After building and testing a Microsoft® SQL ServerTM 2000 application, you must distribute it to customers. You must also be able to distribute the SQL Server components required by your application. There are two options for how you distribute SQL Server components with your application:

- Distribute the SQL Server 2000 relational database engine and client components with your application.
- Distribute only the SQL Server 2000 client components if your customers will already have an instance of SQL Server to which they can connect from your application.

The licensing terms controlling the redistribution of SQL Server components are defined in the file Redist.txt, which is located on your SQL Server compact disc.

Distributing the SQL Server 2000 Desktop Engine

The SQL Server 2000 Desktop Engine is a redistributable version of the relational database engine in SQL Server 2000. It allows an application that uses the SQL Server relational database engine to install the engine as a part of the application setup process. The Desktop Engine is designed so that an application can use it to store data without requiring any database administration from the end user. The Desktop Engine is designed to manage its configuration and resource usage dynamically, minimizing the requirement for administration of the engine after it has been installed. The Desktop Engine does not include SQL Server utilities or tools that have graphical user interfaces. The application setup is coded to install the engine. After the Desktop Engine has been installed, either the application setup or the application use the standard SQL Server APIs (SQL-DMO, Transact-SQL, and so on) to create and configure the database, and the application uses the SQL Server APIs to perform any needed administration.

The Desktop Engine Setup uses the Windows Installer service that ships with Microsoft Windows® 2000 and is available on Windows NT® 4.0, Windows 98, and Windows 95. For more information about Windows Installer, see the MSDN

Library at <u>Microsoft Web site</u>.

There are several methods an application setup program can use for installing the SQL Server Desktop Engine:

- For applications that use the Windows Installer technology in their setup programs, the SQL Server 2000 Desktop Engine is supplied as a set of Windows Installer merge modules. You can use these merge modules to integrate the SQL Server 2000 Desktop Engine Setup into the application's Windows Installer setup. For more information, see Merging the Desktop Engine into Windows Installer.
- For applications that do not use Windows Installer-based setup programs, the SQL Server 2000 Desktop Engine provides a Setup.exe file. This Setup.exe program operates as a shell that calls Windows Installer to install the SQL Server 2000 Desktop Engine merge modules. For more information, see <u>SQL Server 2000 Desktop Engine Setup</u>.
- Applications that do not use Windows Installer-based setup programs can also call Windows Installer directly. For more information, see Executing Windows Installer Directly.

SQL Server 2000 Desktop Engine Setup installs the SQL Server client components, so you do not have to distribute the client components separately if you install SQL Server 2000 Desktop Engine.

Before attempting to install an instance of the Desktop Engine, an application setup process should first ensure that the proposed instance name has not already been used by another instance of SQL Server, and the computer does not already have more than 16 instances (or 15 named instances) installed. SQL Server 2000 supplies two API functions to perform these checks. For more information, see Desktop Engine Installation API.

Distributing the SQL Server Client Components

If your customers already have an installed instance of SQL Server to which they can connect, your application setup only has to install the SQL Server client components. This only needs to be done if your application setup detects that

client components are not installed on the client:

- If your application requires only the relational database client connectivity components needed to support ADO, OLE DB, or ODBC, SQL Server 2000 supplies a SQL Server-specific Microsoft Data Access Components (MDAC) redistribution file named Sqlredis.exe. This file installs the OLE DB and ODBC core components, the Microsoft OLE DB Provider for SQL Server, the SQL Server ODBC driver, and the default SQL Server client Net-Libraries. The license terms that govern the use of the redistribution file are in the file Redist.txt on the SQL Server 2000 compact disc. For more information about running Sqlredis.exe, see sqlredis Software Distribution Executable.
- If your application uses other SQL Server client components, such as PivotTable® Service or SQL-DMO, your application setup must also install the files needed by these additional components after installing the MDAC redistributable files. The list of which files must be copied over and which files must be registered are in the file Redist.txt on the SQL Server 2000 compact disc. The components covered by Redist.txt are:
 - Distributed Component Object Model 95 (DCOM95)
 - PivotTable Service
 - Decision Support Objects (DSO)
 - English Query
 - Replication
 - Data Transformation Services (DTS)
 - Distributed Management Objects (DMO)

