


# Phoneme Perception Test Installation Program

Command Line Parameters

## Table of contents

**1 INTRODUCTION ..... 2**

**2 PHONEME PERCEPTION TEST INSTALLER COMMAND LINE PARAMTERS ..... 2**

2.1 AVAILABLE PROPERTIES ..... 2
2.2 EXAMPLES ..... 5

**3 SETUP.EXE COMMAND-LINE PARAMETERS (INSTALLSHIELD HELP) ..... 5**

3.1 BUILT-IN COMMAND-LINE PARAMETERS ..... 5
3.1.1 Silent Installations ..... 7
3.1.2 Special Installation Modes ..... 9
3.1.3 Passing Data to the Installation ..... 11
3.1.4 Download and Cache Locations (Basic MSI and InstallScript MSI Projects) ..... 12
3.1.5 Debugging ..... 13
3.1.6 SMS Data ..... 14
3.1.7 Miscellaneous ..... 15
3.2 USER-DEFINED COMMAND-LINE PARAMETERS (INSTALLSCRIPT PROJECTS) ..... 10

**4 MSI COMMAND-LINE OPTIONS (INSTALLSHIELD HELP/ MICROSOFT) ..... 18**

## 1 INTRODUCTION

The Phoneme Perception Test installer (InstallShield project type: Basic MSI) basically consists of an MSI and a setup.exe. The latter is responsible for the check and installation of the prerequisite components which again can be found in a separate directory 'ISSetupPrerequisites'. Both, the MSI as well as the setup.exe, react to command line parameters. Therefore, when launching the installer via the setup.exe, one has to respect a special syntax in order to have the arguments for the MSI passed through to it (-> Chapter 3, /v : pass arguments to Msiexec).

The setup.exe command line parameters are described in chapter 3. The msi command line options are described in chapter 4. In addition, Phoneme Perception Test installer package specific properties can be set in order to configure the installation itself. These properties are described in the following chapter, together with some examples. All this information is subject to change without further notice.

## 2 PHONEME PERCEPTION TEST INSTALLER COMMAND LINE PARAMETERS

### 2.1 Available Properties

The Phoneme Perception Test MSI takes the following properties into account:

| | |
|---|---|
| CHECKOFF | <>1: default, checks turned on<br>1: the following checks are turned off<br>- required operating system<br>- admin rights<br>- prerequisites (windows installer, .net framework, runtime libs…)<br>- block for Noah versions < 3.5 |

C_LANGUAGE Phoneme Perception Test language selection. Not all listed languages might be supported at all times. Please verify the actual availability in the 'User Settings' dialog, country drop-down menu.

| | |
|---|---|
| Danish | da-DK |
| Dutch | nl-NL |
| English | en-US |
| French | fr-FR |
| German | de-DE |
| Italian | it-IT |
| Japanese | ja-JP |
| Korean | ko-KR |
| Norwegian | nb-NO |
| Polish | pl-PL |
| Portuguese | pt-BR |
| Russian | ru-RU |
| Spanish | es-ES |
| Swedish | sv-SE |

C_NOAHINSTALLATION 0: standalone installation
1: Noah installation (default setting but not considered if Noah not found)

C_INSTALLDIR Phoneme Perception Test installation directory

## 2.2 Examples

Precondition: Current directory is where the installation program resides.

**a)** The following command will install Phoneme Perception Test…
- unattended/silent
- will set the Phoneme Perception Test language to French
- will register the Phoneme Perception Test in Noah (Noah needs to be installed)
- will set the installation directory to C:\Program Files\Phonak\Phoneme Perception Test
- will produce a log file PhonemePerceptionTestInstall.log in the Temp directory

Setup.exe /s /v"/qn /l*v %temp%\PhonemePerceptionTestInstall.log C_LANGUAGE=fr-FR C_NOAHINSTALLATION=1 C_INSTALLDIR=\"C:\Program Files\Phonak\Phoneme Perception Test\""

**b)** The following command will install the Phoneme Perception Test…
- unattended, with a basic user interface
- will set the Phoneme Perception Test language to German
- will register the Phoneme Perception Test in Noah (Noah needs to be installed)
- will set the installation directory to C:\Program Files\Phonak\Phoneme Perception Test

Setup.exe /s /v"/qb C_LANGUAGE=de-DE C_NOAHINSTALLATION=1 C_INSTALLDIR=\"C:\Program Files\Phonak\Phoneme Perception Test\""

