



Using Run-Time Command-Line Options

InstallShield MultiPlatform 11.5

The following command-line options, listed alphabetically, can be used when running an installation or uninstallation. You can use these options whether you invoke the wizard directly through a Java command, or execute it using an installation launcher—the launcher passes these arguments to the wizard. When you execute the wizard using an installation launcher, additional command-line options are available (see [Using Installation and Uninstallation Launcher Command-Line Options](#)).



NOTE

To add a classpath to the JAR file at run time, type the following:

```
java -cp additionalClass setup.jar run
```

Where `additionalClass` is the file name of the class that you want to add to `setup.jar` when it executes (for example, `myLib.zip`). To add more than one classpath, separate each with a semicolon (";") if you are developing on Windows, separate each with a colon (":") if you are developing on UNIX. Do not include any spaces between each classpath.

Available Run-Time Command-Line Options

Run-Time Command-Line Options

Option	Purpose	Example Usage(s)
-awt	Specifies to use the AWT interface mode when displaying the dialogs. This option only applies to 5.x project types ; in other project types, it has no effect.	<code>./setup.bin -awt</code>
-condenser (Premier Edition)	<p>Enables you to launch a project that can be condensed in condenser mode. You can specify which locales and platforms to include, include/exclude specific features, and specify the type of output. Both defined and custom platforms and locales can be specified.</p> <p>For additional information about this option, see Using Condenser Command-Line Options.</p>	<p>The example below defines the following:</p> <p>The <code>!!</code> extended information indicator is optional if the platform does not contain any Extended Info.</p> <ul style="list-style-type: none"> The supported Spanish and French locales The AIX Power and HP-UX 11i platforms, both of which do not have any extended information Exclude Feature22 <pre>java -cp setup.jar run -condenser -G condenserLocales=es;fr -G condenserPlatforms= "@@displayName=AIX Power Platform;name=AIX;version= .;arch=POWER_RS POWER_PC\ ppc;parent=AIX Platform!!@@displayName= HP-UX 11i;name=HP-UX;version= .\.11\.11;arch=PA.RISC!!" -P Feature22.active=false</pre>
-console (Premier Edition)	Specifies to use the console interface mode, where messages during installation appear on the Java console, and the wizard is run in console mode. This option overrides the Distributions' Show Console property if it is "False" and automatically includes the -is:javaconsole Java option. This option is required to run a distribution that was set up to use a console distribution, for example, Win32 Console Launcher and Windows IA64 Console Launcher.	<code>java -cp setup.jar run -console</code> <code>setupwin32Console.exe -console</code>
-G global Wizard Property="value"	Enables you to set a global wizard property in a command line or response/options file .	<code>java -cp setup.jar run -G replaceExistingResponse="yesToAll"</code>
-goto beanID	Forces the wizard to jump or "go to" the specified bean upon startup.	<code>java -cp setup.jar run -goto bean45</code>
-log #! <code>fileName></code> <code>@eventType;</code> <code>eventType</code>	<p>Initializes logging for the wizard. The parameters for this option have the following definitions:</p> <ul style="list-style-type: none"> <code>#</code>—Echoes the display to standard output. <code>!fileName></code>—The name of a file to save the log information in. If you specify <code>!</code> without a file name, the default log file name is used. <code>@eventType;</code> <code>eventType</code>—A list of the events to log, each separated by a semicolon. <code>ALL</code> enables the logging of all events; <code>NONE</code> disables logging and clears the log file. 	<code>java -cp setup.jar -log #!</code> <code>/opt/projects/logfile @ALL run</code>



In most UNIX shells, the semicolon character (;) has a specific meaning to the OS, and results in an error when used in a command line. This problem can be avoided if the option is surrounded by single quotes (') or another escape character.

For an installation that requires [multiple media](#), this option specifies the particular media number and location, where # represents the media number and *location* represents the path to or mounting point of that piece of media. Multiple location parameters can also be specified. Multiple instances of this command-line option would be required if a [silent installation](#) involved multiple media—the [response/options file](#) would contain one or more instances of this option to indicate the location of each piece of media.

-media #=*location*

```
java -cp setup.jar run -media 1=D:/ 2=E://
setup.jar -media 1=/usr/mount1 2=/usr/mount2
```



This option only works with the installation JAR file—it does not work with installation launcher distributions.

-options-record *response*
FileName

Specifies that [automatically generate a record-type response/options file should be generated automatically](#) for the project after the completion of the installation or uninstallation.



There is no space between **-options** and **-record**.

```
java -cp setup.jar run -options
-record recordFile.txt
```

-options-template
response
FileName

Specifies that a response/options options-type file should be generated automatically for the project now that can be used to provide user input during an installation or uninstallation.