- SQL Namespace (SQL NS)
- DB-Library

SQL Server 2000 Desktop Engine and Windows Installer

The primary way to install the SQL Server 2000 Desktop Engine is to use the Windows Installer service in Microsoft® Windows® 2000 and available for Windows NT® 4.0, Windows 95, and Windows 98. You can use the following methods to install the Desktop Engine using Windows Installer:

- Microsoft SQL Server[™] 2000 provides a set of Windows Installer merge modules that can be used to install an instance of the Desktop Engine. These can be integrated into a standard Windows Installer setup procedure.
- An application setup program that calls Windows Installer directly can install an instance of the Desktop Engine.

The Desktop Engine also produces a number of return codes that indicate the success or failure of an attempt to install an instance of the Desktop Engine using Windows Installer.

Merging the Desktop Engine into Windows Installer

The Microsoft® SQL Server™ 2000 Desktop Engine is available as a set of Windows Installer merge modules, which can be used by independent software vendors (ISVs) to install an instance of the SQL Server 2000 Desktop Engine during their own setup process. These merge modules can be merged into the ISV setup program using available Windows Installer setup development tools. The SQL Server 2000 Desktop Engine merge modules are located in the /Sqlmsde/Msm folder on the SQL Server 2000 compact disc. For more information about Windows Installer, see the MSDN Library at Microsoft Web site.

You can reduce the disk footprint of your application by customizing the SQL Server 2000 Desktop Engine Setup so that it does not install SQL Server 2000 components that are not used by your application. You can do this by removing the merge modules for the component from your application Windows Installer setup. You must include all of the SQL Server 2000 Desktop Engine merge modules in your application setup, except for the modules listed here.

Optional SQL Server 2000	Merge Modules for that
Component	Component
SQL Server database replication	Repl*.msm
components	
SQL Distributed Management	Dmo*.msm
Objects (SQL-DMO)	

You can remove the replication merge modules, the SQL-DMO merge modules, or both.

You can add entries in the [Options] section of your Windows Installer .ini file that customizes the installation of the instance of the Desktop Engine. These are the settings and their descriptions.

TARGETDIR="executable_folder_path"

Specifies the folder where the Desktop Engine executable files are installed. The default for *executable_folder_path* is C:\Program Files\Microsoft SQL Server.

DATADIR="data_folder_path"

Specifies the folder where the Desktop Engine system databases are built. The default is the same folder used for the Desktop Engine executable files.

INSTANCENAME="instance_name"

Specifies the name for the instance. If no instance name is specified, the instance is installed as a default instance.

COLLATION="collation_name"

Specifies the SQL Server collation that will be used as the default collation for this instance of the Desktop Engine. For information about a list of collation names, see <u>Windows Collation Name</u> and <u>SQL Collation Name</u>.

CALLBACK=Dllname!CallbackFunctionName

Specifies the name of the DLL containing the Desktop Engine Windows Installer callback function, and the name of the callback function. For more information, see <u>Desktop Engine Windows Installer Callback Functions</u>.

USEDEFAULTSAPWD

Specifies that setup assign the default password (NULL) to the **sa** login. You must specify either **USEDEFAULTSAPWD** or **SAPASSWORD**, but not both.

SAPASSWORD="sa_password"

Specifies that setup assign a password to the **sa** login. You must specify either **USEDEFAULTSAPWD** or **SAPASSWORD**, but not both.

Note Settings that contain spaces should be enclosed with quotation marks in the .ini file entries. For example:

TARGETDIR="C:\Program Files\Microsoft SQL Server\Instance1"

For more information about what values to specify for the .ini file settings, see <u>SQL Server 2000 Desktop Engine Setup</u>.

Executing Windows Installer Directly

An application setup program can call the Windows Installer executable file (Msiexec.exe) to install the Microsoft® SQL ServerTM 2000 Desktop Engine. This is typically done when the application setup program has a DLL available that contains a Windows Installer callback function that can process the return codes generated by the Desktop Engine Windows Installer merge modules.