**c)** The following command will uninstall Phoneme Perception Test with a reduced user interface and produce a log file C:\ PhonemePerceptionTestUnInstall.log

Setup.exe /s /x /v"/qr /l*v %temp%\PhonemePerceptionTestUnInstall.log"

Note: when the setup.exe is called and no log file defined, a log file %temp%\ PhonemePerceptionTestInstall.log is automatically created

## 3 SETUP.EXE COMMAND-LINE PARAMETERS (INSTALLSHIELD HELP)

**InstallShield 2010**

Like your compiled .msi file, Setup.exe can accept a number of command-line parameters. Update.exe (available only for Basic MSI and InstallScript MSI projects) accepts nearly all of the same command-line parameters. Using these parameters, end users can specify such data as the language that the installation should run in and whether to launch Setup.exe silently. End users can also pass parameters through Set-up.exe to the included .msi file.

**Note**

Command-line options that require a parameter must be specified with no space between the option and its parameter. For example, **Setup.exe /v"ALLUSERS=2"** is valid, while **Setup.exe /v "ALLUSERS=2"** is not.

Quotation marks around an option's parameter are required only if the parameter contains spaces. If a path within a parameter contains spaces, you may need to use quotation marks within quotation marks, as in the following example:

**Setup.exe /v"INSTALLDIR=\"c:\My Files\""**

**Project**

Some of the command-line options apply to only certain project types. Project-specific limitations are listed for each option.

### 3.1 Built-In Command-Line Parameters

This section describes valid command-line parameters for Setup.exe. The parameters are organized into the following categories:

- Silent Installations
- Special Installation Modes
- Passing Data to the Installation
- Download and Cache Locations (Basic MSI and InstallScript MSI Projects)
- Debugging
- SMS Data
- Miscellaneous

Silence Installation

| Parameters for Special Installation Modes | | |
|---|---|---|
| **Parameter** | **Project Type** | **Description** |
| **/p** : specify password | Basic MSI, InstallScript MSI | If you selected the Password protect Setup.exe option in the Release Wizard for your release, the end user must specify the password with the /p option at run time during a silent installation. A typical command is Setup.exe /s /p"password". |
| **/r** : record mode | InstallScript, InstallScript MSI | In order to run an InstallScript MSI or InstallScript project installation program in silent mode, you must first run Setup.exe with the /r option to generate a response file, which stores information about the data entered and options selected by the user at run time.<br><br>Running an InstallScript MSI or InstallScript installation program with the command Setup.exe /r displays all the runtime dialogs, and stores the data in a file called Setup.iss, created inside the system's Windows folder. To specify an alternative response file name and location, use the /f1 option, described below.<br><br>Basic MSI projects do not create or use a response file for silent installations. |
| **/s** : silent mode | Basic MSI, InstallScript, InstallScript MSI | For an InstallScript MSI or InstallScript project, the command Setup.exe /s runs the installation in silent mode, by default based on the responses contained in a response file called Setup.iss in the same directory. (Response files are created by running Setup.exe with the /r option.) To specify an alternative file name or location of the response file, use the /f1 option.<br><br>The command Setup.exe /s also suppresses the Setup.exe initialization dialog for a Basic MSI installation, but it does not read a response file. To run a Basic MSI installation silently, run the command line Setup.exe /s /v/qn. (To specify the values of public properties for a silent Basic MSI installation, you can use a command such as Setup.exe /s /v"/qn INSTALLDIR=D:\Destination".)<br><br>Using this command-line parameter to launch an installation that includes an InstallShield prerequisite does not automatically run the prerequisite installation silently. You may also need to specify a valid silent command-line parameter for the InstallShield prerequisite in the **Specify the command line for the application when the setup is running in silent mode** setting on the Application to Run tab in the InstallShield Prerequisite Editor.<br><br>For more information, see Specifying Command-Line Parameters for an InstallShield Prerequisite. |

| | | |
|---|---|---|
| **/f1** : specify alternative response file name and path | InstallScript, InstallScript MSI | Using the /f1 option enables you to specify where the response file is (or where it should be created) and what its name is, as in Setup.exe /s /f1"C:\Temp\Setup.iss". Specify an absolute path; using a relative path gives unpredictable results. The /f1 option is available both when creating a response file (with the /r option) and when using a response file (with the /s option). |
| **/f2** : specify alternative log file name and path | InstallScript, InstallScript MSI | When running an InstallScript MSI or InstallScript installation in silent mode (using the /s option), the log file Setup.log is by default created in the same directory and with the same name (except for the extension) as the response file. The /f2 option enables you to specify an alternative log file location and file name, as in Setup.exe /s /f2"C:\Setup.log". Specify an absolute path; using a relative path gives unpredictable results. |

