LabelElementType" minOccurs="0" r
<xs:element name="Text" type="TextElementType" minOccurs="1" maxOccu
```

Complex Type: LabelElementType

 Copy Code

```
<xs:complexType name="LabelElementType">
  <xs:sequence>
    <xs:element name="Text" type="TextElementType" minOccurs="1" r
  </xs:sequence>
</xs:complexType>
```

Complex Type: TextElementType

 Copy Code

```
<xs:complexType name="TextElementType" mixed="true">
  <xs:sequence>
    <xs:element name="Link" type="HyperLinkType" minOccurs="0" ma
  </xs:sequence>
</xs:complexType>
```

See Also

Concepts

[Specify work item form controls](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Layout XML Element Reference

[Example](#) [See Also](#) [Send Feedback](#)

You can use the **Layout** element to define how the elements on the work item form appear. You can define more than one layout to support different clients, such as the Windows client for Visual Studio or the web client for Team Web Access.

To add elements to a form, you modify the definition for a work item type. See [Modify or add a custom work item type \(WIT\)](#).

The **Layout** element is a required child element of the **FORM** element. For more information about how to use the **Layout** element, see [Design the work item form](#).

Schema Hierarchy

WITD

WORKITEMTYPE

FORM

Layout

```
<Layout Target="ClientName" MinimumSize="(width,height)" Padding="(left, right, top, bottom)" ControlSpacing="distance" Labels="true | false" HideReadOnlyEmptyFields="true | false">
  <Group> . . . </Group>
  <Control> . . . </Control>
  <TabGroup> . . . </TabGroup>
  <Splitter> . . . </Splitter>
```

</Layout >

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Attributes

Attribute	Description
Target	<p>Optional Layout attribute.</p> <p>Specifies the name of the client to which the layout applies. The type is xs:string. Specify one of the following strings:</p>
	<ul style="list-style-type: none"><li data-bbox="665 1218 941 1470">• WinForms: Applies the form to Team Explorer and Team Explorer Everywhere.<li data-bbox="665 1512 941 1680">• Web: Applies the form to Team Web Access. <p>Optional Layout attribute.</p>

MinimumSize

Specifies the minimum size in pixels for the form. The syntax is specified in (width, height). The attribute type is SizeType.

Pattern value: $\wedge (\d+\, \d+)\$$

Pattern value example: (100,100)

Optional Layout attribute.

Padding

Specifies the amount of space in pixels around the inside border of the form. The attribute type is SizeType.

Pattern value: $\wedge (\d+\, \d+\, \d+\, \d+)\$$

Pattern value example: (2,0,2,0)

Optional Layout attribute.

Specifies the amount of space in pixels around the outside border of the form.

Margin	<p>The attribute type is SizeType.</p> <p>Pattern value: ^\(\d+\,\d+\,\d+\,\d+\)\$</p> <p>Pattern value example: (2,0,2,0)</p> <p>Optional Layout attribute.</p>
ControlSpacing	<p>Specifies the vertical offset of controls defined in the form.</p> <p>Optional Layout attribute.</p>
LabelSpacing	<p>Specifies the number of pixels between the label and the edit region of the control.</p> <p>Optional Layout attribute.</p>
HideControlBorders	<p>Specify a value of True to hide control borders, and False to display control borders.</p> <p>Optional Layout attribute.</p>

HideReadOnlyEmptyFields	Specify a value of true to hide read-only and empty fields, and false to display these fields.
-------------------------	--

Child Elements

Element	Description
Group	Optional. Defines a group of elements to display together on the form.
Control	Optional. Defines a work item field, text, hyperlink, or other control element to display in a work item form.
TabGroup	Optional. Contains one or more Tab elements.
Splitter	Optional. Defines a splitter and its orientation on the form between sibling form elements.

Parent Elements

Element	Description
---------	-------------

FORM

Required. Contains the
Layout element that defines
how the elements on the work
item type form are displayed.

Remarks

The Layout element is a required child element of FORM. The Layout element specifies a choice, and at least one of the child elements must be defined: Group, Control, TabGroup, or Splitter. You can define more than one child element.

`minOccurs="1"`

`maxOccurs="unbounded"`

Example



```
<FORM>
  <Layout>
    <Group>
      <Column PercentWidth="36">
        <Group>
          <Control FieldName="System.Title" Type="FieldControl" La
          <Control FieldName="System.AreaPath" Type="WorkItemClass
          <Control FieldName="Microsoft.VSTS.Common.ProductUnit" 1
          <Control FieldName="Microsoft.DevDiv.BusinessUnit" Type=
          </Group>
        </Column>
      <Column PercentWidth="33">
        <Group>
          <Control FieldName="Microsoft.DevDiv.SubTitle" Type="Fie
          <Control FieldName="System.IterationPath" Type="WorkIter
          <Control FieldName="Microsoft.DevDiv.Other" Type="FieldC
          </Group>
        </Column>
      <Column PercentWidth="31">
        <Group>
          <Control FieldName="Microsoft.DevDiv.Type" Type="FieldCo
          <Control FieldName="System.AssignedTo" Type="FieldContro
          <Control FieldName="System.State" Type="FieldControl" La
          </Group>
        </Column>
      </Group>
    </Layout>
  </FORM>
```

Element Information

Namespace <http://schemas.microsoft.com/VisualStudio/2008/workitemtracking/t>

Schema Name Work Item Type Definition

Validation File WorkItemTypeDefinition.xsd

Element Type LayoutType Complex Type

Can Be Empty No

See Also

Reference

[Control XML element reference](#)

Concepts

Specify work item form controls

[All FORM XML elements reference](#)

Design the work item form



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

LinksControlOptions Elements

[Example](#) [See Also](#) [Send Feedback](#)

You use the **LinksControlOptions** element to define the options for controlling what links can be added to a work item and the default columns that you want to appear for the list of links in a work item. When you add a links control to a work item form, you can specify filters that restrict the types of links that users can be view and create and the types of work items to which users can create links. Specifically, you use the following elements as indicated:

- **WorkItemLinkFilters:** Use to restrict the types of links between work items in the current team project.
- **ExternalLinkFilters:** Use to restrict the types of links that can be created between work items and other objects, such as changesets, hyperlinks, or version controlled files.
- **WorkItemTypeFilters:** Use to restrict the link relationships that users can create based on work item type and whether the work item is defined in the current team project.

In each of these elements, you can specify **Filter** elements to specify the link types or work item types that you want to include or exclude. For more information about how to use these elements, see [Define link controls to restrict link relationships](#).

The **LinksControlOptions** element is a child element of the **CONTROL** element.

Schema Hierarchy

WITD

WORKITEMTYPE

FORM

Layout

Group

Column

Tab

Control

```
<xs:element name="LinksControlOptions" type="LinksControlOptionsType" />
<xs:element name="WorkItemLinkFilters" type="WorkItemLinkFiltersType" />
<xs:element name="Filter" type="WorkItemLinkFilterElementType" minOccurs="1" />
<xs:element name="ExternalLinkFilters" type="ExternalLinkFiltersType" />
<xs:element name="Filter" type="ExternalLinkFilterElementType" minOccurs="1" />
<xs:element name="WorkItemTypeFilters" type="LinksControlWorkItemTypeFiltersType" />
<xs:element name="Filter" type="LinksControlWorkItemTypeFilterElementType" />
<xs:element name="LinkColumns" type="LinkColumnsType" minOccurs="0" />
<xs:element name="LinkColumn" type="LinkColumnType" minOccurs="1" />
```

```
<LinksControlOptions>
  <WorkItemLinkFilters FilterType="include | exclude | includeAll | filterOn" />
  <Filter LinkType="linkTypeRefName" FilterOn="reversename | filterOn" />
</WorkItemLinkFilters>
  <ExternalLinkFilters FilterType="include | exclude | includeAll | filterOn" />
  <Filter LinkType="externalLinkName" />
</ExternalLinkFilters>
  <WorkItemTypeFilters Scope=" project | all" FilterType=" include | exclude | includeAll | filterOn" />
  <Filter WorkItemType="workItemTypeReferenceName" />
</WorkItemTypeFilters>
  <LinkColumns>
    <LinkColumn RefName="reference name" | LinkAttribute="link attribute" />
  </LinkColumns>
</LinksControlOptions>
```

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Child Elements and Attributes

Element	Attribute	Description
LinksControlOptions		<p>Optional Control element when type="LinksControl".</p> <p>Provides a container for elements that define what links can be added to a work item and default columns to appear for the list of link relationships in a work item.</p> <p>The element type is complex type: LinksControlOptionsType.</p> <p>Optional LinksControlOptions element.</p>
WorkItemLinkFilters		<p>Provides a container for one or more Filter elements that specify the link filter criteria used for link types that are defined for the project collection and that create relationships between work items.</p> <p>The element type is complex type: WorkItemLinkFiltersType.</p> <p>Required WorkItemLinkFilters attribute.</p> <p>Defines the method used to filter the set of</p>

types provided in the set of Filter elements.
element type is simple type:
[LinksControlFilterKind](#).

Valid values that you can use are:

FilterType

- **exclude:** Use to disallow links to be created from those link types listed in the Filter elements.
- **excludeAll:** Use to disallow all link types.
- **include:** Use to allow only those link types listed in the Filter elements.
- **includeAll:** Use to allow links to be created from all link types.

Required WorkItemLinkFilters element when the FilterType is exclude or include.

When the FilterType is excludeAll or includeAll, no Filter elements should be specified.

Filter

Specifies the types of links that are to be included or excluded from the set of links that can be created for the work item. The element type is simple type:
[WorkItemLinkFilterElementType](#).

Required Filter attribute.

Specifies the reference name for a type of link. Valid types include the following system defined link types:

- Storyboard

LinkType

- System.LinkTypes.Dependency
- System.LinkTypes.Hierarchy
- System.LinkTypes.Related

In addition, you can specify the reference name for custom link types defined for the team project collection. For more information, see [Link type element reference](#).

Optional Filter attribute.

Specifies the type of filter to apply to the link type. The element type is simple type: [LinksControlFilterOnKind](#).

Valid values are:

- forwardname: Use to filter on the forward name defined for a type of link.
- reversename: Use to filter on the reverse name defined for a type of link.

FilterOn

If unspecified, then both the forward and reverse names are used to filter the link type.

Note

If the link type topology is Network, the forward and reverse names are the same. For more information, see [Link type element reference](#).

Optional LinksControlOptions element.

ExternalLinkFilters

Provides a container for one or more Filter elements that specify the link types that use can specify to create relationships to object are not work items, such as changesets, hyperlinks, and files under version control. element type is complex:
[ExternalLinkFiltersType](#).

Optional ExternalLinkFilters attribute.

Defines the method that is used to filter the of link types provided in the set of Filter elements. The element type is simple:
[LinksControlFilterKind](#).

You can use the following values:

FilterType

- exclude: Use to disallow links to be cr from those link types that appear in the Filter elements.
- excludeAll: Use to disallow all link ty
- include: Use to allow only those link t listed in the Filter elements.
- includeAll: Use to allow links to be cr from all link types.

If unspecified, all links to external work ite are excluded.

Required ExternalLinkFilters element when FilterType is exclude or include.

When the FilterType is excludeAll or includeAll, no Filter elements should be specified.

Filter

The element type is complex:
[ExternalLinkFilterElementType](#).

Specifies the name of the link types that will be included or excluded from the set of links that users can create for the work item.

Required Filter attribute.

Specifies the reference name for a type of link to exclude or include. The attribute type is simple: [ExternalLinkTypeName](#). You can specify the following link types:

LinkType

- Fixed in Changeset
- Result Attachment
- Source Code File
- Test Result
- Workitem Hyperlink

Optional LinksControlOptions element.

Provides a container for one or more Filter elements that specify the filter criteria to apply to work item types.

WorkItemTypeFilters

The element type is complex:
[LinksControlWorkItemTypeFiltersElement](#)

Optional WorkItemTypeFilters attribute.

Defines the scope of the filter applied to the

of work item types provided in the set of Filter elements. The element type is simple type: [LinksControlWorkItemTypeFilterScopeKind](#)

Scope

Valid values that you can use are:

- all: Use to allow links to be created to work item types specified in the Filter
- project: Use to allow links to be created only to those work item types that are defined for the current project.

If unspecified, links to all types of work item are allowed.

Required WorkItemTypeFilters attribute.

Defines the method that is used to filter the of work item types provided in the set of Filter elements. The element type is simple: [LinksControlWorkItemTypeFilterKind](#).

You can use the following values:

FilterType

- exclude: Use to disallow the work item types in the Filter elements.
- excludeAll: Use to disallow all work item types.
- include: Use to allow the work item types in the Filter elements.
- includeAll: Use to allow all work item types.

If you do not specify any of these values, links to all work item types are allowed.

Filter

Optional WorkItemTypeFilters element when the FilterType is exclude or include.

When the FilterType is excludeAll or includeAll, you should not specify any Filter elements.

Specifies the types of work items that are to be included or excluded from the set of work item types that can be linked to. The element type is a complex type:

[LinksControlWorkItemTypeFilterElement](#)

Required Filter attribute.

Specifies the reference name of a work item type to be filtered. The attribute type is: `typelib:NonEmptyPlainConstant`.

WorkItemType

Minimum length: 1; maximum length: 255.

Pattern value: `^[^\\]*$`

Pattern value example: Task

Optional LinksControlOptions element.

LinkColumns

Provides a container for one or more LinkColumn elements. The element type is a complex type: [LinkColumnsType](#).

Required LinkColumns element.

Specifies the work item fields and link type attributes displayed for the list of links defined for a work item. The element type is a complex type.

LinkColumn

type: [LinkColumnType](#).

This column list is the default display. The can add and remove columns from the link

The order in which the LinkColumn elements are listed defines the order in which the columns are displayed in the work item form.

Optional LinkColumn attribute. Specify RefName or LinkAttribute, but not both.

RefName

Specifies the reference name corresponding to a valid work item field for the team project collection. The attribute type is `typelib:ReferenceFieldName`.

Optional LinkColumn attribute. Specify RefName or LinkAttribute, but not both.

LinkAttribute

Specifies the reference name corresponding to any attribute for a valid link type for the team project collection. The attribute type is `typelib:ReferenceFieldName`.

Parent Elements

Element	Description
---------	-------------

Control	Required. Defines a field, text, hyperlink, or other control element to appear on the work item form.
---------	---

Remarks

LinksControlOptions is an optional child element of Control element where ControlType="LinksControl". You can use the LinksControlOptions element to support the following actions:

- Specify the restrictions that govern the creation of relationships to other work items based on the link type and, optionally, the forward and reverse names of the link types.
- Specify the restrictions that govern the creation of relationships to objects other than work items, such as changeset, hyperlink, and so on, based on external link types.
- Specify the restrictions that govern the creation of relationships to other work items based on work item type and, optionally, the team project where those work items are defined.
- Specify the default column fields that you want to display information for the link list.

Example

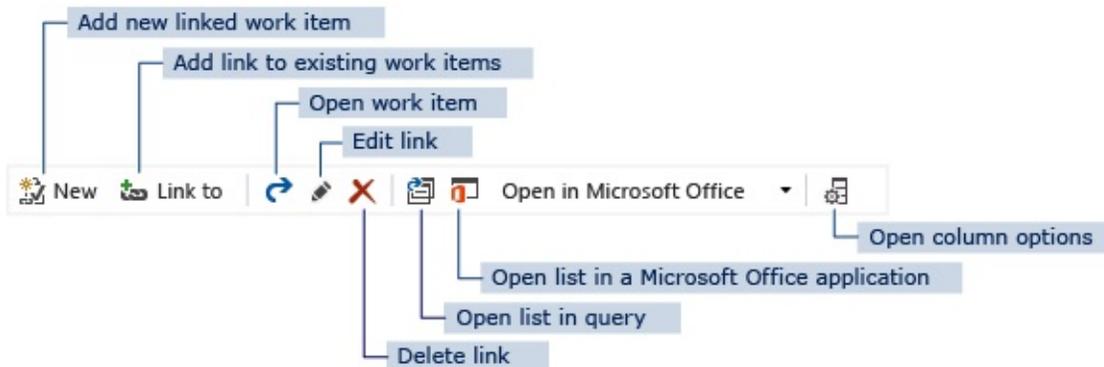
The following example specifies how to enable the creation of links whose type is `Microsoft.VSTS.Common.TestedBy` and show the Forward Name for this link type. The filter options disallow creation of links to all external link types. In addition, displayed default column fields for the listed links correspond to the following friendly names: ID, Work Item Type, Title, Assigned To, State, and the Comment field defined for the link. For more examples, see [Define link controls to restrict link relationships](#).

 **Copy Code**

```
<Control Type="LinksControl" Name="TestedBy" Label="&Work items
  <LinksControlOptions>
    <WorkItemLinkFilters FilterType="include">
      <Filter LinkType="Microsoft.VSTS.Common.TestedBy" FilterOn=
    </WorkItemLinkFilters>
    <WorkItemTypeFilters FilterType="include">
      <Filter WorkItemType="Test Case" />
    </WorkItemTypeFilters>
    <ExternalLinkFilters FilterType="excludeAll"/>
    <LinkColumns>
      <LinkColumn RefName="System.ID" />
      <LinkColumn RefName="System.WorkItemType" />
      <LinkColumn RefName="System.Title" />
      <LinkColumn RefName="System.AssignedTo" />
      <LinkColumn RefName="System.State" />
      <LinkColumn LinkAttribute="System.Links.Comment" />
    </LinkColumns>
  </LinksControlOptions>
</Control>
```

LinksControl toolbar buttons

Each Control element of Type="LinksControl" provides a toolbar, which contains the buttons shown in the next illustration:



These buttons become available only after you perform a specific action:

- The button to create a work item that is linked to the open work item (💡) becomes available only after you save the open work item.
- The buttons to open the list of work items in a query (📄) and in a Microsoft Office client (🏢) become available only when at least one work item is listed in the links control tab.
- The buttons to open a work item (🔄), edit a link (✎), and delete a link (✖) become available only after you click one or more work items listed in the links control tab.

The links control that is displayed is the same for both Team Web Access and Team Explorer, except when it is configured to only render Storyboard links. In that case, the toolbar only contains those controls to add a new link, open the linked item, and delete the link. Also, the Team Web Access version displays the Start Storyboarding link within the control menu.

DESCRIPTION **STORYBOARDS** TEST CASES TASKS

Start storyboarding



Title

Schema Definitions

-

[LinksControlOptionsType](#)

- [WorkItemLinkFiltersType](#)
- [WorkItemLinkFilterElementType](#)
- [ExternalLinkFiltersType](#)
- [ExternalLinkFilterElementType](#)
- [LinksControlFilterKind](#)
- [LinksControlFilterOnKind](#)
- [ExternalLinkTypeName](#)
- [LinksControlWorkItemTypeFiltersElementType](#)
- [LinksControlWorkItemTypeFilterElementType](#)
- [LinksControlWorkItemTypeFilterScopeKind](#)
- [LinksControlWorkItemTypeFilterKind](#)
- [LinkColumnsType](#)
- [LinkColumnType](#)

LinksControlOptionsType

 Copy

```
<xs:complexType name="LinksControlOptionsType">
  <xs:all>
    <xs:element name="LinkColumns" type="LinkColumnsType" minOccurs="1" maxOccurs="1"/>
  </xs:all>
</xs:complexType>
```

```

        <xs:element name="WorkItemLinkFilters" type="WorkItemLinkFilterElement'
        <xs:element name="ExternalLinkFilters" type="ExternalLinkFilterElement'
        <xs:element name="WorkItemTypeFilters" type="LinksControlWorkItemFilter'
    </xs:all>
</xs:complexType>

```

WorkItemLinkFiltersType

 **Copy Code**

```

<xs:complexType name="WorkItemLinkFiltersType" >
  <xs:sequence>
    <xs:element name="Filter" type="WorkItemLinkFilterElementType" minOccurs="1" maxOccurs="1" />
  </xs:sequence>
  <xs:attribute name="FilterType" type="LinksControlFilterKind" />
</xs:complexType>

```

WorkItemLinkFilterElementType

 **Copy Code**

```

<xs:complexType name="WorkItemLinkFilterElementType" >
  <xs:attribute name="LinkType" type="typelib:ReferenceFieldName" />
  <xs:attribute name="FilterOn" type="LinksControlFilterOnKind" />
</xs:complexType>

```

ExternalLinkFiltersType

 **Copy Code**

```

<xs:complexType name="ExternalLinkFiltersType" >
  <xs:sequence>
    <xs:element name="Filter" type="ExternalLinkFilterElementType" minOccurs="1" maxOccurs="1" />
  </xs:sequence>
  <xs:attribute name="FilterType" type="LinksControlFilterKind" />
</xs:complexType>

```

ExternalLinkFilterElementType

Copy Code

```
<xs:complexType name="ExternalLinkFilterElementType" >  
  <xs:attribute name="LinkType" type="ExternalLinkTypeName" use="required"/>  
</xs:complexType>
```

LinksControlFilterKind

Copy Code

```
<xs:simpleType name="LinksControlFilterKind">  
  <xs:restriction base="xs:string">  
    <xs:enumeration value="include" />  
    <xs:enumeration value="exclude" />  
    <xs:enumeration value="includeAll" />  
    <xs:enumeration value="excludeAll" />  
  </xs:restriction>  
</xs:simpleType>
```

LinksControlFilterOnKind

Copy Code

```
<xs:simpleType name="LinksControlFilterOnKind">  
  <xs:restriction base="xs:string">  
    <xs:enumeration value="forwardname" />  
    <xs:enumeration value="reversename" />  
  </xs:restriction>  
</xs:simpleType>
```

ExternalLinkTypeName

Copy Code

```
<xs:simpleType name="ExternalLinkTypeName">  
  <xs:restriction base="xs:string">  
    <xs:minLength value="1"/>  
    <xs:maxLength value="128"/>  
  </xs:restriction>  
</xs:simpleType>
```

LinksControlWorkItemTypeFiltersElementType

 [Copy Code](#)

```
<xs:complexType name="LinksControlWorkItemTypeFiltersElementType" >
  <xs:sequence>
    <xs:element name="Filter" type="LinksControlWorkItemTypeFilter" />
  </xs:sequence>
  <xs:attribute name="Scope" type="LinksControlWorkItemTypeFilterScopeKind" />
  <xs:attribute name="FilterType" type="LinksControlWorkItemTypeFilterKind" />
</xs:complexType>
```

LinksControlWorkItemTypeFilterElementType

 [Copy Code](#)

```
<xs:complexType name="LinksControlWorkItemTypeFilterElementType" >
  <xs:attribute name="WorkItemType" type="typelib:NonEmptyPlainConsistentWorkItemTypes" />
</xs:complexType>
```

LinksControlWorkItemTypeFilterScopeKind

 [Copy Code](#)

```
<xs:simpleType name="LinksControlWorkItemTypeFilterScopeKind">
  <xs:restriction base="xs:string">
    <xs:enumeration value="project" />
    <xs:enumeration value="all" />
  </xs:restriction>
</xs:simpleType>
```

LinksControlWorkItemTypeFilterKind

 [Copy Code](#)

```
<xs:simpleType name="LinksControlWorkItemTypeFilterKind">
  <xs:restriction base="xs:string">
    <xs:enumeration value="include" />
    <xs:enumeration value="exclude" />
    <xs:enumeration value="includeAll" />
  </xs:restriction>
</xs:simpleType>
```

```
    </xs:restriction>
</xs:simpleType>
```

LinkColumnsType

 **Copy Code**

```
<xs:complexType name="LinkColumnsType" >
  <xs:sequence>
    <xs:element name="LinkColumn" type="LinkColumnType" minOccurs="1" maxOccurs="1" />
  </xs:sequence>
</xs:complexType>
```

LinkColumnType

 **Copy Code**

```
<xs:complexType name="LinkColumnType" >
  <!-- Must be one or the other, but not both -->
  <xs:attribute name="LinkAttribute" type="typelib:ReferenceField" />
  <xs:attribute name="RefName" type="typelib:ReferenceFieldName" />
</xs:complexType>
```

Element Information

Namespace <http://schemas.microsoft.com/VisualStudio/2008/workitemtracking/>t

Schema Name Work Item Type Definition

Validation File WorkItemTypeDefinition.xsd

Element Type LinksControlOptionsType

Can Be Empty Not applicable

See Also

Concepts

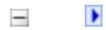
[Link type element reference](#)

Specify work item form controls

Define link controls to restrict link relationships

Other Resources

[Define a custom link type](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Link and Param XML Elements Reference

[See Also](#) [Send Feedback](#)

You can add a hyperlink to a field or a standalone label on a work item form by using the **Link** element. You use the **Link** element in the following instances to:

- Add a hyperlink to a field label
- Add a hyperlink to a standalone label or portions of informational text
- Specify the URL for content to be displayed within a work item form

To add elements to a form, you modify the definition for a work item type. See [Modify or add a custom work item type \(WIT\)](#).

The **Link** element is either a child element of the **Control** element, or a child element of the **WebpageControlTarget** or **WebpageControlOptions** elements. For more information about these elements, see [Control XML element reference](#) and [WebpageControlOptions XML Elements Reference](#).

Schema Hierarchy

WITD

WORKITEMTYPE

FORM

Layout

Group

Column

Tab

Control

```
<Link URLRoot="URLRoot" URLPath ="URLPathWithParameters">  
  <Param Index="IndexValue " Value="ParamValue " Type ="Original |  
</Link>
```

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Child Elements and Attributes

Element Attribute	Description
Link	Optional Control element when type="FieldControl".
	Optional LabelText element.
	Optional Text element.
	Optional WebpageControlOptions element.
	Container element for defining a hyperlink or URL. The element type is complex type: HyperLinkType .
	Required Link attribute.
	Specifies the protocol and the server name of the URL to which label links. Supported protocols are HTTP, HTTPS and MAILTO. You can either hardcode the specified server name, or you can use one of the following macros and optionally append additional address information to a macro: Note Each macro links to the URL that is defined and enabled for the team project of the work item form for which the Link element is defined. If any of the resources are not provisioned or enabled for the team project, a "page not found" error will appear.

- UrlRoot
- **@PortalPage:** The URL of the SharePoint site for the current team project (for example, `http://serverName/sites/collectionName/TeamProjectName`).
 - **@ProcessGuidance:** The URL of the SharePoint site for process guidance for a team project.
 - **@ReportManagerUrl:** The URL of the server that hosts Report Manager for the current team project. This provides a link to the folder view for the reports (for example, `http://serverName/Reports/Pages/Folder.aspx?ItemPath=/collectionName/teamProjectName`).
 - **@ReportServiceSiteUrl:** The URL of the server that hosts Server Reporting Services for the current team project. This provides the base path for the team project (for example, `http://serverName/Reports/Pages/reportservice.aspx?ItemPath=/collectionName/teamProjectName`).

To determine the hardcoded value for the first two macros, you open the project portal settings for a team project. For more information, see [Configure or redirect process guidance](#).

The attribute type is simple type: [NonBlankString](#).

Optional Link attribute.

UrlPath Specifies a URL path for the hyperlink. You use this attribute to specify variable parameters that are determined at run time. The attribute type is simple type: [NonBlankString](#).

Optional Link element. Required when UrlPath is specified.

Param Specifies a value to be used to determine the URL of the hyperlink when UrlPath is specified.

The element type is complex type: [HyperLinkParamType](#).

Required Param attribute.

Index Specifies the index of the parameter where the value must be formatted into the URL string. It is not required to order the values sequentially; however, they must match the number of index in the URLPath string.

The attribute type is PositiveInteger. The first index number be 0.

Required Param attribute.

Value Specifies the number associated with the value for a URL parameter in the hyperlink.

Value Specifies a value that must be formatted into the URLPath string. Values must be valid reference names of fields used in the work item type definition.

The @Me variable is supported. It is replaced with the name of the current team member that is viewing the work item form. Valid entries are not case sensitive

The attribute type is xs:string.

Optional Param attribute.

Specifies that the original value of the field must be used for the parameter. The attribute type is simple type: [HyperLinkParamValueKind](#).

Valid values are as follows:

Type

- Original: Specifies to use the original parameter value to determine the URL path.

- Current: Specifies to use the current parameter value to determine the URL path.

If this attribute is not specified, the current value of the field as a parameter

Parent Elements

Element	Description
Control	Required. Defines a field, text, hyperlink, or other control element to appear on the work item form.
LabelText	Optional Control element.
LabelText	Container element for a label to appear on the work item form.
LabelText	Optional LabelText element.

Text

Container element for the information or label to appear on the work item form.

Optional Control element.

WebpageControlOptions

Container element that specifies the options for the Web page control.

For more information, see [Displaying Web Content within a Work Item Form](#).

Remarks

The default behavior of the Link element is to open the hyperlink in the work item form or Web page. If the hyperlink is authored to open in a new window (target="_blank"), then an external browser is opened.

You can combine plain and hyperlinked text by adding multiple **Text** elements in a **LabelText** element.

The LabelText element is only recognized by clients that are running the latest version of Team Explorer. Therefore, you must specify the Label attribute in a Control element when you specify a child Link element. For clients that are running the latest version of Team Explorer, the text provided by the a LabelText element appears in the work item form. For clients that are running previous versions of Team Explorer, the text specified by the Control element Label attribute appears in the work item form.

Example: Specifying a Hyperlink with Parameters

Description

The following example shows how you can define a hyperlink that links to the process guidance for the Iteration topic:

```
http://serverName/sites/collectionName/projectName/render.aspx?  
wit=bug&topic=Iteration
```

The name of the project is derived from the value assigned to the System.Project field.

Code

 **Copy Code**

```
<Link UrlRoot="http://serverName" UrlPath="sites/{0}/render.aspx?wit  
  <Param Index="0" Value="System.Project" />  
</Link>
```

Example: Adding a Hyperlink to a Standalone Label

Description

The following example shows how you can assign a hyperlink to the text "work item guidance" on a work item form.

Code

 **Copy Code**

```
<Control type="LabelControl">
  <LabelText>
    <Text>
      <Link UrlRoot="http://www.msn.com" />
      work item guidance
    </Text>
  </LabelText>
</Control>
```

Example: Adding a Hyperlink to a Field

Description

The following example shows how you can add a hyperlink to the label for a field on a work item form.

Code

 Copy

```
<Control Type="FieldControl" FieldName="System.Title" LabelPosition=  
  <LabelText>  
    <Text>  
      <Link UrlRoot="http://www.live.com/" />  
      This text comes from the inner Label Text  
    </Text>  
  </LabelText>  
</Control>
```

Schema Definitions

- [HyperLinkType](#)
- [NonBlankString](#)
- [HyperLinkParamType](#)
- [HyperLinkParamValueKind](#)

Complex Type: HyperLinkType

 **Copy Code**

```
<xs:complexType name="HyperLinkType" >
  <xs:sequence>
    <xs:element name="Param" type="HyperLinkParamType" minOccurs="1" maxOccurs="1" />
  </xs:sequence>
  <xs:attribute name="UrlRoot" type="NonBlankString" use="required" />
  <xs:attribute name="UrlPath" type="NonBlankString" use="optional" />
</xs:complexType>
```

HyperLinkParamType

 **Copy Code**

```
<xs:complexType name="HyperLinkParamType" >
  <xs:attribute name="Index" type="PositiveInteger" use="required" />
  <xs:attribute name="Value" type="xs:string" use="required" />
  <xs:attribute name="Type" type="HyperLinkParamValueKind" use="optional" />
</xs:complexType>
```

NonBlankString

 **Copy Code**

```
<xs:simpleType name="NonBlankString">
  <xs:restriction base="xs:string">
```

```
<xs:pattern value=".*\S+.*"/>
</xs:restriction>
</xs:simpleType>
```

HyperLinkParamValueKind

 **Copy Code**

```
<xs:simpleType name="HyperLinkParamValueKind">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Original"/>
    <xs:enumeration value="Current"/>
  </xs:restriction>
</xs:simpleType>
```

Element Information

Namespace <http://schemas.microsoft.com/VisualStudio/2008/workitemtracking/>t

Schema Name Work Item Type Definition

Validation File WorkItemTypeDefinition.xsd

Element Type HyperLinkType, HyperLinkParamType

Can Be Empty Not applicable

▣ **See Also**

Concepts

Specify work item form controls

Design the work item form



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Tab XML element reference

[Example](#) [See Also](#) [Send Feedback](#)

You use the **Tab** element to cluster a group of fields or to support one or more special controls, such as the controls that link work items, display the work item history, or to attach files.

```
<xs:element name="Tab" type="TabType" minOccurs="1" maxOccurs="unbo
```

```
<Tab Label="LabelText" Padding="(left, top, right, bottom)" Margin='  
  <Group> . . . </Group>  
  <Control> . . . </Control>  
  <TabGroup> . . . </TabGroup>  
  <Splitter> . . . </Splitter>  
</Tab>
```

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Attributes

Attribute	Description
	Required TabType attribute.
Label	The label for a set of grouped elements.
	Optional TabType attribute.
Padding	The amount of space in pixels around the inside border of the control.
	Optional TabType attribute.
Margin	The amount of space in pixels around the outside border of the control.

Child Elements

Element	Description
----------------	--------------------

Group	Optional. Defines a group of elements to display together on the form.
Control	Optional. Defines a field that is to appear on the form.
TabGroup	Optional. Contains one or more Tab elements.
Splitter	Optional. Defines a splitter and its orientation on the form between sibling form elements.

Parent Elements

Element	Description
TabGroup	Optional. Contains one or more Tab elements.

Remarks

Tab is a required child element of TabGroup.

minOccurs="1"

maxOccurs="unbounded"

Example

```
<FORM>
  <Layout>
    . . .
    <TabGroup>
      <Tab Label="Planning">
        <Group Label="Status" Padding="(0,0,0,3)">
          <Column PercentWidth="100">
            <Control FieldName="Microsoft.DevDiv.Importance'
            <Control FieldName="Microsoft.DevDiv.Commitment'
            . . .
          </Column>
        </Group>
      </Tab>
      <Tab Label="Description">
        <Group>
          <Column PercentWidth="100">
            <Control FieldName="System.Description" Type="Hi
          </Column >
        </Group>
      </Tab>
    . . .
    </TabGroup>
  </Layout>
</FORM>
```

Element Information

Namespace <http://schemas.microsoft.com/VisualStudio/2005/workitemtracking/t>

Schema Name Work Item Type Definition

Validation File WorkItemTypeDefinition.xsd

Element Type TabType Complex Type

Can Be Empty Yes

See Also

Concepts

[All FORM XML elements reference](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

WebpageControlOptions XML Elements Reference

[Example](#) [See Also](#) [Send Feedback](#)

You can display content defined for a Web page or within a CDATA tag on a work item form by using the **WebpageControlOptions** element. You can include either the **Link** or the **Content** child elements to specify either a URL or HTML for the content to appear in the form.

To add elements to a form, you modify a work item type. See [Modify](#) or add a custom work item type (WIT).

The **WebpageControlOptions** element is a child element of the **CONTROL** element.

Schema Hierarchy

WITD

WORKITEMTYPE

FORM

Layout

Group

Column

TabGroup

Tab

Control

```
<WebpageControlOptions AllowScript="true | false" ReloadOnParamChange="true | false">
  <Link UrlRoot="UrlRoot" UrlPath="UrlPathWithParameters">
    <Param Index="IndexValue " Value="Param Value " Type="Original">
  </Link>
  <Content>
    <![CDATA[Contents of HTML]]>
  </Content>
</WebpageControlOptions>
```

Attributes and Elements

The following sections describe attributes, child elements, and parent elements.

Child Elements and Attributes

Element	Attribute	Description
WebpageControlOptions		Optional Control element when type="WebpageControl".
		Container element for defining whose contents will be rendered work item form.
		Optional Link element. Requires UrlPath is specified.
		Specifies a value to be used to the URL of the hyperlink when specified.
		The element type is complex type WebpageControlOptionsType .
		Optional WebpageControlOptions attribute.
	AllowScript	Specifies whether JavaScript on Web page is allowed to run (true) or not (false). The attribute type is xs:boolean.
	All other settings such as ActiveX based on the Internet Explorer	

defined for the client.

Optional WebpageControlOpti attribute.

ReloadOnParamChange Specifies whether to reload (true) contents of the Web page when parameter in the work item for changed (true) or not (false). The type is xs:boolean.

The default value is true.

Optional WebpageControlOpti element. This element is mutually exclusive with the Content element.

Link

Specifies the URL for the Web page rendered in the work item form.

For more information, see [Link XML Elements Reference](#).

Optional WebpageControlOpti element.

This element is mutually exclusive with the Link element.

Content

Specifies the HTML content that is rendered by the Web page control. Content is specified within a CDATA section. For example:

```
<![CDATA[Click here for detailed  
Guidance</a></b>]]>
```

The element type is xs:string.

Note

The HTML content is not valid to rendering it in the work item

Parent Elements

Element	Description
Control	Required parent element. Defines a field, text, hyperlink, or other control element to appear on the work item form.
	Optional LabelText element.
Text	Container element for the information or label to appear on the work item form.
	For more information, see LabelText and Text XML Elements Reference .

Remarks

The Link and Content elements are mutually exclusive.

The AllowScript and ReloadOnParamChange attributes are not allowed when you use the Content element.

When the Web page control is configured to display raw HTML, all scripting is disabled. Any hyperlinks in the content that are clicked are opened in the Visual Studio Web browser if the work item form is shown in Visual Studio or in Internet Explorer.

Several toolbar buttons are added to the work item form for a tab that contains WebpageControl. These options include the following functions:

- Back and forward navigation
- Stop content loading and refresh content
- Return to content home page
- Open the content in a new window or in an external browser

In addition, if the Web page to be loaded contains untrusted content, then the icon is blocked and the blocked icon appears.

The **WebpageControlOptions** element is only recognized by clients that are running the latest version of Team Explorer. For previous versions, a red area is displayed and a message appears indicating that the control could not be found. For more information, see Provide help text, hyperlinks, or web content on a work item form.

Example

The following example shows how to load a work item form with HTML defined in a CDATA tag.

 **Copy Code**

```
<Control Type="WebpageControl">
  <WebpageControlOptions>
    <Content>
      <![CDATA[Click here for detailed <a href="http://www.mic
    </Content>
  </WebpageControlOptions>
</Control>
```

Schema Definitions

WebpageControlOptionsType

 **Copy Code**

```
<xs:complexType name="WebpageControlOptionsType">
  <xs:choice minOccurs="1" maxOccurs="1">
    <xs:element name="Link" type="HyperLinkType"/>
    <xs:element name="Content" type="xs:string"/>
  </xs:choice>
  <xs:attribute name="AllowScript" type="xs:boolean" use="optional"/>
  <xs:attribute name="ReloadOnParamChange" type="xs:boolean" use="optional"/>
</xs:complexType>
```

NonBlankString

 **Copy Code**

```
<xs:simpleType name="NonBlankString">
  <xs:restriction base="xs:string">
    <xs:pattern value=".*\S+.*"/>
  </xs:restriction>
</xs:simpleType>
```

Element Information

Namespace <http://schemas.microsoft.com/VisualStudio/2008/workitemtracking/t>

Schema Name	Work Item Type Definition
Validation File	WorkItemTypeDefinition.xsd
Element Type	WebpageControlOptionsType
Can Be Empty	Not applicable

See Also

Concepts

Specify work item form controls
Design the work item form

Other Resources

[Provide help text, hyperlinks, or web content on a work item form](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Categories XML element reference

[Example](#) [See Also](#) [Send Feedback](#)

You use the **CATEGORIES** element to define one or more categories for grouping work item types.

