Visual Studio .NET Academic Assignment Manager Source Package

Visual Studio .NET Academic Assignment Manager Source Package

Visual Studio .NET Academic Assignment Manager provides a way for you to create and manage course information and assignments and enter and track grades, all from within the Visual Studio .NET integrated development environment (IDE). Visual Studio .NET Academic Assignment Manager Source Package allows you to customize the Assignment Manager server and client.

With the source files, you can customize Assignment Manager to:

- Integrate with other academic products
- Include automatic cheat detection
- Integrate with the NT user authentication used by your school
- Coordinate the user lists with the student list used by the registrar's office

In This Documentation Set

Installing Assignment Manager Source Package
: Find out about the hardware and software requirements for installing and modifying Assignment Manager Source Package. This section also provides an explanation of the various files that compose the source package.

Customizing Assignment Manager Source Package
: Find out about how to modify the source files for Assignment Manager and deploy the modified version.

Visual Studio .NET Academic Assignment Manager Faculty Client

Copyright

are to specification level 2.0. This product is not approved or endorsed by Sun Microsystems, Inc.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Visual Studio .NET Academic Assignment Manager Source Package

Installing Assignment Manager Source Package

After you have downloaded Visual Studio .NET Academic Assignment Manager Source Package from the MSDN Academic Alliance Web site, you can install it on a development machine. Be sure to check the hardware and software requirements for installing and using the Assignment Manager Source Package. For more information, see Assignment Manager Source Package System Requirements.

To install Assignment Manager Source Package

1. Double-click AssignmentManagerSource.exe.

   A message box appears, with information about the source package.

2. In the message box, click **OK**.
3. In the next dialog, choose the location where you want the files copied to or accept the default location and click **Unzip**.

   When the files have been copied, the readme file is displayed.

Next, create a virtual directory for the Web application portion of Assignment Manager. You need to have Internet Information Services (IIS) installed to use and customize Assignment Manager. For more information, see Assignment Manager Source Package System Requirements.

To create a virtual directory for Assignment Manager

1. On the **Start** menu, choose **Run**.
2. Enter `inetmgr` and then click **OK**.
3. In the **Internet Information Services (IIS) Manager**, expand <*machine name*> **(local computer)** and then expand **Web Sites**.
4. Right-click **Default Web Site**, select **New**, and then select **Virtual Directory**.

The **Virtual Directory Creation Wizard** appears.

5. Click **Next**.
6. In the **Alias** text box, type AMWebDev and then click **Next**.

Note Assignment Manager Source Package cannot be installed on a machine that already has a Web application that uses the alias "amweb".

1. In the **Directory** box, enter the path to the location where \...\Assignment Manager Server\Web\AMWeb\ was installed to, or click **Browse** to locate the path and then click **Next**. By default, this location is c:\ AssignmentManagerSource\Assignment Manager Server\Web\AMWeb.
2. Choose the level of access permissions for the virtual directory and then click **Next**.
3. Click **Finish**.

Next, install an Assignment Manager deployment dialog.

To install the deployment dialog

1. In the installation directory for the source package, copy the file VsdCustomTextPwd.wid. By default, this location is c:\AssignmentManagerSource\Assignment Manager Server\setup\dialogs\VsdCustomTextPwd.wid.
2. Browse to \...\Microsoft Visual Studio .NET 2003\Common7\Tools\Deployment\VsdDialogs\1033 and paste VsdCustomTextPwd.wid in the directory. This path depends on where you chose to install Visual Studio .NET 2003.

After you have successfully installed the source package, you can then begin modifying the code. For more information, see Customizing Assignment Manager Source Package.

Visual Studio .NET Academic Assignment Manager Source and Visual Studio .NET 2002 Compatibility

Visual Studio .NET Academic Assignment Manager Source cannot be modified in Visual Studio .NET 2002, nor can modified versions of Visual Studio .NET Academic Assignment Manager be accessed by machines with Visual Studio .NET 2002 installed. Assignment Manager Source is compatible only with Visual Studio .NET 2003.

See Also

Assignment Manager Source Package System Requirements | Source Package Components

---

Visual Studio .NET Academic Assignment Manager Source Package

Assignment Manager Source Package System Requirements

This topic includes hardware requirements and software requirements for installing and modifying Visual Studio .NET Academic Assignment Manager Source.