Special Installation Modes

| Parameters for Special Installation Modes | | |
|---|---|---|
| **Parameter** | **Project Type** | **Description** |
| **/a** : administrative installation | Basic MSI, InstallScript MSI | The /a parameter does not work with Update.exe. Update.exe launches a patch that accesses and modifies an existing cached .msi file on the system, and an administrative installation does not cache the .msi file.<br><br>The /a option causes Setup.exe to perform an administrative installation. An administrative installation copies (and uncompresses) your data files to a directory specified by the user, but it does not create shortcuts, register COM servers, or create an uninstallation log.<br><br>If an installation contains InstallShield prerequisites and you want to extract them from Setup.exe, add a path after the /a parameter to extract the prerequisites to that location. A sample command is Setup.exe /a"C:\temp". |
| **/j** : advertise mode | Basic MSI, InstallScript MSI | The /j option causes Setup.exe to perform an advertised installation. An advertised installation creates shortcuts, registers COM servers, and registers file types, but does not install your product's files until the user invokes one of these "entry points." |
| **/x** : uninstall mode | Basic MSI, InstallScript MSI | The /x option causes Setup.exe to uninstall a previously installed product. |
| **/uninst** : uninstall product | InstallScript, InstallScript MSI (if the InstallScript user interface UI style is the traditional style, which uses the InstallScript engine as an external UI handler) | This parameter does not apply to InstallScript MSI projects in which the InstallScript UI style is the new style (which uses the InstallScript engine as an embedded UI handler). To learn more, see Using the InstallScript Engine as an External vs. Embedded UI Handler for InstallScript MSI Installations.<br><br>The /uninst option causes Setup.exe to execute only the event handler function OnUninstall, whose default code<br><br>uninstalls the previously installed product. |
| **/removeonly** : uninstall product | InstallScript, InstallScript MSI | The /removeonly option sets the REMOVEONLY system variable equal to a non-zero value. The default code for the OnMaintUIBefore event handler function conditionally displays the SdWelcomeMaint dialog box, depending on the value of REMOVEONLY. |

**/instance**=<InstanceId> : Basic MSI specify which instance should be installed or uninstalled

The /instance=<InstanceId> option—which is available for Basic MSI projects that support the installation of multiple instances of a product—lets you specify which instance you want to install, update, or uninstall. <*InstanceId*> represents the value of the InstanceId property that identifies the instance. Whenever you use this option and include a valid InstanceId value, the installation suppresses the instance selection dialog.

For example, the following sample command line installs the instance that has 2 as the value of the InstanceId property:

Setup.exe /instance=2

Use *default* to identify the instance that is installed by the base installation package, as in the following example:

Setup.exe /instance=default

To specify the instance that you want to uninstall, include the /x option with the /instance=<InstanceId> option.

For more information, see Naming an Instance and Run-Time Behavior for Installing Multiple Instances of a Product.