```
<CATEGORIES>
  <CATEGORY name="category display name" refname="category referenc
    <DEFAULTWORKITEMTYPE name="work item type reference name" />
    <WORKITEMTYPE name="work item type reference name" />
  </CATEGORY>
</CATEGORIES>
```

Attributes and elements

Child elements and attributes

Element	Attribute	Description
CATEGORIES		Required element within the XML categories file.
		Container element for specifying one or more CATEGORY elements that are defined for a team project.
		Required CATEGORIES child element.
CATEGORY		Specifies a named category group that contains one default DEFAULTWORKITEMTYPE element and zero or more WORKITEMTYPE elements.
		Required CATEGORY attribute.
	name	The attribute type is typelib:FriendlyName.
		Specifies the display name of the category. Must be between 1 to 254 characters and unique within the team project.

	<p>Required CATEGORY attribute.</p>
	<p>The attribute type is typelib:ReferenceName.</p>
	<p>For more information, see Naming conventions for work item tracking objects.</p>
	<p>Required CATEGORY element.</p>
<p>DEFAULTWORKITEMTYPE</p>	<p>Specifies a work item type to be tagged as belonging to the named category and to be used as the default type for the category.</p>
	<p>Required DEFAULTWORKITEMTYPE attribute.</p>
	<p>The attribute type is WorkItemTypeRef.</p>
<p>name</p>	<p>Specifies the reference name of a valid work item type for the team project that will be used as the default type for the category.</p>
	<p>Optional CATEGORY element.</p>
<p>WORKITEMTYPE</p>	

Specifies a work item type to be tagged as belonging to the named category.

Required WORKITEMTYPE attribute.

name The attribute type is WorkItemTypeRef.

Specifies the reference name of a valid work item type for the team project that will belong to the named category.

Remarks

CATEGORIES is the root element of the categories schema.

A category is defined within the **CATEGORIES** set of **CATEGORY** elements that is stored and used by a team project. Each **CATEGORIES** element must have at least one **CATEGORY** element defined.

A category cannot be empty. Each **CATEGORY** element must have at least one **DEFAULTWORKITEMTYPE** or **WORKITEMTYPE** element defined.

Each category has a friendly name and a reference name that must be unique within the team project. Each category friendly name must meet the following requirements:

- Names can have up to 254 Unicode characters.
- Names must not be empty.
- Names cannot have leading or trailing white spaces.
- Names cannot contain backslash (\) characters.
- Names cannot contain two consecutive white spaces.

Category friendly names are not localized and cannot be made into a token, whereas the names for work item types are localized and can be made into a token.

For reference naming conventions, see [Naming conventions for work item tracking objects](#).

Agile tools, category requirements, and restrictions

You can specify which types of work items are considered as backlog items or as task items. For example, the Scrum process template assigns the Bug and Product Backlog Item types to the Requirements Category, whereas the Agile

template assigns User Story and the CMMI template assigns Requirement. All default process templates assign the Task type of work item to the Task Category.

When assigning work item types to categories, consider the following operational notes:

- To use the backlog and task boards, you must assign at least one work item type to the Requirements Category and one work item type to the Task Category.
- You cannot assign the same work item type to both the Requirements Category and to the Task Category.
- If you include more than one work item type in the Requirements Category or the Task Category, the type assigned to the DEFAULTWORKITEMTYPE element appears as the default type on the Agile backlog and board pages.
- For all work item types that you assign to a category that is referenced in the ProcessConfiguration file, you must assign the workflow states to a valid metastate as described in Process Configuration XML element reference. Several Team Foundation clients reference category and metastate assignments defined in the ProcessConfiguration file.

Example

The following example lists the default categories XML file for the Visual Studio Scrum 2013.3 process template.

 Copy C

```
<?xml version="1.0" encoding="utf-8"?>
<cat:CATEGORIES xmlns:cat="http://schemas.microsoft.com/VisualStudio
  <!-- Usage: for resilience in Test system. Even if WITs have differ
  <CATEGORY name="Bug Category" refname="Microsoft.BugCategory">
    <DEFAULTWORKITEMTYPE name="Bug" />
  </CATEGORY>
  <CATEGORY name="Feature Category" refname="Microsoft.FeatureCatego
    <DEFAULTWORKITEMTYPE name="Feature" />
  </CATEGORY>
  <CATEGORY name="Requirement Category" refname="Microsoft.Requireme
    <DEFAULTWORKITEMTYPE name="Product Backlog Item" />
    <WORKITEMTYPE name="Bug" />
  </CATEGORY>
  <CATEGORY name="Test Case Category" refname="Microsoft.TestCaseCat
    <DEFAULTWORKITEMTYPE name="Test Case" />
  </CATEGORY>
  <CATEGORY name="Shared Step Category" refname="Microsoft.SharedSte
    <DEFAULTWORKITEMTYPE name="Shared Steps" />
  </CATEGORY>
    <CATEGORY name="Shared Parameter Category" refname="Microsoft.Sha
    <DEFAULTWORKITEMTYPE name="Shared Parameter" />
  </CATEGORY>
  <CATEGORY name="Code Review Request Category" refname="Microsoft.(
    <DEFAULTWORKITEMTYPE name="Code Review Request" />
  </CATEGORY>
  <CATEGORY name="Code Review Response Category" refname="Microsoft.
    <DEFAULTWORKITEMTYPE name="Code Review Response" />
  </CATEGORY>
  <CATEGORY name="Feedback Request Category" refname="Microsoft.Fee
    <DEFAULTWORKITEMTYPE name="Feedback Request" />
  </CATEGORY>
  <CATEGORY name="Feedback Response Category" refname="Microsoft.Fee
    <DEFAULTWORKITEMTYPE name="Feedback Response" />
  </CATEGORY>
  <CATEGORY name="Test Plan Category" refname="Microsoft.TestPlanCat
    <DEFAULTWORKITEMTYPE name="Test Plan" />
  </CATEGORY>
  <CATEGORY name="Test Suite Category" refname="Microsoft.TestSuiteC
```

```
<DEFAULTWORKITEMTYPE name="Test Suite" />
</CATEGORY>
<CATEGORY name="Task Category" refname="Microsoft.TaskCategory">
  <DEFAULTWORKITEMTYPE name="Task" />
</CATEGORY>
<CATEGORY name="Hidden Types Category" refname="Microsoft.HiddenCa
  <DEFAULTWORKITEMTYPE name="Code Review Request" />
  <WORKITEMTYPE name="Code Review Response" />
  <WORKITEMTYPE name="Feedback Request" />
  <WORKITEMTYPE name="Feedback Response" />
  <WORKITEMTYPE name="Shared Steps" />
  <WORKITEMTYPE name="Shared Parameter" />
  <WORKITEMTYPE name="Test Plan" />
  <WORKITEMTYPE name="Test Suite" />
</CATEGORY>
</cat:CATEGORIES>
```

Element information

Namespace <http://schemas.microsoft.com/VisualStudio/2008/workitemtracking/>

Schema Name Categories

Validation File Categories.xsd

Element Type CATEGORIES, CATEGORY, DEFAULTWORKITEMTYPE, WORKITEMTYPE

Can Be Empty Yes

See Also

Other Resources

Use categories to group work item types



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Customize global workflow

[See Also](#) [Send Feedback](#)

You define a global workflow when you want to maintain a set of work item fields that multiple types of work items share. If you use a global workflow, you can define fields and global lists that are available to all work item types in either a team project or a team project collection.

In this topic

-

[Contents of a Global Workflow](#)

- [Defining and Managing Global Workflow](#)
- [Example Global Workflow XML File](#)

▣ Contents of a Global Workflow

You can define a global workflow in a similar way that you customize a work item type. To customize or create a global workflow, you define or modify an XML file.

📌 Note

No global workflows are defined for any Microsoft Solutions Framework (MSF) process templates. However, when you install Team Foundation Server Extensions for Project Server, a global workflow is imported to the team project collections that you configure to participate in data synchronization. For more information, see Synchronize Team Foundation Server with Project Server.

A global workflow can contain the following elements:

- **FIELD** (Definition). You can specify the definitions for fields and all qualifying and conditional rules for each field. These rules are contained within a **FIELDS** parent element.
- **GLOBALLIST**. You can specify one or more global lists as part of the global workflow. These lists are contained within a **GLOBALLISTS** parent element.

If you specify a field that is not currently defined for the team project or the collection, the field is created when the global workflow is imported. You cannot change the properties of fields that have already been defined for a type of work item. For example, you cannot change the attributes that are associated with the friendly name, the reference name, or reporting by modifying the global workflow.

A global workflow cannot contain the following elements and definitions:

- **HELPTTEXT**. You cannot specify help text for a field.
- **WORKFLOW**. You cannot specify a workflow that consists of states,

transitions, and reasons. Workflow is always scoped to a type of work item.

- **FORM.** You cannot specify the layout of the work item form because it is always scoped to a type of work item.

Global workflow elements belong to the
<http://schemas.microsoft.com/VisualStudio/2008/workitemtracking/globalworkfl>
namespace.

[Back to top](#)

Defining and Managing Global Workflow

Task	Topic
Look up the syntax of a global workflow element. You define global workflow by using the FIELD (Definition) and GLOBALLIST elements and their child elements.	

[Global Workflow XML Element Reference](#)

Create, modify, export, or import a global workflow. You manage global workflow by using the following commands in the **witadmin** command-line tool:

- **exportglobalworkflow**: Exports the XML file that define a global workflow.
- **importglobalworkflow**: Imports the XML file that defines a global workflow.

[Import and export global workflow \[witadmin\]](#)

[Back to top](#)

Example Global Workflow XML File

The following example shows how you can define two fields within the global workflow by using the **FIELD** (Definition) element and its supported rule elements. For more information, see

[All FIELD XML elements reference.](#)



```
<?xml version="1.0" encoding="utf-8"?>
<GLOBALWORKFLOW>
  <FIELDS>
    <FIELD name="Project Server Enterprise Project" refname="Micro
      <ALLOWEDVALUES not="[global]\Project Collection Service Acc
        <LISTITEM value="PsProject" />
      </ALLOWEDVALUES>
      <DEFAULT not="[global]\Project Collection Service Accounts'
    </FIELD>
    <FIELD name="Project Server Requested Project GUID" refname="M
      <ALLOWEDVALUES not="[global]\Project Collection Service Acc
        <LISTITEM value="3aabbaa9-f204-4748-897c-47904155bc61" /
      </ALLOWEDVALUES>
      <DEFAULT not="[global]\Project Collection Service Accounts'
      <WHEN field="Microsoft.Sync.ProjSrv.ProjectName" value="PSF
        <ALLOWEDVALUES not="[global]\Project Collection Service
          <LISTITEM value="3aabbaa9-f204-4748-897c-47904155bc61
        </ALLOWEDVALUES>
          <COPY from="value" value="3aabbaa9-f204-4748-897c-479041
        </WHEN>
      </FIELD>
    </FIELDS>
  </GLOBALWORKFLOW>
```

[Back to top](#)

See Also

Concepts

[All FIELD XML elements reference](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Global Workflow XML Element Reference

[See Also](#) [Send Feedback](#)

By using global workflow, you can minimize the work that is required to define and update work item fields and global lists that many types of work items and team projects share. With global workflow, you can define and update fields and global lists that apply to all types of work items in a team project or a team project collection.

If you want only to manage global lists for a collection, see

[Define global lists.](#)

Global Workflow Syntax Structure

You define the global workflow by using the GLOBALWORKFLOW element, which supports **FIELDS** (Definition) or **GLOBALLISTS** elements as children, but not both.

Specify FIELDS:

 **Copy Code**

```
<?xml version="1.0" encoding="utf-8"?>
<GLOBALWORKFLOW>
  <FIELDS>
    <FIELD> . . . </FIELD>
  </FIELDS>
</GLOBALWORKFLOW>
```

Specify GLOBALLISTS:

 **Copy Code**

```
<?xml version="1.0" encoding="utf-8"?>
<GLOBALWORKFLOW>
  <GLOBALLISTS>
    <GLOBALLIST> . . . </GLOBALLIST>
  </GLOBALLISTS>
</GLOBALWORKFLOW>
```

FIELD (Definition) Element

You use the following syntax to define the data fields within a global workflow. This syntax shows the **FIELD** (Definition) element format and all optional child elements. For more information, see

[FIELD \(Definition\) element reference](#) and [All FIELD XML elements reference](#).

Note

You cannot specify the **HELPTEXT** element for a field that you define in a global workflow.

```
<FIELD name="fieldDisplayName" refname="fieldReferenceName" type="Si
syncnamechanges="true | false" reportingname="reportingDisplayName"
reportable="Dimension | Detail | Measure" formula="avg" >
  <ALLOWEDVALUES> . . . </ALLOWEDVALUES>
  <ALLOWEXISTINGVALUE />
  <CANNOTLOSEVALUE />
  <COPY />
  <DEFAULT />
  <EMPTY />
  <FROZEN />   <MATCH />
  <NOTSAMEAS />
  <PROHIBITEDVALUES /> . . . </PROHIBITEDVALUES>
  <READONLY />
  <REQUIRED />
  <SERVERDEFAULT />
  <SUGGESTEDVALUES /> . . . </SUGGESTEDVALUES>
  <VALIDUSER />
  <WHEN>> . . . </WHEN>
  <WHENNOT> . . . </WHENNOT>
  <WHENCHANGED> . . . </WHENCHANGED>
  <WHENNOTCHANGED> . . . </WHENNOTCHANGED>
</FIELD>
```

[Back to top](#)

GLOBALLIST and LISTITEM Child Elements

The following table describes the **GLOBALLIST** and **LISTITEM** elements. You specify these elements as child elements of the **GLOBALWORKFLOW** element. You can use these elements to enumerate a list of values that appears to the user as a pick list or a drop-down menu of items. For more information, see

[Define pick lists.](#)

Element	Description and Syntax
GLOBALLIST	<p>Defines a set of LISTITEM elements that are stored for a team project collection or a team project.</p> <p>globalListName: A string of text that contains between 1 and 255 characters.</p> <p> Copy Code</p> <pre data-bbox="435 1289 1057 1388"><GLOBALLIST name="globalListName"> <LISTITEM> . . . </LISTITEM> </GLOBALLIST></pre>
LISTITEM	<p>Defines a valid value that appears in the list.</p> <p> Copy Code</p>

```
<LISTITEM value="listName" />
```

LISTITEM is a required child element
of **GLOBALLIST**.

See Also

Concepts

[All FIELD XML elements reference](#)

Customize work tracking objects to support your team's processes

Other Resources

[Customize global workflow](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Link type element reference

[See Also](#) [Send Feedback](#)

You use different link types to form link relationships between different work item types (WITs). There are three categories of link types: system-defined, process-template defined, and user-defined. The link types defined in the default process templates support link relationships among the test management WITs.

Each link type defines the link labels, topology type, and restrictions that are used when links between work items are constructed. For example, the parent-child link type defines two labels (Parent and Child), supports a hierarchical or tree topology, and prevents circular references from being created between work items.

To customize or create a link type, use **witadmin importlinktype** to import the link type definition file to the project collection that hosts your team project.

Syntax structure

You can define additional link types by adding them to the link types that are defined for a team project collection. You can define valid link types for use in your process based on the structure that the link type definition schema provides. A link type is defined by the following XML syntax in the link types XML file:

 **Copy Co**

```
<LinkTypes>
  <LinkType ReferenceName="LinkTypeName" ForwardName="ForwardName"
</LinkTypes>
```

The descriptions in the following table apply to the previous syntax:

Attribute	Description
ReferenceName	Name of the link type. This name is used internally when you create a link between two work items.
ForwardName	Name of the link at the source work item. This name appears when you add links to the source work item.
ReverseName	Name of the link at the target work item. This name appears when a listing of the links at

the target work item appears.

TopologyType

Specifies the DirectedNetwork, Network, Tree, or Dependency topology. The first three topologies are directional, and you use them to define subordinate or sequential relationships. You use Network to define relationships between peers or where no implied subordination exists.

Link directionality is determined by the assignments made to the ForwardName and ReverseName attributes. If you create a custom link and assign the same name to the forward and reverse names, you should set the link type to Network because it is the only non-directional topology.

Requirements

LINKTYPES is a set of **LINKTYPE** elements that are stored and used by a team project collection.

Each link type has a reference name and two optional friendly names, or name labels, which must be unique within the project collection. Each link type name must meet the following requirements:

- Names can have up to 254 Unicode characters.
- Names must not be empty.
- Names cannot have leading or trailing white spaces.
- Names cannot contain backslash (\) characters.
- Names cannot contain two consecutive white spaces.

System-defined link types

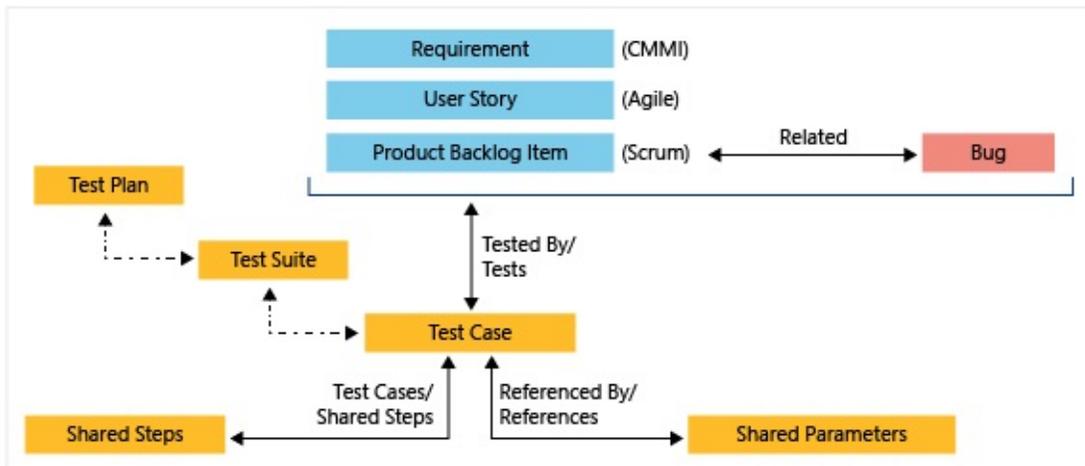
The following table summarizes the system-defined link types. You cannot modify these link types.

Forward Name	Reverse Name	Reference name	Topology
Successor	Predecessor	System.LinkTypes.Dependency	Dependency
Child	Parent	System.LinkTypes.Hierarchy	Tree
Related	Related	System.LinkTypes.Related	Network

In addition, there are additional link types used to link work items to other objects in the team project or other elements, such as a web page or network location. For example, the Storyboard link type links work items to storyboards or any file located on a network share. For more information, see [Link work items to support traceability](#).

Process-template defined link types

When you create a team project using one of the default process templates that TFS provides, the following link types are added to your team project. These link types are specifically designed to support interaction with tracking work items in Team Foundation and Microsoft Test Manager.



Forward Name	Reverse Name	Link type reference name
Affects (See Note 1)	Affected By	Microsoft.VSTS.Common.Affects
Referenced By (See Note 2)	References	Microsoft.VSTS.TestCase.SharedParameterReferencedBy
Tested By	Tests	Microsoft.VSTS.Common.TestedBy
Test Case	Shared	Microsoft.VSTS.TestCase.SharedStepReferencedBy

Steps

Notes

1. The Affects/Affected By link type links Change Requests to Requirements. This link type is only added to team projects that are created with a CMMI process template.
2. The Referenced By/References link type links Shared Parameter with Test Case to support running tests with different data. This link type is added to a team project when you update the team project using the Configure Features wizard or you create a team project based on a TFS 2013.2 process templates or later version.

Link type definition files are defined in the WorkItem Tracking\LinkTypes folder of the Template.zip file. The following definition for the Microsoft.VSTS.Common.TestedBy link type is defined in the TestedBy.xml file:

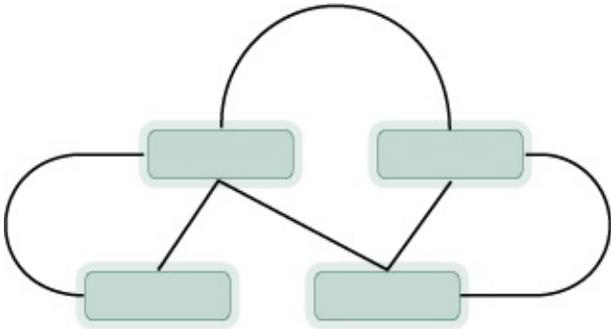


```
<?xml version="1.0" encoding="utf-8"?>
<LinkTypes>
  <LinkType ReferenceName="Microsoft.VSTS.Common.TestedBy" ForwardL
</LinkTypes>
```

When you create the team project by using the corresponding process template, the link type definition is imported into the project collection.

Link restrictions and topologies

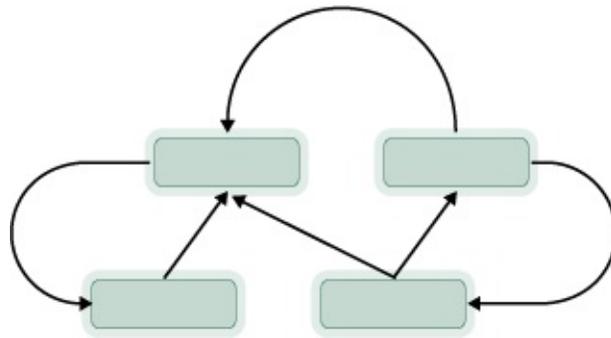
The topology types described in the following table determine the restrictions placed on the usage of each link type.

Topology type	Illustration
<p>Network: You can use network links to create basic relationships between work items that are non-restrictive. The link is the same at both end points. Circular relationships are allowed.</p>	 <p>The diagram shows four light green rounded rectangular nodes arranged in a 2x2 grid. The top-left and top-right nodes are connected by a curved line above them. The bottom-left and bottom-right nodes are connected by a curved line below them. The top-left node is connected to the bottom-left node by a straight line. The top-right node is connected to the bottom-right node by a straight line. Additionally, the top-left node is connected to the bottom-right node by a diagonal straight line, and the top-right node is connected to the bottom-left node by a diagonal straight line.</p>
<p>Example usage: Use a network link, such as Related, to record a relationship between two features that</p>	

might share dependencies.

Directed Network: You can use directed network links to create relationships between work items that indicate directionality. The link name is different at the end points. Circular relationships are allowed.

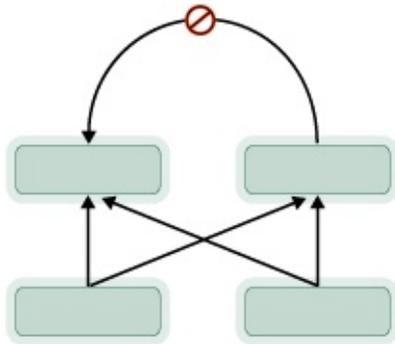
Example usage: Use a directed network link to record a relationship between two features that might share dependencies and which you want to distinguish from each other in some



way.

Dependency:
You can use dependency links to create relationships between work items that have directionality and to restrict circular relationships. The link name is different at the end points.

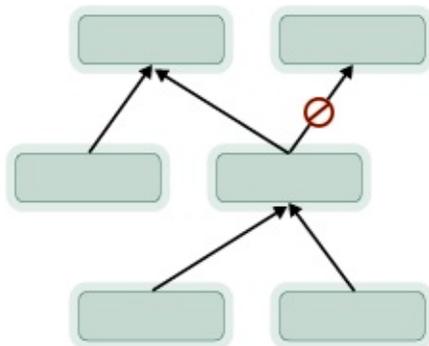
In the illustration, you cannot create a dependent link to a work item that contains dependent link relationships to the same work items.



Example usage: Use a dependency link to record

the features that must be completed to deliver a user requirement.

Tree: You can use tree links to create multi-level hierarchical relationships among work items. Tree links support multi-level hierarchical views, have directionality, and restrict circular relationships. The link name is different at the end points. Tree links are the only type of link that is supported by the Tree of Work Items query.



In the illustration, you cannot

assign two
parents to a
child.

Example
usage: Use a
tree link to
record tasks
and subtasks
from your
team that
must be
completed to
deliver a
feature.

Link types schema definition

The following code lists the schema definition for link types:

 [Copy Code](#)

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
  id="WorkItemLinkTypeDefinition"
  elementFormDefault="unqualified"
  attributeFormDefault="unqualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://schemas.microsoft.com/VisualStudio/2005/workitemtrac
  xmlns="http://schemas.microsoft.com/VisualStudio/2005/workitemtrac
  xmlns:mstns="http://schemas.microsoft.com/VisualStudio/2005/workit
  xmlns:typelib="http://schemas.microsoft.com/VisualStudio/2005/work
  version="1.0">
  <xs:import namespace="http://schemas.microsoft.com/VisualStudio/2
  <xs:simpleType name="TopologyTypes">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Network" />
      <xs:enumeration value="DirectedNetwork" />
      <xs:enumeration value="Dependency" />
      <xs:enumeration value="Tree" />
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="LinkTypeType">
    <xs:attribute name="ReferenceName" type="typelib:ReferenceName"
    <xs:attribute name="Topology" type="TopologyTypes" use="optional
    <xs:attribute name="ForwardName" type="typelib:FriendlyName" use
    <xs:attribute name="ReverseName" type="typelib:FriendlyName" use
  </xs:complexType>
  <xs:complexType name="LinkTypesType">
    <xs:sequence>
      <xs:element name="LinkType" type="LinkTypeType" minOccurs="1'
    </xs:sequence>
  </xs:complexType>
  <xs:element name="LinkTypes" type="LinkTypesType" />
</xs:schema>
```

Q & A

Q: How do I add a custom link type?

A: Define an XML definition file according to the content provided in this topic and then use **wiadmin importlinktype** to import the link type definition file to the project collection that hosts your team project.

Q: What link type is used to link test cases to test suites and test suites to test plans?

A: Using Team Web Access or Test Manager, you can view which test cases are defined for a test suite, and which test suites are defined for a test plan. However, these objects aren't linked to each other through link types.

See Also

Concepts

Customize work tracking objects to support your team's processes

Other Resources

[Link work items to support traceability](#)



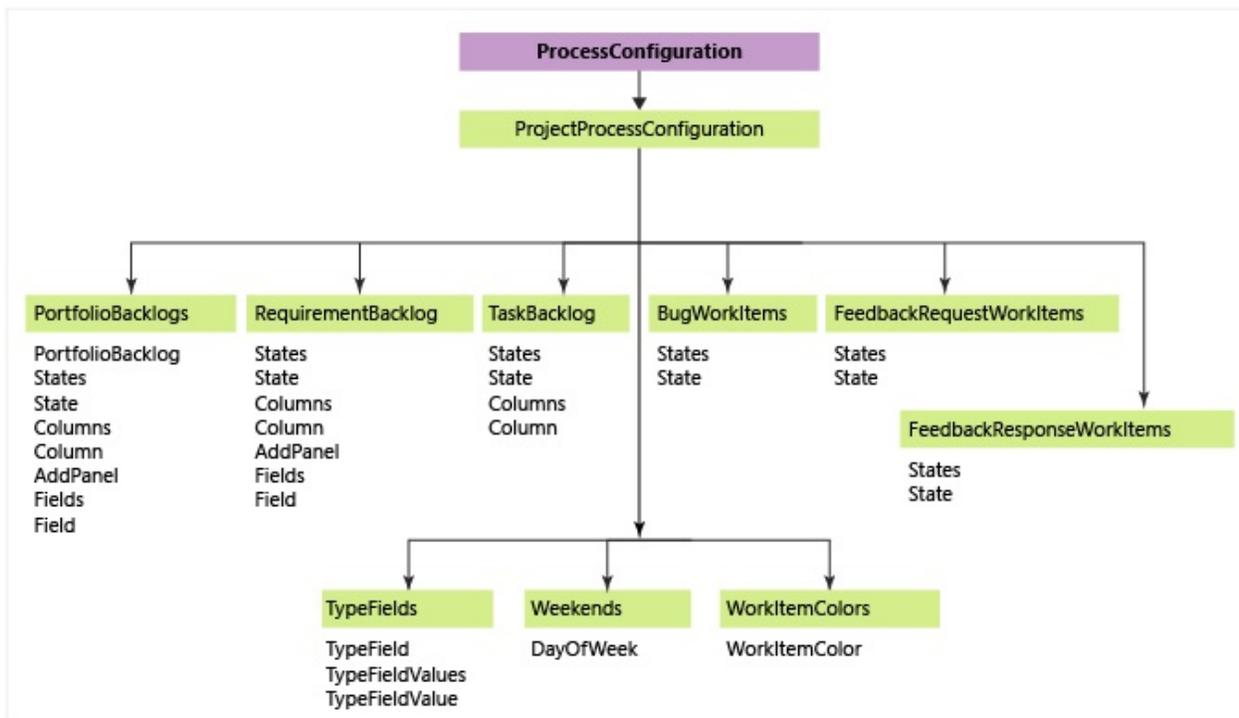
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Process configuration XML element reference

[Send Feedback](#)

Process configuration defines the default configuration and functional capabilities that your teams can access using the Agile planning tools.

These tools, which you view through Team Web Access (TWA), display a filtered set of work items based on the configuration made to the PortfolioBacklog, RequirementBacklog, and TaskBacklog sections of the process configuration XML definition file. In addition, process configuration defines the workflow states-to-metastate mappings for all work item types (WITs) that require mapping.



To learn more, see [Configure and customize Agile planning tools for a team project](#).

Areas that you can customize:

-

[Configure a backlog page](#)

- [Map metastates for a category of work item types](#)
- [Customize the default columns and column sequence](#)
- [Customize the quick add panel](#)
- [Change the number of work items that can appear on the task board](#)
- [Map metastates for tool-specific work item types](#)
- [Assign fields used in agile planning tools and charts](#)
- [Specify weekend days](#)
- [Change the color for a work item type](#)

To update the process configuration, you export the XML definition file, edit it, and then import the file. You use the **witadmin** command line tool to import and export the file.



Note

The syntax samples shown in this topic correspond to the default assignments defined in the Visual Studio Scrum 2013 process template. To access the latest version of the process templates, install the latest version of TFS and download the templates using the Process Template Manager.

Configure a backlog page

There are three types of backlog pages: product backlog, iteration or sprint backlog, and portfolio backlog. You can customize each backlog page in the following ways:

- **Metastate mappings:** Map workflow states to metastates. These mappings support the display of all agile planning pages, including the Kanban board and task board.
- **Quick add panel:** Specify the WITs and work item fields that appear for quickly adding items to the backlog.

To change the types of work items that are considered backlog items or tasks, you add them to the corresponding category. For an example, see [Add bugs to the task board or backlog](#).

- **Column fields:** Define the default fields and column sequence.

You configure the backlog pages within the XML sections that appear in the following sample:

 **Copy Code**

```
<PortfolioBacklogs>
  <PortfolioBacklog category="Microsoft.FeatureCategory" pluralName
  . . .
  </PortfolioBacklog>
</PortfolioBacklogs>
<RequirementBacklog category="Microsoft.RequirementCategory" pluralName
  . . .
</RequirementBacklog>
<TaskBacklog category="Microsoft.TaskCategory" pluralName="Tasks" si
  . . .
</TaskBacklog>
```

Element

PortfolioBacklogs Optional. Container element for portfolio backlog pages.

Optional. Up to five instances.

Container element that defines the metastate mappings, def

```
<PortfolioBacklog category="PortfolioCategory" pa
  <States> . . . </States>
  <Columns> . . . </Columns>
  <AddPanel> . . . </ AddPanel>
</PortfolioBacklog >
```

PortfolioBacklog

Assign values to the attributes as described:

- category: Specify the name of a category that you have associated with this backlog type.
- parent: Specify the name of the category that represents
- pluralName: Specify the plural label to use when referring to Initiatives, or Epics.
- singularName: Specify the singular label to use when referring to Initiative, or Epic.

Required. One instance only.

Container element that defines the metastate mappings, def displays all active items in the team's backlog.

RequirementBacklog

```
<RequirementBacklog category="RequirementCategory"
  <States> . . . </States>
  <Columns> . . . </Columns>
  <AddPanel> . . . </ AddPanel>
</RequirementBacklog >
```

Required. One instance only.

Container element used to customize the layout of pages th

TaskBacklog

```
<TaskBacklog category="Microsoft.TaskCategory" p1  
.  
.</TaskBacklog >
```

By default, the task board is restricted to a total of 500 wor
attribute.

Implementation notes

- The values assigned to CategoryName must correspond to a category group defined for the team project. You specify category groups in the definition file for Categories. For more information, see

[Categories XML element reference.](#)

- You use portfolio backlogs to view the rollup of backlog items at lower levels and to view progress across several teams. New and upgraded team projects contain one level labeled Features. You can add up to four additional levels.

Note

Use of the portfolio pages may require you to be granted Advanced access. For details, see [Change access levels.](#)

For information about using portfolio backlog pages, see [Work with portfolio backlogs.](#)

- The product backlog represents a list of requirements for the product that you are developing. Backlog items correspond to a specific type of work item based on the process template used to create your team project, such as product

backlog item, user story, or requirement. If you use different types of work items or capture your requirements using two or more types of work items, then you can customize the product backlog page to support your usage.

For information about using the product backlog pages, see [Create your backlog](#).

- The sprint or iteration backlog pages display both the set of requirements that you and your team have committed to in a specific iteration cycle and the tasks that you have linked to those requirements. Tasks must be linked to requirements using the child link type. Because the types of work items that appear on these pages correspond to the same types that appear on the product backlog page, much of the customization work that you do for the product backlog page will define the functionality of the task backlog pages.

For information about using the sprint backlog pages, see [Work in sprints](#).