Hardware Requirements

Requirement	Enterprise Architect
Processor	PC with a Pentium II-class processor, 450 MHz *Recommended*: Pentium III-class, 600MHz
Available Hard Disk Space	1.5 MB on the installation drive
Operating System	Windows® 2000, Windows XP, or Windows Server 2003[1], [2]

[1]Windows XP Home does not support local Web application development; local Web application development is only supported in the Professional or Server versions of Windows.

[2]Microsoft Windows 2000 Datacenter Server is not a supported operating system.

Software Requirements

To edit the Assignment Manager source files, you need to have certain software installed.

Software	Use
Internet Information Services 5.0 or 6.0	Needed to modify the server portion of Assignment Manager.
ASP.NET version 2003	Needed to modify and run the server portion of Assignment Manager.
Visual Studio .NET 2003	Recommended to modify the Source Package. For information on requirements, see "Visual Studio .NET Hardware Requirements" in the MSDN Library.

See Also

Installing Assignment Manager Source Package

Visual Studio .NET Academic Assignment Manager Source Package

Customizing Assignment Manager Source Package

The source package includes all of the files you need to modify Assignment Manager. Two solution files, AssignmentManager.sln for the server portion of Assignment Manager and AMClients.sln for the student and faculty user interface, include the projects required to modify and rebuild Assignment Manager. These projects and solutions have been coded using C#.

With the source files, you can customize Assignment Manager to:

- Integrate with other academic products
- Include automatic cheat detection
- Integrate with the NT user authentication used by your school
- Coordinate the user lists with the student list used by the registrar office

In This Section

Source Package Components
: Provides a listing of some of the files provided with the source package and their uses.

Editing Source Package Solutions
: Provides the steps for opening source package solutions in Visual Studio .NET.

Deploying a Custom Assignment Manager
: Provides the steps for deploying a modified version of Assignment Manager.

Localizing Assignment Manager
: Provides the steps for changing the English text strings to the language of your choice.

Related Sections

Installing Assignment Manager Source Package

Visual Studio .NET Academic Assignment Manager Source Package

Source Package Components

Assignment Manager Source Package contains all of the source code for the Assignment Manager Server tool. When you download and install the source package, you have the pieces you need to customize the Assignment Manager server and clients.

- **AssignmentManager.sln** Installed at c:\ AssignmentManagerSource\Assignment Manager Server\ by default.
- **AMClients.sln** Installed at c:\ AssignmentManagerSource\Assignment Manager Clients\ by default.

AssignmentManager.sln

AssignmentManager.sln includes four project files: ActionService, AMInstall, AMWeb, and AMSetup. You can edit these files to customize the server portion of Assignment Manager.

ActionService.csproj

This project contains the Assignment Manager Service that monitors MSMQ and processes Auto Build and Auto Check requests. In addition, this project includes code for a timer for the service to check for notifications.

Some of the files related to monitoring MSMQ are listed below.

ServiceInstaller.cs
: The file contains code that runs during Assignment Manager setup to install the service on the client machines and start the service.

ServerAction.cs
: This file contains the event that runs when a message appears in the MSMQ. In addition, this file handles parsing the message and firing the correct process-either Auto Build or Auto Check.

IActionService.cs
: This file contains the interface for the different processes that the server can run, Auto Build and Auto Check.

AutoBuild.cs
: This file contains the code for the Auto Build process that builds the student's project.

AutoCheck.cs
: This file contains the code for the Auto Check process that runs the student's executable against the expected output provided by the faculty and compares the results.

Some of the files related to the service timer are listed below.

RequestListener.cs
: This file is the primary file for starting both functions of the ActionService. This file creates the MSMQ when started and then starts the process listening to it. In addition, the file starts the timer and sends out

notifications.

AMWeb.csproj

This project contains the .aspx pages and code that display the Assignment Manager Server Web interface. In addition, this project contains code for all the actions a user can do, including data input, as well as Auto Build and Auto Check interactions with MSMQ.

AMVersion.xml Contains the version information of the Assignment Manager Server. Clients reference this to tell if they are connecting to the correct version of Assignment Manager.