Passing Data to the Installation

| Parameters for Passing Data to the Installation | | |
|---|---|---|
| **Parameter** | **Project Type** | **Description** |
| **/v** : pass arguments to Msiexec | Basic MSI, InstallScript MSI | Use the /v option to pass command-line options and values of public properties through to Msiexec.exe.<br><br>If you want to pass more than one argument to Msiexec.exe, you can use the /v option multiple times at the command line, once for each argument. For example:<br><br>Setup.exe /v"/l*v c:\test.log" /v"MYPROPERTY1=value1" /v"/qb"<br><br>As an alternative, you can pass multiple arguments through the /v option as in the following example:<br><br>Setup.exe /v"/l*v c:\test.log "MYPROPERTY1=value1 /qb" if you pass the /v parameter at the command prompt when launching Setup.exe, any parameters that are specified for the CmdLine keyname in Setup.ini are ignored. To learn more, see Setup.ini. |
| **/v**"ISSCRIPTCMDLINE=\"<Option1> <Option2>\"" : pass arguments that should be passed to the script | Basic MSI projects that have InstallScript custom actions | This option specifies command-line parameters to be passed to the script. Any property supported by InstallScript MSI (where appropriate) can be specified. (The most common ones are /d and /z.)<br><br>For example, the following indicates that you want to debug the script, and that the CMDLINE variable should contain TEST.<br><br>Setup.exe /v"ISSCRIPTCMDLINE=\"-d -zTEST\""<br><br>Note that as shown above, when you want to specify that a double quote character is not a delimiter for the command line but a delimiter for the property, use \".<br><br>Note also that as with any public Windows Installer property, this property should be specified with all uppercase letters. |
| **/z** : pass arguments to CMDLINE variable | InstallScript MSI | The /z option is used to pass data to the InstallScript system variable CMDLINE, as in Setup.exe /z"My Custom Data", after which the variable CMDLINE would contain the string "My Custom Data". |

Download and Cache Locations (Basic MSI and InstallScript MSI Projects)

| Parameters for Download and Cache Locations | | |
|---|---|---|
| **Parameter** | **Project Type** | **Description** |
| **/ua** : specify URL for InstMsiA.exe<br>**/uw** : specify URL for InstMsiW.exe | Basic MSI, InstallScript MSI | In the Release Wizard, you can specify download locations for the Windows Installer installers InstMsiA.exe and InstMsiW.exe. A user can specify an alternative URL at run time using the /ua and /uw options, as in the following example:<br><br>Setup.exe /uw"http://www.otherlocation.com/engines"<br><br>The file name is not necessary.<br><br>You must specify the full URL with the parameters. |
| **/um** : specify URL to .msi package | Basic MSI, InstallScript MSI | In the Release Wizard, for a Web Downloader build, you can specify a download location for your .msi database. A user can specify an alternative URL using the /um option, as in the following example:<br><br>Setup.exe /um"http://www.otherlocation.com/packages/product.msi"<br><br>You must specify the full URL with the parameter. |
| **/b** : cache installation locally | Basic MSI, InstallScript MSI | In the Release Wizard, for a Downloader build, you can specify whether to cache the contents of a compressed package on the local system. With the /b option, the user can specify the directory in which to cache the installation files, as in the following example: Setup.exe /b"C:\CacheDirectory"<br><br>You must specify the full URL with the parameter. |

Debugging

| Parameters for Debugging | | |
|---|---|---|
| **Parameter** | **Project Type** | **Description** |
| **/d** : debug InstallScript | Basic MSI projects with InstallScript custom actions, InstallScript, InstallScript MSI | For an InstallScript project, running the command Setup.exe /d runs the installation program with the InstallScript Debugger.<br><br>Debugging InstallScript code requires the debug-information file Setup.dbg to be available. To debug an InstallScript project on a system other than the development system:<br><br>1. Copy the InstallScript Debugger executable file ISDbg.exe (located in the System folder of your InstallShield distribution) to the test system, and register it by launching the executable with the /REGSERVER command-line option.<br><br>2. Copy Setup.dbg to the test system.<br><br>3. Run Setup.exe with the command Setup.exe /d"<path>", where *path* is the directory containing Setup.dbg.<br><br>For a Basic MSI project, the following command runs your InstallScript custom actions in the InstallScript Debugger:<br><br>Setup.exe /v"ISSCRIPTDEBUG=1 ISSCRIPTDEBUGPATH=\"path-to-Setup.dbg\"" |
| **/debuglog** : generate a log file for debugging | Basic MSI, InstallScript MSI | The /debuglog parameter lets you generate a log file for Setup.exe.<br><br>To generate a log named InstallShield.log in the same directory as the Setup.exe file, pass just the command-line parameter. Note that this does not work if the Setup.exe file is in a read-only location. For example: setup.exe /debuglog<br><br>To specify the name and location of the log file, pass the path and name, as in the following example:<br><br>setup.exe /debuglog"C:\PathToLog\setupexe.log" |