☒ **Map workflow states to metastates**

Most WITs require their workflow states to be mapped to a metastate. Workflow states define how a work item progresses from first activation or creation to closed or complete. For example, the states defined for the Scrum product backlog item define a progression of four states, from New, Approved, Committed, to Done, and also includes a fifth state, Removed, to account for a state removed from the backlog without being implemented.

Metastates, on the other hand, determine how the agile planning tools treat each workflow state. The primary metastates used by the backlog and task board are Proposed, InProgress, and Complete.

By associating each workflow state to a metastate, the background operations performed to display the backlog and task boards know how to correctly interpret the status of each work item. For example, the following mappings are defined for the Scrum product backlog pages

 **Copy Code**

```
<RequirementBacklog category="Microsoft.RequirementCategory" pluralName="Requirements"
  <States>
    <State value="New" type="Proposed" />
    <State value="Approved" type="Proposed" />
```

```

        <State value="Committed" type="InProgress" />
        <State value="Done" type="Complete" />
    </States>
</RequirementBacklog >

```

There are three categories of metastates: Agile, Bug, and Feedback. The following table describes the mapping attributes and values.

Element	Description
	Required. Assigns a workflow state to a metastate.
	 Copy Code
	<pre><State type="TypeName" value="ValueName"/></pre>
	Valid values for TypeName correspond to a value assigned to a STATE within the WORKFLOW section of those WITs assigned to the category group.
	Valid values for ValueName correspond to one of the following enumerated values:
	<ul style="list-style-type: none"> • Agile: Use for all work item types. <ul style="list-style-type: none"> ◦ Proposed: Indicates work items that are new, not yet committed, or not yet being worked on. ◦ InProgress: Indicates work items that have been committed or are actively being worked on. ◦ Complete: Indicates work items that have been implemented. For the Kanban

State

board to be valid, at least one workflow state must be mapped to the Complete metastate.

Once a workflow state transitions to a state that is associated with the Complete metastate, the associated work item will fall off the product backlog items page. However, it will continue to be listed on the Kanban board.

Work items in a workflow state that are not mapped to one of the supported metastates do not appear on the backlog or board pages.

- **Bug:** Use only for work item types grouped within the Bug Category. In addition to the Agile metastates, includes the Resolved metastate which indicates bugs that have been resolved.

Note

You can only assign the Resolved metastate to a workflow state specified under the BugWorkItems element.

- **Feedback:** Use only for work item types grouped within the Feedback Request or Feedback Response categories. Requested, Received, Reviewed, and Declined.

Specifies a collection of **State** elements that associate WIT workflow states with metastates.

Required element for the following parent elements:

- BugWorkItems
- PortfolioBacklog
- RequirementBacklog
- TaskBacklog
- TestPlanWorkItems
- TestSuiteWorkItems
- FeedbackRequestWorkItems
- FeedbackResponseWorkItems

States

≡ Customize the default columns and column sequence

You can add or remove columns, change the sequence of the columns, or change the column width for the pages that display a backlog page. Changes you make to the page through the Column Options dialog persist until you change them again. The following section of code simply defines the default column set and sequence.

Order	Work Item Type	Title	State	Eff...	Iter...	Tags
1	Product Backlog Item	Add recent items	New		Fabri...	
2	Bug	Avatar doesn't show up in...	New		Fabri...	
3	Product Backlog Item	Change preferences	New		Fabri...	website-related
4	Product Backlog Item	Remember login	New		Fabri...	website-related

📄 Copy Code

```
<Columns>
  <Column refname="Microsoft.VSTS.Common.Priority" width="400" />
  <Column refname="System.Title" width="400" />
  <Column refname="System.State" width="100" />
</Columns>
```

```

    <Column refname="Microsoft.VSTS.Scheduling.Effort" width="50" />
    <Column refname="System.IterationPath" width="200" />
</Columns>

```

Element	Description
Columns	Specifies a collection of Column elements. Required element for the backlog page elements: PortfolioBacklog, RequirementBacklog, and TaskBacklog.
Column	Specifies a field to appear as a column on a backlog page.
	 Copy Code
	<pre><Column refname="FieldReferenceName" width="FieldWidth" /></pre>

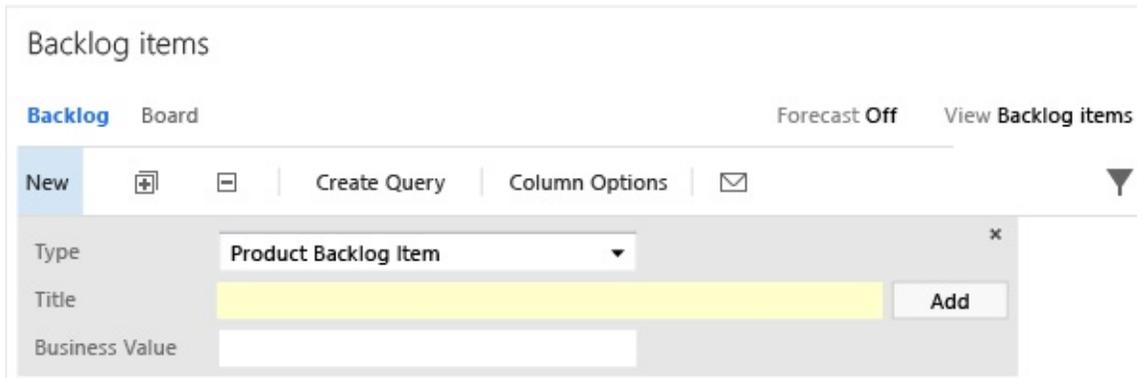
Task board column headings

The column headings that appear on the task board page correspond to the workflow states assigned to the default WIT assigned to the Task Category. The column sequence corresponds to the natural progression of the workflow transitions, moving from left to right. To modify the column layout, you modify the workflow for the WIT assigned to the Task Category. The workflow states defined for the default task type in the Task Category must be assigned to a valid metastate as described in

[Map metastates for a category of work item types.](#)

☰ Customize the quick add panel

You can add fields for any quick add panel. For example, the following example adds Business Value to the product backlog panel.



The panel only displays fields that are included in the FIELDS section of the WIT definition for the WIT selected. For example, if you select the bug WIT, then only Title displays, because Business Value isn't defined for bugs. To add another WIT to the panel, you add it to the Requirements Category as described here.

The following code corresponds to the default assignments defined in the Visual Studio Scrum and MSF for Agile process templates.

Copy Code

```
<AddPanel>
  <Fields>
    <Field refname="System.Title" />
  </Fields>
</AddPanel>
```

Element	Description
AddPanel	Container element used to specify the "quick add" experience, the fields to appear within the panel area where new backlog items are defined.
Fields	Specifies a collection of Field elements.
	Specifies a work item field to appear within

the panel for the product backlog page.

Copy Code

Field

```
<Field refname="FieldReferenceName"/>
```

The same field should appear on the work item form of each WIT included in the category for the backlog.

Change the number of work items that can appear on the task board

For performance reasons, the task board is restricted to display a maximum of 500 work items. When you open the task board, all work items are loaded into cache. Limiting the number of work items may yield quicker load times. You can change this limit by specifying a value for the `workItemCountLimit` attribute of the `TaskBacklog` element.

For example, you can increase the limit by adding `workItemCountLimit="800"`:

Copy Code

```
<TaskBacklog category="Microsoft.TaskCategory" pluralName="Tasks" s:  
.  
.  
</TaskBacklog>
```

Map metastates for tool-specific work item types

Metastate mappings are defined for additional WIT categories. For the Scrum process template, this includes mappings for the feedback request and response categories. For the MSF Agile and CMMI process templates, it also includes mappings for the bug category. (Scrum includes bugs in the Requirement Category and therefore defines the metastate mappings within the RequirementBacklog section.)

```
<FeedbackRequestWorkItems category="Microsoft.FeedbackRequestCategory"
  <States>
    <State value="Active" type="InProgress" />
    <State value="Closed" type="Complete" />
  </States>
</FeedbackRequestWorkItems>
<FeedbackResponseWorkItems category="Microsoft.FeedbackResponseCategory"
  <States>
    <State value="Active" type="InProgress" />
    <State value="Closed" type="Complete" />
  </States>
</FeedbackResponseWorkItems>
```

The following table describes the additional elements used to define the metastate mappings for tool-specific work item types. See

[Map metastates for a category of work item types](#) for information about assigning the actual state values and types. The CategoryName must correspond to a category defined for the team project.

Element

Optional. Container element that defines the meta: how these mappings are used in the display of TW

Explorer updates the bug state as developers move
Write new code for a user story.

BugWorkItems

```
<BugWorkItems category="CategoryName" plus  
  <States>  
    . . .  
  </States>  
</BugWorkItems>
```

Required. Do not customize.

Container element that defines the metastate map

FeedbackRequestWorkItems

```
<FeedbackResponseWorkItems category="Cate  
  <States>  
    . . .  
  </States>  
</FeedbackRequestWorkItems>
```

Required. Do not customize.

Container element that defines the metastate map

FeedbackResponseWorkItems

```
<FeedbackResponseWorkItems category="Cate  
  <States>  
    . . .  
  </States>  
</FeedbackResponseWorkItems>
```

Only required when you customize the workflow :
of Test Manager installed with Visual Studio 2013

Container element that defines the metastate map

TestPlanWorkItems

```
<TestPlanWorkItems category="Microsoft.Te
  <States>
    <State type="InProgress" value="Des:
    <State type="InProgress" value="Test
    <State type="Complete" value="Signe
  </States>
</TestPlanWorkItems>
```

Only required when you customize the workflow of Test Manager installed with Visual Studio 2013

Container element that defines the metastate map

TestSuiteWorkItems

```
<TestSuiteWorkItems category="Microsoft.Te
  <States>
    <State type="Proposed" value="Author
    <State type="InProgress" value="Test
    <State type="Complete" value="Comple
  </States>
</TestSuiteWorkItems>
```

To map metastates for TestPlanWorkItems or TestSuiteWorkItems, you must upgrade your application-tier server to TFS 2013.3. Afterwards, you can customize the workflow state of test plans and test suites.

To learn more, see [Import and export process configuration](#).

Assign fields used in agile planning tools and charts

You can change the work item fields that are used in calculating capacity, burndown charts, forecasting, and velocity. Any change you make to one of the default assignments should correspond to a change made to the WIT used to define and capture information for that value.

For example, if you change the refname assigned to type="Activity" then you should include the same field in the WIT definition assigned to the Task Category which captures the activity information.

 Copy Code

```
<TypeFields>
  <TypeField refname="System.AreaPath" type="Team" />
  <TypeField refname="Microsoft.VSTS.Scheduling.RemainingWork" type="Team" />
  <TypeField refname="Microsoft.VSTS.Common.BacklogPriority" type="Team" />
  <TypeField refname="Microsoft.VSTS.Scheduling.Effort" type="Effort" />
  <TypeField refname="Microsoft.VSTS.Common.Activity" type="Activity" />
  <TypeField refname="Microsoft.VSTS.Feedback.ApplicationStartInfo" type="Team" />
  <TypeField refname="Microsoft.VSTS.Feedback.ApplicationLaunchInfo" type="Team" />
  <TypeField refname="Microsoft.VSTS.Feedback.ApplicationType" type="Team" />
  <TypeFieldValues>
    <TypeFieldValue value="Web application" type="WebApp" />
    <TypeFieldValue value="Remote machine" type="RemoteMachine" />
    <TypeFieldValue value="Client application" type="ClientApplication" />
  </TypeFieldValues>
</TypeField>
</TypeFields>
```

Element	Description
TypeFields	Required. Specifies a collection of TypeField elements.
	Required. Specifies the reference name of a field whose value

The fields you specify should correspond to the fields that you feature information.

 **Copy Co**

```
<TypeField refname="FieldReferenceName" type="NameOf
```

Specify the format only when type="RemainingWork". You can specify a TimeUnitString that you want to have appear on the capacity board on the task board.

For agile planning pages:

- **Activity:** Used to support the capacity-by-activity feature. Assigned to the Task Category.

Note: The values displayed in the Agile planning tool Capacity for the field in all team projects within the project collection that appear for Capacity on the sprint backlog pages, you specify projects for the field assigned to type="Activity".

- **Effort:** Used to calculate the team velocity. Specify the same Requirement Category that you use to capture the estimated amount of work that a backlog item requires to implement.
- **Order:** Used to define the sort order for items on the backlog page according to the ascending order as defined by the field.

Note

You can move items by dragging them up or down the list. The background process updates the field assigned to the type.

- **RemainingWork:** Used to calculate remaining work and by the WIT assigned to the Task Category which you use to capture measurement that remain to finish a task.

The value that you specify for format is used on the backlog

TypeField

work is reported. For example, when reporting capacity-b next to the column heading for the task states on the task

For TimeUnitString, specify any text string that you want or days.

For example, the following values are all valid:

```
format="{0} h"
```

```
format="{0} hours"
```

```
format="hours {0}"
```

```
format="time {0}"
```

- **Team:** Used to associate the backlog pages with a team. To decouple teams from area paths, you can specify a different support Agile planning tools.

For the feedback request form:

Note

You should not have to change the default assignments made if assignments correspond to the fields used to capture the correct the Feedback Request Category.

- **ApplicationStartInformation:** Used to capture the path to
- **ApplicationLaunchInstructions:** Used to capture launch in
- **ApplicationType:** Used to capture the type of application. values specified in the WIT definition for the feedback request pick list.

Required for the TypeFieldValue when type="ApplicationType

TypeFieldValues

Specifies a collection of **TypeFieldValue** elements which are u

Required. Do not customize.

Specifies the name of an application type to appear on the feed

 **Copy Co**

```
<TypeFieldValue value="ApplicationTypeName" type="Ty
```

TypeFieldValue The default assignments correspond to the allowed values spec request form.

 **Copy Co**

```
<TypeFieldValues>  
  <TypeFieldValue value="Web application" type="web  
  <TypeFieldValue value="Remote machine" type="Remo  
  <TypeFieldValue value="Client application" type=""  
</TypeFieldValues>
```

Implementation notes

- If you change a field within the **TypeFields** section, you should make the corresponding change in the WIT definition. For example, if you change the fields assigned to capture work Effort, then you should make the same change in the WIT definitions for the product backlog item and bug (for Scrum).
- You can look up the reference name for a field using this index.

Assign non-working days

The capacity planning and burndown charts reference the non-working days. The following non-working days are defined within each TFS process template.

Copy Code

```
<Weekends>  
  <DayOfWeek>Saturday</DayOfWeek>  
  <DayOfWeek>Sunday</DayOfWeek>  
</Weekends>
```

Element	Description
	Required child of the Weekends element.
	Specifies a day of the week that corresponds to a non-working day.

Copy Code

```
<DayOfWeek>NameOfADay</DayOfWeek>
```

DayOfWeek

Valid names correspond to the English days of the week: Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, and Saturday.

Note

You must specify the day of a week in English, regardless of the installed language of Team Foundation Server.

Optional. Container element used to specify non-working days.

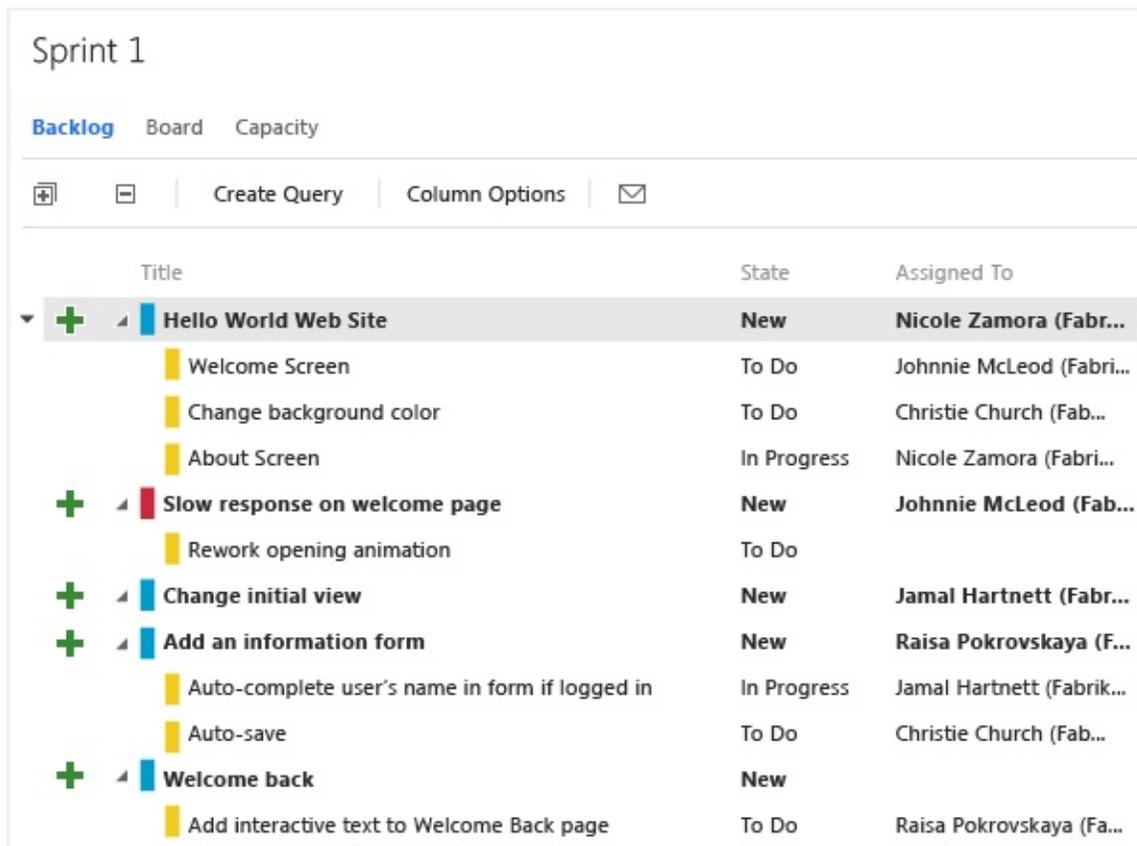
Weekends

Specify non-working days when you want to account for non-working days in the calculation of capacity and burndown charts.

Some note about how this appears in the burndown chart and cannot be removed.

Change the color for a work item type

At a glance, you can differentiate WITs when viewing a query result or backlog based on the color assigned to the WIT.



Title	State	Assigned To
+ ▸ ■ Hello World Web Site	New	Nicole Zamora (Fabr...
■ Welcome Screen	To Do	Johnnie McLeod (Fabr...
■ Change background color	To Do	Christie Church (Fab...
■ About Screen	In Progress	Nicole Zamora (Fabr...
+ ▸ ■ Slow response on welcome page	New	Johnnie McLeod (Fab...
■ Rework opening animation	To Do	
+ ▸ ■ Change initial view	New	Jamal Hartnett (Fabr...
+ ▸ ■ Add an information form	New	Raisa Pokrovskaya (F...
■ Auto-complete user's name in form if logged in	In Progress	Jamal Hartnett (Fabrik...
■ Auto-save	To Do	Christie Church (Fab...
+ ▸ ■ Welcome back	New	
■ Add interactive text to Welcome Back page	To Do	Raisa Pokrovskaya (Fa...

The following color assignments are defined within the Scrum process template.

Copy Code

```
<WorkItemColors>
  <WorkItemColor primary="FF009CCC" secondary="FFD6ECF2" name="Proc
  <WorkItemColor primary="FF773B93" secondary="FFEEE2F2" name="Feat
  <WorkItemColor primary="FFF2CB1D" secondary="FFF6F5D2" name="Task
  <WorkItemColor primary="FFCC293D" secondary="FFFAEAE5" name="Bug'
  <WorkItemColor primary="FFFF9D00" secondary="FFFCEECF" name="Code
  <WorkItemColor primary="FFFF9D00" secondary="FFFCEECF" name="Code
```

```
<WorkItemColor primary="FFFF9D00" secondary="FFFCEECF" name="Feed" />
<WorkItemColor primary="FFFF9D00" secondary="FFFCEECF" name="Feed" />
<WorkItemColor primary="FFFF9D00" secondary="FFFCEECF" name="Impediment" />
<WorkItemColor primary="FFFF9D00" secondary="FFFCEECF" name="Share" />
<WorkItemColor primary="FFFF9D00" secondary="FFFCEECF" name="Test" />
</WorkItemColors>
```

Element	Description
WorkItemColors	Optional. Container element for specifying colors for work item displays. The primary color is used in board displays, such as Kanban board. The secondary color is used in board displays, such as Scrum board.  Copy Code
WorkItemColor	<WorkItemColor primary="HexColorCode" secondary="HexColorCode" />
	To get hexadecimal colors, use this HTML color picker tool .

Q & A

Q: How do I customize other functions that appear on the Agile planning tools in TWA?

A: Some customizations can be done through the user interface. Others require editing process configuration or other team project objects. For an overview, see [Configure and customize Agile planning tools for a team project](#).

Q: Do you want to work with two or more portfolio backlogs?

A: The default experience supports one level of portfolio backlog. You can add up to five levels as described in [Add a backlog to Agile portfolio management](#).

Q: Do you want to add or change the WITs that appear on your task board or product backlog?

A: If you've added a custom WIT and want to add that to either the backlog or task board, you can. You just can't have them appear in both places. Learn how by reading [Add bugs or other work item types to backlogs or boards](#).

Q: Do you want to see a worked example for importing and exporting process configuration?

A: An example is provided here: [Import and export process configuration \[witadmin\]](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Process template plug-ins: Index to XML element definitions

[See Also](#) [Send Feedback](#)

By using the index of elements in this topic, you can look up the syntax structure and find examples of how to use each element of the schema definition for process templates.

Important

The schema definition for process templates uses a mix of camel-case and all capitalized elements. If you encounter errors when validating your type definition files, check the case structure of your elements. Also, the case structure of opening and closing tags must match according to the rules for XML syntax.

Alphabetical index

- activateFeatures
- CATEGORIES
- CATEGORY
- checkin_note
- Children
- Datasources
- DEFAULTWORKITEMTYPE (Category)
- Dependencies(ProcessTemplate)
- dependencies (Task)
- dependency (ProcessTemplate)
- dependency (task)
- description
- documentLibraries
- documentLibrary
- exclusive_checkout
- feature
- File
- files
- folder (Portal)
- folder (Reporting)
- folders (Portal)
- folders (Reporting)
- get_latest_on_checkout
- Group
- group (ProcessTemplate)
- Groups (Groups)
- groups (ProcessTemplate)
- HYPERLINK
- LINKTYPE
- LINKTYPES
- member
- members
- parameter
- parameter
- permission
- Permission (Work Item Query)
- permission
- plugin
- plugins
- Portal
- ProcessTemplate (root)
- ProcessTemplate (taskXm)
- Property (Classification)
- property (Reporting)
- property (Classification)
- property (Reporting)

- FIELD (Work Item Instance)

- FIELD

- FIELDS

- Metadata

- name

- Node

- Nodes

- QUERIE

- Query

- QueryFc

- Referenc

- report

Elements that define the root plug-in for process templates

The ProcessTemplate file contains two main parent elements within the **ProcessTemplate** root element: **metadata** and **groups**. The syntax for each of the following elements and their child elements is described in

[Process template XML elements reference](#).

- **metadata**: description, nname, plugin, plugins
- **groups**: dependencies, dependency, group, tasklist

Elements that define plug-in tasks

You use the **tasks** element and its child elements to define the artifacts to be uploaded or specific elements to be created for each plug-in. You can access the topic that provides detailed information and examples for each plug-in and element in the following table.

Plug-in	Related topic	Elements
All	Define the tasks to process a plug-in	dependencies, dependency, ProcessTemplate, task, tasks, taskXml
Build	Define the initial configuration of Team Foundation Build	Permission, ProcessTemplate
Classification	Define the initial areas and iterations in the classification plug-in	Children, Node, Nodes, properties, property
Groups and Permissions	Configure initial groups, teams, members, and permissions	group, groups, member, members, permission, permissions
Lab	Define the initial configuration of Lab Management	

Permission, ProcessTemplate

Portal

[Define the project portal plug-in for a process template](#)

activateFeatures, documentLibraries, documentLibrary, feature, file, files, folder, folders, Portal, site

Reports

[Add reports to the process template](#)

datasources, folder, folders, parameter, parameters, properties, property, reference, report, reports, ReportingServices, site

Test Management

[Define the initial configuration of Test Manager](#)

dependencies, dependency, TestConfigurations, TestResolutionState, TestSettings, TestVariables

Version Control

[Define the initial configuration of Team Foundation version control](#)

checkin_note, exclusive_checkout, get_latest_on_checkout, permission

Work Item Tracking

[Define objects for tracking work items using the work Item tracking plug-in](#)

CATEGORIES, FIELD, GLOBALLISTS, HYPERLINK, LINKTYPE, LINKTYPES, Permission, Query, QueryFolder, QUERIES, WI, WORKITEMS, WORKITEMTYPE, WORKITEMTYPES

■ **Type definition elements that define objects for tracking work**

In the Work Item Tracking plug-in, you can specify several type definition files and query definition files to upload. You can access the topic that provides detailed information and examples for each object and element in the following table:

Object	Related topic	Elements
Categories		

[Categories XML element reference](#)

CATEGORIES, CATEGORY

Global lists

[Define global lists](#)

GLOBALLIST, GLOBALLISTS, LISTITEM

Link types

Define a custom link type

LinkType, LinkTypes

Work item queries

[Define a work item query to add to a process template](#)

TeamFoundationServer, TeamProject, Wiql, WorkItemQuery

Work item types

[All WITD XML elements reference](#)

Top level elements only: **DESCRIPTION, FIELDS (Definition), FORM, GLOBALLISTS, WITD, WORKFLOW, WORKITEMTYPE**

See Also

Concepts

[Localization and globalization of WITD child elements](#)

[All WITD XML elements reference](#)

[Customize work tracking objects to support your team's processes](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Overview of process template files

[See Also](#) [Send Feedback](#)

You customize the initial configuration of team projects for Visual Studio Team Foundation Server (TFS) by customizing one or more process template files. You can customize several types of template files, which include the root XML file, plug-in files, and XML definition files. By customizing these files, you can define the initial configuration of all team projects that are created from the process template.

Go here to learn more about the default TFS process templates. Go here to learn about process template plug-in files.

You can customize elements of an existing team project by changing how its reports, work item types, and project portal function. In this case, your changes affect the way people work in that project, but they do not affect existing or future projects. For more information, see [An end-to-end view of what you can configure and customize in Visual Studio TFS](#).

In this topic

- [The root process template file](#)
- [Plug-in files](#)
- [New Team Project Wizard and plug-in file interaction](#)
- [XML definition files for test management](#)
- [Files that define objects used to track work items](#)
- [Process guidance files](#)

■ **The root process template file**

You customize the root XML file, ProcessTemplate.xml, to define the sequence of plug-in files to be processed and the dependencies of each plug-in. The ProcessTemplate.xml file contains all task groups that must run to successfully create a team project. Each task group references a subordinate XML plug-in file (often in a subfolder) where the specific tasks are defined. In general, you specify one task group for each plug-in.

When the New Team Project Wizard creates a team project, it runs the set of plug-ins that are defined in the ProcessTemplate.xml file. For more information, see

[Define the root tasks using the process template plug-in file File.](#)

Plug-in files

Plug-in files define the tasks that will be run and the screens that appear when a project lead creates a team project by using the New Team Project wizard. Each plug-in file defines one or more tasks. Tasks either specify to upload a file, set permissions for a group, or define some other configuration element. Each plug-in can be modified to customize a process template. Except for the Classification plug-in, plug-ins can also be deleted from the process template.

Each plug-in reads the set of tasks and dependencies to obtain the list of tasks that it must run and determine whether the requisite preceding tasks have been run. By using these definition files, you specify the initial configurations that each plug-in must implement.

A plug-in uploads files or configures data for a specific functional area. For example, the Work Item Tracking plug-in sets up work item types, queries, and initial work items for a new team project. TSF process templates include the plug-in files that are described in the following table. For information about how to customize each of these files, see [Customize a process template](#).

Folder	Plug-in file	Description
Build	Build.xml	Defines the tasks to configure the initial security permissions that are assigned to identities for Team Foundation Build and to upload the build template

		files.
Classification	Classification.xml	Defines the initial iterations and areas of a team project.
Groups and Permissions	GroupsandPermissions.xml	Defines the initial security groups of a team project and their permissions.
Lab	Lab.xml	Defines the tasks to configure the initial security permissions that are assigned to identities for Visual Studio Lab Management.
Reports	ReportsTasks.xml	Defines the initial reports for a team project and sets up the report site.

Test
Management

TestManagement.xml

Defines the test management files to upload that will create the initial test variables, configurations, settings, and resolution states of a team project. These settings are used by Microsoft Test Manager.

Version
Control

VersionControl.xml

Defines the initial security permissions for version control, check-in notes for a team project, and whether exclusive check-out is required.

WorkItem
Tracking

WorkItems.xml

Defines the initial work item types, queries, and work item instances of a team project.

Windows
SharePoint
Services

WssTasks.xml

Defines the project portal for the team based on a template for a SharePoint site. Also defines template files and process guidance.

■ New Team Project Wizard and plug-in file interaction

Project leads create team projects by using the New Team Project Wizard. The screens that appear in the wizard are determined by the plug-ins that are used. For example, if a process template does not include the plug-in for Windows SharePoint Services, no screen appears to ask the project lead for information about the project portal.

After the project lead finishes the wizard and clicks Finish, the wizard makes calls to the plug-ins to perform the work of creating the team project. The order in which the plug-ins are called is determined by the XML process definition files.

XML definition files for test management

The following table describes the XML definition files that are used to configure the initial states for Microsoft Test Manager. These files are located in the Test Management folder. Microsoft Test Manager is available with Visual Studio Ultimate and Microsoft Visual Studio Test Professional. For more information, see

[Define the initial configuration of Test Manager](#) and Testing the application.

File	Description
testconfiguration	Configures the default test configurations (hardware and software) that are defined for a team project. After the project is created, you can delete these configurations and create other configurations.
testresolutionstate	Defines the test resolution states that are used by Test Runner and Microsoft Test Manager.

 **Note**

You cannot change these states after the team project is created.

testsettings

Specifies the initial test settings files. Only one file is specified, localrun.testsettings.

testvariable

Defines the initial test variables. After the project is created, you can modify these variables and create other variables.

Files that define objects that are used to track work items

Team members use work items to track work. You might want to expand the selection of work items that is provided by Visual Studio Application Lifecycle Management (ALM) so that it better serves the processes that your team uses and the ways that you communicate. To do that, you can author new work item types. A work item type is a template from which new work items are created. You can also modify existing work item types.

The following table describes the XML definition files that are used to configure objects that are used to track work items. These files can be customized for a process template or afterward for an individual team project.

Folder	File	Description
Classification	FileMapping	Specifies the file that defines how fields are mapped from Office Project to Team Foundation. For more information, see

[Map Microsoft Project fields to Team Foundation fields.](#)

WorkItem Tracking/Work Item Types

MSF Agile Software Development :

- User Story, Bug, and Task
- Code Review Request, Code Review Response, and Review
- Feedback Request and Feedback Response
- Issue
- Test Case and Shared Steps

MSF for Capability Maturity Model Integration (CMMI) Process Improvement:

- Requirements, Change Request, Bug, and Task
- Code Review Request, Code Review Response, and Review
- Feedback Request and Feedback Response
- Issue and Risk
- Test Case and Shared Steps

Visual Studio Scrum :

- Product Backlog Item, Bug, and Task
- Code Review Request, Code Review Response, and Review
- Feedback Request and Feedback Response
- Impediment
- Test Case and Shared Steps

Defines the fields, workflow, and form for each type of work item. For more information about the work items that are defined for the MSF process templates, see the following topics:

- Scrum process template work item types and workflow
- Agile process template work item types and workflow

- CMMI process template work item types and workflow
- Scrum process template work item types and workflow
- Work with team project artifacts, choose a process template

For more information about how to customize or create work item types, see [Customize work tracking objects to support your team's processes](#).

WorkItem Tracking/LinkTypes

MSF Agile Software Development:

- SharedSteps, TestedBy

MSF for CMMI Process Improvement:

- Affects
- SharedSteps, TestedBy

Visual Studio Scrum:

- Storyboard
- SharedSteps, TestedBy

Defines additional types of links that are used to create relationships among work items.

For more information, see [Define a custom link type](#).

WorkItem Tracking

Global Lists

Note

No global lists are defined in the default process templates.

Global lists define a list of menu items that is shared across work item types and team projects. You can define global lists within work item types that you upload with your process template. For more information, see [Define global lists](#).

WorkItem Tracking

Categories.xml

- Bug Category
- Code Review Request Category and Code Review Response Category
- Feature Category
- Feedback Request Category and Feedback Response Category
- Requirement Category
- Shared Step Category and Test Case Category
- Task Category
- Hidden Types Category

Defines groups that associate a type of work item with a category. For more information, see [Use categories to group work item types](#).

Several of these categories are required to support the process configuration used by the Agile backlog and task boards available from Team Web Access. For more information, see [Configure and customize Agile planning tools for a team project](#).

WorkItem Tracking/Process

Process Configuration

Defines the configuration for the backlog and task board pages that appear in Team Web Access. For more information, see [Configure and customize Agile planning tools for a team project](#).

WorkItem Tracking/Queries

Team work item queries (.wiq)

Defines an initial set of team queries that support finding work items. You can add or modify these queries. For more information, see [Add work item queries to a process template](#).

Process guidance files

Process guidance files are uploaded to a team project portal. These files specify links to process guidance content, project management workbooks, and sample templates and documents. All of these objects can be customized for a process template or afterward when they are defined for a team project.

The following table describes the process guidance files that are included in the MSF process templates.

Note

Process template files do not include dashboards or Microsoft Excel reports. These files are added to the team project, depending on selections that are made when a team project is created. If no SharePoint site is provisioned for the team project, no dashboards or Office Excel reports are available. For more information about these artifacts, see the following topics: Dashboards, Dashboards (CMMI), Excel reports, and Excel reports (CMMI).

Folder	File type	Description
		Defines process guidance URL files. These files define the URL that is opened when a team member clicks  in

Windows SharePoint Services/Process Guidance/Supporting Files .htm files a work item form. The TFS process templates contain links to topics on MSDN. These files can be customized to point to other resources for process guidance.

Windows SharePoint Services/Shared Documents .xlsx files Defines Microsoft Excel workbooks to support bug triage and track issues.

Provides sample documents and templates as a starting point for creating work products.
.docx,

Windows SharePoint Services/Samples and Templates .dotx, .xlsx files, These files are uploaded to the team project portal when a SharePoint site is provisioned for the team project.

See Also

Concepts

Work with team project artifacts, choose a process template

Customize a process template



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define the root tasks using the process template plug-in file [File](#)

[See Also](#) [Send Feedback](#)

By customizing ProcessTemplate.xml, you can add, remove, or change the sequence in which process template plug-in files are processed. The ProcessTemplate.xml is the root file that defines the entire process template and all subordinate XML files. This file contains all of the task groups that must run to successfully create a team project. Each task group references a subordinate XML file (often in a subfolder) where the specific tasks are defined. In general, you specify one task group for each plug-in.

Go [here](#) to learn more about the default TFS process templates. Go [here](#) to learn about process template plug-in files.

In this topic

-

[Structure of the ProcessTemplate.xml file](#)

- [Define the plug-ins and task groups](#)
- [Example ProcessTemplate file](#)

Structure of the ProcessTemplate.xml file

In each ProcessTemplate.xml file, you define metadata and task groups. The metadata describes the template and specifies the plug-ins to be processed, as the following example shows

```
<metadata>
  <name>MSF for Agile Software Development 2013</name>
  <description>This template is flexible and will work great for n
  <plugins>
    <plugin name="Microsoft.ProjectCreationWizard.Classification"
    <plugin name="Microsoft.ProjectCreationWizard.Reporting" wizar
    <plugin name="Microsoft.ProjectCreationWizard.Portal" wizardPa
  . . .
  </plugins>
</metadata>
```

The task groups specify the dependencies and tasks to be performed for each plug-in, as the following example shows:

```
<groups>
  <group id="Classification" description="Structure definition for
    <dependencies></dependencies>
    <taskList filename="Classification\Classification.xml" />
  </group>
  <group id="Groups" description="Create groups and assign permis:
    <dependencies>
      <dependency groupId="Classification" />
    </dependencies>
    <taskList filename="Groups and Permissions\GroupsandPermissior
  </group>
  . . .
</groups>
```

[Back to top](#)

Define the plug-ins and task groups

You customize the ProcessTemplate.xml file by using the **ProcessTemplate** element and its child elements. For more information about the syntax of each element, see

[Process template XML elements reference.](#)

You define the plug-ins and root tasks to be processed by specifying the following information in the indicated sequence:

1. name. You use this element to assign a label to the process template. In the New Team Project Wizard, you specify the process template based on its name, as the following example shows:

Copy Code

```
<name>Simplified Process</name>
```

This element is required. The name appears not only in the process template list in the New Team Project Wizard but also in the Process Template Manager dialog box. The name of each process template must be unique within a team project collection, and each template name must contain no more than 254 Unicode characters. If you upload a template that is the same name as an existing template, the existing template will be overwritten.