When the application setup program calls Msiexec.exe, the application must provide the same set of Desktop Engine options as those that can be defined in the [Options] section of the Windows Installer .ini file. For information about a list of these options, see Merging the Desktop Engine into Windows Installer.

The Desktop Engine installation sample shows using the Win32® **CreateProcess** function to execute Msiexec.exe. The sample application provides a set of Windows Installer options, including a **CALLBACK** option, in the **CreateProcess szBuf** parameter. For more information about the sample, see <u>Desktop Engine Installation Sample</u>.

SQL Server 2000 Desktop Engine Setup

The Microsoft® SQL Server™ 2000 Desktop Engine includes a Setup.exe file that an application setup process can use to install the SQL Server 2000 Desktop Engine. You can define settings for customizing the way Setup.exe installs SQL Server 2000 Desktop Engine. These settings can be specified through either command prompt switches or an .ini file.

Syntax

```
setup [/?]
 [ /i package_file
     [ /settings ini_file ]
     [ [ TARGETDIR="executable_folder_path"]
         [ DATADIR="data_folder_path" ]
          [ INSTANCENAME="instance_name" ]
          [ COLLATION="collation_name" ]
          [ CALLBACK=Dllname!CallbackFunctionName ]
         [ USEDEFAULTSAPWD | SAPASSWORD="sa_password" ]
       1
   1
   | [ /x package_file ]
 ]
 [/L*v [filename]]
 [/qn|/qb]
```

Arguments

Displays a syntax summary of the **setup** switches. **setup** displays more switches than it accepts; use only the switches documented in this topic.

/i package_file

Specifies the name of the Windows Installer installation package file (an .msi file) to be used to install an instance of the SQL Server 2000 Desktop Engine. Place the .msi file in the same folder as setup.exe. If /i is not specified, copy all 16 of the .msi files from the \Sqlmsde\Setup folder on the SQL Server 2000 compact disc to the same folder as Setup.exe.

settings ini_file_name

Specifies the name of an .ini file containing settings for TARGETDIR, DATADIR, INSTANCENAME, COLLATION, USEDEFAULTSAPWD, and SAPASSWORD. If /settings is specified, these options should be set in the .ini file, not in the command prompt switches. Place the .ini file in the same folder as setup.exe.

TARGETDIR="executable_folder_path"

Specifies the folder where the Desktop Engine executable files are installed. The default for *executable_folder_path* is C:\Program Files\Microsoft SQL Server.

DATADIR="data_folder_path"

Specifies the folder where the SQL Server system databases are built. The default is the same folder used for the Desktop Engine executable files.

INSTANCENAME="instance_name"

Specifies the name for the instance. If no instance name is specified, the instance is installed as a default instance.

COLLATION="collation_name"

Specifies the SQL Server collation that will be used as the default collation for this instance of the Desktop Engine. For information about a list of collation names, see <u>Windows Collation Name</u> and <u>SQL Collation Name</u>.

CALLBACK=Dllname!CallbackFunctionName

Specifies the name of the DLL containing the Desktop Engine Windows Installer callback function, and the name of the callback function. For more information, see <u>Desktop Engine Windows Installer Callback Functions</u>.

USEDEFAULTSAPWD

Specifies that setup assign the default password (NULL) to the **sa** login.

SAPASSWORD="sa_password"

Specifies that setup assign a password to the **sa** login. When /**i** is specified, you must specify either /**USEDEFAULTSAPWD** or /**SAPASSWORD**, but not both.

/**x** package_name

Specifies the name of the Windows Installer installation package file (an .msi file) to use when uninstalling an instance of SQL Server 2000 Desktop Engine. You must specify the name of the same installation package file that was used to install the instance of the Desktop Engine. Place the .msi file in the same folder as setup.exe.

/**L*v** [filename]

Specifies that a verbose log be created. If *filename* is specified, the log is stored in the file specified.

/qn

Specifies that setup run with no user interface.

/qb

Specifies that setup show only the basic user interface. Only dialog boxes displaying progress information are displayed. Other dialog boxes, such as the dialog box that asks users if they want to restart at the end of the setup process, are not displayed.