AssemblyInfo.cs Contains assembly information for the amweb.dll assembly that is built.

Error.aspx A page to display errors. For some errors, the user is redirected to this page.

Login.aspx The Assignment Manager log on page. This file handles changing the user's password when they first log in.

logoff.aspx Logs user out of Assignment Manager.

web.config Contains Web server settings for the Assignment Manager virtual directory.

AssignmentManager folder

The files in this folder contain the background code that handles database interactions and interactions with MSMQ.

AssignmentList.cs Represents the data structure used for storing a list of assignments. This class is used often when displaying a list of assignments to users, such as on assignments.aspx

AssignmentM.cs Represents an assignment. This class is the middle level between the Web pages and database. The class also contains functions for adding, modifying, and deleting assignments from the database.

Constants.cs Stores constants for using the middle-tier layer

(\AssignmentManager).

CourseM.cs Represents a course. Similar to AssignmentM.cs in that this class is the middle level between Web pages and the database, and therefore contains all the functions for manipulating the database.

DatabaseCall.cs Abstracts database connections. This file stores all of the code that is common across all database connections, making it possible to call a stored procedure in Assignment Manager by using just a few lines of code.

ImportUsers.cs Handles the commit method for committing an 'import users' transaction to the database.

LSAUtil.cs Accesses the Local Security Authority (LSA) where the database connection string is stored.

MessageM.cs Handles all of the work involved in sending SMTP messages to users.

RoleM.cs Handles database interactions for roles and permissions, including setting and modifying.

ServiceControl.cs Accesses the Windows Services on the machine to start or stop the Assignment Manager service. This class is used in settings.aspx.

StudentAssignmentM.cs Similar to AssignmentM.cs, but for student assignments and submissions. Handles database interaction and is used by the Web pages when displaying information.

UserList.cs Similar to AssignmentList. This class is a data structure used to store a list of users.

UserM.cs Handles users, including loading information from the database and storing information to the database. This class is accessed by the Web pages for that information.

Common folder

This folder contains classes used by multiple Web pages.

Constants.cs Contains constants used across Web pages.

Functions.cs Contains functions commonly used across Web pages.

Faculty folder

This folder contains the .aspx files and C# code files for the Faculty Web pages. These Web pages are the interface that users interact with.

AddCourse.aspx Contains no UI. This page is accessed via the Faculty Tools and adds a course, then redirects to a new page.

AddEditAssignment.aspx Modifies the properties on an assignment or adds a new assignment.

AddEditCourse.aspx Modifies the properties for a course.

AddEditUser.aspx Modifies the properties for a user or adds a new user.

AddResource.aspx Contains code for a small modal dialog box that displays when a user clicks the Add resource link on addeditcourse.aspx.

Assignments.aspx Displays a list of assignments to the user.

ChangePassword.aspx Changes a user's password.

ConfirmAssignmentDelete.aspx Contains code for a modal confirmation dialog box that displays when the user chooses to delete an assignment from the AddEditAssignment.aspx page.

ConfirmResourceDelete.aspx Contains code for a modal confirmation dialog box that displays when the user chooses to delete a resource from the AddEditcourse.aspx page.

ConfirmUserDelete.aspx Contains code for a modal confirmation dialog box that displays when the user chooses to delete a user from AddEditUser.aspx page.

DeleteCourse.aspx Contains no UI. This file is accessed via the Faculty Tools to delete a course, then redirects the user's browser to a different page.

GradeSubmissions.aspx Contains the UI for faculty to grade a user's submission.

ImportForm.aspx Contains the first page of import users, where the user is prompted for a file with user information in it.

ImportFormPreview.aspx Contains the second page of import users, where the user is asked to select which fields match to which columns.

Results.aspx Contains the final page of import users, where the user is shown the results of the import. For example, "20 users imported successfully."

Settings.aspx Contains code for Assignment Manager-wide settings, such as enable or disable the auto build service and project upload size.

submissions.aspx Lists user submissions.

UploadDownload.aspx Contains code for uploading and downloading a student submission.

users.aspx Lists users in the current course.

workwithcourse.aspx Contains no UI. Redirects to the correct page.

Images folder

This folder contains all of the images used in the Web pages.