2. description. You use this element to define the text that describes the process template in the New Team Project Wizard, as the following example shows:



```
<description>Choose the simple process for projects that require
```

This text appears on the Select a Process Template screen in the New Team Project Wizard when a project lead or administrator is deciding which

process template to use. In general, try to describe what kinds of projects for which the process template is useful in terms of team size, length, cost, and other factors.

3. **Plug-ins.** You use the **plugins** and **plugin** elements to define the set of plug-ins that the process template uses. You must make sure that if any subordinate XML file references a plug-in, that plug-in appears in this list. You must also specify the wizardPage attribute (a true or false value) to indicate whether the plug-in requires a page on the New Team Project Wizard. You must specify the set of a plug-ins that will be processed when you create a team project and whether the plug-in should have a visible page in the New Team Project Wizard.

The following XML shows the correct values to use for each of the nine plug-ins that Visual Studio Application Lifecycle Management (ALM) includes.

```
<plugins>    <plugin name="Microsoft.ProjectCreationWizard.Class:
```

4. **Task Groups.** You use the **groups** and **group** elements to define task groups. Each task group defines information that is necessary to process each plug-in. Each task group specifies the file that contains the set of tasks to process and optional dependencies that the group has on other task groups. The most common and easiest strategy is to create one **taskList** element per group per plug-in.

The following example shows how a task group creates the default groups and permissions. The specific tasks are defined in the referenced file (GroupsandPermissions.xml). This task group has a dependency that the tasks in the Classification group successfully complete.

```
<group id="Groups" description="Create groups and assign permis:
```

For each task group, you define the following information:

1. ID. A unique identification that is assigned to the plug-in.
2. Description. The text that describes the plug-in tasks in the New Team Project Wizard.
3. Completion message. The message to appear in the New Team Project Wizard when all tasks for the plug-in successfully complete.
4. (Optional) Failure message. The message to appear in the New Team Project Wizard when one or more tasks for the plug-in fails to complete.
5. (Optional) Plug-in dependencies. You use the **dependencies** and **dependency** elements to define the set of dependencies for a plug-in to be processed. A dependency corresponds to a plug-in that must successfully complete before the current plug-in tasks can be processed.
6. Task file. You use the **tasklist** element to specify the file that contains the set of tasks to process the plug-in.

[Back to top](#)

Example Process Template file

The following syntax lists the contents of the ProcessTemplate.xml file for Microsoft Visual Studio Scrum.

```
<?xml version="1.0" encoding="utf-8"?>
<ProcessTemplate>
  <metadata>
    <name>Microsoft Visual Studio Scrum 2013</name>
    <description>This template is for teams who follow the Scrum met
<version type="6B724908-EF14-45CF-84F8-768B5384DA45" major="3" minor
  <plugins>
    <plugin name="Microsoft.ProjectCreationWizard.Classification"
    <plugin name="Microsoft.ProjectCreationWizard.Reporting" wizardPa
    <plugin name="Microsoft.ProjectCreationWizard.Portal" wizardPa
    <plugin name="Microsoft.ProjectCreationWizard.Groups" wizardPa
    <plugin name="Microsoft.ProjectCreationWizard.WorkItemTracking
    <plugin name="Microsoft.ProjectCreationWizard.VersionControl"
    <plugin name="Microsoft.ProjectCreationWizard.TestManagement"
    <plugin name="Microsoft.ProjectCreationWizard.Build" wizardPag
    <plugin name="Microsoft.ProjectCreationWizard.Lab" wizardPage=
  </plugins>
</metadata>
<groups>
  <group id="Classification" description="Structure definition for
    <dependencies>
</dependencies>
    <taskList filename="Classification\Classification.xml" />
  </group>
  <group id="Groups" description="Create groups and assign permis
    <dependencies>
      <dependency groupId="Classification" />
    </dependencies>
    <taskList filename="Groups and Permissions\GroupsAndPermissio
  </group>
  <group id="WorkItemTracking" description="Work item definitions
    <dependencies>
      <dependency groupId="Classification" />
      <dependency groupId="Groups" />
    </dependencies>
    <taskList filename="WorkItem Tracking\WorkItems.xml" />
  </group>
  <group id="VersionControl" description="Creating version contro
```

```

    <dependencies>
      <dependency groupId="Classification" />
      <dependency groupId="Groups" />
      <dependency groupId="WorkItemTracking" />
    </dependencies>
    <taskList filename="Version Control\VersionControl.xml" />
  </group>
  <group id="Build" description="Build default processes uploading"
    <dependencies>
      <dependency groupId="VersionControl" />
      <dependency groupId="Groups" />
    </dependencies>
    <taskList filename="Build\Build.xml" />
  </group>
  <group id="Lab" description="Creating Lab." completionMessage="l
    <dependencies>
      <dependency groupId="Classification" />
      <dependency groupId="Groups" />
      <dependency groupId="WorkItemTracking" />
      <dependency groupId="Build" />
    </dependencies>
    <taskList filename="Lab\Lab.xml" />
  </group>
  <group id="TestManagement" description="Test Management default
    <dependencies>
      <dependency groupId="Classification" />
      <dependency groupId="Groups" />
      <dependency groupId="WorkItemTracking" />
    </dependencies>
    <taskList filename="Test Management\TestManagement.xml" />
  </group>
  <group id="Reporting" description="Project reports uploading." c
    <dependencies>
      <dependency groupId="Classification" />
      <dependency groupId="WorkItemTracking" />
      <dependency groupId="VersionControl" />
    </dependencies>
    <taskList filename="Reports\ReportsTasks.xml" />
  </group>
  <group id="Portal" description="Creating project portal site" c
    <dependencies>
      <dependency groupId="Classification" />
      <dependency groupId="Reporting" />
    </dependencies>
    <taskList filename="Windows SharePoint Services\WssTasks.xml"
  </group>
</groups>
</ProcessTemplate>

```

See Also

Concepts

[Overview of process template files](#)

Customize a process template

Other Resources

[Process template XML elements reference](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Process template XML elements reference

[See Also](#) [Send Feedback](#)

You can customize the root file for a process template to add or remove plug-in files. The root file defines all the plug-ins and all the tasks that the New Team Project Wizard will process. For the process templates for Microsoft Solutions Framework (MSF), the root file is named `ProcessTemplate` and located in the container folder for the process template files. For more information, see

[Overview of process template files.](#)

The parent element of the root plug-in file is the **ProcessTemplate** element, which must have only one **metadata** and one **groups** element defined. You must assign a unique name to each process template within a team project collection, and the name cannot be longer than 124 Unicode characters.

Caution

If you upload a template with the same name as an existing template, the existing template will be overwritten.

In this topic

- [ProcessTemplate syntax structure](#)
- [Metadata element reference](#)
- [Groups element reference](#)

ProcessTemplate syntax structure

The root plug-in for process templates must be specified in its own file in the **ProcessTemplate** container element and conform to the schema for process templates, which is defined in the ProcessTemplate.xsd and ProcessTemplateItem.xsd files. You can download the schema files for process templates from the following page on the Microsoft website:

[Process Template and Work Item Schemas for Visual Studio Team Foundation.](#)

The following example shows the structure of the root **ProcessTemplate** element. For a more extensive example, see [Define the root tasks using the process template plug-in file File](#).

 **Copy Code**

```
<?xml version="1.0" encoding="utf-8"?>
<ProcessTemplate>
  <metadata> . . . </metadata>
  <groups> . . .</groups>
</ProcessTemplate>
```

Metadata element reference

The following syntax shows the structure of the **metadata** element and its child elements.

 **Copy Code**

```
<metadata>
  <name>ProcessTemplateName</name>
  <description>ProcessTemplateDescription </description>
  <plugins>
    <plugin name="Microsoft.ProjectCreationWizard.pluginID" wizard
    . . .
  </plugins>
</metadata>
```

Element

Syntax

 **Copy Code**

description <description>ProcessTemplateDescription</description>

 **Copy Code**

metadata <metadata>
 <name> . . . </name>
 <description> . . . </description>

```
<plugins> . . . </plugins>  
</metadata>
```

 **Copy Code**

name

```
<name>ProcessTemplateName</name>
```

 **Copy Code**

plugin

```
<plugin name="Microsoft.ProjectCreationWizard.PluginName  
wizardPage="true | false" />
```

 **Copy Code**

plugins

```
<plugins>
  <plugin> . . . </plugin>
</plugins>
```

 **Copy Code**

version

```
<version type="ID" major="Number1" minor="Number2 " />
```

Groups element reference

The following syntax shows the structure of the **groups** element and its child elements.

 **Copy Code**

```
<groups>
  <group id="groupId" description="GroupDescription" completionMess:
    <dependencies>
      <dependency groupId="dependentGroupId" />
      . . .
    </dependencies>
    <taskList filename="RelativePathAndFileName" />
  </group>
  . . .
</groups>
```

Element	Syntax	Description
	 Copy Code	Required child element of group .
dependencies	<pre><dependencies> <dependency>. . . </dependency> </dependencies></pre>	Specifies dependencies that the group has on other groups.
	 Copy Code	Optional child element dependencies .
dependency	<pre><dependency groupId="groupId" /></pre>	Specifies the ID of another task group which this group depends. The other must complete its task before this task group.

group

```
<group id="TaskGroupId"
  description="TaskGroupDescription"
  completionMessage="SuccessMessage"
  failureMessage="FailureMessage">
  <dependencies> . . . </dependencies>
  <taskList> . . . </taskList>
</group>
```

Copy Code

can start.

Required child elements of **groups**.

Identifies a set of tasks for a plug-in to run during project creation. The following values are valid for each attribute:

- **id:** Specifies a unique ID that identifies a task group. If another task group depends on this group, it refers to this ID. You must assign a unique value for the ID within the plug-in file.
- **description:** Specifies the message to include during team project creation, that the task group is being processed.
- **completionMessage:** Specifies the message to include during team project creation, that the task group is completed.

successfully.
Maximum length
256 characters

- failureMessage
Specifies the
message to include
during team project
creation, that the
task group failed to
complete
successfully.

Required child element
of **ProcessTemplate**

Container element
defines the set of task
groups that will run
during project creation

Specifies an XML file
that contains the list
of tasks to run. The list
of tasks for a group must
always be in a separate
XML file. For more
information about how to
specify these files,

 **Copy Code**

groups

```
<groups>  
  <group> . . . </group>  
</groups>
```

 **Copy Code**

tasklist

```
<taskList filename="filePath" />
```

[Define the tasks to process a plug-in.](#)

See Also

Concepts

[Overview of process template files](#)

[Define the tasks to process a plug-in](#)

[Define dependencies for task groups and tasks in plug-in files](#)

Customize a process template

[Define the root tasks using the process template plug-in file File](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

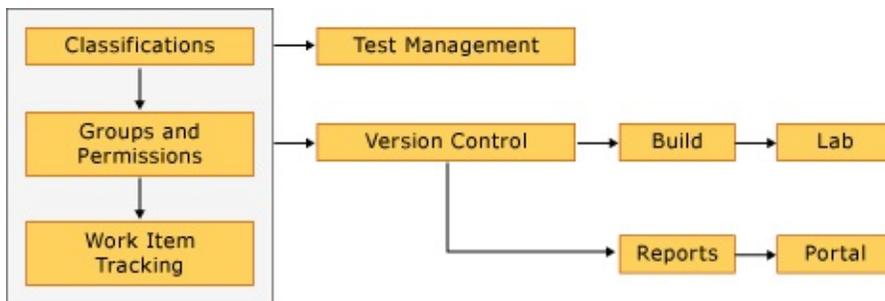
Define dependencies for task groups and tasks in plug-in files

[See Also](#) [Send Feedback](#)

When you add a task group or a task to a plug-in file, you must add them in the correct sequence and declare any dependencies that the group or task has on the successful completion of other task groups or tasks. A task can depend on other tasks, requiring other tasks to complete before primary task can run. For example, the task to create work item queries cannot run until all the tasks to create work item types have successfully completed. Therefore, the task to upload work item queries depends on the task to upload the types of work items.

Required plug-ins and plug-in dependencies

The following illustration shows the dependent relationships that are defined for the Team Foundation Server (TFS) process templates. As the illustration shows, the plug-in for Work Item Tracking depends on the successful completion of the plug-in for Groups and Permissions, which depends on the successful completion of the plug-in for Classifications. The plug-ins for both Test Management and Version Control depend on the completion of the first three plug-ins. The plug-ins for Build, Lab, Reports, and Portal depend on the completion of the plug-in for Version Control.



Tip

If you define your task groups and tasks in each plug-in file in the same order as their dependencies, you make reading the file and eliminating dependency problems easier.

Dependencies can be used to indicate the order in which plug-ins run in the New Team Project Wizard. Some plug-ins are always dependent on other plug-ins. Also, not all plug-ins are required for a process template.

The following table identifies which plug-ins depend on other plug-ins. Only the plug-in for Classifications is required. All other plug-ins are optional. For more information about each plug-in, see

[Overview of process template files.](#)

Plug-in	Dependent Plug-ins
Build	Groups, VersionControl
Classifications	None
Groups	Classifications
Lab	Build, Classifications, Groups, WorkItemTracking
Portal	Classifications, Reporting, VersionControl, and WorkItemTracking
Reporting	Classifications, VersionControl, and WorkItemTracking
TestManagement	Classifications, Groups, and WorkItemTracking
	Classifications,

VersionControl Groups, and
WorkItemTracking

WorkItemTracking Classifications and
Groups

Defining task group dependencies

Use the dependency element in tasks and groups to indicate when a dependency exists. The following example shows how to use the dependency element to specify that the WorkItemTracking task group depends on the Classification and Groups task groups.

Copy Code

```
<group id="WorkItemTracking"
  description="Workitem definitions uploading."
  completionMessage="Workitem definitions uploaded.">
  <dependencies>
    <dependency groupId="Classification"/>
    <dependency groupId="Groups"/>
  </dependencies>
  <taskList filename="WorkItem Tracking\WorkItems.xml"/>
</group>
```

Note

The groupId element references the id element value in the other group.

The following table describes the elements that you use to define task group dependencies. For more information about how to define task groups, see

[Process template XML elements reference.](#)

Element	Syntax	Description
	 Copy Code	Required child element of group .
dependencies	<pre><dependencies> <dependency>. . . </dependency> </dependencies></pre>	Specifies the other groups on which a

group
depends.

Optional child
element of
dependencies.

 **Copy Code**

dependency

```
<dependency groupId="groupId" />
```

Specifies the
ID of another
task group on
which this
group
depends. The
other group
must complete
its tasks
before this
task group can
start.

Defining task dependencies

The following example shows how to use the **dependency** element to specify that the WorkItems task depends on the task with an ID of "WITs."

 **Copy Code**

```
<task
  id="WIS"
  name="WorkItems"
  plugin="Microsoft.ProjectCreationWizard.WorkItemTracking"
  completionMessage="work items uploaded"
  completionDescription = "Processing the actual work items used by
  <dependencies>
    <dependency taskId="WITs" />
  </dependencies>
```

The following table describes the elements that you use to define task dependencies. For more information about how to define tasks, see

[Define the tasks to process a plug-in.](#)

Element	Syntax	Description
	 Copy Code	Optional child element of task .
dependencies	<pre><dependencies> <dependency>. . . </dependency> </dependencies></pre>	Specifies the other tasks on which a task depends.
		Optional child element of dependencies .

dependency

 **Copy Code**

```
<dependency taskId="taskId" />
```

Specifies the ID of another task on which this task depends. The other task must complete its tasks before this task can start.

See Also

Concepts

Customize a process template

[Define the tasks to process a plug-in](#)

[Define the root tasks using the process template plug-in file File](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define the tasks to process a plug-in

[See Also](#) [Send Feedback](#)

You use the **tasks** element and its child elements to define the artifacts to be uploaded or specific elements to be created for a plug-in. A task specifies work that must occur to configure something specific to the process in a new team project. A number of the tasks that you define upload the work item types, work item queries, documents, and reports for a team project. Other tasks that you can define configure other areas of Visual Studio Application Lifecycle Management, such as Team Foundation Build, Visual Studio Lab Management, Test Manager, and Team Foundation version control.

In this topic

-

[Tasks Syntax Structure](#)

- [Tasks Element Reference](#)
- [Index to taskXML Elements by Function](#)
- [Example: Defining Types of Work Items](#)

Tasks Syntax Structure

You can specify several tasks within a plug-in, and these tasks may have dependencies on one another. The tasks that you define for each plug-in must conform to the schema definition for **tasks**, defined in the Tasks.xsd file, and be specified in its own file using the **tasks** container element.

The following syntax shows the high-level structure of the **tasks** element and its child elements.

 **Copy Code**

```
<tasks>
  <task>
    <dependencies>
      <dependency />
      .
      .
      .
    </dependencies />
    <taskXml>
      .
      .
      .
    </taskXml>
  </task/>
</tasks>
```

[Back to top](#)

Tasks Element Reference

The following table describes each element that can define the tasks for a plug-in.

Element	Syntax
	 Copy Code
dependencies	<pre><dependencies> <dependency>. . . </dependency> </dependencies></pre>
	 Copy Code
dependency	<pre><dependency taskId="taskId" /></pre>
	 Copy Code
	<pre><ProcessTemplate Type="FileType" Filename="LocalPath Description="FileDescription" ServerPath="UploadLocation" /></pre>
	Where each attribute is defined as follows:
ProcessTemplate	<ul style="list-style-type: none">• Type: The type of file that is being uploaded. The following are valid: Default and Upgrade for build template files and C# template files.

- **Filename:** The local directory path and name of the file to
- **Description:** A short text string that describes the file to
- **ServerPath:** The path of the server that is running Team Server where the file should be uploaded. The only valid is \$\$\$PROJECTNAME\$\$/BuildProcessTemplates.

 **Copy Code**

task

```
<task id="task id" name="name of task"
  plugin="Microsoft.ProjectCreationWizard.<PluginName>"
  completionMessage="SuccessfulCompletionMessage"
  completionDescription="InProgressMessage">
  <dependencies>. . . </dependencies>
  <taskXml>. . . </taskXml>
</task>
```

tasks

```
<tasks>  
  <task>. . . </task>  
</tasks>
```

 **Copy Code**

taskXml

```
<taskXml> . . . </taskXml>
```

 **Copy Code**

[Back to top](#)

Index to taskXML Elements by Function

For more information about the specifics of the XML markup elements that you can specify for each plug-in, see one of the following topics:

- Build:

[Define the initial configuration of Team Foundation Build](#)

- Classifications: [Define the initial areas and iterations in the classification plug-in](#)
- Groups and Permissions: [Configure initial groups, teams, members, and permissions](#)
- Lab: [Define the initial configuration of Lab Management](#)
- Portal: [Define the project portal plug-in for a process template](#)
- Reports: [Add reports to the process template](#)
- Test Management: [Define the initial configuration of Test Manager](#)
- Version Control: [Define the initial configuration of Team Foundation version control](#)
- Work Item Tracking: [Define objects for tracking work items using the work item tracking plug-in](#)

[Back to top](#)

Example: Defining Types of Work Items

The following example shows a task that creates six types of work items by referencing the definition files for each type.

 **Copy Code**

```
<task
  id="WITs"
  name="WorkItemType definitions"
  plugin="Microsoft.ProjectCreationWizard.WorkItemTracking"
  completionMessage="WorkItemTypes created"
  completionDescription = "Processing work item types used by work
<taskXml>
  <WORKITEMTYPES>
    <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Bu
    <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Sl
    <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\T
    <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Te
    <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Us
    <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Is
  </WORKITEMTYPES>
</taskXml>
</task>
```

[Back to top](#)

See Also

Concepts

[Define dependencies for task groups and tasks in plug-in files](#)

[Overview of process template files](#)

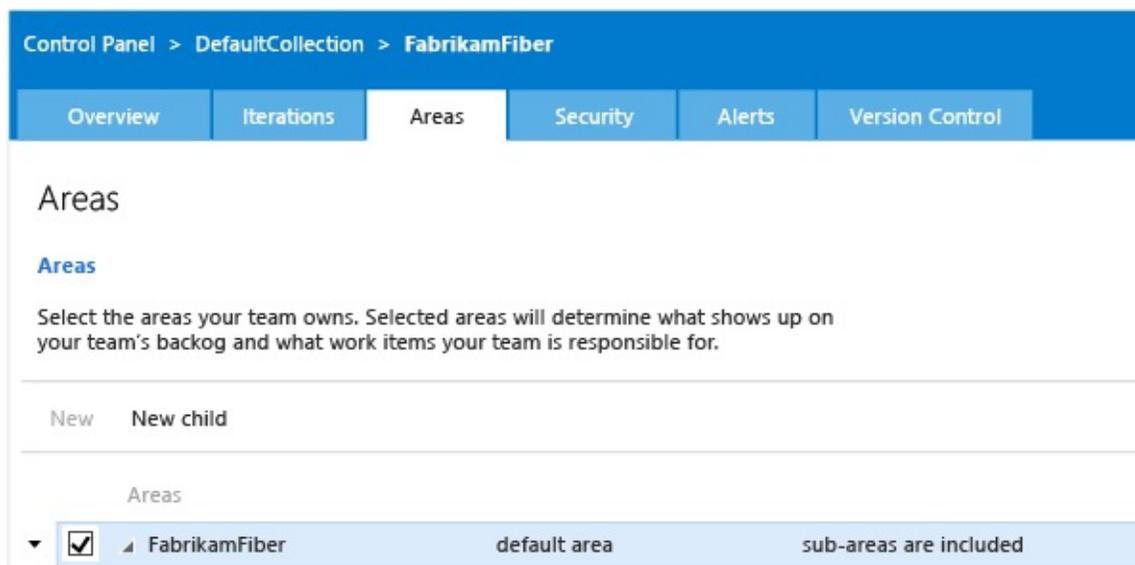
- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define the initial areas and iterations in the classification plug-in

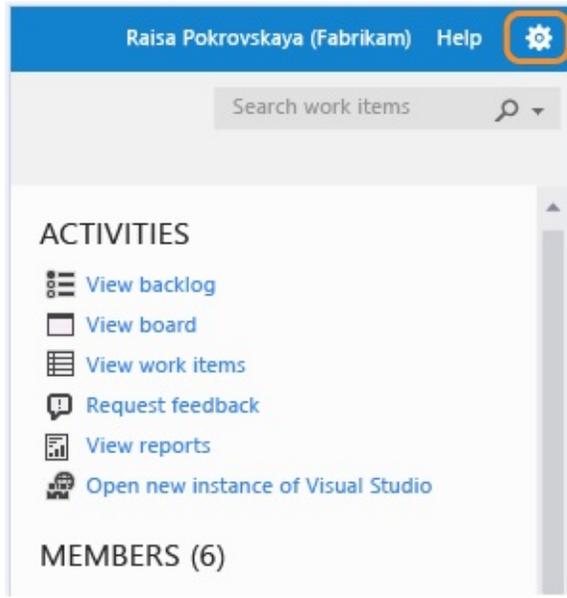
[See Also](#) [Send Feedback](#)

Classifications facilitate tracking, grouping, and reporting on work based on useful categories, such as related product features and project milestones. You define areas to organize work items into logical, physical, or functional categories. You define iterations to group work items into milestones or time cycle categories. For example, a team could organize the product work into a client area, a server area, and an extensibility area. Iterations that you define determine how many times the team will repeat a particular set of major activities, such as plan, develop, and test.

The classifications that you define appear on the Areas and Iterations pages for the new team project, as the following illustration shows. You access this page from the Team Web Access administration pages. To open this page, go [here](#).



Access these pages from the Team Web Access (TWA) administration context. To open the administration context, choose the  gear Settings icon.



To learn how to connect to TWA, go [here](#).

After a team project has been created, you modify the areas and iterations using TWA. You can also define permissions to control access to a project area or iteration. For guidelines and naming restrictions that apply to classifications, see [Add and modify area and iteration paths](#). For information about the fields that track areas and iterations, see

[Areas and iterations field reference](#)

In addition to areas and iterations, you specify the mapping file for Microsoft Project to upload within the Classification plug-in.

⚠️ Important

By default, process templates for Microsoft Solution Framework (MSF) define no area nodes and three iteration nodes, which are named Iteration 1, Iteration 2, and Iteration 3. The iteration nodes are referenced in the definition of work item queries, and the tasks are defined in the WorkItemTracking and Portal plug-ins.

Classification plug-in name and location

The following table lists the name of the file, the folder, and the plug-in for the process templates for Microsoft Solutions Framework (MSF):

File name:	Classification.xml
Folder name:	Classification
Plug-in name:	Microsoft.ProjectCreationWizard.Classification

Note

You can change the names of the XML file and the folder but not the plug-in. Team Foundation Server does not include a mechanism for the deployment of client-side plug-ins, policies, or other modifications. If you want to deploy this kind of functionality, you must use your own distribution and installation program.

Classification tasks and dependencies

The Classification plug-in file must conform to its schema definition, as specified in the `Css.xsd` file, and the plug-in must be specified in its own file.

In the XML file, specify one or more tasks and their dependencies. In general, you only need one task to specify all the iterations and nodes that a team project requires.

Note

If you modify a process template's areas or iterations, make sure that you do not break any assignments that are used for work item queries or work items that are defined in the process template. For example, the `Iteration1Backlog.wiq` work item query for the MSF process template for agile software development references Iteration 1. For more information, see

[Add work item queries to a process template.](#)

■ Specifying the initial product areas

Specify the root node of areas by using the following syntax:

Copy Code

```
<Node StructureType="ProjectModelHierarchy" Name="Area" xmlns="">
```

Specify zero or more children nodes for as many areas as you want for your process. Use the Node element to specify each area and set the StructureType attribute to ProjectModelHierarchy.

Copy Code

```
<Node StructureType="ProjectModelHierarchy" Name="NodeName" xmlns=""
```

The following example shows how to specify two areas, Client and Server:

Copy Code

```
<?xml version="1.0" encoding="utf-8" ?>
<tasks>
  <task
    id="UploadStructure"
    name="Creating project structure"
    plugin="Microsoft.ProjectCreationWizard.Classification"
    completionMessage="Portfolio project structure created.">
    <taskXml>
      <Nodes>
        <Node StructureType="ProjectModelHierarchy" Name="Teams"
          <Children>
            <Node StructureType="ProjectModelHierarchy" Name='
              <Node StructureType="ProjectModelHierarchy" Name='
            </Children>
          </Node>
        </Nodes>
      </taskXml>
    </task>
  </tasks>
```

▣ Specifying the initial iteration paths

Specify the root node for iterations by using the following syntax:

Copy Code

```
<Node StructureType="ProjectLifecycle" Name="NodeName" xmlns="">
```

Specify zero or more children nodes for as many iterations as you want for your process. Use the **Node** element to specify each iteration and set the StructureType attribute to ProjectLifecycle.

The following example shows how to specify four iterations: Milestone 1, Milestone 2, Beta, and RTM.

Copy Code

```
<?xml version="1.0" encoding="utf-8" ?>
<tasks>
  <task
    id="UploadStructure"
    name="Creating project structure"
    plugin="Microsoft.ProjectCreationWizard.Classification"
    completionMessage="Team project structure created.">
    <taskXml>
      <Nodes>
        <Node StructureType="ProjectLifecycle" Name="Iteration"
          <Children>
            <Node StructureType="ProjectLifecycle" Name="Miles
            <Node StructureType="ProjectLifecycle" Name="Miles
            <Node StructureType="ProjectLifecycle" Name="Beta'
            <Node StructureType="ProjectLifecycle" Name="RTM">
          </Children>
        </Node>
      </Nodes>
    </taskXml>
  </task>
</tasks>
```

■ Specifying the field mapping file for Microsoft Project

Specify the file that contains the field mappings for Microsoft Project in the **properties** section of the Classification.xml file. Use the property element to specify the file to upload, which is labeled FileMapping.xml for MSF process templates. This file contains the mappings between fields in Microsoft Project and fields in Team Foundation. This file is uploaded to the database for the team project. The following example shows the syntax structure for this entry:

Copy Code

```
<properties>
  <property name="MSPROJ"
            value="Classification\FileMapping.xml"
            isFile="true" />
</properties>
```

To customize this file, see

[Map Microsoft Project fields to Team Foundation fields.](#)

Classification plug-in element reference

The following table describes the elements that you use to define the initial product areas and iterations. Specify these elements within a taskXml container element in the Classification plug-in file. For information about this element, see [Define the tasks to process a plug-in](#).

[Define the tasks to process a plug-in](#).

⚠ Caution

The Css.xsd schema file does not define the property or properties elements. When you upload the process template, the Process Template Manager validates these elements before storing them in Team Foundation Server.

Define areas and iterations by specifying a tree path of nested nodes and children nodes. For more information, see [Add and modify area and iteration paths](#).

		 Copy Code
Children	<pre><Children> <Node> . . . </Node> </Children></pre>	
		 Copy Code
Node	<pre><Node StructureType="ProjectLifecycle ProjectModelHierar <Node> . . . </Node> <Children> . . . </Children> </Node></pre>	

 **Copy Code**

Nodes

```
<Nodes>
  <Node> . . . </Node>
</Nodes>
```

 **Copy Code**

properties

```
<properties>
  <property />
</properties>
```

 **Copy Code**

```
<property name="property name" value="Classification\FileN;
```

Where the following definitions apply for each attribute:

property

- name: Required. Defines the name of the property. The only valid
- value: Required. Defines the relative path to the field mapping fil
- isFile: Optional. Specifies whether the property is defined in a fil
exist, or process template validation fails.

See Also

Concepts

[Add and modify area and iteration paths](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define the project portal plug-in for a process template

[See Also](#) [Send Feedback](#)

By using the portal plug-in, you can define the initial document libraries, library structure, and documents of a team project's portal. You can also include the tasks that create dashboards and Excel reports that are designed to work with the Team Foundation Server (TFS) process templates.

The tasks that you specify in the plug-in for SharePoint Products are run only when you create a SharePoint site when you create a team project. For more information about site requirements and administration, see SharePoint Products requirements for Team Foundation Server.

Portal plug-in name and location

The following table summarizes the names of the portal plug-in file, the folder, and the plug-in for the TFS process templates.

File name:	WssTasks.xml
Folder name:	Windows SharePoint Services
Plug-in name:	Microsoft.ProjectCreationWizard.Portal

Note

You can change the names of the XML file and the folder but not the plug-in. TFS does not include a mechanism for the deployment of client-side plug-ins, policies, or other modifications. If you want to deploy this kind of functionality, you must use your own distribution and installation program.

Portal creation tasks and dependencies

In the portal plug-in file, specify one or more tasks and their dependencies. You specify the following key tasks:

- Create the site by using a specified site template
- Create document libraries
- Create folders, and upload files to the document libraries
- Activate dashboard features

For an example of a task that specifies a simple project portal, see the `WssTasks.xml` file in one of the TFS process templates. You can download the latest process templates from TFS.

■ Create the site and specify the site template

You must specify a site template on which the project portal is based. The site template also must be available on the Team Foundation Server SharePoint portal. Site templates are not included in the process template.

Specify the site template using the site element.

 **Copy Code**

```
<site template="" language=""/>
```

The following table describes the attributes for the **site** element.

Attribute	Description
	Specifies which template to use. You must specify a template that is defined on the server that hosts SharePoint Products for the team projects that will be created.
template	To use the features that are available with the current version of the process templates for MSF, specify the following string: Team Foundation Server Project Portal.

language

- Specifies a locale ID to indicate which language version of the site template to use. The English version is 1033.

The following example shows how to reference the English version of the current version of the MSF Agile process template, which supports dashboards.

```
<tasks>
  <task id="SharePointPortal" name="Create Sharepoint Portal" plugin="Microsoft.SharePoint.WebPartPages" >
    <dependencies />
    <taskXml>
      <Portal>
        <site template="Team Foundation Server Project Portal" language="1033" />
      </Portal>
    </taskXml>
  </task>
</tasks>
```

Note

All other tasks defined within the portal plug-in depend on the site creation task because the project portal must be created before you can create additional document libraries or copy files.

Process template performance

The files that you specify in WssTasks.xml are included as part of the process template when it is uploaded. The size of the process template affects how long it takes to create a new team project. Larger process templates take longer to create new team projects. Therefore you should consider alternate strategies to provide files when process template performance degrades.

Use the site template

You can include files as part of the SharePoint site template. This moves files out of the process template and into the site template and improves performance when creating new team projects.

However, there is a benefit to listing files in WssTasks.xml. You can bind Microsoft Project and Microsoft Excel files to a query using the queryid attribute described previously. This assists team members when they open the files because they will already be connected to the correct Team Foundation Server and query.

■ Use an alternate website

If you have a large collection of files that are shared amongst multiple team projects, you can use an alternate Web site to host the files and link to the files from the project portal or Web pages. This strategy also increases performance when creating team projects since the files do not need to be copied to the project portal. For more information, see [Configure or add a project portal](#).

Create document libraries

After the project portal is created, you can specify that additional document libraries be created. Use the **documentLibrary** element to specify a document library.

Copy Code

```
<documentLibraries>  
  <documentLibrary name="" description="" isProcessGuidance="true"  
</documentLibraries>
```

Note

Versioning for a document library cannot be enabled or disabled in the process template. You can enable or disable versioning via the Windows SharePoint Services site template, or after project creation.

The following table describes the attributes for the documentLibrary element.

Attribute	Description
name	Specifies the name of the document library that appears on the project portal.
description	Provides a description of the document library that appears on the project portal.

Specifies whether
the document
library is used to
support process
guidance content.

The following example shows how to create a document library named Development that the development team can use to store documents.

```
<documentLibraries>  
  . . .  
  <documentLibrary name="Process Guidance" description="How to make  
  . . .  
</documentLibraries>
```

■ Create the folder structure of a document library

After the project portal is created, you can specify additional folders to create. You can also specify files to copy such as template files.

Use the **folder** element to create a new folder.

 **Copy Code**

```
<folders>  
  <folder documentLibrary="" name=""/>  
</folders>
```

The following table describes the attributes for the folder element.

Attribute	Description
documentLibrary	Identifies which document library to create the folder in.
name	Specifies the name of the folder.

The following example shows how to create a folder named Trip Reports in the Development document library where developers can store trip reports from conferences or customer visits.

 **Copy Code**

```
<folders>
```

```
<folder documentLibrary="Development"  
name="Trip Reports"/>  
</folders>
```

Specify files to upload to document libraries

You must also copy files into document libraries and folders. The source files are located under the Windows SharePoint Services folder. You must specify a target on the project portal.

Copy Code

```
<files>  
  <file source="" documentLibrary="" target="" queryId="" />  
</files>
```

The following table describes the attributes for the file element.

Attribute	Description
source	A relative path to the local copy of the file. Generally, the source path name always begins with Windows SharePoint Services, which is the name of the folder where the portal plug-in is defined.
documentLibrary	The name of the document library into which the file will be copied.

target

A relative path to the target on the project portal where the file is copied.

queryId

Specifies the name of a work item query that is bound to the file. When the file is opened for the first time, it is populated with the results of running the query. This attribute is optional, and the specified query must exist in the workitems.xml file.

 **Note**

queryId only applies to .xls files.

The following example shows how to copy a template file named Project Checklist into the Project Management folder. This example also specifies that when the file is first open, it should be populated with the results of the Project Checklist work item query.

 **Copy Code**

```
<files>
  <file source="Windows SharePoint Services\Templates\Project Checkl
    documentLibrary="Project Management"
    target="Project Checklist.xls"
    queryId="Project Checklist" />
</files>
```

Note

You could include folders and files as part of the site template, and you would not have to list them in the XML.

Note

If you specify .exe files and the SharePoint site does not support .exe files, you will not be able to create a team project successfully by using the process template.

Process guidance content and support files

Process guidance is content that documents the processes to be followed by team members who work on a software project. Work items, reports, and queries can all change during the lifecycle of a team project, and they can be different between team projects. Process guidance content provides details about a team project, such as information about how to complete work item fields, examples of healthy and unhealthy reports, and descriptions of the queries. Process guidance also provides details about the process to follow on a team project, such as roles to assume and activities to complete.

From the MSDN library, you can access the process guidance content for most artifacts defined by each of the default TFS process templates. You can browse the content available on MSDN from one of the following topics: Visual Studio Scrum, MSF for Agile, and MSF for CMMI.