Remarks

SQL Server 2000 Desktop Engine Setup is a tool that applications can use to install the SQL Server 2000 Desktop Engine as a part of the typical installation process. It is not intended as a tool for installing the database engine from the

SQL Server 2000 compact disc. To install SQL Server 2000 from a compact disc, use the autorun.exe component in the root folder of the SQL Server 2000 compact disc. For more information about installing SQL Server from the compact disc, see Overview of Installing SQL Server 2000.

Applications that do not use Windows Installer as their setup mechanism can use SQL Server 2000 Desktop Engine Setup. Applications that use Windows Installer should instead use the SQL Server 2000 Desktop Engine Merge Modules. For more information about using the merge modules, see <u>SQL Server 2000 Desktop Engine and Windows Installer</u>.

To run SQL Server 2000 Desktop Engine Setup, the application setup process must copy these files to the same folder:

- Setup.exe from the \Sqlmsde folder on the SQL Server 2000 compact disc.
- Sqlrun.cab from the \Sqlmsde\Setup folder on the SQL Server 2000 compact disc.
- Either all of the SqlrunXX.msi installation package files from the \Sqlmsde\Setup folder on the SQL Server 2000 compact disc, or a custom installation package file. For more information, see Managing Desktop Engine Installation Package Files.
- Optionally, an .ini file with settings for the TARGETDIR, DATADIR, INSTANCE NAME, COLLATION, USEDEFAULTSAPWD, and SAPASSWORD options.

The application then executes setup.exe at the point in the installation process where the Desktop Engine should be installed. The setup process of an application typically will use one of two mechanisms to execute the Desktop Engine Setup:

• Execute Setup.exe from a batch command file or from the command prompt. When using this method, setup switches are specified in the same way they would be specified for any other command prompt

utility.

Execute Setup.exe from within the application setup executable file
using a method similar to the Win32® CreateProcess() function. In this
case, setup switches are specified in a character string passed in using
the lpCommandLine paramter. For information about using
CreateProcess() to execute Setup.exe, see Desktop Engine Installation
Sample.

If neither /**qn** nor /**qb** is specified, Setup.exe displays all user interface dialog boxes.

Whenever an application needs to uninstall an instance of the Desktop Engine, it must call the SQL Server 2000 Desktop Engine Setup using the /x switch to specify the same .msi file that was used to install that instance. For example, to uninstall a named instance installed with a custom installation package file named MyCustom.msi, execute:

Setup /x MyCustom.msi

Settings Using an .ini File

Optionally, the TARGETDIR, DATADIR, INSTANCE NAME, COLLATION, USEDEFAULTSAPWD, and SAPASSWORD options can be specified using an .ini file. The first line in the .ini file must contain the string [Options]. The format of the .ini file is shown in this example:

[Options]

TARGETDIR="C:\Program Files\Microsoft SQL Server\Mssql\$MyIns DATADIR="C:\Program Files\Microsoft SQL Server\Mssql\$MyInstan INSTANCENAME=MyInstance

Note Settings that contain spaces should be enclosed with quotation marks.

Recommended Settings for TARGETDIR and DATADIR

Because your application is installing an instance of SQL Server 2000, it should typically specify the same default folders that SQL Server 2000 uses for instance

installations. You can specify different locations if you want to install the instance in a non-default location.

When installing a default instance, the recommended default settings are:

TARGETDIR="C:\Program Files\Microsoft SQL Server\Mssql\Binn" DATADIR="C:\Program Files\Microsoft SQL Server\Mssql\Data"

When installing a named instance, the recommended default settings are:

TARGETDIR="C:\Program Files\Microsoft SQL Server\Mssql\$MyIns DATADIR="C:\Program Files\Microsoft SQL Server\Mssql\$MyInstan INSTANCENAME="MyInstance"

Managing Desktop Engine Installation Package Files

The Microsoft® SQL Server™ 2000 Desktop Engine Setup uses two main files: a .cab file containing the files to be installed, and an installation package (.msi file) that defines the installation process. In addition, if an application setup program uses the Win32® **CreateProcess()** function to call the Windows Installer program (Msiexec.exe), the setup program must specify an installation package file.