There is no space between **-options** and **-template**.

```
java -cp setup.jar run -options-template templateFile.txt
```

-options *response*
FileName

Specifies that a response/options file should be used to execute the installation or uninstallation that contains command-line options, one command per line, that set specified properties for the installation or uninstallation.

```
java -cp setup.jar run -options /opt/projects/responseFile
```



A response/options file is usually used when a silent installation is run (see the next option).

-P *product*
BeanID
.property
Name(*.sub*
property name)
=value

Specifies properties for a product bean. The Bean ID property of the product bean is specified first, followed by the property and optional subproperty, followed by the new value for the property or subproperty. This option can be specified multiple times, and can be used in Universal Installer and Universal Static Suite projects.

```
java -cp setup.jar run -P bean56.displayName
=SuperSoftware
```

-searchpath *paths*
SeparatedBySemicolons

Specifies additional paths to search for [assembly](#) data files (assembly.dat), which are then added to the classpath so that the referenced assemblies can be found at run time. As a result, you can provide assemblies manually in a command line. If any path could include a space(s), enclose the entire path list in double-quotes, as illustrated in this example.

```
java -cp setup.jar -searchpath "X:/Shared/Assemblies;
bin/my assembly directory"
```

-SP *subinstaller*
ArchiveName
.product
BeanID
.property
Name
(.subproperty
*Name)
=
*value**

Specifies properties for a product bean within a Universal Dynamic Suite where *subinstallerArchive* *Name* represents the path to the subinstaller within the suite (assemblies/uuid/version/assembly.dat), followed by the Bean ID property of the product bean, followed by the property name and optional subproperty, followed by the new value for the property or subproperty. This option can be specified multiple times.

```
java run -SP
"assemblies/056d238eeab8af865a18de5b5ab4c72e/1.00.000/assembly
bean56.active=false
```

-silent

Specifies to install or uninstall the product in silent mode, where the installation or uninstallation is performed with no end-user interaction. This option automatically includes the Java [-is:silent](#) option, which only prevents the appearance of the Launcher UI. Using **-silent** results in a totally silent installation.

```
java -cp setup.jar run -silent
```



The **-silent** option basically renders all of the dialog logic useless—only the **execute()** method is called during a silent installation.

-SW *subinstaller*
ArchiveName
.wizardBean
ID.property
Name(*.sub*
property
Name)
=
value

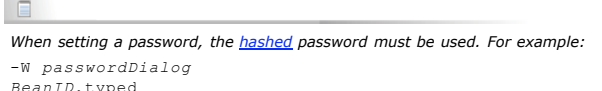
Specifies properties for a wizard bean within a Universal Dynamic Suite where *subinstallerArchive* *Name* represents the path to the subinstaller within the suite (assemblies/uuid/version/assembly.dat), followed by the Bean ID property of the wizard bean, followed by the property name and optional subproperty, followed by the new value for the property or subproperty. This option can be specified multiple times.

```
java run -SW
"assemblies/056d238eeab8af865a18de5b5ab4c72e/1.00.000/assembly
bean2.contentType=text
```

-swing

Specifies to use the Java Swing interface mode when displaying the dialogs. This option only applies if the project was built with Swing enabled; otherwise, it has no effect.

```
setup.exe -swing./setup.bin
-swing
```

<p>-V <i>Variable</i> <i>FieldName</i> <i>=value</i></p>	<p>Sets the value of a variable property, for example, in a control in a dialog, the exit code or output of Execute Process Actions, etc.</p>	<pre>java -cp setup.jar run -V myTextFieldVariable="yes"</pre>
<p>-vpd <i>path</i> (Premier Edition)</p>	<p>Sets the location of the VPD registry on the target machine. A relative or absolute path can be specified. If the path contains a space(s), enclose it within double-quotes. The specified path is appended to the platform-specific location.</p>	<pre>java -cp setup.jar run -vpd "/mySoftware/Killer Apps" java -cp setup.jar run -W bean2.contentType=text</pre>
<p>-W <i>beanID</i> <i>.property</i> <i>Name(.sub</i> <i>propertyName)=value</i></p>	<p>Specifies properties for a wizard bean. The Bean ID property is specified first, followed by the property and optional subproperty, followed by the new value for that property or subproperty. This option can be specified multiple times in one command.</p>	<p></p> <p><i>When setting a password, the hashed password must be used. For example:</i></p> <pre>-W passwordDialog BeanID.typed PasswordHash= "hashedPassword"</pre>

See Also

[Performing a Command-Line Build](#)
[Using Installation and Uninstallation Launcher Command-Line Options](#)
[Using Java Command-Line Options](#)

InstallShield MultiPlatform Help Library
28 November 2005

maCrovision
[copyright](#)
[contact](#)