To support access to process guidance from the work item forms in Team Explorer, a set of .htm files are uploaded to the Process Guidance folder within the Documents SharePoint library. These files specify URLs to the MSDN content that is opened when a team member chooses the process guidance  icon within a work item form. These files are uploaded based on the file tasks defined within the portal plug-in. For example, the Agile process template defines the following file tasks:

Activating dashboard features

Dashboards show project data, support investigation, and help teams quickly perform common tasks. Dashboards display several Excel reports and Team Web Access Web parts.

You use the **activateFeatures** element to cause the creation of the dashboards and Excel reports. You must include the following code in the portal plug-in file, within the **Portal** element, to activate the creation of the dashboards and Excel reports for a team project.

To activate dashboard features that are designed for use with the process template for Visual Studio Scrum:

Copy Code

```
<Portal>
. . .
<activateFeatures>
  <!-- TfsDashboardScrumWss -->
  <feature featureId="71EC0D67-21AB-4560-A825-9D976DA09D04"
  <!-- TfsDashboardAgileQuickLaunch -->
  <feature featureId="1D363A6D-D9BA-4498-AD1A-9874ACA5F827"
</activateFeatures>
. . .
</Portal>
```

To activate dashboard features that are designed for use with the process template for MSF Agile software development:

Copy Code

```
<Portal>
. . .
<activateFeatures>
  <!-- TfsDashboardAgileMoss -->
  <feature featureId="0D953EE4-B77D-485b-A43C-F5FBB9367207"
  <!-- TfsDashboardAgileQuickLaunch -->
  <feature featureId="1D363A6D-D9BA-4498-AD1A-9874ACA5F827"
</activateFeatures>
```

```
· · ·  
</Portal>
```

To activate dashboard features that are designed for use with the process template for MSF CMMI process improvement:

 **Copy Code**

```
<Portal>  
· · ·  
<activateFeatures>  
  <!-- TfsDashboardCmmiMoss -->  
  <feature featureId="3D0BA288-BF8E-47F0-9680-7556EDEF6318"  
  <!-- TfsDashboardCmmiProcessDocLibraries -->  
  <feature featureId="8610B95B-063F-4FB5-837C-BCF2FE9423C6"  
</activateFeatures>  
· · ·  
</Portal>
```

See Also

Concepts

Customize a process template

[Overview of process template files](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Add reports to the process template

[See Also](#) [Send Feedback](#)

By using the Reporting plug-in, you can specify the folder structure and set of reports that will appear under the  Reports node for a new team project.

Important

The Reporting plug-in requires that the team project collection where you will create a team project has been configured with SQL Server Analysis Services and SQL Server Reporting Services. For more information, see

[Add reports to a team project.](#)

You specify two main tasks by using the Reporting plug-in. First, you create the reporting site by using the **site** element. Second, you specify the folder structure and the reports to be uploaded to the site by using the **folder** and **report** elements. The reports that you upload have an .rdl extension and are designed for use with Reporting Services. For an overview of the reports that the TFS process templates provide, see Reports (SQL Server Reporting Services).

After a team project is created from the process template, you can add, remove, rename, and change the folder structure for reports. For more information, see View, upload, and organize reports (Reporting Services Reports).

■ Reporting plug-in name and location

The following table summarizes the names of the file, the folder, and the plug-in for the MSF process templates.

File name:	ReportsTasks.xml
Folder name:	Reports
Plug-in name:	Microsoft.ProjectCreationWizard.Reporting

■ Note

You can change the names of the XML file and the folder but not the plug-in. Visual Studio Team Foundation Server does not include a mechanism for the deployment of client-side plug-ins, policies, or other modifications. If you want to deploy this kind of functionality, you must use your own distribution and installation program.

In the XML file, you specify one or more tasks and their dependencies. For an example of a task that specifies a set of reports, see the ReportsTasks.xml file in the MSF for Agile Software Development process template.

Reporting plug-in syntax structure

The Reporting plug-in file must conform to the schema definition for ReportingServices, which is defined in the Rosetta.xsd file, and be specified in its own file.

The following syntax shows the structure of the Reporting plug-in. For a description of each element, see

[ReportingServices Child Elements](#) later in this topic.



```
<?xml version="1.0" encoding="utf-8"?>
<tasks>
  <task id="Site" plugin="Microsoft.ProjectCreationWizard.Reporting"
    <dependencies />
    <taskXml>
      <ReportingServices>
        <site> </site>
      </ReportingServices>
    </taskXml>
  </task>
  <task id="Populate Reports" plugin="Microsoft.ProjectCreationWizar
    <dependencies>
      <dependency taskId="Site" />
    </dependencies>
    <taskXml>
      <ReportingServices>
        <folders> . . . </folders>
        <reports> . . . </reports>
      </ReportingServices>
    </taskXml>
  </task>
</tasks>
```

■ Creating the Reports site

For the reports to run correctly, you must create a reporting site. The reporting site will have a link to it on the project portal home page, and the link will be labeled  Reports. To create the reporting site, use the **site** element, as the following example shows:

Copy Code

```
<taskXml>
  <ReportingServices>
    <site></site>
  </ReportingServices>
</taskXml>
```

Defining folders

You can create folders on the reporting site by using the **folder** element. Use the path attribute to specify the relative path name of the new folder. The folder appears on the project site and under the Reports folder in Team Explorer.

The following example creates four top-level folders and two sub-folders on the reporting site. The top-level folders are labeled Bugs, Builds, Project Management, and Tests. The two sub-folders are located under Project Management and are labeled Visual Studio and Visual Studio ALM.

Copy Code

```
<taskXml>
  <ReportingServices>
    <folders>
      <folder path="Bugs" />
      <folder path="Builds" />
      <folder path="Project Management" />
      <folder path="Project Management/Visual Studio" />
      <folder path="Project Management/Visual Studio ALM" />
      <folder path="Tests" />
    </folders>
  </ReportingServices>
</taskXml>
```

■ Specify the reports to upload

Reports are defined by .rdl files. To add reports to the reporting site, copy the report .rdl files into a folder under the Reports folder in the process template. Then use the report element to describe the necessary properties and data sources for the report.

The following example specifies that the Remaining Work.rdl file will be uploaded to the Project Management folder. You must specify the **ExplicitProject** parameter and the data sources for each report. The data sources in the following example correspond to the names that are automatically assigned to the Analysis Services cube and data warehouse relational database.

📌 Note

The names of the reporting services data sources are Tfs2010ReportsDS and Tfs2010OlapReportsDS. Even though you might have installed or upgraded to TFS 2012 or TFS 2013, these names, which were assigned to the data sources for a TFS 2010 installation, continue to be used.



```
<taskXml>
  <ReportingServices>
    <reports>
      <report name="Remaining Work" filename="Reports\Remaining Wc
      <parameters>
        <parameter name="ExplicitProject" value="" />
      </parameters>
      <datasources>
        <reference name="/Tfs2010OlapReportDS" dsname="TfsOlapRep
        <reference name="/Tfs2010ReportDS" dsname="TfsReportDS" />
      </datasources>
    </report>
    . . .
  </reports>
</ReportingServices>
</taskXml>
```

ReportingServices element reference

The following syntax shows the structure of the **ReportingServices** element and its child elements. You specify these elements within the **taskXml** container element and only for the Reporting plug-in.

Copy Code

```
<ReportingServices>
  <folders>
    <folder />
    .
    .
    .
  </folders>
  <reports>
    <report>
      <parameters>
        <parameter />
      </parameters>
      <datasources>
        <reference />
      </datasources>
    </report>
    .
    .
    .
  </reports>
</ReportingServices>
```

The following table describes the elements that you use to specify a report to upload to a team project and that can be accessed through SQL Server Reporting Services.

Element	Syntax
datasources	<pre><datasources> <reference /> </datasources></pre>

Copy Code

folder

 **Copy Code**

```
<folder path="FolderName" />
```

folders

 **Copy Code**

```
<folders>
  <folder path="FolderName" />
</folders>
```

parameter

 **Copy Code**

```
<parameter name="ParameterName" value="ParameterValue" />
```

Where each attribute has the following definition:

- **name:** The name of a parameter that you want to specify.
- **value:** The value to assign to the parameter.

At a minimum, you must define the following parameter to automatically use the project that contains your report:

 **Copy Code**

```
<parameter name="ExplicitProject" value="" />
```

parameters

```
<parameters>  
  <parameter />  
</parameters>
```

 **Copy Code**

 **Copy Code**

properties

```
< properties >  
  < property />  
</ properties >
```

 **Copy Code**

```
<property name="PropertyName" value="PropertyValue"
```

property

Where each attribute has the following definition:

- **name:** The name of a property that you want to specify. For more information, see View, organize, and configure reports in the Reporting Services Manager.
- **value:** The value to assign to the property.

 **Copy Code**

```
<reference name="RelationalDBName" dsname="OLAPDBN:
```

Where each attribute has the following definition:

- **name:** The name of the data source in Team Foundation Server. In Team Foundation Server 2010, this value is either

reference

Tfs2010OlapReportDS for the Analysis Services cube c
Tfs2010ReportsDS for the relational data warehouse.

- **dsname:** The name of the database resource. Specify Tf for the Analysis Services cube, and TfsReportsDS for th warehouse.

For more information about the names of reporting data sour reports after upgrade to Team Foundation Server 2010.

 **Copy Code**

```
<report name="ReportName" filename="ReportFilePath"
folder="FolderName" cacheExpiration="CacheDuration"
  <parameters>. . .</parameters>
  <datasources>. . .</datasources>
  <properties>. . .</properties >
</report>
```

Where each attribute has the following definition:

report

- **name:** The name of the report to display on the reportin Team Explorer.
- **filename:** A relative path under the local Reports folder get the .rdl report file.
- **folder:** A relative path name of the location to add the r reporting site.
- **cacheExpiration:** The default number of minutes for w is cached.

 **Copy Code**

reports

```
<reports>
  <report />
  . . .
```

```
</reports>
```

 **Copy Code**

```
ReportingServices <ReportingServices>  
    <folders>. . .</folders>  
    <reports>. . .</reports>  
</ReportingServices>
```

 **Copy Code**

```
site <site> </site>
```

See Also

Concepts

Customize a process template

[Overview of process template files](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Configure initial groups, teams, members, and permissions

[See Also](#) [Send Feedback](#)

By using the plug-in file for Groups and Permissions, you can configure the initial security settings for a team project. You accomplish this by defining tasks that create security groups, nest groups, define groups as teams, configure initial team settings, assign members to groups, and allow or deny specific permissions to each group. In addition to performing these tasks, you can specify the initial security settings for collection-level, project-level, and project-classification areas.

Microsoft process templates assign several permissions to default groups. You can modify these assignments by customizing the plug-in file for Groups and Permissions. For more information about this plug-in, see [Define groups, teams, and permissions using the Groups and Permissions Plug-in](#).

In this topic

- [Define and assign permissions to groups](#)
- [Group macros and default groups](#)
- [Nest groups and assign members to groups](#)
- [Define a team](#)
- [Assign collection-level permissions](#)
- [Assign project-level permissions](#)
- [Assign permissions to control area paths](#)
- [Assign permissions to control iteration paths](#)

For information about how to configure the initial security settings for a team project's functional areas, such as Team Foundation Build, Team Foundation version control, and Visual Studio Lab Management, see [Control access to functional areas](#).

For information about how to customize types of work items to allow or deny access to groups or users, see [Apply a rule to a work item field](#).

For more information about how to administer users and groups and control access for Visual Studio Application Lifecycle Management (ALM), see [Manage users or groups in TFS](#).

Define and assign permissions to groups

You can use the **group** and **member** elements to specify a new security group in Team Foundation Server and add members to that group. You can use the **group permission** element to assign permissions to a group and to members of that group. You must encapsulate each of these elements within their corresponding container elements: **groups**, **members**, and **permissions**. You use the following syntax structure for each of these elements:

Copy Code

```
<group name="Group Name" description="Description of Group"></group>  
<member name="MemberName"></member>  
<permission name="PermissionName" class="ClassName" allow="True | False"></permission>
```

The following table describes the attributes for the **group**, **member**, and **group permission** elements. You use these elements only in the Groups and Permissions plug-in file.

Element	Attribute	Description
group	name	Specifies the name of the group that you are creating.
	isTeam	Indicates if the group is a team (true) or not (false).
	description	Describes the purpose of the group to other users.

Specifies the name of a group that you are adding as a member of another group. You can create groups and pre-populate them with any of the following types of members:

member name

- Default groups that are defined in Team Foundation Server
- Project groups that have been previously created in the groupsandpermissions.xml file (for example, [\$\$PROJECTNAME\$\$]\Contributors)
- Groups and users who are defined in Active Directory, which you specify by using the following format:
 - DOMAIN\USERNAME
 - DOMAIN\GROUPNAME

For information about the format to use when you specify default groups, see

[Group macros and default groups](#) later in this topic.

permission

name

Identifies which permission is being applied. For a list of the supported permissions, see the following sections later in this topic:

- [Assign collection-level permissions](#)
- [Assign project-level permissions](#)
- [Assign permissions to control area paths](#)

- [Assign permissions to control iteration paths](#)

class

Identifies the class, or area, where the group permission is granted. The following values are valid:

- **NAMESPACE**: Specifies collection-level permissions.
- **PROJECT**: Specifies project-level permissions.
- **CSS_NODE**: Specifies permissions for viewing and managing area paths for a team project.
- **ITERATION_NODE**: Specifies permissions for viewing and managing iteration paths for a team project.
-

allow

Uses a **true** or **false** value to indicate whether the permission is allowed or denied.

path

Identifies the node of the area path or iteration path where the permission is being applied. This attribute is valid only when **class** is set to **CSS_NODE** or **ITERATION_NODE**.

Group macros and default groups

The following table lists the macros that you can use to specify a default group that is defined in Team Foundation Server.

Note

You can specify the macros in the following table only in the plug-in for Groups and Permissions. You cannot specify these macros when you assign permissions by using the plug-ins for build, version control, or lab management.

Default groups	Macro
Project Collection Administrators	[SERVER]\\$\$PROJECTCOLLECTIONADMINGROUP\$\$ [SERVER]\\$\$TEAMFOUNDATIONADMINGROUP\$\$ \$\$COLLECTIONADMINGROUP\$\$
Project Collection Service Accounts	[SERVER]\\$\$PROJECTCOLLECTIONSERVICESGROUP\$\$
Project Collection Build Service Accounts	[SERVER]\\$\$PROJECTCOLLECTIONBUILDSERVICESGROUP\$\$ \$\$COLLECTIONBUILDSERVICESGROUP\$\$
Project Collection	[SERVER]\\$\$PROJECTCOLLECTIONBUILDADMINGROUP\$\$

Build Administrators \$\$COLLECTIONBUILDADMINISTRATORSGROUP\$\$

 \$\$PROJECTADMINGROUP\$\$

Project Administrators [\$\$PROJECTNAME\$\$]\\$\$PROJECTADMINGROUP\$\$

 [\$\$PROJECTNAME\$\$]\Builders

 \$\$CREATOR_OWNER\$\$

Project creator @creator

Default team @defaultTeam

Example: Nest groups and assign members to groups

The following example shows how to configure groups that are named TestGroup1, TestGroup2, and TestGroup3. In this example, you add TestGroup1 as a member of TestGroup2. For this code to be valid, you must define TestGroup1 before you define TestGroup2.

```
<task id="GroupCreation1">
  <taskXml>
    <groups>
      <group name="TestGroup1" description="Test group 1. Contains
        <permissions>
          <permission name="GENERIC_READ" class="PROJECT" allow="true" />
        </permissions>
      </group>
      <group name="TestGroup2" description="Test group 2. Contains
        <permissions>
          <permission name="GENERIC_READ" class="PROJECT" allow="true" />
        </permissions>
        <members>
          <member name="TestGroup1" />
          <member name="$$PROJECTADMINGROUP$$" />
        </members>
      </group>
      <group name="TestGroup3" description="Test group 3. Contains
        <permissions>
          <permission name="GENERIC_READ" class="PROJECT" allow="true" />
        </permissions>
        <members>
          <member name="DOMAIN\USER" />
          <member name="DOMAIN\GROUP" />
          <member name="[$$PROJECTNAME$$]\$$PROJECTADMINGROUP$$" />
          <member name="[SERVER]\$$PROJECTCOLLECTIONBUILDSERVICES" />
        </members>
      </group>
    </groups>
  </taskXml>
</task>
```

Define a team

In addition to creating groups, you can assign a group as a team. Creating a team project also creates a default team. If you have several teams that want to organize their work separately from the other teams, then you can either define these teams within the Groups and Permissions plug-in file, or you can configure them after you create the team project. See [Add another team or a hierarchy of teams](#).

The following example shows how to configure a group as a team. In this example, you specify the group, Dream Team, as a team and add the team project creator as a member of the team. Whatever iteration paths that you specify for the team must be defined in the Classifications plug-in file. See

[Define the initial areas and iterations in the classification plug-in.](#)

 **Copy Code**

```
<group name="Dream Team" isTeam="true" description="Next generation
  <permissions>
    <permission name="GENERIC_READ" class="PROJECT" allow="true" />
  </permissions>
  <members>
    <member name="@creator"/>
  </members>
  <teamSettings areaPath="Area">
    <iterationPaths backlogPath="Iteration">
      <iterationPath path="Release 1\Sprint 1" />
      <iterationPath path="Release 1\Sprint 2" />
      <iterationPath path="Release 1\Sprint 3" />
      <iterationPath path="Release 1\Sprint 4" />
      <iterationPath path="Release 1\Sprint 5" />
      <iterationPath path="Release 1\Sprint 6" />
    </iterationPaths>
  </teamSettings>
</group>
```

Assign collection-level permissions

You can assign collection-level permissions by using the group **permission** element and the `NAMESPACE` class. These permissions control access to resources that are available across team projects. You can set collection-level permissions for only the following categories of users:

- Collection-level users and groups, such as Project Collection Administrators
- Project-level groups that have been added to the collection level on your server that is running Team Foundation
- Custom groups that you create and add to the collection level

For the format to use when you specify groups, see

[Group macros and default groups](#) earlier in this topic.

Note

You can set these permissions by right-clicking the server in Team Explorer and then clicking Security, by opening and using the administration console for Team Foundation, or by using the **TFSSecurity** and **tf** command-line tools. For more information, see Collection-Level Groups, [Change groups and permissions with TFSSecurity](#), and Permission Command.

The following example shows how to grant collection-level permissions to the project administrators for a team project.



```
<group name="PROJECTADMINGROUP" description="Members of this group c
  <permissions>
    <permission name="GENERIC_READ" class="NAMESPACE" allow="true
    <permission name="WORK_ITEM_WRITE" class="NAMESPACE" allow="t
    <permission name="MANAGE_LINK_TYPES" class="NAMESPACE" allow=
    <permission name="MANAGE_TEMPLATE" class="NAMESPACE" allow="t
```

```
    <permission name="MANAGE_TEST_CONTROLLERS" class="NAMESPACE"
  </permissions>
</group>
```

The following table describes the collection-level permissions that you can assign.

Note

By default, no collection-level permissions are assigned in the MSF process templates.

Permission	Description
DIAGNOSTIC_TRACE	Alter trace settings. Can change the trace settings for gathering more detailed diagnostic information about Web services for Team Foundation Server.
CREATE_PROJECTS	Create new projects. Can create projects in the team project collection.
	Edit collection-level information. Can edit collection-level permissions for users and groups in the

GENERIC_WRITE

team project collection. Users who have this permission can perform the following tasks:

- Add, remove, or rename a collection-level application group from the collection in Team Foundation Server.

Note

You cannot remove default collection-level groups, such as Project Collection Administrators.

- Add or remove a user or group in Windows user or another application group in Team Foundation Server (at the server level).
- Change

collection-level permissions for users and groups.

Additionally, users who have this permission can modify permissions for version control, and they have write access to all files in version control unless their access is explicitly denied by other permissions.

MANAGE_TEMPLATE

Manage process templates. Can download, create, edit, and upload process templates to the team project collection.

MANAGE_TEST_CONTROLLERS

Manage test controllers. Can register and de-register test controllers for the team project collection.

MANAGE_LINK_TYPES

Manage work item link types. Can add, remove, and change

GENERIC_READ

the types of links for work items.

View collection-level information.

Can view membership of collection-level groups and the permissions of those users.

Assign project-level permissions

You can assign project-level permissions in the Groups and Permissions plug-in file. You assign these permissions by using the group **permission** element and the PROJECT class. These permissions control access to a single project's resources. You can grant access to users and groups in Windows, groups in Team Foundation, and groups that you have previously defined in the Groups and Permissions plug-in file. For the format to use when you specify groups, see

[Group macros and default groups](#) earlier in this topic.

The following example shows how to grant several permissions to the Contributors group for a team project.

 Copy C

```
<group name="Contributors" description="Members of this group can ac
  <permissions>
    <permission name="GENERIC_READ" class="PROJECT" allow="true" ,
    <permission name="DELETE_TEST_RESULTS" class="PROJECT" allow='
      <permission name="PUBLISH_TEST_RESULTS" class="PROJECT" allow
      <permission name="VIEW_TEST_RESULTS" class="PROJECT" allow="t
      <permission name="MANAGE_TEST_ENVIRONMENTS" class="PROJECT" ;
      <permission name="MANAGE_TEST_CONFIGURATIONS" class="PROJECT"
    </permissions>
  </group>
```

The following table describes the project-level permissions that you can assign and indicates the default assignments that are made in the MSF process templates.

Permission	Description	Readers	Contributo
	View project-level		

GENERIC_READ

information.

Can view membership of project-level groups and the permissions of those members. ✓



VIEW_TEST_RESULTS

View test runs.

Can view test plans in this node. ✓



MANAGE_TEST_CONFIGURATIONS

Manage test configurations. Can create and delete test configurations for the team project.



MANAGE_TEST_ENVIRONMENTS

Manage test environments. Can create and delete test environments for the team project.



PUBLISH_TEST_RESULTS

Create test runs. Can add and remove test results and add or modify



test runs for the team project.

DELETE_TEST_RESULTS

Delete test runs. Can delete a scheduled test for the team project.



DELETE

Delete team project. Can delete from Team Foundation Server the project for which the user has this permission.

GENERIC_WRITE

Edit project-level information. Can edit project-level permissions for users and groups in Team Foundation Server.

Assign permissions to control area paths

You can assign permissions that control access to area definitions by using the group **permission** element and the `CSS_NODE` class. These permissions control access to a single project's classification structure. You can grant access to users and groups in Windows, groups in Team Foundation, and groups that you have previously defined in the Groups and Permissions plug-in file. For information about the format to use when you specify groups, see

[Group macros and default groups](#) earlier in this topic.

The following example shows how to grant several permissions to the Contributors group for a team project.

 Copy C

```
<group name="Contributors" description="Members of this group can ac
  <permissions>
    <permission name="GENERIC_READ" class="CSS_NODE" allow="true"
    <permission name="WORK_ITEM_READ" class="CSS_NODE" allow="true"
    <permission name="WORK_ITEM_WRITE" class="CSS_NODE" allow="tru
    <permission name="MANAGE_TEST_PLANS" class="CSS_NODE" allow="t
  </permissions>
</group>
```

The following table describes the permissions that you can assign to control access to the hierarchical structure for the project's area and iteration nodes. The table also indicates the default assignments that are made in the MSF process templates.

Note

Some operations for tracking work items require multiple permissions. For example, you need multiple permissions to delete a node.

Permission	Description	Readers	Contributors	Bi Admin
GENERIC_READ	View this node. Can view the security settings for an area node.	✓	✓	✓
WORK_ITEM_READ	View work items in this node. Can view, but not change, work items that are assigned to an area node.	✓	✓	✓
WORK_ITEM_WRITE	Edit work items in this node. Can edit work items that are assigned to an area node.		✓	✓
MANAGE_TEST_PLANS	Manage test plans. Can create and edit test plans that are assigned to an area node. If test plans have not been run, you can also delete them.		✓	✓
	Create and order child nodes. Can create area nodes. Users who have			

CREATE_CHILDREN

both this permission and the **GENERIC_WRITE** permission can move or re-order any child area node.

Delete this node. Can delete area nodes.

DELETE

Users who have both this permission and the **GENERIC_WRITE** permission for another node can delete area nodes and reclassify existing work items from the deleted node. If the deleted node has child nodes, those nodes are also deleted.

GENERIC_WRITE

Edit this node. Can set permissions for and rename area nodes.

Assign permissions to control iteration paths

You assign permissions that control access to iteration paths by using the group **permission** element and the **ITERATION_NODE** class. These permissions control access to the milestone releases or iterations for a single project. You can grant access to users and groups in Windows, groups in Team Foundation, and groups that you have previously defined in the Groups and Permissions plug-in file. For information about the format to use when you specify groups, see

[Group macros and default groups](#) earlier in this topic.

The following example shows how to grant several permissions to the Contributors group for a team project:

 Copy C

```
<group name="Contributors" description="Members of this group can ac
  <permissions>
    <permission name="GENERIC_READ" class="ITERATION_NODE" allow='
    <permission name="GENERIC_WRITE" class="ITERATION_NODE" allow=
    <permission name="CREATE_CHILDREN" class="ITERATION_NODE" allc
  </permissions>
</group>
```

The following table describes the permissions that you can assign to control access to the hierarchical structure for the project's iteration nodes. Because the MSF process templates do not specify any **ITERATION_NODE** permissions, all team members can create, view, and delete iteration nodes.

Note

Some operations for tracking work items require multiple permissions. For example, you need multiple permissions to delete a node.

Permission	Description
GENERIC_READ	View this node. Can view the security settings for a node.
CREATE_CHILDREN	Create and order child nodes. Can create iteration nodes. Users who have both this permission and the GENERIC_WRITE permission can move or re-order any iteration node.
DELETE	Delete this node. Can delete iteration nodes. Users who have both this permission and the GENERIC_WRITE permission for another node can delete iteration nodes and reclassify existing work items from the deleted node. If the deleted node has child nodes, those nodes are

GENERIC_WRITE

also deleted.

Edit this node. Can set permissions for iteration nodes and rename nodes.

See Also

Concepts

[Define groups, teams, and permissions using the Groups and Permissions Plug-in](#)

[Control access to functional areas](#)

[Manage users or groups in TFS](#)

[Permission reference for Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define groups, teams, and permissions using the Groups and Permissions Plug-in

[See Also](#) [Send Feedback](#)

You can define security groups to control access to functional areas within a team project. In addition to the default security groups in Visual Studio Team Foundation Server, you can configure a team project's initial groups, group members, and security permissions by customizing the Groups and Permissions plug-in. With this plug-in, you can define groups, teams, add groups and users as members to groups, and grant permissions to the groups.

This topic describes the syntax structure of the **groups**, **iterationPath**, **members**, **permissions**, and **teamsettings** elements that are used in the file for the Groups and Permissions plug-in. For more information about how to use these elements, see

[Configure initial groups, teams, members, and permissions.](#)

■ Name and location of Groups plug-in

The Groups and Permission plug-in is defined by the GroupsandPermissions.xml plug-in file, which must conform to the schema definition that is defined in the Gss.xsd file. You can download the schema files for process templates from the following page on the Microsoft website:

[Process Template and Work Item Schemas for Visual Studio Team Foundation.](#)

The following table summarizes the names of the file, the folder, and the plug-in for the process templates for Microsoft Solutions Framework (MSF).

File name:	GroupsandPermissions.xml
Folder name:	Groups and Permissions
Plug-in name:	Microsoft.ProjectCreationWizard.Groups

■ Note

You can change the names of the XML file and the folder but not the plug-in. Visual Studio Team Foundation Server does not include a mechanism for the deployment of client-side plug-ins, policies, or other modifications. If you want to deploy this kind of functionality, you must use your own distribution and installation program.

In the Groups and Permissions plug-in, you specify one or more tasks and their dependencies within the taskXml element. Generally, you specify one task per security group to create for your process. For more information about how to

specify tasks, see [Define the tasks to process a plug-in.](#)

Define groups

You use the **group** element to specify a new security group in Team Foundation Server.

 **Copy Code**

```
<group name="GroupName" description="GroupDescription"></group>
```

The following example shows how to create a group that is named Reader:

 **Copy Code**

```
<task id="GroupCreation1"
  name="Create Groups and Permissions"
  plugin="Microsoft.ProjectCreationWizard.Groups"
  completionMessage="Groups and Permissions created.">
  <taskXml>
    <groups>
      <group name="Readers"
        description="A group for users who have read access"
        <permissions>
          <!-- permissions -->
        </permissions>
      </group>
    </groups>
  </taskXml>
</task>
```

Define members

You use the **member** element to assign a group as a member of a security group in Team Foundation Server.

Copy Code

```
<member name="MemberName" ></member>
```

Note

A group that is a team (**isTeam="true"**) cannot be a member of a group.

The following example shows how to add TestGroup1 as a member of TestGroup2.

Copy Code

```
<task id="GroupCreation1"
  <taskXml>
    <groups>
      <group name="TestGroup1" description="Test group 1. Contair
        <permissions>
          <permission name="GENERIC_READ" class="PROJECT" allow="t
        </permissions>
      </group>
      <group name="TestGroup2" description="Test group 2. Contair
        <permissions>
          <permission name="GENERIC_READ" class="PROJECT" allow="t
        </permissions>
        <members>
          <member name="TestGroup1" />
          <member name="$$PROJECTADMINGROUP$$" />
        </members>
      </group>
    </groups>
  </taskXml>
</task>
```

Define teams and team settings

Within the default Groups and Permissions plug-in file, the `@defaultTeam` macro creates the default team at the root area path. You can change this structure by including additional area paths within the Classification plug-in file. By using the **teamsettings** element, you can pre-configure the iterations assigned to a team. The plug-in uses the following code snippet. In this example, three iterations are defined for the default team.

⚠ Important

You must assign iteration paths that correspond to paths defined in the Classification plug-in file. See

[Define the initial areas and iterations in the classification plug-in.](#)

📄 Copy Code

```
<group name="@defaultTeam">
  <permissions>
    <permission name="GENERIC_READ" class="PROJECT" allow="true" />
  </permissions>
  <members>
    <member name="@creator"/>
  </members>
  <teamSettings areaPath="Area">
    <iterationPaths backlogPath="Iteration">
      <iterationPath path="Iteration 1" />
      <iterationPath path="Iteration 2" />
      <iterationPath path="Iteration 3" />
    </iterationPaths>
  </teamSettings>
</group>
```

You can also define additional teams within a team project. You do this by defining a group and assigning the `isTeam` attribute to true. The following example shows how to define a team and its permissions, members, and initial sprint assignments. Specify the default team settings for a team project.

📄 Copy Code

```
<group name="Dream Team" isTeam="true" description="Next generation
  <permissions>
    <permission name="GENERIC_READ" class="PROJECT" allow="true" ,
  </permissions>
  <members>
    <member name="@creator"/>
  </members>
  <teamSettings areaPath="Area">
    <iterationPaths backlogPath="Iteration">
      <iterationPath path="Iteration 1" />
      <iterationPath path="Iteration 2" />
      <iterationPath path="Iteration 3" />
    </iterationPaths>
  </teamSettings>
</group>
```

Define permissions

You must specify permissions for each group that you create. You use the **permission** element for this purpose.

Copy Code

```
<permission name="PermissionName" class="ClassName" allow="true" | fa
```

The following example shows how to grant permissions to the Reader security group so that members can view information about a team project, but they cannot modify that information.

Copy Code

```
<group name="Readers" description="A group for users who have read a
  <permissions>
    <permission name="GENERIC_READ" class="PROJECT" allow="true" />
    <permission name="GENERIC_READ" class="CSS_NODE" allow="true" />
    <permission name="WORK_ITEM_READ" class="CSS_NODE" allow="true" />
  </permissions>
</group>
```

Groups element reference

The following table describes the elements that you use to define the initial groups and permissions for a team project. You specify these elements within a taskXml container element in the Groups and Permissions plug-in file. For information about this element, see

[Define the tasks to process a plug-in.](#)

⚠ Caution

The Gss.xsd schema file does not define the property or properties elements. When you upload the process template, the Process Template Manager validates these elements before storing them in Team Foundation Server.

The groups and group (Groups and Permission) elements are distinct from the groups and group (Process Template) elements. For information on the latter pair of elements, see [Process template XML elements reference](#).

Copy Code

```
<group name="GroupName" isTeam="true | false" description="...">
  <permissions> . . . </permissions>
  <members> . . . </members>
</group>
```

The following definitions apply for each attribute:

group

- **name:** Required. Specifies the name of the group. The name can be up to 255 characters long.
- **isTeam:** Optional. Identifies the group as a team, which supports members to organize their work within a team project.
- **description:** Required when the group is not a team. Specifies the description of the group. The description is displayed within the security pages.

 **Copy Code**

groups

```
<groups>
  <group> . . . </group>
</groups>
```

 **Copy Code**

iterationPath

```
<iterationPath path="IterationName" />
```

 **Copy Code**

iterationPaths

```
<iterationPaths backlogPath="BacklogPathName">
. . . </iterationPaths>
```

 **Copy Code**

```
<member name="MemberName" ></member>
```

member

For information about how to specify default groups, see [Group M Defined in Team Foundation Server](#).

 **Copy Code**

members

```
<members>  
  <member> . . . </member>  
</members>
```

Copy Code

```
<permission name="PermissionName" class="ClassName" allow="true" />
```

Where the following definitions apply for each attribute:

permission

- **name:** Required. Specifies the name of the permission. For more information, see the table in [Configure initial groups, teams, members, and permissions](#). The class and name combination you can specify as a permission.
- **class:** Required. Identifies the class, or area, where the group is located. The following values are valid: `NAMESPACE` (collection-level), `CSS_NODE` (area node) and `ITERATION_NODE` (iteration node).
- **allow:** Optional. Specifies a true or false value that indicates whether to allow the permission.

Copy Code

permissions

```
<permissions >  
  <permission> . . . </permissions>  
</permissions >
```

Copy Code

```
teamsettings <teamSettings areaPath="Area">  
    .  
    .  
    .  
</teamSettings>
```

See Also

Concepts

[Configure initial groups, teams, members, and permissions](#)

[Control access to functional areas](#)

[Manage users or groups in TFS](#)

Customize a process template

Other Resources

[Apply a rule to a work item field](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Control access to functional areas

[See Also](#) [Send Feedback](#)

You can configure the initial security settings for the following functional areas for a team project: team queries, Team Foundation version control, Team Foundation Build, and Visual Studio Lab Management. The process templates for Microsoft Solutions Framework (MSF) assign several permissions to default security groups. You can modify these assignments by customizing the plug-in file for the appropriate functional area.

For information about how to configure security groups for Visual Studio Team Foundation Server, see [Configure initial groups, teams, members, and permissions](#).

For more information about how to administer users and groups and control access to Visual Studio Application Lifecycle Management (ALM), see [Manage users or groups in TFS](#).

Assign permissions to functional areas

You can use the functional **permission** element to allow or deny permissions for functional areas to a security group in Team Foundation Server, a Windows group, or a Windows identity. You use this element in the plug-in files for work item tracking, Team Foundation version control, Team Foundation Build, and Lab Management. You must encapsulate the permission element within its corresponding container: the **permissions** element. You use the following syntax structure for the functional **permission** element:

 **Copy Code**

```
<permission allow="PermissionName" identity="GroupName"/>  
<permission deny="PermissionName" identity="GroupName"/>  
<permission allow="PermissionName" deny="PermissionName" identity="(
```

The following table describes the attributes for the functional **permission** element:

Attribute	Description
	Identifies the permissions that are granted. You specify permissions as comma-delimited text.
allow	For the names of the permissions that have been defined for each functional area, see the following sections later in this topic: <ul style="list-style-type: none">•

[Assigning Version Control Permissions](#)

- [Assigning Build Permissions](#)
- [Assigning Lab Management Permissions](#)

deny

Identifies the permissions that are revoked. You specify permissions as comma-delimited text.

Note

Denied permissions take precedence over allowed permissions.

identity

Specifies the security group in Team Foundation Server, the Windows group, or the Windows identity to which the permissions are applied. For the format to use when you specify groups, see "Default Groups Defined in Team Foundation Server" in [Configure initial groups, teams, members, and permissions](#).

The following example shows how to grant permissions to allow the Contributors group to view builds and build definitions and to queue builds and edit build quality.

Copy Code

```
<taskXml>  
  <permission allow="Read, PendChange, Checkin, Label, Lock" identi:  
</taskXml>
```

Note

During runtime, if a permission cannot be found for an identity, the permission is searched for in any other groups to which the identity belongs. If the permission cannot be found, the permission is denied by default.