All applications must use a different Desktop Engine installation package file for each instance of the SQL Server Desktop Engine installed on a computer. If one installation package file is used to install one instance, that installation package file cannot be used to subsequently install another instance of the Desktop Engine.

Here are the ways for an application to provide a Desktop Engine installation package file:

- Copy the 16 SqlrunXX.msi files from the \Sqlmsde\Setup folder on the SQL Server 2000 compact disc to the folder containing Setup.exe. There are only 16 .msi files on the compact disc because Microsoft supports a maximum of 16 SQL Server 2000 instances per computer.
- The application setup folder can contain a custom Desktop Engine installation package file.

If an application copies all 16 of the SqlrunXX.msi files into the folder with the Desktop Engine Setup and does not specify a /i parameter, the Desktop Engine Setup dynamically chooses one of the 16 .msi files that has not yet been used to install an instance. This is the recommended approach, and is the easiest to implement.

Each Desktop Engine installation package file contains globally unique identifiers (GUIDs) that uniquely identify the file. GUIDs are incorporated in the installation package file at the time the file is created. The Desktop Engine Setup uses the GUIDs to determine whether a particular installation package file has been already used to install an instance of the Desktop Engine on a computer.

This means that if a user gets two different copies of the same installation package file, they can use only one of them to install an instance of the Desktop Engine on a computer. For example, two different vendors could decide to minimize the size on their application's setup folders by using only the Sqlrun04.msi file from the SQL Server 2000 compact disc, and not copying all 16 of the installation package files supplied by SQL Server. Only one of the applications could install an instance of the Desktop Engine successfully. When the second application attempted to install an instance of the Desktop Engine, it would receive an error stating that the sqlrun04.msi had already been used to install an instance of the Desktop Engine. Applications that need to minimize the number of files they distribute with their application can eliminate the potential for such an error by creating a custom Desktop Engine installation package file.

Each time an application needs to uninstall an instance of the Desktop Engine, it must call the SQL Server 2000 Desktop Engine Setup using the /x switch to specify the same .msi file that was used to install that instance.

Creating a Custom Desktop Engine Installation Package File

Creating a custom Desktop Engine installation package file that will never conflict with an installation package distributed by another application requires using Microsoft Visual Studio®, and the Windows Installer Software Developers Kit (Windows Installer SDK). For more information about the Windows Installer SDK, see the MSDN Library at Microsoft Web site. SQL Server 2000 provides the following Desktop Engine installation package files to use as templates when creating a custom installation package file:

- Sample.msi. This installation package file supports only new installs, it does not support upgrading an instance of the Microsoft Data Engine (MSDE) version 1.0 to the SQL Server 2000 Desktop Engine.
- SampleUpg.msi. This installation package file can be used for both installing new instances of the SQL Server 2000 Desktop Engine and upgrading instances of MSDE 1.0 to SQL Server 2000 Desktop Engine.

The default location for both of these files is in the folder C:\Program Files\Microsoft SQL Server\80\Tools\Devtools\Samples\Desktop.

To generate a custom Desktop Engine installation package file

- 1. Copy the file Sample.msi or SampleUpg.msi to your folder as *YourName*.msi, where *YourName* is the name for your custom installation package file.
- 2. Using the GUIDGEN tool from Visual Studio, generate a GUID.
- 3. Using the ORCA tool from the Windows Installer SDK, edit *YourName*.msi. On the Summary Information menu, replace the current ProductID value with the GUID you generated in GUIDGEN.
- 4. Use GUIDGEN to generate a second GUID.
- 5. In ORCA, edit YourName.msi, go to the Property Table and replace the current ProductCode value with the new GUID. Save *YourName*.msi.

Desktop Engine Windows Installer Return Codes

The Microsoft® SQL Server™ 2000 Desktop Engine Windows Installer components can generate a set of integer return codes. All of the methods for installing the Desktop Engine will eventually use the Windows Installer service. Even the Desktop Engine Setup program uses the Windows Installer server. Any Desktop Engine Setup can therefore generate any of the standard Windows Installer return codes, or error codes. For more information about the Windows Installer error codes, see the MSDN Library at Microsoft Web site. If a Desktop Engine Setup process does not include a callback function, Windows Installer will return only one of two return codes. It returns 0 if the installation is successful, and 1603 if any error is encountered. You must include a callback function in your setup before Windows Installer will return any other return codes.