Scripts folder

This folder contains client-side scripts used in the Web pages.

assnMan.css A cascading style sheet (.css) file that defines the look and feel of the Assignment Manager server Web site.

assnMan.js A file, used for faculty pages, containing code for the descriptions that appear when the user pauses the mouse (mouses over) on menu items.

assnManStudent.js A file, used for student pages, containing code for the descriptions that appear when the user pauses the mouse (mouses over) on menu items.

Download.js Script for downloading files that is used from UploadDownload.aspx.

Miscellaneous.js Additional JScript functions.

popUp.js A file used to handle the modal dialog pages, such as ConfirmResourceDelete.aspx.

Upload.js Handles uploads from UploadDownload.aspx.

Student folder

This folder contains the .aspx files and C# code files for the Student Web pages. These Web pages are the user interface that the user interacts with.

AddCourse.aspx Contains no UI. A Web page for adding a course.

AssignmentGrade.aspx Web page for viewing the grade of a submission and its associated details.

Assignments.aspx Web page that lists the assignments for the current course.

ChangePassword.aspx Web page for changing the current user's password.

CourseInfo.aspx Displays course information.

UploadDownload.aspx Handles uploads and downloads, using Upload.js and Download.js in the \scripts directory.

WorkWithCourse.aspx Contains no UI. Redirects the user to the correct page when connecting for the first time from the client tools.

UserControls folder

This folder contains custom controls that are used in many of the Web pages.

faculty.ascx The user control for the border of faculty pages.

student.ascx The user control for the border of student pages.

goBack.ascx The user control for the Back and Help links.

AMInstall.csproj

This project contains a custom action that runs during installation of Assignment Manager to setup the database, add the AssignmentManager user group, as well as setting file and folder permissions and configuring IIS for Assignment Manager.

Some of the files included in this project:

AMInstall.cs
This file contains the majority of the code that drives the custom action.

AMUser.cs
This file contains the code that creates the AssignmentManager user group.

SecurityPermissions.cs
This file contains code that defines file and folder permissions.

AMSetup.vdproj

A deployment project that builds a distributable .msi file.

AMClients.sln

AMClients.sln includes five project files, FacultyClient, StudentClient, FacultyClientSetup, StudentClientSetup, and AddinResources. You can edit these files to customize the user interface for the Faculty and Student clients.

FacultyClient.csproj

This project contains all of the code for the Faculty Client, which allows the faculty features to appear in Visual Studio .NET.

***Dialog.cs**
: These files contain code for Windows dialogs.

***Command.cs**
: These files contain code for the menu and command items that appear on the Visual Studio Tools menu, which are used to display the Windows dialogs.

Facultytools.cs
: This file contains code that handles the functionality for the Faculty tools.

StudentClient.csproj

This project contains all of the code for the Student Client, which allows the Student features to appear in Visual Studio .NET.

***Dialog.cs**
: These files contain code for Windows dialogs.

***Command.cs**
: These files contain code for the menu and command items that appear on the Visual Studio Tools menu, which are used to display the Windows dialogs.

Studenttools.cs
: This file contains code that handles the functionality of the Student Client, including loading courses, managing the Start Page tabs.

FacultyClientSetup.vdproj

This deployment project builds the .msi used to install the Faculty Client.

StudentClientSetup.vdproj

This deployment project builds the .msi used to install the Student Client.

AddinResources.vcproj

This project creates an assembly that stores resources used for the Add-in names and descriptions, which display in the Add-in Manager dialog box.

See Also

Customizing Assignment Manager Source Package

Visual Studio .NET Academic Assignment Manager Source Package

Editing Source Package Solutions

Assignment Manager Source Package includes two solution files, AssignmentManager.sln and AMClients.sln. AssignmentManager.sln is the solution you use to modify the server portion of Assignment Manager. AMClients.sln allows you to customize the faculty and student user interfaces. Both the server and client portions of Assignment manager are written using C#. Use Visual Studio .NET 2003 to edit these solution files.

To open source package solutions

1. Launch Visual Studio .NET 2003.
2. On the **File** menu, choose **Open Solution**.
3. Browse to the location where you saved the source package. By default, the solutions can be found at c:\AssignmentManagerSource\Assignment Manager Server\AssignmentManager.sln and c:\AssignmentManagerSource\Assignment Manager Clients\AMClients.sln.
4. Select the solution you intend to modify and then click **Open**.