SMS Data

| Parameters for SMS Data | | |
|---|---|---|
| **Parameter** | **Project Type** | **Description** |
| **/m** : generate .mif file | InstallScript, InstallScript MSI | The /m option causes Setup.exe to generate an SMS Management Information Format (.mif) file. Following is a typical command:<br><br>Setup.exe /m"SampleApp"<br><br>Including the ".mif" file extension is not necessary. |
| **/m1** : specify serial number in .mif file | InstallScript, InstallScript MSI | Using the /m1 parameter (along with /m) enables you to specify a serial number to be written to the .mif file. A typical command is as follows:<br><br>Setup.exe /m"SampleApp" /m1"1234-5678" |
| **/m2** : specify locale string in .mif file | InstallScript, InstallScript MSI | Using the /m2 parameter (along with /m) enables you to specify a locale string to be written to the .mif file. A typical command is as follows:<br><br>Setup.exe /m"SampleApp" /m2"ENU" |

Miscellaneous

| Parameters for Miscellaneous | | |
|---|---|---|
| **Parameter** | **Project Type** | **Description** |
| **/delayedstart**:<number of seconds> : delay initialization of the installation | InstallScript | Specifies the amount of time (in seconds) by which initialization of the installation is delayed after Setup.exe is launched.<br><br>Using the -delayedstart option is recommended when manually launching an additional installation after reboot (for example, by using the RunOnce key). The delay allows the operation system to initialize completely; this prevents the problems—such as Remote Procedure Call (RPC) errors—that can occur if an installation initializes before the operating system has initialized completely. The recommended delay length is 30 seconds.<br><br>Note that this option is not needed when the installation starts automatically after reboot (for example, due to a call to SdFinishReboot before reboot). |
| **/runfromtemp** : always clone and run from the temporary directory | Basic MSI, InstallScript, InstallScript MSI | This parameter allows the setup author to always clone the setup and run it from the temporary directory, even if the setup does not meet the conditions for running from the temporary directory. This parameter is ignored if the setup is a self-extracting executable file (.exe). |
| **/clone_wait** : wait for the cloned setup process to complete before exiting | InstallScript | This parameter indicates that the original setup should wait for the cloned setup process to complete before exiting. |
| **/extract_all**:<path> : package's files should not be run but simply extracted | InstallScript | Specifies that a self-extracting package's files should not be run but simply extracted to the location that is specified by <path>. |
| **/h** : clone release to a temporary location and run from that location | Basic MSI, InstallScript MSI | The build engine automatically creates an installation that supports Setup.exe cloning in cases where cloning is required (for example, multi-disk installations). If you need to do this manually, pass /h to Setup.exe and it will clone itself to a temporary location and run from that location. |
| **/hide_**progress : suppress all progress dialogs | Basic MSI, InstallScript, InstallScript MSI | Suppresses the display of any small and standard progress dialogs that might be shown during initialization. The small progress dialog is usually used for installations that display a splash screen during initialization, since a standard-size progress dialog does not leave any space for the splash screen. Specifying the /hide_progress option hides the small progress dialog for those installations, so end users would see just the splash screen without any progress indication.<br><br>For InstallScript installations: If you specify /hide_progress and include a splash screen in your InstallScript installation, the length of time that the splash screen is displayed depends on whether SmallProgress=Y or SmallProgress=N is specified in Setup.ini.<br><br>• If SmallProgress=Y is specified, the splash screen is shown for as long as the progress dialog would have been displayed if /hide_progress was not specified. |