Assign permissions for work item queries

In the workitems plug-in file, you can assign permissions that control access to team query folders. Query folder permissions are specific to queries and query folders. You can grant access to users and groups in Windows or to default groups that are defined for Team Foundation Server.

You assign these permissions by using the functional **permission** element, as the following example shows:



```
<Permission allow="Read, Contribute, Delete, ManagePermissions, FullControl" />
```

Note

After the team project is created, you can set permissions by right-clicking a query folder or query in Team Explorer and then clicking Security. For more information, see Set permissions on queries.

The following table describes the permissions that control access to query folders and queries. It also indicates the default assignments that are made in the MSF process templates. By default, the creators or owners of queries and query folders have full control of managing the queries that they created or own.

Permission	Description	Readers, Contributors, Builders	Creator Owners, Project Administrator Group, Project Collection Administrators
------------	-------------	---------------------------------	--

Read	Read. Can view and run a query or view a query folder and its contents ✓	✓
Contribute	Contribute. Can view and edit a query or query folder and its contents	✓
Delete	Delete. Can view, edit, and delete a query or query folder and its contents	✓
ManagePermissions	Manage Permissions. Can manage permissions for a query or query folder and its contents	✓
	Full	

FullControl

Control.
Can view,
edit, delete,
and manage
permissions
for a query
or query
folder and
its contents



Assign permissions for Version Control

You can assign permissions that control access to source code files and folders by changing the plug-in file for version control. Version control permissions are specific to source code files and folders. You can grant access to users and groups in Windows or default groups that are defined for Team Foundation Server.

You assign these permissions by using the functional **permission** element, as the following example shows:

 **Copy Code**

```
<permission allow="Read, PendChange, Checkin, Label, Lock, Merge" id
```

Note

After the team project is created, you can set these permissions by right-clicking the folder or file in Source Control Explorer, clicking Properties, and clicking the Security tab. On that tab, you can click the user or group for which you want to change permissions and then edit the permissions that are listed in Permissions. You can also set these permissions by using the **tf** command-line tool for version control or the TFSSecurity command-line tool. For more information, see

[Permission reference for Team Foundation Server.](#)

The following table describes the permissions that control access to source code files and folders. It also indicates the default assignments that are made in the MSF process templates.

Permission	Description	Readers	Contributors	Builders	Administrators	Project Group
------------	-------------	---------	--------------	----------	----------------	---------------

Read

Read. Can display the contents of a file or folder.

If a user has Read permissions for a folder but not the files that it contains, the user can display the names and properties of those files, but the user cannot open them.



PendChange

Check out. Can check out and make a pending change to an item. Examples of pending changes include adding, editing, renaming,



	deleting, undeleting, branching, and merging a file.			
Merge	Merge. Can merge changes into the path for which they have permissions.	✓	✓	✓
Checkin	Check in. Can check in items and revise any committed changeset comments. Pending changes are committed when the user checks in the item.	✓	✓	✓
Label	Label. Can label items.	✓	✓	✓
Lock	Lock. Can lock an item so that other users cannot	✓	✓	✓

update it.

ReviseOther

Revise another user's changes. Can change the contents of someone else's changeset comments and check-in notes.



UnlockOther

Unlock another user's changes. Can remove someone else's lock.



UndoOther

Undo another user's changes. Can undo someone else's pending changes.



Administer labels. Can

LabelOther	modify someone else's label.	✓
AdminProjectRights	Manage permissions. Can configure security settings for version control.	✓
CheckinOther	Check in another user's changes. Can perform a check-in as another user. This permission is required for conversion utilities.	✓
	Manage branch. Users who have this permission for a given path can convert any	

ManageBranch

folder under that path into a branch. Users who have this permission for a branch can also edit its properties, re-parent it, and convert it to a folder.



Users who have this permission can branch this branch only if they also have the Merge permission for the target path. Users cannot create branches from a branch for which they do not have the Manage Branch permission.

Assigning permissions for Build

You can assign permissions that control access to build activities by changing the Build plug-in file. You can grant access to users and groups in Windows and groups in Team Foundation Server. For information about the format to use when you specify groups, see "Default Groups Defined in Team Foundation Server" in

[Configure initial groups, teams, members, and permissions.](#)

You assign these permissions by using the functional **permission** element, as the following example shows:



```
<Permission allow="ViewBuildDefinition, QueueBuilds, ViewBuilds, Ed:
```

Note

After the team project is created, you can set these permissions by opening the project in Team Explorer, right-clicking Builds, and then clicking Security. You can apply permissions to a specific build definition by right-clicking the build definition and then clicking Security. If you want to apply permissions to a build folder, right-click it, and then click Security. Additionally, you can set these permissions by using the TFSSecurity command-line tool. For more information, see [Permission reference for Team Foundation Server](#).

The following table describes the permissions that you can assign that control access to the build functions of a team project. The table also indicates the default assignments that are made in the MSF process templates.

Note

The Override check-in validation by build permission should be assigned only to service accounts for build services and to build administrators who are responsible for the quality of the code. For more information, see Check in to a

folder that is controlled by a gated check-in build process.

Permission	Description	Readers	Contributors	E Admin
ViewBuildDefinition	View build definition. Can view the build definitions that have been created for the team project.	✓	✓	✓
ViewBuilds	View builds. Can view the queued and completed builds for this team project.	✓	✓	✓
EditBuildQuality	Edit build quality. Can add information about the quality of the build through the interface for Team Foundation Build.		✓	✓

QueueBuilds	Queue builds. Can add a build to the queue through the interface for Team Foundation Build or at a command prompt.	✓	✓
DeleteBuildDefinition	Delete build definition. Can delete build definitions.		✓
DeleteBuilds	Delete builds. Can delete a completed build.		✓
DestroyBuilds	Destroy builds. Can permanently delete a completed build.		✓
	Edit build definition.		

EditBuildDefinition

Can create and modify build definitions.



ManageBuildQualities

Manage build qualities. Can add or remove build qualities, such as Ready for Deployment, Rejected, or Under Investigation. For more information, see Add or remove build quality values.



ManageBuildQueue

Manage build queue. Can cancel, re-prioritize, or postpone queued builds.



Retain indefinitely. Can mark a build so that

RetainIndefinitely

it will not be automatically deleted by any applicable retention policy.



StopBuilds

Stop builds. Can stop a build that is in progress.



OverrideBuildCheckInValidation

Override check-in validation by build. Can commit a changeset that affects a gated build definition without triggering the system to shelve and build the changes first. For more information, see Check in to a folder that is controlled by a gated check-in build

process.

Update build information.
Can add information about the quality of a build.

UpdateBuildInformation

This permission should be assigned only to service accounts.

Assign permissions for Lab Management

You can control access to activities in Lab Management by changing the Lab plug-in file. Permissions for Lab Management are specific to virtual machines, environments, and other resources. You can grant access to users and groups in Windows and groups in Team Foundation Server. You assign these permissions by using the functional **permission** element, as the following example shows:



```
<permission allow="Read, Create, Write, Edit, Start, Stop, ManageSna
```

Note

You can set permissions for Lab Management by using the **TFSLabConfig** command-line tool. To display information about a specific lab resource, you must have the Read permission for that resource. To delete a location, you must have the Delete Lab Locations permission for that location. For more information, see

[TFSLabConfig Permissions Command.](#)

The following table describes the permissions that you can assign to control access to Visual Studio Lab Management. The table also indicates the default assignments that are made in the MSF process templates.

Permission	Description	Readers	Contributors	Project Collection Build Service Accounts group
------------	-------------	---------	--------------	---

Read

View lab resources. Can view information for the various resources for Lab Management, which include collection host groups, project host groups, and environments.



Create

Import virtual machine. Can import a virtual machine from a Virtual Machine Manager (VMM) library share.

This permission differs from Write because users can create an object in Lab Management but not write anything to the VMM host group or library share.



Write environment and

Write	<p>virtual Machines. Can create environments. Users who have this permission for a project library share can store environments and virtual machines.</p>	✓	✓	✓
Edit	<p>Edit Environment and Virtual Machines. Can edit environments and virtual machines. The permission is checked for the object that is being edited.</p>	✓	✓	✓
Start	<p>Start. Can start an environment.</p>	✓	✓	✓
Stop	<p>Stop. Can stop an environment.</p>	✓	✓	✓
Pause	<p>Pause. Can pause an environment.</p>		✓	✓

ManageSnapshots

Manage snapshots. Can perform all snapshot management tasks, which include taking a snapshot, reverting to a snapshot, renaming a snapshot, deleting a snapshot, and reading a snapshot.



Delete

Delete environments and virtual machines. Can delete environments and virtual machines. The permission is checked for the object that is being deleted.

Manage lab locations. Can edit the locations of resources for

ManageLocation

Lab Management, which include collection host groups, collection library projects, project host groups, and project library shares.

This permission for collection-level locations (collection host groups and collection library shares) also allows a user to create project-level locations (project host groups and project library shares).

DeleteLocation

Delete lab locations. Can delete the locations of resources for Lab Management, which include collection host groups,

collection
library shares,
project host
groups, and
project library
shares.

ManageChildPermissions

Manage Child
Permissions.
Can change the
permissions of
all the child
objects of Lab
Management.
For example, if
a user has this
permission for a
team project
host group, that
user can change
permissions for
all the
environments
under that
group.

ManagePermissions

Manage
Permissions.
Can modify the
permissions for
an object of Lab
Management.
This permission
is checked for
the object whose
permissions are
being modified.

EnvironmentOps

Environment Operations. Can start, stop, pause, and manage snapshots, in addition to performing other operations on an environment.

See Also

Concepts

[Configure initial groups, teams, members, and permissions](#)
[Permission reference for Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define objects for tracking work items using the work Item tracking plug-in
[See Also](#) [Send Feedback](#)

By using the plug-in for tracking work items, you define a team project's initial objects for tracking work. These objects include types of work items, work item queries, categories, link types, and instances of work items. After you create a team project, you can modify these objects by using the **witadmin** command-line tool.

In the plug-in file, you specify one or more tasks and their dependencies. In general, you specify one task for each major upload activity, such as link types, work item types, queries, and categories.

Important

You must specify the tasks to upload files and define queries in a specific sequence: link types first, then work item types, and then queries. Each definition file for these objects depends on the definitions that are specified in the tasks that precede them. In general, you should maintain the task sequence that is defined in the process template that you are customizing. For more information, see

[Define dependencies for task groups and tasks in plug-in files.](#)

■ Plug-in name and location

The following table summarizes the name and location of the file for the process templates for Microsoft Solutions Framework (MSF), in addition to the name of the plug-in.

File name:	WorkItems.xml
File location:	WorkItem Tracking folder
Plug-in name:	Microsoft.ProjectCreationWizard.WorkItemTracking

■ Note

You can change the name and location of the XML file but not the name of the plug-in. Visual Studio Team Foundation Server does not include a mechanism for the deployment of client-side plug-ins, policies, or other modifications. If you want to deploy this kind of functionality, you must use your own distribution and installation program.

For specific information about each type of object that you can define, see one of the following topics:

-

[Add type definitions for work items to a process template](#)

- [Add link type definitions to a process template](#)
- [Add type definitions for work item categories to a process template](#)

- [Add work item queries to a process template](#)
- [Add a work item instance to a process template](#)

■ Work item tracking plug-in syntax structure

The plug-in file for work item tracking must conform to the schema definition that is defined in the WorkItemMethodology.xsd file, and the plug-in must be specified in its own file. You can download the schema files for process templates from the following page on the Microsoft website:

[Process Template and Work Item Schemas for Visual Studio Team Foundation.](#)

The following syntax shows the high-level structure of the WorkItemTracking plug-in. Four tasks are specified, one each to upload the definitions for link types, types of work items, queries, and categories. If you were to add definitions work item instances, you would specify an additional task for each.

For a description of each element, see [Element reference](#) later in this topic.

```
<?xml version="1.0" encoding="utf-8"?>
<tasks>
  <task id="LinkTypes" name="LinkType definitions" plugin="Microsoft
    <taskXml>
      <LINKTYPES>
        <LINKTYPE />
        . . .
      </LINKTYPES>
    </taskXml>
  </task>
  <task id="WITs" name="WorkItemType definitions" plugin="Microsoft.
    <dependencies>
      <dependency taskId="LinkTypes" />
    </dependencies>
    <taskXml>
      <WORKITEMTYPES>
        <WORKITEMTYPE />
        . . .
      </WORKITEMTYPES>
    </taskXml>
  </task>
```

```

<task id="Queries" name="Stored Query Definitions" plugin="Microsc
  <dependencies>
    <dependency taskId="WITs" />
  </dependencies>
  <taskXml>
    <QUERIES>
      <Permission />
      . . .
      <QueryFolder >
        <Query />
      . . .
      </QueryFolder>
      . . .
    </QUERIES>
  </taskXml>
</task>
<task id="Categories" name="Categories definitions" plugin="Microsc
  <dependencies>
    <dependency taskId="WITs" />
  </dependencies>
  <taskXml>
    <CATEGORIES fileName="WorkItem Tracking\Categories.xml" />
  </taskXml>
</task>
</tasks>

```

Element reference

The following table describes the elements that you use to upload work item tracking objects. You specify these elements within a **taskXml** container element in the WorkItemTracking plug-in file. For information about the task, dependency, and taskXml elements, see

[Define the tasks to process a plug-in.](#)

Element	Syntax
CATEGORIES	 Copy Code <pre><CATEGORIES fileName="CategoriesFilePathName" /></pre>
FIELD	 Copy Code <pre><FIELD refname="FieldReferenceName" value="Value" /></pre>

HYPERLINK

 **Copy Code**

```
<HYPERLINK URL="URL" relativePath="false" />
```

LINKTYPE

 **Copy Code**

```
<LINKTYPE fileName="LinkTypeFilePathName" />
```

LINKTYPES

 **Copy Code**

```
<LINKTYPES>
  <LINKTYPE />
  .
  .
  .
</LINKTYPES>
```

Permission

 **Copy Code**

```
<permission allow="ListOfPermissions" identity='
```

Query

 **Copy Code**

```
<Query name="QueryName" fileName="QueryFilePath"
```

QueryFolder

 **Copy Code**

```
<QueryFolder name="
FolderName
">
```

QUERIES

 **Copy Code**

```
<QUERIES>
.
.
.
</QUERIES>
```

WI

 **Copy Code**

```
<WI type="WorkItemType">
  <FIELD> . . .</FIELD>
  <HYPERLINK> . . .</HYPERLINK>
</WI>
```

 **Copy Code**

WORKITEMS

```
WORKITEMS>  
  <WI>  
    <FIELD> . . .</FIELD>  
    <HYPERLINK> . . .</HYPERLINK>  
  </WI>  
</WORKITEMS >
```

 **Copy Code**

WORKITEMTYPE

```
<WORKITEMTYPE fileName="WITFilePathName" />
```

 **Copy Code**

WORKITEMTYPES

```
<WORKITEMTYPES>  
  <WORKITEMTYPE />  
  .  
  .  
  .  
</WORKITEMTYPES>
```

See Also

Concepts

Customize a process template

Customize work tracking objects to support your team's processes

[Overview of process template files](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Add type definitions for work items to a process template

[See Also](#) [Send Feedback](#)

The work item types (WITs) that you define for a process template provide the foundation for all tracking, monitoring, and reporting on the development of a product and its features. A WIT defines the data fields, the workflow, and the work item form for an item of work that will be tracked in a team project. Types of work items include bugs, user stories, and tasks. You can first customize the type definitions to add fields, change the workflow, or modify the work item form. Also, you can add or remove WITs from a process template.

You specify the type definitions for a process template for several work items as a task within the WorkItemTracking plug-in. This task is required because work item types support tracking and reporting work. You specify each type definition file to upload within the **taskXml** element. The plug-ins for test management, reports, and the portal depend on the successful upload of the definitions for WITs.

The Team Foundation Server (TFS) process templates define six or more types of work items. The types and the fields that are defined within them are referenced in the definitions of categories, work item queries, and reports. Therefore, the task to upload the definitions for WITs must successfully complete before the tasks to upload categories, work item queries, and reports.

Also, the task to upload the definition files for link types must precede the task to upload the type definitions for WITs. For more information, see

[Define dependencies for task groups and tasks in plug-in files.](#)

After a team project is created by using the process template, you can add, remove, rename, and change the definitions of WITs by using the **witadmin** command-line tool. For more information, see [Import, export, and manage work](#)

[item types \[witadmin\]](#).

Defining WITs

Each type definition must be specified in its own file in the **WITD** container element. Each definition must also conform to the schema definition for work item types, which is defined in the `workitemtypedefinition-02.xsd` and `workitemtypedefinition-02.xsd` files. You can download the schema files for tracking work items from the following page on the Microsoft website:

[Process Template and Work Item Schemas for Visual Studio Team Foundation.](#)

The following example shows the high-level syntax structure that defines a work item type.

 **Copy Code**

```
<WITD application="work item type editor" version="1.0">
  <WORKITEMTYPE name="Bug">
    <DESCRIPTION>Bug work items are used to track defects in the
    <GLOBALLISTS> . . . </GLOBALLISTS>
    <FIELDS> . . . </FIELDS>
    <WORKFLOW> . . . </WORKFLOW>
    <FORM> . . . </FORM>
  </WORKITEMTYPE>
</WITD>
```

You can customize or create type definition files in the `TypeDefinitions` folder. For more information about how to define a type of work item, see [All WITD XML elements reference](#) and [Customize work tracking objects to support your team's processes.](#)

Specifying definitions of WITs to upload

To upload a WIT definition in the process template, you specify the **WORKITEMTYPE** element. The filename attribute is a relative path of the type definition file. For example, the following syntax specifies that the Bug.xml file will be uploaded.

Copy Code

```
<WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\filename.>
```

The following example shows how to specify a task that creates the following WITs: bug, issue, shared steps, task, test case, and user story. Because the definitions for several WITs contain filters for the **TestedBy** link type, the **LinkTypes** task must be completed before the **WITs** task can be completed.

```
<task id="WITs" name="WorkItemType definitions" plugin="Microsoft.Pr
  <dependencies>
    <dependency taskId="LinkTypes" />
  </dependencies>
  <taskXml>
    <WORKITEMTYPES>
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Bu
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Is
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Cc
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Cc
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Fe
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Fe
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Fe
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Sh
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\T
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Te
      <WORKITEMTYPE fileName="WorkItem Tracking\TypeDefinitions\Us
    </WORKITEMTYPES>
  </taskXml>
</task>
```

WORKITEMTYPES element reference

The following table describes the elements that you use to upload the type definitions for work items. You specify these elements within a **taskXml** container element in the WorkItemTracking plug-in file.

Note

By using the **WORKITEMTYPE** (WorkItemTracking) element, you specify a type definition file to upload. By using the **WORKITEMTYPE** (WITD) element, you specify the name of a type to define. For more information, see

[All WITD XML elements reference.](#)

Element	Syntax	Req of WO
WORKITEMTYPE	 <code><WORKITEMTYPE fileName="WITFilePathName" /></code>	Spec nam cont defir
WORKITEMTYPES	 <code><WORKITEMTYPES> <WORKITEMTYPE /> . . . </WORKITEMTYPES></code>	Req of th Wor plug Con of WO

elen
spec
to up

See Also

Reference

[Import, export, and manage work item types \[witadmin\]](#)

Concepts

[All WITD XML elements reference](#)

[Define objects for tracking work items using the work Item tracking plug-in](#)

Other Resources

[Modify or add a field to support queries, reports, and workflow](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Add link type definitions to a process template

[See Also](#) [Send Feedback](#)

You can define additional types of links that team members can create between work items for a team project. A type of link defines the labels and rules that govern the relationships or links between work items of that type. The link types that you specify supplement the default types for hierarchical, dependent, and related links. You specify the definition file for each type of link in a separate file, or you can specify them all in one file. You then specify the definition files to upload within the **taskxml** element of the WorkItemTracking plug-in.

The Team Foundation Server (TFS) process templates define two types of links that are referenced in the definitions of several types of work items. These are in addition to the system defined link types. For more information, see

[Link type element reference](#).

The type definitions for links must be uploaded before the type definitions for work items. For more information, see [Define dependencies for task groups and tasks in plug-in files](#).

After a team project is created, you can add, remove, rename, and change the types of links for a team project collection by using the **witadmin** command-line tool. For more information, see [Manage link types \[witadmin\]](#).

Define link types

You must specify each link type definition using the **LinkType** element. In addition, each definition must conform to the schema definition for link types, which are defined in the `workitemlinktypedefinition-01.xsd` file. You can download the schema files for tracking work from the following page on the Microsoft website:

[Process Template and Work Item Schemas for Visual Studio Team Foundation.](#)

The following example shows the syntax structure that defines the **TestedBy** link type.

 Copy

```
<?xml version="1.0" encoding="utf-8"?>
<LinkTypes>
  <LinkType ReferenceName="Microsoft.VSTS.Common.TestedBy" ForwardName="TestedBy" />
</LinkTypes>
```

You can customize or create definition files for types of links in the `LinkTypes` folder. For more information about how to define a type of link, see [Define a custom link type](#).

[Back to top](#)

Specify link type definitions to upload

To upload a link type definition in the process template, you specify the **LINKTYPE** element within the **taskxml** element. The filename attribute is a relative path of the definition file for the link type. For example, the following syntax specifies that the TestedBy.xml file will be uploaded.

Copy Code

```
<LINKTYPE fileName="WorkItem Tracking\LinkTypes\TestedBy.xml" />
```

The following example shows how to specify a task that creates the two types of links that are defined in the process template for MSF for Agile Software Development v5.0. These types correspond to the **SharedStep** and **TestedBy** link types.

```
<task id="LinkTypes" name="LinkType definitions" plugin="Microsoft.F
  <taskXml>
    <LINKTYPES>
      <LINKTYPE fileName="workItem Tracking\LinkTypes\SharedStep.
      <LINKTYPE fileName="workItem Tracking\LinkTypes\TestedBy.xr
    </LINKTYPES>
  </taskXml>
</task>
```

[Back to top](#)

LINKTYPES element reference

The following table describes the elements that you use to upload link type definitions. You specify these elements within a **taskXml** container element in the WorkItemTracking plug-in file.

Note

You specify the file that defines link types to upload by using the **LINKTYPE** (WorkItemTracking) element. You specify the name of a link type to define by using the **LinkType** (Definition) element.

Element	Syntax	Description
LINKTYPE	 Copy Code <pre><LINKTYPE fileName="LinkTypeFilePathName" /></pre>	Required child element of LINKTYPES . Specifies the file and name of the link type that contains the type definition to upload.
LINKTYPES	 Copy Code <pre><LINKTYPES> <LINKTYPE /> . . </LINKTYPES></pre>	Optional child element of the taskXml container element in the WorkItemTracking plug-in. Contains a collection of LINKTYPE elements that specify a

definition fi
upload.

See Also

Reference

[Manage link types \[witadmin\]](#)

Concepts

[Define objects for tracking work items using the work Item tracking plug-in](#)

Other Resources

[Define a custom link type](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Add type definitions for work item categories to a process template

[See Also](#) [Send Feedback](#)

You can add categories to your process template. A category associates a group label with one or more types of work items. Categories are useful when your team projects contain similar work item types that are named differently. You specify the category definitions in one file, and then you specify that file to upload within the **taskxml** element of the WorkItemTracking plug-in.

The Team Foundation Server (TFS) process templates define over ten categories that are used to support Agile planning tools, test case management, and other features. You can customize or create a category definition file. For the MSF process templates, the category file is defined in the WorkItem Tracking folder. For more information, see [Use categories to group work item types](#).

After a team project is created from the process template, you can export and import categories for a team project collection by using the **witadmin** command-line tool. For more information, see [Import and export categories \[witadmin\]](#).

Define categories

The file that defines the categories must conform to the schema definition for categories, defined in the categories-01.xsd file, and must be specified in its own file in the **WITD** container element.

The following example shows the syntax structure that defines a category that is named Requirement Category and that is associated with the **User Story** work item type.

 **Copy Code**

```
<CATEGORIES>
  <CATEGORY refname="Microsoft.RequirementCategory" name="Requirement Category" >
    <DEFAULTWORKITEMTYPE name="User Story" />
  </CATEGORY>
  .
  .
  .
</CATEGORIES>
```

Specify a category definition file to upload

To upload a set of category definitions, you specify the **CATEGORIES** element within the **taskxml** element. The filename attribute is a relative path of the category definition file. For example, the following syntax specifies that the categories.xml file will be uploaded.

Copy Code

```
<CATEGORIES fileName="WorkItem Tracking\categories.xml" />
```

The following example shows how to specify a task that uploads a categories file. Because each category specifies a default work item type, the task to upload the category definition file depends on the successful completion of the **WITs** task which uploads the type definitions for work items.

```
<task id="Categories" name="Categories definitions" plugin="Microsoft
  <dependencies>
    <dependency taskId="WITs" />
  </dependencies>
  <taskXml>
    <CATEGORIES fileName="WorkItem Tracking\Categories.xml" />
  </taskXml>
</task>
```

CATEGORIES element reference

The following table describes the **CATEGORIES** element that you use to upload the category definition file. You specify this element within a **taskXml** container element in the WorkItemTracking plug-in file.

Note

You specify a definition file to upload using the **CATEGORIES** (WorkItemTracking) element. You specify the set of categories to define using the **CATEGORIES** (Definition) element.

Element	Syntax	D
		Optic elem Work plug-
	 Copy Code	Spec and r
CATEGORIES	<code><CATEGORIES fileName="CategoriesFilePathName" /></code>	file t the c defin uplo: Work plug- proce

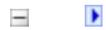
See Also

Reference

[Import and export categories \[witadmin\]](#)

Concepts

[Define objects for tracking work items using the work Item tracking plug-in](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Add work item queries to a process template

[See Also](#) [Send Feedback](#)

By adding work item queries to your process template, you can define the initial set of shared queries and query folder structure for a team project. All team members use queries to find the bugs, tasks, and other work items on which they must take action.

Work item queries specify criteria for generating a list of work items, such as a list of active bugs or closed tasks. Files for work item queries have a .wiq extension and are stored in the Queries subfolder of the WorkItem Tracking folder for Microsoft Solutions Framework (MSF) process templates.

Important

Microsoft process templates define several queries. For more information, see [Shared queries \(Scrum\)](#), [Shared queries \(Agile\)](#), and [Shared queries \(CMMI\)](#). Query definitions depend on the fields and types of work items in the process template. Also, some workbooks that are uploaded within the Portal plug-in depend on the query definitions.

You specify the query definitions to upload as a task within the WorkItemTracking plug-in. This task may be required because several artifacts in a process template may depend on a query. In addition, the task to upload queries depends on the successful completion of the task for uploading work item types. You specify each query file to upload within the **taskXml** element.

In this topic

-

[Creating a File for a Work Item Query](#)

- [Specifying Queries to Upload](#)
- [QUERIES Element Reference](#)

After a team project is created from the process template, you can add, remove, and change queries by using the Query Editor. For more information, see [Query for work items](#).

For information about how to assign query permissions to groups, see [Control access to functional areas](#). For information about how to create query folders, see [Define objects for tracking work items using the work Item tracking plug-in](#).

Creating a File for a Work Item Query

Each query definition must be specified in its own file with an extension of .wiq, using the **WorkItemQuery** parent element, and conform to the schema that is defined in the wiq.xsd file. You can download the schema files for tracking work items from the following page on the Microsoft website:

[Process Template and Work Item Schemas for Visual Studio Team Foundation.](#)

The following example shows the high-level syntax structure that defines a work item query:

 **Copy Code**

```
<WorkItemQuery Version="1">
<TeamFoundationServer>collectionURL</TeamFoundationServer>
<TeamProject>TeamProjectName</TeamProject>
  <Wiq1>
    WorkItemQueryLanguage
  </Wiq1>
</WorkItemQuery>
```

For more information about how to create a work item query to add to a process template, see [Define a work item query to add to a process template.](#)

[Back to top](#)

■ Specifying Queries to Upload

To include the work item queries in the process template, create one or more tasks in the workitems.xml file, which you can find in the \WorkItem Tracking folder, which is in the folder to which you downloaded your process template. Use the **Query** element to specify the file for the work item query. For example, the following XML specifies the query that is defined in the ActiveBugs.wiq file to be uploaded and named Active Bugs.

Copy Code

```
<Query name="Active Bugs" fileName="WorkItem Tracking\Queries\Active
```

You add the set of queries to upload as a task in the WorkItemTracking plug-in.

The following example shows how to specify a task to create a query folder that is named Product Management and upload a query that is named All User Stories to that folder.

```
<task id="Queries" name="Stored Query Definitions" plugin="Microsoft
  <dependencies>
    <dependency taskId="WITs" />
  </dependencies>
  <taskXml>
    <QUERIES>
      <QueryFolder name="Product Management" >
        <Query name="All User Stories" fileName="WorkItem Track:
      </QueryFolder>
    </QUERIES>
  </taskXml>
</task>
```

For more information, see

[Define objects for tracking work items using the work Item tracking plug-in.](#)

[Back to top](#)

QUERY Elements

The following syntax shows the structure of the **QUERIES** element and its child elements.

 **Copy Code**

```
<QUERIES>
  <Permission />
  <QueryFolder >
    <Query />
  </QueryFolder>
</QUERIES>
```

The following table describes the elements that you use to specify the query folder structure, permissions, and queries to upload. You specify these elements within a **taskXml** container element in the WorkItemTracking plug-in file.

Element	Syntax
Permission	 Copy Code <code><permission allow="ListOfPermissions" identity="GroupNa</code>
	 Copy Code

```
<Query name="QueryName" fileName="QueryFilePathName" />
```

Query

As the following example shows, you can upload the query that is "Active Bugs" and that is defined in the ActiveBugs.wiq file:

```
<Query name="Active Bugs" fileName="WorkItem  
Tracking\Queries\ActiveBugs.wiq" />
```

 **Copy Code**

QueryFolder

```
<QueryFolder name="FolderName">  
  <Query />  
</QueryFolder>
```

 **Copy Code**

QUERIES

```
<QUERIES>  
  .  
  .  
  .  
</QUERIES>
```

[Back to top](#)

See Also

Concepts

Query for work items

[Define objects for tracking work items using the work Item tracking plug-in](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Visual Studio Application Lifecycle Management

Define a work item query to add to a process template

[See Also](#) [Send Feedback](#)

You can create a work item query (.wiq) file most easily in Team Explorer. In an existing team project, you can use Team Explorer to create all work item queries that you want to add to your process template and then follow the steps in this topic to save each query as a .wiq file. For more information about how to create work item queries, see [Query for work items](#).

Requirements

- To create a query, you must be a member of the Readers group, or your View work items in this node permission must be set to Allow for each team project in the query.
- To save a query as a team query, you must have the appropriate permissions described in [Set permissions on queries](#).

Save a work item query as a .wiq file

1. In Team Explorer, open the query that you want to save..
2. On the File menu, choose Save <name of query> [Query] As.
3. In the Save Query As dialog box, choose File, and specify a location and file name for the .wiq file. As an alternative, choose Browse, specify a file name, browse to the location to save the file, and then choose Save.
4. Copy the file to the \WorkItem Tracking\Queries folder, which is in the folder to which you downloaded your process template.
5. Open the .wiq file in a text editor.
6. Remove the <TeamFoundationServer> and <TeamProject> elements that associate the query with a specific server that is running Visual Studio Team Foundation Server and a specific team project, as the following example shows:

Copy Code

```
<TeamFoundationServer>CollectionURL</TeamFoundationServer>  
<TeamProject>ProjectName</TeamProject>
```

Note

You must edit the .wiq file in a text editor and remove the <TeamFoundationServer> and <TeamProject> elements that associate the query with a specific server and team project. Otherwise, the query will not work correctly if the process template is uploaded to a different server. As an alternative, use macros where you can so that your query does not contain the explicit name of the current team project or any other values that are specific to a certain environment.

7. Remove any additional elements that are specific to a user.

■ Create a work item query that references an iteration path

You can define a query that references a specific iteration path by including the macro for the team project, `$$PROJECTNAME$$`, and the name of an iteration path that is defined in the Classification plug-in file. For example, the following syntax specifies a query that includes only those work items whose iteration path is under Iteration 1.

Copy Code

```
AND [Source].[System.IterationPath] UNDER '$$PROJECTNAME$$\Iteratic
```

When the team project is created, the macro is replaced with the name of the team project.

By using this macro, you can define workbooks that reference specific iteration paths. The process template for MSF agile software development contains an iteration-specific query, `Iteration1Backlog.wiq`, that supports the iteration-specific workbook, `Iteration Backlog.xlsm`.

When you upload iteration-specific queries, the task to process the `Classification.xml` file must complete before the task to process the query files. For more information, see

[Define the initial areas and iterations in the classification plug-in](#) and [Define the root tasks using the process template plug-in file File](#).

■ Example of an iteration-specific work item query

The following example shows the `Iteration1Backlog` work item query, which supports the `Iteration Backlog` workbook.

Copy Code

```

<?xml version="1.0" encoding="utf-8"?>
<WorkItemQuery Version="1">
  <Wiq1>
    SELECT [System.Id],
           [System.WorkItemType],
           [System.Title],
           [System.State],
           [System.AssignedTo],
           [Microsoft.VSTS.Scheduling.RemainingWork],
           [Microsoft.VSTS.Scheduling.CompletedWork],
           [Microsoft.VSTS.Scheduling.StoryPoints],
           [Microsoft.VSTS.Common.StackRank],
           [Microsoft.VSTS.Common.Priority],
           [Microsoft.VSTS.Common.Activity],
           [System.IterationPath],
           [System.AreaPath]
    FROM WorkItemLinks
    WHERE (Source.[System.TeamProject] = @project
           AND [Source].[System.AreaPath] UNDER @project
           AND [Source].[System.IterationPath] UNDER '$$PROJECTNAME$$\]
           AND (
               Source.[System.WorkItemType] = 'User Story'
               OR Source.[System.WorkItemType] = 'Task'
           )
           )
           AND [System.Links.LinkType] = 'System.LinkTypes.Hierarchy-For
           AND [Target].[System.WorkItemType] = 'Task'
           ORDER BY [Microsoft.VSTS.Common.StackRank], [Microsoft.VSTS.C
           mode(Recursive)
  </Wiq1>
</WorkItemQuery>

```

WorkItemQuery element reference

The following syntax shows the structure of the **WorkItemQuery** element and its child elements.

 **Copy Code**

```
<WorkItemQuery Version="1">
  <TeamFoundationServer>collectionURL </TeamFoundatonServer>
  <TeamProject>TeamProjectName </TeamProject>
  <Wiql>
    WorkItemQueryLanguage
  </Wiql>
</WorkItemQuery>
```

Element

Syntax

Opt

 **It**

In ;
to j

 **Copy Code**

TeamFoundationServer <TeamFoundationServer>collectionURL
</TeamFoundatonServer>

Spe
foll

htt

If n
UR

htt

The
of 2

Op

⚠️ In

 **Copy Code**

In ;
to j

TeamProject

```
<TeamProject>TeamProjectName</TeamProject>
```

Spe

The
len;

Rec

 **Copy Code**

Spe
con
a te
nur

Wiql

```
<Wiql>WorkItemQueryLanguage</Wiql>
```

The
SQ
loc.

See Also

Concepts

Query for work items

[Define objects for tracking work items using the work Item tracking plug-in](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Add a work item instance to a process template

[See Also](#) [Send Feedback](#)

By adding work items to your process template, you define a set of tasks or other items that act as reminders for each project manager to perform after the team project is created. For example, you can create one task each to assign permissions to team members, define product areas and milestones, set up build definitions, and determine the branch strategy for version control.

 **Note**

Defining work item instances is optional. The process templates for Microsoft Solutions Framework (MSF) do not include any work item instances.

In this topic

-

[Defining Work Item Instances](#)

- [WORKITEMS Element Reference](#)

Defining Work Item Instances

To define a work item instance, you specify the work item type along with field values for those work items. You use the **WI** and **FIELD** elements that are contained within the **WORKITEMS** element. You specify the set of work item instances within a **taskXml** element in the WorkItemTracking plug-in file. Each work item instance must conform to the schema definition for work items, which is defined in the WorkItemMethodology.xsd file. You can download the schema files for tracking work items from the following page on the Microsoft website:

[Process Template and Work Item Schemas for Visual Studio Team Foundation.](#)

The type attribute for the **WI** element specifies which work item type is being created, such as task, user story, or issue. You must assign values to each required field based on the type of work item.