These return codes specific to the Desktop Engine are returned only if you have specified a Desktop Engine Windows Installer callback function.

Return		
Code	#define Label	Message Tex
50035	ERROR_CHARS_NOT_SUPPORTED	The text spec
		characters no
		the Windows
50043	ERROR_DUP_INSTANCE_NAME	An installatic
		same instance
		exists.
50047	ERROR_INVALID_INSTNAME_SYNTAX	Instance nam
		from 1-16 ch
		must follow t
		SQL Server r
		identifiers.
60001	ERROR_CONFIGURE_SERVER_FAILURE	ConfigServer
60002	ERROR_INSTALL_CATALOG_STP_FAILURE	InstallCatalo
60003	ERROR_INSTALL_DTC_FAILURE	InstallDTC is
60004	ERROR_INSTALL_PERFMON_FAILURE	InstallPerfmc

60005	ERROR_INSTALL_SQLAGENT_SECURITY_FAILURE	InstallSQLA _{
		failed.
60006	ERROR_INSTALL_SQLREDIS_FAILURE	InstallSQLRe
60007	ERROR_OVER_MAX_INSTANCES	The number (
		over the max
		PickInstance(
60009	ERROR_UPGRADE_FAILURE	UpgradeData

Desktop Engine Windows Installer Callback Functions

The Microsoft® SQL Server™ 2000 Desktop Engine Windows Installer functionality supports callback functions that an application setup program can use to tailor its response to the return codes raised by the Desktop Engine Windows Installer.

The setup program developer can code a Desktop Engine callback function as a function in a Microsoft Visual C++® DLL. The callback function is coded the same as a Windows Installer Custom Action Type 1 callback function, except that the Desktop Engine callback function must take an additional UINT parameter that will contain the Desktop Engine exit code, or return code:

UINT __stdcall MyCallbackFunction(MSIHANDLE hinstall, UINT uExitCode)

For more information about coding Windows Installer Custom Action Type 1 DLLs, see the MSDN Library at <u>Microsoft Web site</u>. For a list of the return codes specific to the Desktop Engine, see <u>Desktop Engine Windows Installer</u> Return Codes.

Use the Desktop Engine Windows Installer CALLBACK option to invoke the callback function during setup:

CALLBACK=Dllname!CallbackFunctionName

Specifies the name of the dll containing the Desktop Engine Windows Installer callback function, and the name of the callback function.

Desktop Engine Installation API

Before attempting to install an instance of the Microsoft® SQL Server™ 2000 Desktop Engine, an application setup process should first validate the proposed installation using the Desktop Engine Installation API functions:

- Use **NumInstalledInstances** to ensure the computer does not already have more than 16 instances installed.
- Use **IsInstanceNameValid** to ensure the name for the new instance of SQL Server has not already been used by an existing instance.

To use these functions in a setup program you are developing, you must first:

- Copy the file Validateinstance.h from C:\Program Files\Microsoft SQL Server\80\Tools\DevTools\Samples\Desktop to a folder in your compiler include path. Validateinstance.h contains the function prototypes for the Desktop Engine Setup API functions.
- Copy the files Desktopenginedeploy.lib and Sqdedev.lib from C:\Program Files\Microsoft SQL Server\80\Tools\DevTools\Samples\Desktop to a folder in your compiler library path.

If the application setup program requires logic to interpret the Desktop Engine return codes, also copy the file Sqlserrors.h from C:\Program Files\Microsoft SQL Server\80\Tools\DevTools\Samples\Desktop to a folder in your compiler include path. Sqlserror.h includes the #defines creating the symbolic names you can use to reference the Desktop Engine Windows Installer return codes.

NumInstalledInstances

Returns the number of instances of Microsoft® SQL ServerTM installed on a computer.