| | | |
|---|---|---|
| | | • If SmallProgress=N is specified, the splash screen is shown for the length of time specified in the SplashTime key; thus, using /hide_progress and SmallProgress=Y at the same time is not recommended.<br><br>For InstallScript MSI installations: If you include a splash screen, the installation automatically switches to the small progress dialog, and the splash screen is shown only during the time that the progress dialog is displayed. Note that this is true even if /hide_progress is specified. Therefore, it is recommended that you avoid using /hide_progress with a splash screen in InstallScript MSI installations. |
| **/hide_splash** : suppress splash screen | InstallScript, InstallScript MSI | Suppresses the display of the splash screen if one is included. |
| **/hide_usd** : suppress update dialog for multi-instance installations | InstallScript | Suppresses display of the dialog that is displayed by an update-enabled installation to let the end user select which instance of your product will be updated. This dialog is displayed by default when an update-enabled installation detects multiple previous installations. When this command-line option is used and an update-enabled installation detects multiple previous installations, the installation updates the first previous installation that it finds. |
| **/ig** : specify the value of the system variable INSTANCE_GUID | InstallScript | Specifies the value of the system variable INSTANCE_GUID; for example, -ig{722C7440-B317-4B3B-AECA-0199EA4E7CDB}. If this option is not used, the installation automatically assigns a value to INSTANCE_GUID (for multi-instance installations, this value is a newly generated GUID; for standard installations, this value is the same as the value of PRODUCT_GUID). This option is useful if you have created an installation launcher—that is, a custom application that runs before your installation does to perform pre-setup tasks, such as determining the instance GUID that you want to use for the installation. Do not specify anything other than a valid GUID with this option. |
| **/installfromweb** : specify the full path (or URL) to the installation media files (Disk1) | InstallScript | This option is similar to the media_path option except that this option forces the installation to behave like a launched One-Click Install installation, even if the path to the media files is not a URL. Use this option if you are launching the installation from a Web page manually. In addition, this option is added automatically if the built-in Setup.ocx file is used to launch the installation. |
| **/media_path** : specify the full path (or URL) to the installation media files (Disk1) | InstallScript | This option indicates that the installation should look for the Disk1 files in the location that is specified. Note that only the Setup.exe file needs to be in the original launch location; the installation obtains all other required files from the specified location (including Setup.exe, which must be present in the media location). You can specify a URL as the path to the media files; in this case, the installation behaves like a launched One-Click Install installation, which always shows the security dialog. To learn more about the behavior of One-Click Install installations, see One-Click Install Installations in InstallScript |

| | | |
|---|---|---|
| | | Projects. |
| **/L** : Language ID | Basic MSI, InstallScript, InstallScript MSI | This option indicates that the installation should run in the specified language as specified. You can specify the language ID as either a hexadecimal or decimal number. If you specify the hexadecimal number, be sure to proceed<br><br>the value with 0x. For example, **-l0x0407** or **-l1031** indicates that the installation should be run in German. Note that if you specify a language ID that is not supported by the installation or you specify an invalid<br><br>language ID, the parameter is ignored. Also note that if this parameter is specified and it is valid, the language dialog (if enabled) is automatically suppressed. |
| **/w** : wait | Basic MSI, InstallScript MSI | For a Basic MSI project, the /w option forces Setup.exe to wait until the installation is complete before exiting.<br><br>If you are using the /w option in a batch file, you may want to precede the entire Setup.exe command-line option with *s*tart /WAIT. A properly formatted example of this usage is as follows: start /WAIT setup.exe /w |
| **/SMS : wait** | InstallScript MSI | For an InstallScript MSI project, Setup.exe automatically waits for the installation to finish before exiting, so this option (used by earlier versions of InstallShield Professional) is no longer necessary. |

3.2 User-Defined Command-Line Parameters (InstallScript Projects)

Along with the command-line switches listed above, -bd, -f, and -zi are command-line switches reserved by InstallShield. User redefinition of these command-line switches, either uppercase or lowercase, can cause errors.

You can define your own command-line arguments, which are copied to the system variable CMDLINE at run time. Like predefined command-line switches, you can pass these arguments directly to Setup.exe, place them in Setup.ini, or (for testing purposes while you are using the InstallShield IDE) place them in the Settings dialog box, which is displayed when you click Settings on the Build menu in InstallShield.

**Note**

Setup.exe initializes correctly even on systems with more than 256 MB of memory and always stays in memory until the setup is complete. Due to the nature of DOS, when you launch Setup.exe from the command line, a DOS prompt is quickly returned although Setup.exe is still in memory.

**See Also**

Creating a Setup Launcher

## 4 MSI COMMAND-LINE OPTIONS (INSTALLSHIELD HELP)

The executable program that interprets packages and installs products is Msiexec.exe. Note that Msiexec also sets an error level on return that corresponds to system error codes. Command-line options are case-insensitive.