The following example specifies a work item task that reminds project administrators to assign team members to one or more security groups. Values are assigned to all required fields. Because work item instances depend on the type definitions for work items, you must specify the task for creating work item instances after the task for creating work item types.

```
<task id="WIT_Instances" name="Work Item tasks definitions" plugin='
  <dependencies>
    <dependency taskId="WITs" />
  </dependencies>
  <taskXml>
    <WORKITEMS>
      <WI type="Task">
        <FIELD refname="System.Title" value="Setup: Set Permission
        <FIELD refname="System.IterationPath" value="$$PROJECTNAME
        <FIELD refname="System.State" value="Active" />
        <FIELD refname="System.Reason" value="New" />
        <FIELD refname="System.Description" value="Using the admir
      </WI>
    </WORKITEMS>
  </taskXml>
</task>
```

[Back to top](#)

WORKITEMS Element Reference

The following syntax shows the structure of the **WORKITEMS** element and its child elements. You specify these elements within a **taskXml** element in the WorkItemTracking plug-in file.

Copy Code

```
<WORKITEMS>
  <WI type="TypeA">
    <FIELD refname="Field1" value="Value1" />
    <FIELD refname="Field2" value="Value2" />
    . . .
    <FIELD refname="FieldN" value="ValueN" />
  </WI>
  . . .
</WORKITEMS>
```

Element	Syntax
	 Copy Code
FIELD	<pre><FIELD refname="FieldReferenceName" value="Value" /></pre>
	 Copy Code
HYPERLINK	<pre><HYPERLINK URL="URL" relativePath="false" /></pre>

WI

 **Copy Code**

```
<WI type="WorkItemType">  
  <FIELD> . . .</FIELD>  
  <HYPERLINK> . . .</HYPERLINK>  
</WI>
```

WORKITEMS

 **Copy Code**

```
<WORKITEMS>  
  <WI>  
    <FIELD> . . .</FIELD>  
    <HYPERLINK> . . .</HYPERLINK>  
  </WI>  
</WORKITEMS >
```

[Back to top](#)

See Also

Concepts

Customize a process template

[Define objects for tracking work items using the work Item tracking plug-in](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Map Microsoft Project fields to Team Foundation fields

[See Also](#) [Send Feedback](#)

You can customize the way in which work item fields that are stored in Team Foundation are mapped to fields that are defined for Microsoft Project. You can change the way specific fields are published and designate the default link type to use when you create hierarchical or tree links and dependency links.

Unlike Microsoft Excel, Microsoft Project uses a limited set of columns, which include predefined columns, such as Task Name, and custom fields. When a user publishes or refreshes work item data in a Microsoft Project file, the field map determines which fields in the work item database match the columns in Microsoft Project.

You can customize the mappings, for example, to support a field that you created or to map fields to predefined columns instead of to custom columns. For complete documentation on the field mapping file for Microsoft Project, see [Add or change how Project fields map to TFS fields](#).

You customize the mapping by modifying the FileMapping.xml file. The following table summarizes the names of the file, the folder, and the plug-in for the process templates for Microsoft Solutions Framework (MSF). The plug-in contains the definition of the task that uploads the file to the team projection collection.

File name:	FileMapping.xml
------------	-----------------

Folder name:	Classification
--------------	----------------

Plug-in name: Microsoft.ProjectCreationWizard.Classification
For more information, see [Define the initial areas and iterations in the classification plug-in.](#)

Specify how fields are mapped

You specify mapped fields by using the **Mapping** element and the following syntax structure:

 **Copy Code**

```
<Mappings>
  <Mapping WorkItemTrackingFieldReferenceName="System.Id"
    ProjectField=""
    ProjectName=""
    ProjectUnits=""
    PublishOnly=""
    IfSummaryRefreshOnly=""/>
</Mappings>
```

The following table describes the attributes of the **Mapping** element.

Attribute	Description
WorkItemTrackingFieldReferenceName	Required. Specifies the reference name of a field in a type of work item.
ProjectField	Required. Specifies the name of a column in Microsoft Project. Specify predefined column names by prefixing "pj" to the name, such as pjTaskName for the Task Name column. Specify custom fields as pjTaskText followed by

a number, such as
pjTaskText11.

ProjectName

Optional. Specifies the name to display as the column name to the user. If this attribute is not specified, the name of the field in the type of work item is used.

ProjectUnits

Optional. Specifies the type of units to use when you map a type of field to Microsoft Project. Valid values are pjMinute, pjHour, pjDay, pjWeek, and pjMonthUnit.

PublishOnly

Optional. If set to true, indicates that the field is published to the work item database but not refreshed. This value is typically used for calculated fields that should not be updated in Team Explorer. If set to false, indicates that the field is both published and refreshed. By default, this attribute is set to false.

By default, only Start

Date and Finish Date have this attribute set to true.

Optional. If set to true, indicates that the field is never published to the work item database but is refreshed from the work item database when the following conditions are also true:

- The row for the field is a summary task in Microsoft Project.
- The summary task has the values of Publish and Refresh set to Yes.
- The summary task contains at least one child task that is bound to Team Foundation Server.

Any updates or calculations that Microsoft Project makes can overwrite the value that is refreshed from the work item database in the project plan. However, the modified value is never saved to

IfSummaryRefreshOnly

the work item database. This attribute is typically used for summary fields that, if published to the work item database, lead to data inconsistencies.

If set to false, indicates that the field may be both published and refreshed. By default, this attribute is set to false.

 **Note**

The IfSummaryRefreshOnly attribute supersedes the PublishOnly attribute. For more information, see

[Customize the Microsoft Project field mapping file.](#)

By default, the **IfSummaryRefreshOnly** attribute is set to true for the Original Estimate, Remaining Work, and Completed Work fields.

Specify the fields for synchronization, links, and attachments

In addition to the **Mapping** element, two other elements determine how fields will synchronize and which field will support links and attachments.

The **SyncField** element specifies which column serves as the synchronization field. The synchronization field is titled "Publish and Refresh" and allows the user to indicate whether a task row is published or refreshed only.

You use the following XML syntax to specify which column maps to the synchronization field. You set the ProjectField attribute to a valid column in Microsoft Project.

 **Copy Code**

```
<SyncField ProjectField="" />
```

The **LinksField** element specifies the links and attachments column. By using the links and attachments column, users can indicate whether a particular task row has links or attachments.

You use the following XML syntax to specify which column maps to the links and attachments field. You set the ProjectField attribute to a valid column in Microsoft Project.

 **Copy Code**

```
<LinksField ProjectField="" />
```

Default field mappings

The following example shows how the MSF for Agile Software Development process template maps fields in work items to columns in Microsoft Project.

```
<?xml version="1.0" encoding="utf-8"?>
<MSProject>
  <Mappings>
    <Mapping WorkItemTrackingFieldName="System.AreaPath" ProjectField="pjTaskText25" />
    <Mapping WorkItemTrackingFieldName="System.AssignedTo" ProjectField="pjTaskText26" />
    <Mapping WorkItemTrackingFieldName="System.Id" ProjectField="pjTaskText27" />
    <Mapping WorkItemTrackingFieldName="System.IterationPath" ProjectField="pjTaskText28" />
    <Mapping WorkItemTrackingFieldName="System.Reason" ProjectField="pjTaskText29" />
    <Mapping WorkItemTrackingFieldName="System.Rev" ProjectField="pjTaskText30" />
    <Mapping WorkItemTrackingFieldName="System.State" ProjectField="pjTaskText31" />
    <Mapping WorkItemTrackingFieldName="System.Title" ProjectField="pjTaskText32" />
    <Mapping WorkItemTrackingFieldName="System.WorkItemType" ProjectField="pjTaskText33" />
    <Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Common.Accepted" ProjectField="pjTaskText34" />
    <Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Common.Assigned" ProjectField="pjTaskText35" />
    <Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Common.Closed" ProjectField="pjTaskText36" />
    <Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Common.Completed" ProjectField="pjTaskText37" />
    <Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Common.Declined" ProjectField="pjTaskText38" />
    <Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Common.InProgress" ProjectField="pjTaskText39" />
    <Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Common.Open" ProjectField="pjTaskText40" />
    <Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Common.Rejected" ProjectField="pjTaskText41" />
    <Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Common.Scheduled" ProjectField="pjTaskText42" />
    <LinksField ProjectField="pjTaskText26" />
    <SyncField ProjectField="pjTaskText25" />
  </Mappings>
</MSProject>
```

See Also

Concepts

[Define the initial areas and iterations in the classification plug-in](#)

Add or change how Project fields map to TFS fields

[Customize the Microsoft Project field mapping file](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define the initial configuration of Team Foundation version control

[See Also](#) [Send Feedback](#)

By using the plug-in for version control, you can configure a team project's initial security permissions, check-out policies, and check-in notes.

In the XML file, you specify one or more tasks and their dependencies. Generally, you need only one task to configure settings for version control. For an example of a task that specifies these settings, see the VersionControl.xml file that is defined for a process template for Microsoft Solutions Framework (MSF).

The following table summarizes the names of the file, the folder, and the plug-in for the process templates for MSF.

File name:	VersionControl.xml
------------	--------------------

Folder name:	Version Control
--------------	-----------------

Plug-in name:	Microsoft.ProjectCreationWizard.VersionControl
---------------	--

Note

You can change the name of the XML file and the folder name but not the name of the plug-in. Team Foundation Server does not include a mechanism for the deployment of client-side plug-ins, policies, or other modifications. If you want

to deploy this kind of functionality, you must use your own distribution and installation program.

Exclusive Check Out

You can control whether multiple people can check out a file at the same time by specifying the **exclusive_checkout** element.

 **Copy Code**

```
<exclusive_checkout required=""/>
```

If the **required** attribute is set to **true**, only one person can check out a file at a time. If this attribute set to **false**, multiple people can check out a file at the same time, and they must reconcile changes when they check in the file.

The following example shows how to require exclusive check-out:

 **Copy Code**

```
<exclusive_checkout required="true"/>
```

Get Latest on Check Out

You use the `get_latest_on_checkout` element to configure the default behavior when a user checks out a file for a team project.

 **Copy Code**

```
<get_latest_on_checkout required=""/>
```

If the **required** attribute is set to **true**, the most recent version of an item, or tip, is downloaded every time that a user checks it out. This behavior resembles the check-out behavior in Visual SourceSafe.

If the **required** attribute is set to **false**, the check-out operation will check out the local version in your workspace. By default, this attribute is set to false.

The following example shows how to specify that the default check-out behavior is to get the most recent version of an item when a user checks it out.

 **Copy Code**

```
<get_latest_on_checkout required="true"/>
```

Check-in Notes

The developer provides check-in notes when he or she checks in code. These notes describe whether the code changes are related to team processes and, if they are, how. For example, a check-in note can indicate whether the change was made because of a security review, and the note can include details about the changes relative to the security review.

You use the following syntax for a `checkin_note` element

 **Copy Code**

```
<checkin_note label="" required="" order=""/>
```

The following table describes the attributes of the `checkin_note` element.

Attribute	Description
label	The label that describes the check-in note. The label appears in the Pending Check-ins dialog box when a user checks in a change.
required	Specifies whether the check-in note is required to have a value. If this attribute is set to true, the check-in note must have a value. If this attribute is set to false, a value is optional.

order	Specifies an ordinal number to indicate in what order to display the check-in notes. This attribute is optional.
-------	--

The following example shows how to customize the check-in notes for MSF for Agile Software Development to create an additional check-in note that is named "Documentation Impact" and that is not required to have a value.

Copy Code

```
<taskXml>
  <checkin_note label="Code Reviewer" required="false" order="1"/>
  <checkin_note label="Security Reviewer" required="false" order="2"/>
  <checkin_note label="Performance Reviewer" required="false" order="3"/>
  <checkin_note label="Documentation Impact" required="false"/>
</taskXml>
```

Permissions

Version control has a specific set of permissions that you can configure for a process template. By specifying permissions, you define what actions security groups and individuals can perform on items that are under version control. For more information, see

[Control access to functional areas.](#)

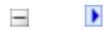
See Also

Concepts

[Configure initial groups, teams, members, and permissions](#)

Other Resources

Use version control



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define the initial configuration of Test Manager

[See Also](#) [Send Feedback](#)

By using the plug-in for Microsoft Test Manager, you can define a team project's initial test management setup. You can define settings such as test variables and test configurations that combine test variables. For example, you can define the hardware platform, the operating system, the browser version, or another type of hardware or software to be used for testing, and then you can define test configurations that combine these variables. In addition, you can customize the default test settings and the list of valid test resolution states. After you create a team project, you can modify each test configuration through Microsoft Test Manager except for test resolution states.

The test management plug-in file is provided with the process templates for Microsoft Solutions Framework (MSF). Microsoft Test Manager is available with Visual Studio Ultimate and Visual Studio Test Professional. For more information, see [Testing the application](#).

Note

You can customize the initial security configuration for test activities by assigning them in the GroupsandPermissions.xml plug-in file. For more information, see [Configure initial groups, teams, members, and permissions](#).

The following table summarizes the names of the file, the folder, and the plug-in for the process templates for MSF.

File name:	testmanagement.xml
------------	--------------------

Folder
name: Test Management

Plug-
in name: Microsoft.ProjectCreationWizard.TestManagement

Note

You can change the names of the XML file and the folder but not the plug-in. Team Foundation Server does not include a mechanism for the deployment of client-side plug-ins, policies, or other modifications. If you want to deploy plug-ins, policies, or other modifications to Team Explorer, you must use your own distribution and installation program.

In this topic

- [Test Management Tasks and Dependencies](#)
- [Defining Test Environments](#)
- [Defining Test Configurations](#)
- [Defining the Resolution States for Test](#)
- [Defining the Default Test Settings](#)

■ Test Management Tasks and Dependencies

In the testmanagement.xml file, you specify one or more tasks and their dependencies. The plug-in file specifies four tasks, and each task uploads a test management file. The testconfiguration.xml file depends on the information that is specified in the testvariable.xml file. For more information about the **task**, **taskXml**, and **dependency** elements, see

[Define the tasks to process a plug-in](#) and [Define dependencies for task groups and tasks in plug-in files](#).

The following code represents the default testmanagement.xml file that is defined for the MSF process templates:

```
<?xml version="1.0" encoding="utf-8"?>
<tasks>
  <task id="TestVariable" name="Default test variables" plugin="Micro
    <taskXml>
      <TestVariables fileName="Test Management\TestVariable.xml" />
    </taskXml>
  </task>
  <task id="TestConfiguration" name="Default test configurations" pi
    <dependencies>
      <dependency taskId="TestVariable" />
    </dependencies>
    <taskXml>
      <TestConfigurations fileName="Test Management\TestConfiguratio
    </taskXml>
  </task>
  <task id="TestSettings" name="Default test settings" plugin="Micro
    <taskXml>
      <TestSettings fileName="Test Management\TestSettings.xml" />
    </taskXml>
  </task>
  <task id="TestResolutionState" name="Default test resolution state
    <taskXml>
      <TestResolutionStates fileName="Test Management\TestResolutio
    </taskXml>
```

```
</task>  
</tasks>
```

Defining Test Configuration Variables

You use the **TestVariable** and **AllowedValue** elements to define test configuration variables. You can define any number of test configuration variables and their valid values. The following values are defined in the testvariable.xml file in the MSF process templates:

- Operating systems:
 - Windows Vista
 - Windows XP
- Default browsers:
 - Internet Explorer 7.0
 - Internet Explorer 8.0
 - Firefox 3.0

After the project is created, you can modify these variables and create other variables. For more information, see Test configurations: specifying test platforms.

The **TestVariable** element must be encapsulated within its corresponding container element: **TestVariables**. You use the following syntax structure for these elements:

 **Copy Code**

```
<TestVariables>
  <TestVariable name="VariableName" name="Operating System" descrip
    <AllowedValue value ="Name of Allowed Value"/>
  </TestVariable>
</TestVariables>
```

You use the following syntax for the test variables that are defined in the MSF process templates:

Copy Code

```
<?xml version="1.0" encoding="utf-8" ?>
<TestVariables>
  <TestVariable
    name="Operating System" description="Default operati
    <AllowedValue value ="Windows 8" />
    <AllowedValue value ="Windows 7" />
    <AllowedValue value ="Windows Vista" />
    <AllowedValue value ="Windows XP" />
  </TestVariable>
  <TestVariable
    name="Browser" description="Default browsers">
    <AllowedValue value ="Internet Explorer 9.0" />
    <AllowedValue value ="Internet Explorer 8.0" />
    <AllowedValue value ="Internet Explorer 7.0" />
    <AllowedValue value ="FireFox 3.0" />
    <AllowedValue value ="Internet Explorer 10.0" />
  </TestVariable>
</TestVariables>
```

Defining Test Configurations

You use the **TestConfiguration** and **TestVariable** elements to define test configurations that combine one or more test configuration variables. One default configuration is defined in the testconfiguration.xml file in the MSF process template: Windows Vista and Internet Explorer 7.0. After the project is created, you can delete these configurations and create other configurations. For more information, see Test configurations: specifying test platforms.

You must encapsulate the **TestConfiguration** element within its corresponding container element: **TestConfigurations**. You use the following syntax structure for these elements:

 **Copy Code**

```
<TestConfigurations>
  <TestConfiguration name="Test Configuration Name" description="
    <TestVariable name="VariableName" value="Variable Value" />
    <TestVariable name="VariableName" value="Variable Value" />
  </TestConfiguration>
</TestConfigurations>
```

The following table describes the attributes for the **TestConfiguration** element.

Attribute	Description
state	Identifies the test configuration as active or inactive.
isdefault	Identifies the test configuration as a default configuration.

You use the following syntax for the test configurations that are defined in the MSF process templates.

 **Copy Code**

```
<?xml version="1.0" encoding="utf-8" ?>
<TestConfigurations>
  <TestConfiguration name="Windows 8"
    description="Default operating system for te
    <TestVariable
      name="Operating System" value="Windows 8" />
  </TestConfiguration>
</TestConfigurations>
```

Defining Resolution States for Test

You use the **TestResolutionState** element to specify the reasons why a test failed. The following states are defined in the `testresolutionstate.xml` in the MSF process template: Needs investigation, Test issue, Product issue, and Configuration issue.

Note

You cannot change these states or add states after the team project is created.

You must encapsulate the **TestResolutionState** element within its corresponding container element: **TestResolutionStates**.

You use the following syntax structure for these elements:

Copy Code

```
<TestResolutionStates>
  <TestResolutionState name="ResolutionName" />
</TestResolutionStates>
```

You use the following syntax for the resolution states that are defined in the MSF process templates.

Copy Code

```
<?xml version="1.0" encoding="utf-8" ?>
<TestResolutionStates>
  <TestResolutionState name="Needs investigation" />
  <TestResolutionState name="Test issue" />
  <TestResolutionState name="Product issue" />
  <TestResolutionState name="Configuration issue" />
</TestResolutionStates>
```

Defining the Default Test Settings for a Local Test Run

You use the **TestSetting** element to specify the name of the file to use when a test is run. The following file is defined in the testsettings.xml in the MSF process template: localrun.testsettings. For more information, see Setting Up Test Machines to Run Tests or Collect Data.

You must encapsulate the **TestSetting** element within its corresponding container element: **TestSettings**. You use the following syntax structure for these elements:

Copy Code

```
<TestSettings>
  <TestSetting name="Name of Test Setting " filename="FileName" />
</TestSettings>
```

You use the following syntax for the testsettings.xml file that is defined in the MSF process templates.

Copy Code

```
<?xml version="1.0" encoding="utf-8" ?>
<TestSettings>
  <TestSetting name="Local Test Run" filename="localrun.testsettir
</TestSettings>
```

See Also

Concepts

Testing the application



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define the initial configuration of Team Foundation Build

[See Also](#) [Send Feedback](#)

You can customize a team project's initial build permissions that are used by Team Foundation Build. The build.xml plug-in file specifies the security permissions for build activities of all team projects that are created with the same process template. The following template files are uploaded to the team project database:

Note

Starting with Team Foundation Server (TFS) 2013, the build.xml plug-in no longer uploads build template files.

The following table summarizes the names of the file, the folder, and the plug-in for the process templates for Microsoft Solutions Framework (MSF).

File name:	Build.xml
Folder name:	Build
Plug-in name:	Microsoft.ProjectCreationWizard.Build

Note

You can change the names of the XML file and the folder but not of the plug-in. Team Foundation Server does not include a mechanism for the deployment of client-side plug-ins, policies, or other modifications. If you want to deploy this kind of functionality, you must use your own distribution and installation program.

In the Build.xml file, you specify one or more tasks and their dependencies. The TSF plug-in file specifies the permissions that are assigned to specific TFS default groups. For information about how to customize the initial security configuration for Lab Management, see [Control access to functional areas](#). For more information about the **task**, **taskXml**, and **dependency** elements, see [Define the tasks to process a plug-in](#) and [Define dependencies for task groups and tasks in plug-in files](#).

The following code represents the default build.xml file that is defined for the TFS process templates:

```
<?xml version="1.0" encoding="utf-8"?>
<tasks>
  <task id="BuildTask" name="Add Build Permissions" plugin="Microsoft
    <dependencies />
    <taskXml>
      <!-- Project-level groups -->
      <Permission allow="ViewBuilds, ViewBuildDefinition" identity='
      <Permission allow="EditBuildQuality, ViewBuilds, QueueBuilds,
      <Permission allow="DeleteBuilds, DestroyBuilds, EditBuildQuali
      <Permission allow="DeleteBuilds, DestroyBuilds, EditBuildQuali

      <!-- Collection-level groups -->
      <Permission allow="EditBuildQuality, ManageBuildQueue, Overric
      <Permission allow="ViewBuildDefinition, EditBuildDefinition, [
      <Permission allow="DeleteBuilds, DestroyBuilds, EditBuildQuali
    </taskXml>
  </task>
</tasks>
```

See Also

Concepts

[Control access to functional areas](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Define the initial configuration of Lab Management

[See Also](#) [Send Feedback](#)

You can customize a team project's initial security configuration for activities that are performed by using Visual Studio Lab Management. The lab.xml plug-in file specifies the security permissions for lab management activities of all team projects that are created with the same process template. It also specifies the task to upload the default template file for Lab Management, which is named LabDefaultTemplate.xaml. This file is uploaded to the database for the team project.

The following table summarizes the names of the file, the folder, and the plug-in for the process templates for Microsoft Solutions Framework (MSF).

File name:	Lab.xml
Folder name:	Lab
Plug-in name:	Microsoft.ProjectCreationWizard. Lab

Note

You can change the names of the XML file and the folder but not the plug-in. Team Foundation Server does not include a mechanism for the deployment of client-side plug-ins, policies, or other modifications. If you want to deploy this

kind of functionality, you must use your own distribution and installation program.

In the Lab.xml file, you specify one or more tasks and their dependencies. The MSF plug-in file specifies one task, which uploads the lab management file, and the permissions that are assigned to specific default groups for Team Foundation. To customize the initial security configuration for Lab Management, see [Control access to functional areas](#). For more information about the **task**, **taskXml**, and **dependency** elements, see [Define the tasks to process a plug-in](#) and [Define dependencies for task groups and tasks in plug-in files](#).

The following code represents the default lab.xml file that is defined for the MSF process templates:

```
<?xml version="1.0" encoding="utf-8"?>
<tasks>
  <task id="LabTask" name="Create Lab area" plugin="Microsoft.Projec
    <dependencies />
    <taskXml>
      <ProcessTemplate Type="Custom" Filename="Lab\Templates\LabDefa
        <permission allow="Read, Create, Write, Edit, Delete, ManagePe
        <permission allow="Read, Create, Write, Edit, Delete, ManageCl
        <permission allow="Read, Create, Write, Edit, Start, Stop, Mar
        <permission allow="Read" identity="[$$PROJECTNAME$$]\@@Reader:
        <permission allow="Read, Write, Edit, Start, Stop, ManageSnap:
      </taskXml>
    </task>
  </tasks>
```

See Also

Concepts

[Configure and administer Lab Management](#)

[Control access to functional areas](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Customize the Microsoft Project field mapping file

[See Also](#) [Send Feedback](#)

You can customize how work item fields that are defined in Team Foundation map to fields in Microsoft Project, and you can change how specific fields are published. Microsoft Project includes predefined fields, such as Task Name, and custom fields. When you publish or refresh work item data in Microsoft Project, the field map determines which fields in the work item database match the fields in Microsoft Project.

For information on using Project and TFS to track work, see [Create your backlog and tasks using Project](#).

To modify the field mappings for a team project, you export and then import the Microsoft Project Mapping File using the TFSFieldMapping command line tool.

Mapping element

To specify a mapping between a work item field and a project column, you use the Mapping element.

You use the following XML syntax to specify a mapping between a work item type field and an Microsoft Project field. The Mapping element is then used to specify a field mapping.

Copy Code

```
<Mapping WorkItemTrackingFieldName=""  
  ProjectField=""  
  ProjectName=""  
  ProjectUnits=""  
  PublishOnly="">  
  IfSummaryRefreshOnly="" />
```

Attributes

The following table describes the attributes that can be used with the Mapping element.

Attribute	Description
WorkItemTrackingFieldName	Required. Specifies the reference name of a field in a work item type.
	Required. Specifies the name of a field in Microsoft Project. Specify predefined field names by prefixing "pj"

ProjectField

to the name, such as pjTaskName for the Task Name column. Specify custom fields as pjTaskText followed by a number, such as pjTaskText11.

ProjectName

Optional. Specifies the name to appear as the column name in Microsoft Project. If you do not specify this attribute, the field name of the work item type is used.

ProjectUnits

Optional. Specifies the type of units to use when you map a field type to Microsoft Project. You can specify the following values: pjMinute, pjHour, pjDay, pjWeek, and pjMonthUnit.

 **Note**

You can specify ProjectUnits only for fields that specify time duration or other time units.

Optional. If set to true,

PublishOnly

indicates that the field is published to the work item database but is not refreshed. This value is typically used for calculated fields that should not be updated in Team Explorer. If set to false, indicates that the field is both published and refreshed. The default value is false.

In the default mapping file, the two fields whose **PublishOnly** attribute is set to true are the Start Date and Finish Date.

Optional. If set to true, indicates that the field is never published to the work item database but is refreshed from the work item database when the following are also true:

- The row for the field is a summary task in Office Project.
- The summary task has the Publish and Refresh values set to Yes.

- The summary task contains at least one child task that is bound to Team Foundation Server.

IfSummaryRefreshOnly

Any updates or calculations that Microsoft Project makes can overwrite the value that is refreshed from the work item database in the project plan. However, the modified value is never saved to the work item database. This attribute is typically used for summary fields that, if published to the work item database, lead to data inconsistencies.

If set to false, indicates that the field may be both published and refreshed. The default value when not specified is false.

Note

The value that is assigned to the IfSummaryRefreshOnly attribute supersedes the value that is assigned to PublishOnly. For more information, see

[Fields that Affect Publishing and Refreshing](#) later in this topic.

In the default mapping file, the **IfSummaryRefreshOnly** attribute is set to true for the Original Estimate, Remaining Work, and Completed Work fields.

The following example shows how to map the work item type field that contains the title to the task name Microsoft Project field:

Copy Code

```
<Mapping WorkItemTrackingFieldName="System.Title"
  ProjectField="pjTaskName"/>
```

The following XML syntax shows a more complex example of how to map the work item type field that has the start date to the start date Microsoft Project field. Because the field is calculated, it is specified to publish and not refresh in order to prevent errors from being introduced by Team Explorer updates.

Copy

```
<Mapping WorkItemTrackingFieldName="Microsoft.VSTS.Schedul:
```

Note

As a best practice, you should map calculated fields in Microsoft Project to read-only fields in Team Foundation. This helps avoid confusion so that team members do not try to change calculated fields.

ReservedField Element

The ReservedField element works to support hierarchical links defined between tasks and is an optional element.

If this element is not specified in the mapping, then Microsoft Project uses the default pjNumber20 field.

SyncField Element

The synchronization field enables you to control the publish and refresh behavior of each task. The field displays as a column with the title Publish and Refresh when you use the Team System Task Sheet view.

You must specify a synchronization field in the Microsoft Project field mapping file. Use the following XML syntax to specify which field is the synchronization field. To specify a synchronization field, use the SyncField element. The ProjectField attribute must be set to a valid Office Project field.

Copy Code

```
<SyncField ProjectField="" />
```

The following example shows how to specify pjTaskText25 as the synchronization field:

Copy Code

```
<SyncField ProjectField="pjTaskText25" />
```

For more information about how to use the synchronization field in Office Project, see [Publish or refresh work items in Project \[redirected\]](#).

ResourceNameSeparator Element

You can use the ResourceNameSeparator to define the character that will distinguish resource names that are in a string. Team Foundation users and resources are synchronized with the users of the Active Directory directory service. The names for users and resources may include a delimiter, such as a comma, to separate the last name, first name, and middle initial of a resource.

This is an optional element. If this element is not specified in the mapping, then Microsoft Project separates resource names by using the default mappings that are defined in the following table:

If Active Directory and the team project list separator is	Then in Office Project replace with
,	;
;	:
.	:

If you want to specify a different character than those listed in the above table, then you can define the ResourceNameSeparator by using the following syntax:

Copy Code

```
<ResourceNameSeparator WorkItemTrackingCharacter="" ProjectCharacter
```

The following example shows how to specify the character "*" as the separator to use in Office Project when the Team Foundation character is "-":

Copy Code

```
<ResourceNameSeparator WorkItemTrackingCharacter="-" ProjectCharacter
```

Hierarchy Link Type

When you create summary tasks in Microsoft Project, a tree link is created between the summary task, the parent, and the subordinate or child tasks. Office Project uses the default `System.LinkTypes.Hierarchy` to create these links.

Dependency Link Type

When you create links between tasks in Microsoft Project, you create a dependent link between the tasks. The predecessor task is assigned a Predecessor link and the successor task is assigned a Successor link. These are the default designations that are defined for the System.LinkTypes.Dependency link type.

Mapping Attributes that Affect Publishing and Refreshing

The following mapping fields and Microsoft Project field values determine whether a value for a work item is published or refreshed:

- The value of the Publish and Refresh value for each work item.
- The value of the **PublishOnly** attribute for a specific mapping field.
- The value of the **IfSummaryRefreshOnly** for a specific mapping field.
- The classification of the task as a summary or parent task.

A parent task is a task that has at least one child task that is published to Team Foundation Server.

For work items that are not summary tasks

The following table indicates whether a work item that is not a summary or parent task is published or refreshed based on the mapping field attributes and the assignment that is made to the Publish and Refresh value of the item.

Publish and Refresh (task level)	PublishOnly Attribute	Field is Published?	Field is Refreshed?
No	True or False	No	No
Refresh	True or False	No	Yes

Only				
Yes	True	Yes	No	
Yes	False	Yes	Yes	

For work items that are summary tasks

If the summary task criteria are met and **IfSummaryRefreshOnly** is true, its value supersedes the **PublishOnly** value. The following table indicates whether a summary or parent task is published or refreshed based on the assignment that is made to its Publish and Refresh value and the mapping field attributes.

Publish and Refresh (task level)	PublishOnly Attribute	IfSummaryRefreshOnly Attribute	Field is Published?	Field is Refreshed?
No	True or False	True or False	No	No
Refresh Only	True	True	No	Yes
Refresh Only	True	False	No	Yes
Refresh Only	False	True or False	No	Yes

Yes	True or False	True	No	Yes
Yes	True	False	Yes	No
Yes	False	False	Yes	Yes

See Also

Concepts

[Link type element reference](#)

Add or change how Project fields map to TFS fields



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Field mappings in Microsoft Project

[See Also](#) [Send Feedback](#)

You can customize how data is mapped between Microsoft Project and Team Foundation Server (TFS) by modifying the Microsoft Project Field Mapping File. The mapping file associates the reference name of a work item tracking field with a Project field. Before you change the Microsoft Project field mappings, you should review the information provided later in this topic to understand how TFS copies data from Project to the work item tracking database.

Important

The supported set of Project field names may be different for different versions of Microsoft Project.

Calculated Fields

Generally, you do not want to refresh calculated fields from the work item database. For example, start and finish dates are typically calculated based on task duration and dependencies. If a team member were to change the start date of a task in Team Explorer, the change could introduce errors in Office Project when the field is refreshed. Therefore you can configure calculated fields so that they are publish only, meaning that Office Project will publish changes for those fields to the work item database, but not update them. For more information about how to configure fields as publish only, see

[Upload or download the Microsoft Project Mapping file \[TFSFieldMapping\]](#).

You can also prevent team members from changing calculated field values in Team Explorer by using the **READONLY** attribute in the work item form definition. For more information about the **READONLY** attribute, see [All FIELD XML elements reference](#).

Note

The Microsoft Solutions Framework (MSF) for Agile Software Development and MSF for CMMI Process Improvement process templates already configure the start and finish dates as publish only mappings, and read-only in Team Explorer.

Mapping Field Types

When you publish tasks from Project to TFS, Team Foundation copies the data to the work item database. Also, when tasks are refreshed, the field data is copied from the work item database to your Project plan. For the field data to be copied correctly, the field types in TFS and Project must be compatible. The following table lists which Project field types are compatible with TFS field types:

TFS Work Item Field Type	Microsoft Project Field Type
DateTime	Datetime.
Double	Work, units, percentages, fractions, and any field type with a range of $\pm 5.0 \times 10^{-324}$ to $\pm 1.7 \times 10^{308}$.
History	Not Supported.
Html	Not Supported.
Integer	Work, units, numbers, and any field type with a range of -2,147,483,648 to 2,147,483,647.

PlainText Not Supported.

Text (maximum of 255 characters.)

⚠ Important

Office Project does not populate the Resource Names field list with names of team members.

Therefore, you must manually add names to the list. When you assign a work item to a resource in Office Project, you should specify the resource by its display name from Active Directory Domain Services (AD DS) or the Address Book. If you assign a work item to a resource by alias or other shortened form of the name, you introduce an inconsistency that can cause validation errors.

String

TreePath Custom outline codes.

For more information about work item field types, see

[Define and modify work item fields](#). For more information about Project field types, see the help about the pjField constant in the [Project 2013 developer documentation](#).

Project Field Names

In Project you can map any field name that starts with pjtask to a field in TFS. For example, you can map pjTaskText10 to a TFS String field. These field names are used in the ProjectField attribute when you specify which TFS work item field maps to a Project field. For a complete list of the Microsoft Project field names, see the help about the pjField constant in the Microsoft Office 2003 Software Development Kit.

Note

If you are using Project Server, there are additional fields that have "Enterprise" in the name, such as pjTaskEnterpriseCost1. TFS does not publish or refresh data to Project Server, unless you integrate it as described in Synchronize Team Foundation Server with Project Server. Therefore, these field names are not supported with the Team Foundation add-in to Project.

See Also

Reference

[Upload or download the Microsoft Project Mapping file \[TFSFieldMapping\]](#)

Concepts

Create your backlog and tasks using Project
Add or change how Project fields map to TFS fields



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Resolve errors received when configuring features

[See Also](#) [Send Feedback](#)

You may be able to resolve errors and warnings that the Configure Features wizard displays. These problems occur because definitions in the team project conflict with definitions in the process templates uploaded to your team project collection. You should change the process template to resolve the error and rerun the wizard. Or, you can change your team project and then rerun the wizard.

In this topic

- [Resolve errors reported by the Configure Features wizard](#)
 - [Resolve errors by modifying a process template](#)
 - [Resolve errors by modifying a team project or team project collection](#)
 - [Issues and resolutions](#)
- [Resolve warnings reported by the Configure Features wizard](#)
 - [Resolve warnings about adding an ACTION statement to a work item type](#)
 - [Add the Storyboard links control](#)

Note

If you encounter problems while performing the following procedures, you might find solutions on the forums for Team Foundation Server (TFS), at the following pages on the Microsoft website: [Work Item Tracking](#) and [Process Templates](#).

Required permissions

- To download and upload process templates, you must be a member of the Project Collection Administrators group. If security permissions are set explicitly, your Manage process template permission for the team project collection must be set to *Allow*.
- To run the **witadmin** command-line tool, you must be a member of one of the following groups: Team Foundation Administrators, Project Collection Administrators, or Project Administrators for the team project.

For more information, see [Permission reference for Team Foundation Server](#).

■ **Resolve errors reported by the Configure Features wizard**

You can resolve an error by modifying the process template used to configure the new features, or by modifying your team project. After you've corrected the error, rerun the wizard.

■ **Resolve errors by modifying a process template**

You modify a process template by performing these steps:

1. Download the process template from Team Foundation Server. See [Download the latest version of the process templates](#).
2. Modify a definition file for a work item type, categories, or process configuration. See [Customize work tracking objects to support your team's processes](#)
3. Upload the process template to Team Foundation Server. See [Upload, download, and delete process templates for a team project collection](#).