Syntax

int NumInstalledInstances ();

Arguments

The function takes no parameters.

Returns

The number of instances of SQL Server installed on the computer.

Remarks

Microsoft does not support more than 16 instances of SQL Server on a computer. Use **NumInstalledInstances** to check the number of existing instances of SQL Server on a computer before installing a new instance.

See Also

Desktop Engine Installation Sample

IsInstanceNameValid

Indicates if a proposed instance name has been used for an instance of Microsoft® SQL ServerTM already installed on the computer.

Syntax

BOOL IsInstanceNameValid (LPTSTR *lpszInstanceName*);

Arguments

LpszInstanceName

Is a pointer to a character string containing an instance name. For more information about the correct format of an instance name, see <u>Instance</u> Name.

Returns

TRUE if the instance name has not been used by any of the instances of SQL Server installed on the computer. FALSE if the instance name has already been used.

Remarks

IsInstanceNameValid cannot be used to check whether a default instance has already been installed. The Desktop Engine Setup will return ERROR_DUP_INSTANCE_NAME if you attempt to install a default instance when one has already been installed on the computer.

See Also

Desktop Engine Installation Sample

sqlredis Software Distribution Executable

sqlredis is a self-extracting executable file that can be executed by application setup programs that must install the Microsoft® SQL Server[™] client connectivity components required to support the OLE DB or ODBC APIs, or any other API built over these two, such as ADO, RDO, or DAO, or the Active Template Libraries (ATL).

Syntax

sqlredis [/?] [/C /T:"temporary_working_folder_path"]

Arguments

/?

Displays a syntax summary of the **sqlredis** switches. This version of **sqlredis** displays more switches than it accepts; use only the switches documented in this topic.

/C /T:"temporary_working_folder_path"

Specifies that **sqlredis** only extract and expand its files into the folder specified in *temporary_working_folder_path*, and not install them. If these switches are not specified, **sqlredis** installs its files and uses the default temp folder on the computer for the extraction process.

Remarks

sqlredis allows you to redistribute the SQL Server 2000 client connectivity components with an application that supports connections to SQL Server 2000, but assumes the customer has separately purchased and installed the SQL Server 2000 relational database engine. **sqlredis** is a self-extracting program that invokes an InstallShield setup; it can be executed anywhere an InstallShield setup program can be executed in the setup process of an application. **sqlredis** is a silent setup program; it displays no dialog boxes that require user input. **sqlredis** is located in the \x86\other folder on the SQL Server 2000 compact disc.

To install the SQL Server connectivity components, execute **sqlredis** using no parameters. Only specify the /C /T:"*temporary_working_folder_path*" switches if you want to see what modules are installed by **sqlredis**.

The SQL Server 2000 version of **sqlredis** accepts different parameters than the **sqlredis** distributed with SQL Server version 7.0 Service Pack 1 and Service Pack 2. Use only the parameters documented in this topic when using the SQL Server 2000 version of **sqlredis**.

Desktop Engine Installation Sample

ConsoleDesktopengine.exe demonstrates how to call either the Microsoft® SQL ServerTM 2000 Desktop Engine Setup or Windows Installer Setup to install an instance of the Desktop Engine.

Default Location

C:\Program Files\Microsoft SQL Server\80\Tools\Devtools\Samples\Desktop

To run the sample

- 1. Copy these files from the sample folder to a folder in your Microsoft Visual C++® include path: sqlserror.h, validateinstance.h.
- 2. Copy these files from the sample folder to a folder in your Visual C++ library path: sqldedev.lib, desktopenginedeploy.lib.
- 3. Using Visual C++ 6.0, open and run consoledesktopengine.dsw to create an executable.
- 4. Copy the contents of the \Sqlmsde\Setup folder on the SQL Server 2000 compact disc to the folder holding ConsoleDesktopengine.exe.
- 5. Optionally, generate a custom Desktop Engine installation package file in the folder containing ConsoleDesktopengine.exe. For more information about generating a custom installation package file, see Managing Desktop Engine Installation Package Files.
- 6. Run ConsoleDesktopengine.exe.

See Also

SQL Server 2000 Desktop Engine Setup