The command-line options in the following table are available with Windows Installer 3.0 and earlier versions. The Standard Installer Command-Line Options are also available beginning with Windows Installer 3.0.

| Option | Parameters | Meaning |
|---|---|---|
| **/I** | *Package*\|*ProductCode* | Installs or configures a product. |
| **/f** | [p\|o\|e\|d\|c\|a\|u\|m\|s\|v] *Package*\|*ProductCode* | Repairs a product. This option ignores any property values entered on the command line. The default argument list for this option is 'omus.' This option shares the same argument list as the **REINSTALLMODE** property.<br><br>**p -** Reinstalls only if file is missing.<br><br>**o -** Reinstalls if file is missing or an older version is installed.<br><br>**e -** Reinstalls if file is missing or an equal or older version is installed.<br><br>**d -** Reinstalls if file is missing or a different version is installed.<br><br>**c -** Reinstalls if file is missing or the stored checksum does not match the calculated value. Only repairs files that have<br><br>**msidbFileAttributesChecksum** in the Attributes column of the File table.<br><br>**a -** Forces all files to be reinstalled.<br><br>**u -** Rewrites all required user-specific registry entries.<br><br>**m -** Rewrites all required computer-specific registry entries.<br><br>**s -** Overwrites all existing shortcuts.<br><br>**v -** Runs from source and re-caches the local package. Do not<br><br>use the **v** reinstall option for the first installation of an application or feature. |
| **/a** | *Package* | Administrative installation option. Installs a product on the network. |
| **/x** | *Package*\|*ProductCode* | Uninstalls a product. |

| | | |
|---|---|---|
| **/j** | [u\|m]*Package*<br><br>*or*<br><br>[u\|m]*Package***/t***Transform List*<br><br>*or*<br><br>[u\|m]*Package***/g***LanguageID* | Advertises a product. This option ignores any property values entered on the command line.<br><br>**u -** Advertises to the current user.<br><br>**m -** Advertises to all users of machine.<br><br>**g -** Language identifier.<br><br>**t -** Applies transform to advertised package. |
| **/L** | [i\|w\|e\|a\|r\|u\|c\|m\|o\|p\|v\|x\|+\|!\|*] *Logfile* | Writes logging information into a logfile at the specified existing path. The path to the logfile location must already exist. The installer does not create the directory structure for the logfile. Flags indicate which information to log. If no flags are specified, the default is 'iwearmo.'<br><br>**i -** Status messages.<br><br>**w -** Nonfatal warnings.<br><br>**e -** All error messages.<br><br>**a -** Start up of actions.<br><br>**r -** Action-specific records.<br><br>**u -** User requests.<br><br>**c -** Initial UI parameters.<br><br>**m -** Out-of-memory or fatal exit information.<br><br>**o -** Out-of-disk-space messages.<br><br>**p -** Terminal properties.<br><br>**v -** Verbose output.<br><br>**x -** Extra debugging information.<br><br>**Windows Installer 2.0:** Not supported. The x option is available with Windows Installer version 3.0.3790.2180 and later.<br><br>**+ -** Append to existing file.<br><br>**! -** Flush each line to the log.<br><br>**"*" -** Wildcard, log all information except for the v and x options. To include the v and x options, specify "/l*vx".<br><br>**Note** For more information about all the methods that are available for setting the logging mode, see Normal Logging in the Windows Installer Logging section. |

**/m** *filename*

**Note** The length of *filename* must be no more than eight characters.

Generates an SMS status .mif file. Must be used with either the install (-i), remove (-x), administrative installation (-a), or reinstall (-f) options. The ISMIF32.DLL is installed as part of SMS and must be on the path.

The fields of the status mif file are filled with the following information:

Manufacturer - **Author**

Product - **Revision Number**

Version - **Subject**

Locale - **Template**

Serial Number - not set

Installation - set by ISMIF32.DLL to "DateTime"

InstallStatus - "Success" or "Failed"

Description - Error messages in the following order: 1) Error messages generated by installer. 2) Resource from Msi.dll if installation could not commence or user exit. 3) System error message file. 4) Formatted message: "Installer error %i", where %i is error returned from Msi.dll.

**/p** *PatchPackage[;patchPackage2…]*

Applies a patch. To apply a patch to an installed administrative image you must combine the following options:

/p <*PatchPackage>[;patchPackage2…]* /a <*Package*>

**/q** n|b|r|f

Sets user interface level.

q , qn - No UI

**qb -** *Basic UI*. Use qb! to hide the **Cancel** button.

**qr -** *Reduced UI* with no modal dialog box displayed at the end of the installation.

**qf -** *Full UI* and any authored FatalError, UserExit, or Exit modal dialog boxes at the end.

**qn+ -** No UI except for a modal dialog box displayed at the end.

**qb+ -** Basic UI with a modal dialog box displayed at the end. The modal box is not displayed if the user cancels the installation. Use qb+! or qb!+ to hide the **Cancel** button.