Once you have edited files, you can then build and debug the customized code. For more information on debugging and building code, see the Visual Studio .NET documentation in the MSDN Library.

See Also

Customizing Assignment Manager Source Package | Deploying a Custom Assignment Manager

---

Visual Studio .NET Academic Assignment Manager Source Package

Deploying a Custom Assignment Manager

After you have modified and tested the changes you have made to Assignment Manager, you can then deploy this custom version of Assignment Manager.

To deploy the Assignment Manager Server

1. In Visual Studio .NET, open AssignmentManager.sln.
2. On the **Build** menu, choose **Build Solution**.

   The following file is created during the build process:

   AMSetup.msi located in \...\Assignment Manager Server\Setup\AMSetup\ <*buildtype*>\.

   For example, c:\Assignment Manager Server\Setup\AMSetup\Debug\AMSetup.msi.

3. The file AMSetup.msi is located by default in \...\Setup\AMSetup\ <buildtype>\, where <buildtype> refers to Debug, Release, or another custom build configuration. Copy this file to the machines where you want to install Assignment Manager.

Users who do not need to have the Assignment Manager server components installed should only install client pieces, as appropriate.

To deploy the Assignment Manager Clients

1. In Visual Studio .NET, open AMClients.sln.
2. On the **Build** menu, choose **Build Solution**.

   The following files are created during the build process:

   For Assignment Manager Faculty Client \...\Assignment Manager Clients\FacultyClientSetup\<*buildtype*>\AMFacultyClient.msi.

For Assignment Manager Student Client \...\Assignment Manager Clients\StudentClientSetup\<*buildtype*>\AMStudentClient.msi.

Note <*buildtype*> refers to Debug, Release, or another custom build configuration.

3. Copy the .msi files to the machines on which you want to install.

See Also

Customizing Assignment Manager Source Package

Visual Studio .NET Academic Assignment Manager Source Package

Localizing Assignment Manager

As part of customizing Assignment Manager, you can localize the user interface text for Assignment Manager Server, Assignment Manager Faculty Client, and Assignment Manager Student Client. You can also localize the user interface text that is displayed in setup.

To localize the text for Assignment Manager Server

1. Open AssignmentManagerStrings.txt, located in *n*:\...\ Assignment Manager Server\Resources\.
2. Change the entries in the file to their translated counterparts.

   For example, change the English BtnCancel=&Cancel to the Spanish BtnCancel=&Cancelar.

3. Run am.bat, also located in *n*:\...\ Assignment Manager Server\Resources\.
4. In Visual Studio .NET, open Assignment Manager.sln.
5. On the **Build** menu, choose **Build Solution**.

Once you have localized the user interface text for the server component, you can then proceed with localizing the text for the two client components.

To localize the text for the Faculty and Student Clients

1. Open AMClientResources.txt, located in *n*:\...\ Assignment Manager Clients\Resources\.
2. Change the entries in the file to their translated counterparts.

   For example, change the English BtnCancel=&Cancel to the Spanish BtnCancel=&Cancelar.

3. Run bldResources.bat, also located in *n*:\...\ Assignment Manager Clients\Resources\.
4. In Visual Studio, open AMClients.sln.
5. On the **View** menu, choose **Resource View**.
6. Expand the nodes until **String Table** is visible.

7. Double-click **String Table** to open the file and then change the entries to their translated counterparts.
8. On the **Build** menu, choose **Build Solution**.

Now you can localize the text for the setup UI.

To localize the text for the installation user interface

1. In Visual Studio .NET, open the deployment package you intend to localize: AMsetup.vdproj, FacultyClientSetup.vdproj, or StudentClientSetup.vdproj.
2. In Solution Explorer, select the project.
3. In the **Properties** window, change the **Localization** value to the language of your choice.
4. In **Solution Explorer**, right-click on the project name and choose **View** and then choose **User Interface**.
5. Select a dialog in the editor window and then open the Properties window to translate the text for that dialog.
6. When you have finished editing the text for each dialog, rebuild the project.
7. Repeat steps 1-6 for each deployment package.

See Also

Customizing Assignment Manager Source Package