See also [Update a customized process template to access new features](#).

■ **Resolve errors by modifying your team project or team project collection**

To resolve an error or warning, you may choose to modify the team project or team project collection using the **witadmin** command. See

[witAdmin: Customize and manage objects for tracking work](#).

You can export a type definition, make changes to field definitions, and then import the type definition as needed. See [Import, export, and manage work item types \[witadmin\]](#).

Issues and resolutions

To resolve the errors listed in the following table, take the corresponding corrective actions, either to the process template or the team project, and then re-run the wizard.

Error	Issue	Process template resolution	Team project resolution
TF400613: The work item type '{1}' specified in category '{0}' does not exist.	A required work item type is missing from your team project because it was renamed, was removed, or was not added. Either rename the specified work item type definition, or import the missing work item	Change the work item type defined in the specified category to specify an existing work item type.	To rename a work item type, use witadmin renamewitd. To add a missing work item type, locate it in the latest process template, and import it using witadmin importwitd.

type from See
the latest
version of
the
process
templates
installed
with the
TFS
upgrade.

[Import, export, and manage work item types \[witadmin\]](#).

TF400614: Category '{0}' does not exist.

A required category is missing from the categories definition file in the process template that was selected to update your team project. Add the missing category.

Add the missing category to the process template. See [Use categories to group work item types](#).

Add the missing category to the team project using witadmin importcategories. See [Import and export categories \[witadmin\]](#)

TF400617: The type of field '{0}' in work item type '{1}' conflicts with the type of the existing field.

The data type of the field defined in the work item type being added does not match the data type defined in the team project collection. Correct the assignment in the work item type definition and then rerun the wizard.

Note

The upgrade Visual Studio Team Foundation Server 2012 changes the Description field (System.Description) from a field type of plain text to HTML, in order to support text formatting and insertion of images and hyperlinks. In the latest version of Team Foundation Server, you can switch the data type between PlainText and HTML.

Download the process template, open the type definition, locate the FIELD assignment, modify the type attribute to match that defined for the collection, and then, upload the process template. See [FIELD \(Definition\) element reference](#),

☑**Tip**

To determine the type assigned to a field, run witadmin listsfields.

Use **witadmin changefield** and specify the type attribute. See [Manage work item fields \[witadmin\]](#).

☑**Note**

You can change the type definition for the team project collection only when the type is PlainText or HTML.

TF400618: The reporting type of field '{0}' in work item type '{1}' conflicts with the reporting type of the existing field.

A reporting attribute assigned to a field in a work item type definition does not match the attribute defined in the team project collection.

Download the process template, open the type definition, locate the FIELD assignment and modify the reportable attribute to match that defined for the collection. Then, upload the process template. See [FIELD \(Definition\) element reference](#),

Use witadmin changefield and specify the reportingtype attribute. See [Manage work item fields \[witadmin\]](#).

TF400619: The SyncNameChanges of field '{0}' in work item type '{1}' conflicts with the SyncNameChanges of the existing field.

The syncnamechanges attribute assigned to a field in a work item type definition does not match the attribute defined in the team project collection. This attribute

specifies whether to update a person name field when that name changes in Active Directory.

Download the process template, open the type definition, locate the FIELD assignment, modify it to match the definition in the collection, and then, upload the process template. See [FIELD \(Definition\) element reference](#).

Use **witadmin changefield** command and specify the /syncnamechanges parameter. See [Manage work item fields \[witadmin\]](#).

TF400620: The friendly name of field '{0}' in work item type '{1}' conflicts with the friendly name of an existing field.

The friendly name assigned to a field in a work item type definition must match that defined in the team project collection.

Download the process template, open the type definition, locate the FIELD assignment, and modify it to match that defined for the collection. Then, upload the process template.

To change the friendly name for the team project collection, use **witadmin changefield** command and specify the /name parameter. See [Manage work item fields \[witadmin\]](#).

TF400621: The reference name of field '{0}' in work item type '{1}' is already in use by a link type.

Each reference name assigned to fields and link types must be unique within a team project collection.

If the link type is active and in use linking work items, download the process template, open the type definition, locate the FIELD assignment, and modify the refname assignment to match that defined for the collection. Then, upload the process template.

Tip

To determine if a link type is in use, create a direct links query, and then filter for all work items that are linked to another work item using that link type.

If the link type is not active nor in use linking work items within the team project collection, delete it. You can use **witadmin deletelinktype** to delete a link type. See [Manage link types \[witadmin\]](#).

TF400624: The friendly name of category '{0}' conflicts with category '{1}'.

You must assign a unique reference name and friendly name to each category of a team project. You can resolve this error by renaming the existing category '{1}' either in the process template or in your team project.

Download the process template, modify the categories file to rename the category, and then upload the process template. See [Use categories to group work item types](#).

Export the category file using **witadmin exportcategories**, modify the friendly name assigned to category '{1}', and import the file. See [Import and export categories \[witadmin\]](#).

TF400654: Unable to configure Planning Tools.

A mapping defined in the CommonConfiguration file specifies a field or state that does not exist in the work item type defined for the team project.

Download the process template, modify the CommonConfiguration file to correct the mapping, and then upload the process template. See [Customizing Process Configuration](#)

Export the work item type using **witadmin exportwitd**, add the missing field or state, and then import the work item type. See [Import, export, and manage work item types \[witadmin\]](#).

To learn more, see the following topics:

- [Overview of process template files](#)
- [Define and modify work item fields](#)
- Change the workflow for a work item type
- Add or modify work item fields to support reporting

Resolve warnings reported by the Configure Features wizard

To resolve the warnings listed in the following table, follow the resolution steps, and then re-run the wizard.

Warning	Issue	Resolution
TF400609: Cannot add the action '{0}' to the work item type '{1}' because the state '{2}' does not exist.	A required STATE is missing.	See

[Resolve errors adding an ACTION statement to a work item type.](#)

TF400610: Cannot add the action '{0}' to the work item type '{1}' because the transition from state '{2}' to '{3}' does not exist.

A required TRANSITION statement is missing.

See [Resolve errors adding an ACTION statement to a work item type.](#)

Storyboarding: Unable to insert tab to work item type '{1}' in category '{2}'.

The TabGroup element is missing in the type definition.

See [Add the Storyboard links control.](#)

Resolve warnings about adding an ACTION statement to a work item type

To support the My Work feature, the Configure Features wizard inserts two ACTION statements into the workflow transition section of the definition for task work item types. If either the states or the transitions for moving from a new to an active state, or from an active to a new state are missing, you will receive a warning.

To resolve the warnings, review the workflow for the indicated work item type. As needed, modify the workflow to match the workflow states and transitions that are defined for the latest version of the process template installed on the upgraded TFS that matches the one used to create your team project. Then, rerun the wizard. For team projects based on the Agile process template, see

[Update the workflow for agile team projects.](#)

To learn more, see the following topics:

- Export and import work item types [redirected]
- Change the workflow for a work item type
- Download the latest version of the process templates
- Support bug update status using My Work

■ **Add the Storyboard links control tab**

With the addition of Storyboarding, you can now link work items to storyboards. To do this, you add the Storyboard links control tab to the definition of the work item type. If the Configure Features wizard didn't or wasn't able to add the tab, then you can add it manually. In the latest versions of the process templates installed on the upgraded TFS, the **Storyboard** links control tab appears on all backlog work item types, such as Product Backlog, User Story, and Requirement.

📌 **Note**

The Configure Features wizard cannot add the Storyboard links control tab if the TabGroup element is missing from the work item FORM section. The following procedure adds just the Storyboard tab. If you want to add the

standard set of tabs for your backlog work item type, see the type definition in the latest version of the process template for your team project. See Download the latest version of the process templates.

1. To run the **witadmin** command-line tool, open a Command Prompt window where either Visual Studio or Team Explorer is installed and enter:

Copy Code

```
cd %programfiles(x86)%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 32-bit edition of Windows, replace **%programfiles(x86)%** with **%programfiles%**.

2. Export the type definition file for the backlog item by substituting your data for the arguments shown:

Copy Code

```
witadmin exportwitd /collection:CollectionURL /p:"ProjectName"
```

Where:

CollectionURL specifies the URL of the team project collection.
ProjectName specifies the name of your team project defined in the process template.
TypeName specifies the name of your backlog item, for example, BacklogItem.

Use the following format for CollectionURL: http://ServerName:Port/CollectionURL
For example: http://srvalm:8080/tfs/DefaultCollection.

3. Open the file using a text editor, such as Notepad.
4. Add this code snippet just before the `</Layout>` end-tag of your backlog type:

Copy Code

```
<TabGroup>  
<Tab Label="Storyboards">  
  <Control Name="StoryboardsControl" Type="LinksControl">  
    <LinksControlOptions>
```

```
<WorkItemLinkFilters FilterType="excludeAll" />
<ExternalLinkFilters FilterType="include">
  <Filter LinkType="Storyboard" />
</ExternalLinkFilters>
<LinkColumns>
  <LinkColumn RefName="System.Title" />
  <LinkColumn LinkAttribute="System.Links.Comment" />
</LinkColumns>
</LinksControlOptions>
</Control>
</Tab>
</TabGroup>
```

5. Save and close the file.
6. Import the type definition file by typing this command, substituting your data for the arguments that are shown:

 **Copy Code**

```
witadmin importwitd /collection:CollectionURL /p:"ProjectName" ,
```

7. Verify that the tab shows up in the backlog item.

See Also

Concepts

[Configure features after a TFS upgrade](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Update a team project based on an MSF v4.2 process template.

[See Also](#) [Send Feedback](#)

If you have upgraded from Visual Studio Team System 2008 Team Foundation Server to Team Foundation Server 2013, you can update your team project manually. If your team project was based on a Microsoft Solutions Framework (MSF) version 4.2 process template, follow the procedures in this topic. After you apply these updates, you'll be able to access the new features described in [Configure features after a TFS upgrade](#) as well as interface with Microsoft Test Manager.

Important

You only have to follow the procedures in this topic if you are upgrading a team project that you created with a process template provided with Visual Studio Team System 2008 Team Foundation Server, or one that does not contain the work item types Test Cases and Shared Steps.

These procedures will only support access to new features available with Team Foundation Server 2012. Additional work is required to add new queries or the latest reports, update customized reports, or access dashboards. For more information, see [Additional information about changes made when upgrading TFS](#).

Update tasks required to access new features:

- 1.

[Rename system fields](#)

- [\(Agile only\) Rename Scenario to User Story](#)

- [Download the latest version of MSF process template](#)
- [Import link types](#)
- [\(Optional\) Apply as needed customizations](#)
- [Import work item types](#)
- [Import the category file](#)
- [Import the process configuration files](#)
- [Verify access to new features](#)

Additional tasks required to interface with Microsoft Test Manager:

1. [Specify the bug type to be created in Microsoft Test Manager](#)
2. [Grant permissions to test team members](#)
3. [Launch Microsoft Test Manager](#)

Requirements

- To download a process template, you must be a member of the Project Collection Administrators group. If the required security permissions are set explicitly, your Manage process template permission for the team project collection must be set to Allow.
- To run the **witadmin** and **tcm** command-line tools, you must be a member of one of the following groups: Team Foundation Administrators, Project Collection Administrators, or Project Administrators for the team project.
- To grant permissions, you must be a member of the administrative group at the level of the group that you want to change. For example, if you want to change permissions for a group or user at the team project collection level, you must be a member of the Project Collection Administrators group for that collection, or your Edit Collection-Level Information permission must be set to Allow.

For more information, see [Permission reference for Team Foundation](#)

[Server.](#)

1. Rename system fields

Because the friendly names of several system fields were renamed in Visual Studio Team Foundation Server 2010, you need to manually rename these fields in your team project collection. System fields that were renamed include System.AreaID, System.IterationID, System.HyperLinkCount, System.ExternalLinkCount, and System.AttachedFileCount.

Perform this task for each team project collection defined on your upgraded Team Foundation Server.

1. Open a Command Prompt window where either Visual Studio 2012 or Team Explorer 2012 is installed and type:

Copy Code

```
cd %programfiles%\Microsoft Visual Studio 12.0\Common7\IDE
```

On a 64-bit edition of Windows, replace **%programfiles%** with **%programfiles(x86)%**.

2. Type each of the following commands, substituting your data for the arguments that are shown, and then choose the ENTER key.

Copy Code

```
witadmin changefield /collection:CollectionURL /n:System.AreaId  
witadmin changefield /collection:CollectionURL /n:System.AttachedFileCount  
witadmin changefield /collection:CollectionURL /n:System.ExternalLinkCount  
witadmin changefield /collection:CollectionURL /n:System.HyperLinkCount  
witadmin changefield /collection:CollectionURL /n:System.RelatedFieldId
```

Use this format for CollectionURL:

http://ServerName:Port/VirtualDirectoryName/CollectionName, for example: **http://srvalm:8080/tfs/DefaultCollection**.

[Back to top](#)

2. (Agile only) Rename the Scenario work item type

To minimize the amount of customizations you need to make, and to comply with future updates to the Agile process template, you should rename the Scenario work item type to User Story.

Note

Of course, renaming the Scenario work item type will require you to update existing reports and queries that reference the Scenario work item type. However, because of the schema changes made to data warehouse with the upgrade to Team Foundation Server 2010, the pre-existing or pre-upgrade reports need to be rewritten to work with the new schema. See

[Locating Reports After the Upgrade to Team Foundation Server 2010.](#)

Perform this task for each team project that you want to update.

- Type the following command, substituting your data for the arguments that are shown, and then choose the ENTER key.

Copy Code

```
witadmin renamewitd /collection:CollectionURL /p:projectName /n
```

Tip

Enclose a parameter in quotes when it contains spaces. For example, specify /p:"My Project X" when your project name contains spaces.

[Back to top](#)

3. Download the latest version of MSF process template

See [Download the latest version of the process templates](#).

Tip

To get access to the latest versions of the default process templates, install the latest quarterly update for Team Foundation Server. Significant updates were made to the workflow for several work item types in the latest quarterly update. These changes support backward transitions so that when you inadvertently drag a work item on the Kanban board or the task board to a resolved or closed state, you can drag it back to an earlier workflow state.

You can obtain the upgrade from the Microsoft download site:

[Quarterly Update for Microsoft Visual Studio Team Foundation Server 2012.](#)

[Back to top](#)

4. Import link types

Import the link types, SharedSteps and TestedBy, located in the LinkTypes folder in the process template that you downloaded in task 3.

Perform this task for each team project collection defined on your upgraded Team Foundation Server.

- Type the following two commands, substituting your data for the arguments that are shown, and then choose the ENTER key.

Copy Code

```
witadmin importlinktype /collection:CollectionURL /f:"Directory\  
witadmin importlinktype /collection:CollectionURL /f:"Directory\  
"
```

For DirectoryPath, specify the location of the LinkTypes folder for the process template that you downloaded. The directory path should follow this structure: Drive:\MSFTemplateFolder\WorkItem Tracking\LinkTypes.

[Back to top](#)

5. (Optional) Apply customizations to the latest versions of work item types

If you have customized any of the following work item types, you should update the latest version of these types with your customizations. The following tables summarize the fields removed and added in the latest versions of each process template.

Agile work item types

Work item type	Removed fields	
Bug	<ul style="list-style-type: none"> Issue (Microsoft.VSTS.Common.Issue) Rank (Microsoft.VSTS.Common.Rank), replaced with Stack Rank Test Name (Microsoft.VSTS.Test.TestName) Test Id (Microsoft.VSTS.Test.TestId) Test Path (Microsoft.VSTS.Test.TestPath) Triage (Microsoft.VSTS.Common.Triage) 	<ul style="list-style-type: none"> Re (M) Se Sta (M) Sy (M)
	<ul style="list-style-type: none"> Baseline Work (Microsoft.VSTS.Scheduling.BaselineWork), replaced with Original Estimate 	
	<ul style="list-style-type: none"> Discipline (Microsoft.VSTS.Common.Discipline), replaced with Activity 	<ul style="list-style-type: none"> Ac
	<ul style="list-style-type: none"> Exit Criteria (Microsoft.VSTS.Common.ExitCriteria) 	<ul style="list-style-type: none"> Or (M)
Task		

User Story (previously named Scenario)	• Issue (Microsoft.VSTS.Common.Issue)	• Sta (M
	• Rank (Microsoft.VSTS.Common.Rank), replaced with Stack Rank	
	• Task Hierarchy (Microsoft.VSTS.Scheduling.TaskHierarchy)	
	• Exit Criteria (Microsoft.VSTS.Common.ExitCriteria)	• Ri
	• Issue (Microsoft.VSTS.Common.Issue)	• Sta (M
	• Rough Order of Magnitude (Microsoft.VSTS.Common.RoughOrderOfMagnitude), replaced with Story Points	• Stc (M

CMMI work item types

Work item type	Removed fields	
Bug	• Baseline Work (Microsoft.VSTS.Scheduling.BaselineWork), replaced with Original Estimate	
	• Estimate (Microsoft.VSTS.CMMI.Estimate)	• Original I (Microsof
	• Issue (Microsoft.VSTS.Common.Issue)	• Severity (
	• Rank (Microsoft.VSTS.Common.Rank), replaced with Stack Rank	• Stack Ran (Microsof
	• Steps to Reproduce (Microsoft.VSTS.CMMI.StepsToReproduce), replaced with Repro Steps	• System Ir (Microsof
	• Test Name (Microsoft.VSTS.Test.TestName)	• Repro ste (Microsof

Task	<ul style="list-style-type: none"> • Test Id (Microsoft.VSTS.Test.TestId) • Test Path (Microsoft.VSTS.Test.TestPath) • Baseline Work (Microsoft.VSTS.Scheduling.BaselineWork), replaced with Original Estimate • Estimate (Microsoft.VSTS.CMMI.Estimate) • Exit Criteria (Microsoft.VSTS.Common.ExitCriteria) • Issue (Microsoft.VSTS.Common.Issue) • Rank (Microsoft.VSTS.Common.Rank), replaced with Stack Rank • Task Hierarchy (Microsoft.VSTS.Scheduling.TaskHierarchy) • Test Name (Microsoft.VSTS.Test.TestName) • Test Id (Microsoft.VSTS.Test.TestId) • Test Path (Microsoft.VSTS.Test.TestPath) • Baseline Work (Microsoft.VSTS.Scheduling.BaselineWork), replaced with Original Estimate • Completed Work (Microsoft.VSTS.Scheduling.CompletedWork) • Estimate (Microsoft.VSTS.CMMI.Estimate), replaced with Scheduling Size 	<ul style="list-style-type: none"> • Original I (Microsoi) • Stack Ra (Microsoi)
Requirement	<ul style="list-style-type: none"> • Exit Criteria 	<ul style="list-style-type: none"> • Original I (Microsoi) • Stack Ra (Microsoi)

- (Microsoft.VSTS.Common.ExitCriteria)
- Issue (Microsoft.VSTS.Common.Issue)
- Rank (Microsoft.VSTS.Common.Rank), replaced with Stack Rank
- Remaining Work (Microsoft.VSTS.Scheduling.RemainingWork)
- Scheduling (Microsoft.VSTS.Scheduling)

The types of customizations you might apply include field additions, additions or changes to pick lists, or additions to workflow reasons. Do not change the workflow states as these are used in process configuration and the Agile planning tools. If you must change the workflow, change it after you have finished the update and follow the guidance about metastate mappings provided here: [Configure and customize Agile planning tools for a team project.](#)

If you use other work item types defined in the process template, and want to update them to the latest versions, then apply any customizations you have made for them as well. Also, if you have defined a custom work item type that you use to track test cases, you should apply customizations from that type to the Test Case work item type provided with the latest process template.

To learn more about working with the artifacts that these process templates provide, see the following topics:

- [MSF for Agile software development for Visual Studio ALM \(v6.0\)](#)
- [MSF for CMMI process improvement for Visual Studio ALM \(v6.0\)](#)

[Back to top](#)

6. Import work item types

Import the following work item types based on the process template you are working with.

- Agile: Bug, Task, User Story, Test Case, Shared Steps, Code Review Request, Code Review Response, Feedback Request, Feedback Response
- CMMI: Bug, Task, Requirement, Test Case, Shared Steps, Code Review Request, Code Review Response, Feedback Request, Feedback Response

Perform this task for each team project that you want to update.

- Type the following command for each work item type that you need to import, substituting your data for the arguments that are shown, and then choose the ENTER key.

Copy Code

```
witadmin importwitd /collection:CollectionURL /p:projectName /f
```

Tip

Specify the name of the xml file and not the friendly name of the work item type. For example, specify CodeReviewRequest.xml for the Code Review Request work item type.

For DirectoryPath, specify the directory location of the TypeDefinitions folder for the process template that you downloaded. The directory path should follow this structure: Drive:\MSFTemplateFolder\ WorkItem Tracking\TypeDefinitions.

- (Optional) Verify the work item types are accessible by opening Team Explorer or Team Web Access. You might have to refresh the cache to see the changes.

[Back to top](#)

7. Import the categories file

Import the category file located in the WorkItem Tracking folder of the process template that you downloaded. Categories support intelligent grouping of work item types. To learn more, see [Use categories to group work item types](#).

- In the Command Prompt window, type the following command, substituting your data for the arguments that are shown, and then choose the ENTER key.

 **Copy Code**

```
witadmin importcategories /collection:CollectionURL /p:projectN:
```

For DirectoryPath, specify the path to the WorkItem Tracking folder for the process template that you downloaded. The directory path should follow this structure: Drive:\MSFTemplateFolder\WorkItem Tracking.

[Back to top](#)

8. Import the process configuration file

The process configuration file determines the layout and features available through the backlog and board pages of Team Web Access. To use these pages, you must import the process configuration file.

- Import the definition file for process configuration.

 Copy (

```
witadmin importprocessconfig /collection:CollectionURL /p:" Pro:
```

For DirectoryPath, specify the path to the Process folder for the process template that you downloaded. The directory path should follow this structure: Drive:\TemplateFolder\WorkItem Tracking\Process.

[Back to top](#)

9. Verify access to new features

Perform the tasks provided in New features enabled for Team Web Access.

Note

You will not have to perform the additional steps to update the workflow for Agile team projects as described here:

[Update the workflow for agile team projects](#). By following the procedures in this topic, you will have applied these changes already.

[Back to top](#)

Additional tasks to interface with Microsoft Test Manager

Perform the following tasks to complete the updates required to interface with Test Manager.

1. Specify the bug type to be created in Microsoft Test Manager

To support the automatic creation of a work item to track code defects or bugs that are found when a test team member uses Test Manager, you must specify the bug type to be used for your existing team project. The **tcm bugfieldmapping** command supports the import and export of a mapping file to the team project. The mapping file defines the type of work item to create and the three data fields to be filled by Test Manager. The three fields are reproducible steps, system information, and the build in which the defect was found. When a tester runs a test and finds a defect, they can create a bug in which the three fields are automatically filled.

1. Open Notepad or a text editor, and copy the following code into the file:

Copy Code

```
<?xml version="1.0" encoding="utf-16"?  
<BugFilerMappings workitemtypetocreate="Bug">  
  <ReproSteps>Microsoft.VSTS.TCM.ReproSteps</ReproSteps>  
  <SystemInformation>Microsoft.VSTS.TCM.SystemInfo</SystemInfo>  
  <BuildFoundIn>Microsoft.VSTS.Build.FoundIn</BuildFoundIn>  
</BugFilerMappings>
```

Note

If the work item type that you use to create code defects is labeled something other than "Bug," replace "Bug" in the previous example with the name of that work item type.

2. Save the file, and label it bugfieldmappings.xml.

3. In the Command Prompt window, type the following command, substituting your data for the arguments that are shown, and then choose the ENTER key.



```
tcm bugfieldmapping /import /mappingfile:"DirectoryPath\bugfield
```

For DirectoryPath, specify the folder where you saved the bugfieldmappings.xml file.

For more information, see

[Customize and manage the test experience \[tcm and Microsoft Test Manager\]](#).

[Back to top](#)

2. Grant permissions to test team members

You must grant permissions to team members who will manage test environments and test configurations, create and view test runs, and perform other tasks.

The following table describes the permissions that control access to test functions and support interfacing with the team project for testing. It also indicates the default assignments that are made in version 5.0 of the MSF process templates, in addition to the recommended permissions to grant manual testers and test leads.

Permission	Description	Scope	Readers	Contributors	Builders	Recommended for n tes
View project-	Can view membership of project-level groups	Project-				

level information	and the permissions of those members.	level	✓	✓	✓	✓
View test runs	Can view test plans in this node.	Project-level	✓	✓	✓	✓
Create test runs	Can add and remove test results and add or modify test runs for the team project.	Project-level		✓	✓	✓
Manage test configurations	Can create and delete test configurations for the team project.	Project-level		✓	✓	
Manage test environments	Can create and delete test environments for the team project.	Project-level		✓	✓	
Delete test runs	Can delete a scheduled test for the team project.	Project-level		✓	✓	

View this node	Can view the security settings for an area node.	Area node	✓	✓	✓	
Manage test plans	Can create and edit test plans that are assigned to an area node. If test plans have not been run, you can also delete them.	Area node		✓	✓	✓
Manage test controllers	Can register and unregister test controllers for the team project collection.	Project collection				

You can grant permissions by following the procedures that are indicated for the specific scope area:

- You can set project-level permissions or area node permissions from the administration page of Team Web Access. See [Managing Permissions and Add and modify area and iteration paths](#).
- You can set project collection permissions from Team Explorer by choosing Team, Team Project Collection Settings, Security, by opening and using the administration console for Team Foundation, or by using the **TFSSecurity** and **tf** command-line tools. For more information, see [Collection-Level](#)

Groups.

For more information, see [Change Permissions for a Group or User](#).

[Back to top](#)

3. Launch Microsoft Test Manager

After you have completed the upgrade tasks that are described earlier in this topic, you can start Microsoft Test Manager, connect to your project, and start to plan your test efforts. For more information, see [Testing the application](#).

[Back to top](#)

Additional information about changes made when upgrading TFS

When you upgrade from Visual Studio Team System 2008 Team Foundation Server to TFS 2012, you receive updates made to both TFS 2010 and TFS 2012. There were a number of architectural changes made with the release of TFS 2010. To learn more about the changes made by upgrading to the latest version of TFS from Visual Studio Team System 2008 Team Foundation Server, see the following resources:

- [Team Foundation Server 2010 Key Concepts](#) (blog post)
- [Updating an Upgraded Team Project to Access New Features](#) (VS ALM 2010 article)
- [Locating Reports After the Upgrade to Team Foundation Server 2010](#) (VS ALM 2010 article)
- [Changes and Additions to the Schema for the Analysis Services Cube](#) (VS ALM 2010 article)
- Changes made to team projects and default process templates during upgrade of Team Foundation Server (VS ALM 2012 article)

See Also

Concepts

Configure features after a TFS upgrade

Other Resources

[witAdmin: Customize and manage objects for tracking work](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

PreEmptive Analytics

[Send Feedback](#)

PreEmptive Analytics provides a set of tools that helps provide insights into application adoptions, user behavior, and software quality.

For more information about these tools and how to use them, see the [PreEmptive Solutions website](#).



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Enable the TCP/IP Protocol for a Database Instance

[See Also](#) [Send Feedback](#)

In a multi-server deployment of Team Foundation Server, you must enable the TCP/IP protocol for the Team Foundation database. By enabling this protocol, you gain support for remote connections such as the connection between the application-tier server and the data-tier server for Team Foundation. If the TCP/IP protocol is not enabled for both the database instance and the SQL Server Native Client, the following errors might appear:

- Error 29108. Team Foundation report server configuration: cannot connect to the Team Foundation data-tier when trying to configure the report server database.
- TF31001: Team Foundation cannot retrieve the list of team projects from Team Foundation Server. The Team Foundation Server returned the following error: The request failed with HTTP status 503: TF30059: Fatal error while initializing Web service.

Required Permissions

To perform these procedures, you must be a member of the sysadmin security group for SQL Server on the data-tier server for Team Foundation.

To enable the TCP/IP protocol for the database instance

1. Log on to the data-tier server on which the database instance is defined.
2. Open the Start menu, point to All Programs, point to Microsoft SQL Server 2005 or Microsoft SQL Server 2008, point to Configuration Tools, and then click SQL Server Configuration Manager.

3. In the tree pane, click SQL Server 2005 Services or SQL Server 2008 Services.
4. In the results pane, verify that, under the State column, Running appears next to the name of each service.

If Stopped appears, right-click the name of the service, and click Start.

5. In the tree pane, click SQL Server 2005 Network Configuration or SQL Server 2008 Network Configuration to expand it, and then click Protocols for MSSQLServer/MyInstanceName.

If you specified the default instance during installation, the instance name will be MSSQLSERVER.

6. In the results pane, verify that, under the Status column, Enabled appears next to the name of the TCP/IP protocol.

If Disabled appears, right-click TCP/IP, and then click Enable.

7. In the tree pane, click SQL Native Client Configuration to expand it, and then click Client Protocols.

8. In the results pane, verify that, under the Status column, Enabled appears next to the name of the TCP/IP protocol.

If Disabled appears, right-click TCP/IP, and then click Enable.

9. In the tree pane, click SQL Server 2005 Services or SQL Server 2008 Services.

10. In the results pane, right-click SQL Server (MSSQLServer/MyInstanceName), and then click Restart.

See Also

Concepts

Team Foundation Server Databases

[Configure and manage TFS resources](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Team Foundation Server concepts

[See Also](#) [Send Feedback](#)

To deploy and manage Visual Studio Team Foundation Server effectively, you must understand how it works and communicates with other components of Team Foundation. As an administrator for Team Foundation Server, you should be familiar with Windows authentication, network protocols and traffic, and the structure of the business network on which Team Foundation Server is installed. You should also have an understanding of groups and permissions in Team Foundation Server. You might also find an understanding of SQL Server, SQL Server Reporting Services, and SharePoint Products useful as you manage Team Foundation Server.

Understanding components and terms

You will be better able to plan, deploy, and manage Team Foundation Server if you understand the following components and terms:

- Application tier, data tier, and client tier: The logical tiers that compose Team Foundation Server. These tiers might all be deployed on the same physical computer, or they might be installed across multiple computers. For more information, see [Team Foundation Server architecture](#).
- Team project collection: The primary organizational unit for all data in Team Foundation Server. Collections can include one or more team projects. For more information, see [Manage team project collections](#).
- Team project: A central point for your team to share team activities that are required to develop a specific software technology or product. Team projects are organized in team project collections. For more information, see [Track work with Visual Studio ALM and TFS](#).
- Team Foundation Server Administration Console: The centralized management tool for TFS administrators to configure and manage resources. For more information, see [Configure and manage TFS resources](#).
- Service accounts: The account or accounts that the Web services and applications in Team Foundation use. Team Foundation Server requires service accounts to perform operations across servers and Web services. These service accounts have specific requirements. For more information, see [Service accounts and dependencies in Team Foundation Server](#).
- SharePoint Products: Software that provides support for team project portals and dashboards. You can include one or more SharePoint Web applications as part of your deployment of Team Foundation Server. To include one of these applications, you must install and configure Team Foundation Server extensions for SharePoint Products, and you must configure permissions across the deployment. For more information, see [SharePoint Products and Team Foundation Server](#).

Share information using the project portal.

- SQL Server and SQL Server Reporting Services: Software that provides a database platform for data warehousing and a business intelligence platform for data integration, analysis, and reporting solutions. TFS stores its data in SQL Server databases. You can also optionally include a server that is running SQL Server Reporting Services and that automatically generates reports for team projects. For more information, see [Manage TFS reports, data warehouse, and analysis services cube](#).

Understanding Team Foundation Server Security

To optimize the security of Team Foundation Server, you should understand the following concepts:

- [Topology](#), which includes where and how servers that are running components of Team Foundation are deployed, the network traffic that passes between Team Foundation Server and clients of Team Foundation, and the services that must run on Team Foundation Server.
- [Authentication](#), which includes the determination of the validity of users, groups, and services in Team Foundation Server.
- [Authorization](#), which includes the determination of whether valid users, groups, and services in Team Foundation Server have the appropriate permissions to perform specific actions.

You should also consider the other components and services on which Team Foundation Server depends.

When you consider security for Team Foundation Server, you must understand the difference between authentication and authorization. Authentication is the verification of the credentials of a connection attempt from a client, server, or process. Authorization is the verification that the identity that is trying to connect has permissions to access the object or method. Authorization occurs only after successful authentication. If a connection is not authenticated, it fails before any authorization checking is performed. If authentication of a connection succeeds, a specific action might still be disallowed because the user or group was not authorized to perform that action.

Topologies, Ports, and Services

The first element of deployment and security for Team Foundation Server is whether the components of your deployment can connect to one another to communicate. Your goal is to enable connections between clients of Team

Foundation and Team Foundation Server and to limit or prevent other connection attempts.

Team Foundation Server depends on certain ports and services so that it can function. You can secure and monitor these ports to help meet business security needs. You must permit network traffic for Team Foundation Server to pass between clients of Team Foundation, the servers that host the logical components of the application tier and the data tier for Team Foundation, computers for Team Foundation Build, and remote clients that are using Team Foundation Server Proxy. By default, Team Foundation Server is configured to use HTTP for its Web services. For a full list of ports and services that Team Foundation Server uses and how they are used within its architecture, see

[Team Foundation Server architecture](#).

You can deploy Team Foundation Server in an Active Directory domain or in a workgroup. Active Directory provides more built-in security features than workgroups provide. You can use Active Directory features to help secure your deployment of Team Foundation Server. For example, you can configure Active Directory to prevent duplicate computer names so that a malicious user cannot spoof the computer name with a rogue server that is running Team Foundation Server. To mitigate the same kind of threat in a workgroup, you must configure computer certificates.

Whether you deploy Team Foundation Server in a workgroup or a domain, you must comply with certain constraints imposed by the requirements of Team Foundation Server itself. For more information about topologies for Team Foundation Server, see [A Simple Team Foundation Server Topology](#), [A Moderate Team Foundation Server Topology](#), [A Complex Team Foundation Server Topology](#), [Understanding Windows SharePoint Services](#), and [Understanding SQL Server and SQL Server Reporting Services](#).

■ Authentication

Security for Team Foundation Server is integrated with and relies on Windows integrated authentication and the security features of the Windows operating system. You can use Windows integrated authentication to authenticate accounts for connections between Team Foundation clients and TFS, for Web services on the servers that host the logical application and data tiers, and for connections

between application-tier and data-tier servers themselves.

Note

You can configure TFS to support Kerberos for mutual authentication of both the client and the server after you install TFS.

You should not configure any SQL Server database connections between Team Foundation Server and SharePoint Products to use SQL Server Authentication because it is not as secure as Windows authentication. When you connect to the database, the user name and the password for the database administrator account are sent in an unencrypted format. Windows integrated authentication does not send the user name and password. Instead, it uses Windows integrated authentication security protocols to transfer service account identity information that is associated with the host Internet Information Services (IIS) application pool to SQL Server.

Authorization

Team Foundation Server authorization is based on users and groups in Team Foundation, the permissions that are assigned directly to both those users and groups, and permissions that those users and groups might inherit by belonging to other groups in Team Foundation Server. Users and groups in Team Foundation can be local users or groups, Active Directory users and groups, or both.

Team Foundation Server is preconfigured with default groups at the server, collection, and project level. You can populate these groups by adding individual users. However, you might find management easier if you populate these groups by using Active Directory security groups. By taking this approach, you can manage group membership and permissions more efficiently across multiple computers or applications, such as SharePoint Products and SQL Server.

Your specific deployment might require that you configure users, groups, and permissions on multiple computers and within several applications. For example, you must configure permissions for users and groups in Reporting Services, SharePoint Products, and Team Foundation Server if you want to include reports and project portals as part of your deployment. In Team Foundation Server, you

can set permissions for each project, for each collection, and across a deployment (at the server level). Additionally, certain permissions are granted by default to any user or group that you add to Team Foundation Server, as that user or group is automatically added to Team Foundation Valid Users. For more information, see

[Manage users or groups in TFS.](#)

Besides configuring permissions for authorization in Team Foundation Server, you might need authorization within version control and work items. You manage these permissions separately at a command prompt, but they are integrated as part of the interface for Team Explorer.

See Also

Tasks

[Add SharePoint products to your deployment](#)

Concepts

[Team Foundation Server architecture](#)

[Team Foundation Server architecture](#)

[Manage users or groups in TFS](#)

Other Resources

[SharePoint Products and Technologies and Team Foundation Server](#)



- Visual Basic C#
- Visual C++
- F#
- HLSL
- JScript

Configure TFS using the administration console [redirected]

[Send Feedback](#)

This topic has been merged with [Configure and manage TFS resources](#).