**qb- -** Basic UI with no modal dialog boxes. Please note that **/qb+-** is not a supported UI level. Use qb-! or qb!- to hide the **Cancel** button.

| | | |
|---|---|---|
| | | Note that the ! option is available with Windows Installer 2.0<br><br>and works only with basic UI. It is not valid with full UI. |
| **/?** or **/h** | | Displays copyright information for Windows Installer. |
| **/y** | *module* | Calls the system function **DllRegisterServer** to self-register modules passed in on the command line. Specify the full path to the DLL. For example, for MY_FILE.DLL in the current folder you can use:<br><br>**msiexec /y .\MY_FILE.DLL**<br><br>This option is only used for registry information that cannot be added using the registry tables of the .msi file. |
| **/z** | *module* | Calls the system function **DllUnRegisterServer** to unregister modules passed in on the command line. Specify the full path to the DLL. For example, for MY_FILE.DLL in the current folder you can use:<br><br>**msiexec /z .\MY_FILE.DLL**<br><br>This option is only used for registry information that cannot be removed using the registry tables of the .msi file. |
| **/c** | | Advertises a new instance of the product. Must be used in conjunction with /t. Available starting with the Windows Installer version that is shipped with Windows Server 2003 and Windows XP Service Pack 1 (SP1). |
| **/n** | *ProductCode* | Specifies a particular instance of the product. Used to identify an instance installed using the multiple instance support through a product code changing transforms. Available starting with the Windows Installer version shipped with Windows Server 2003 and Windows XP SP1. |

The options /i, /x, /f[p|o|e|d|c|a|u|m|s|v], /j[u|m], /a, /p, /y and /z should not be used together. The one exception to this rule is that patching an administrative installation requires using both /p and /a. The options /t, /c and /g should only be used with /j. The options /l and /q can be used with /i, /x, /f[p|o|e|d|c|a|u|m|s|v], /j[u|m], /a, and /p. The option /n can be used with /i, /f, /x and /p.

To install a product from A:\Example.msi, install the product as follows:

**msiexec /i A:\Example.msi**

Only public properties can be modified using the command line. All property names on the command line are interpreted as uppercase but the value retains case sensitivity. If you enter **MyProperty** at a command line, the installer overrides the value of MYPROPERTY and not the value of **MyProperty** in the Property table. For more information, see About Properties.

To install a product with PROPERTY set to VALUE, use the following syntax on the command line. You can put the property anywhere except between an option and its argument.

Correct syntax:

**msiexec /i A:\Example.msi PROPERTY=VALUE**

Incorrect syntax:

**msiexec /i PROPERTY=VALUE A:\Example.msi**

Property values that are literal strings must be enclosed in quotation marks. Include any white spaces in the string between the marks.

**msiexec /i A:\Example.msi PROPERTY="Embedded White Space"**

To clear a public property by using the command line, set its value to an empty string.

**msiexec /i A:\Example.msi PROPERTY=""**

For sections of text set apart by literal quotation marks, enclose the section with a second pair of quotation marks.

**msiexec /i A:\Example.msi PROPERTY="Embedded ""Quotes"" White Space"**

The following example shows a complicated command line.

**msiexec /i testdb.msi INSTALLLEVEL=3 /l* msi.log COMPANYNAME="Acme ""Widgets"" and ""Gizmos."""**

The following example shows advertisement options. Note that switches are not case-sensitive.

**msiexec /JM msisample.msi /T transform.mst /LIME logfile.txt**

The following example shows you how to install a new instance of a product to be advertised. This product is authored to support multiple instance transforms.

**msiexec /JM msisample.msi /T :instance1.mst;customization.mst /c /LIME logfile.txt**

The following example shows how to patch an instance of a product that is installed using multiple instance transforms.

**msiexec /p msipatch.msp;msipatch2.msp /n {00000001-0002-0000-0000-624474736554} /qb**

When you apply patches to a specific product, the /i and /p options cannot be specified together in a command line. In this case, you can apply patches to a product as follows.

**msiexec /i A:\Example.msi PATCH=msipatch.msp;msipatch2.msp /qb**

The **PATCH** property cannot be set in a command line, when /p option is used. If the **PATCH** property is set when the /p option is used, the value of **PATCH** property is ignored and overwritten.

Build date: 5/22/2008

© 2008 Microsoft Corporation. All rights reserved.