

Setting Up (Servoy 5.2 branch only)



Servoy 5.2 instructions

This instruction is only relevant for the Servoy 5.2 branch. See [Setting Up](#) for the instructions for trunk and any other future branches

Servoy is a plugin for the Eclipse platform, so to develop the Servoy codebase an Eclipse SDK environment with Eclipse Plugin development facilities is required.

In order to get started with development on one of the open source Servoy projects, the following is required:

1. A Servoy Developer installation, with the SQL Explorer plugin installed and updated with the latest Servoy Server libraries
2. A separate Eclipse environment with the Eclipse plugin development tools installed
3. Configuration of the Eclipse installation

In This Chapter

- [Quick Start for the Impatient](#)
- [Before Getting Started: 32 Or 64 Bit Installation](#)
- [Installing Servoy Developer](#)
- [Installing Eclipse with the Eclipse Plugin Development Tools](#)
- [Configuration](#)

Quick Start for the Impatient

- Install Servoy Developer 5.2
- Run Servoy Developer and install the SQL Explorer plugin and close Servoy Developer again
- Install an Eclipse distribution that contains the plugin development tools for Eclipse (PDE), for example Eclipse Classic
- Setup a String Substitution variable named 'servoy_install' which points to the installation directory of the previously installed Servoy Developer
- Create a ClassPath variable named 'WEBSTART' that points to javaws.jar of the used JVM
- Checkout the open source Servoy projects from https://subversion.servoy.com/product/branches/v52x_branch/.
- Download the builds of the Servoy Server libraries from http://build.servoy.com:8080/latest/servoy_v52x_branch/ who's name postfix matched the value of the releaseNumber variable in the ClientVersion class of the checked-out source code and place them in {servoyInstall}/developer/plugins. Remove any other version of the same file if it exists.
- Setup a Target Platform by:
 - Opening the correct .target file, located in the targets directory of the 'com.servoy.eclipse.core' project:
 - generic.target when running Eclipse and Servoy Developer both under either a 32 or 64 JVM OR
 - the .target file that matches the platform and JVM on which Servoy Developer runs. In this scenario, also add the JVM that Servoy Developer uses as Execution environment to the Eclipse installation and mark it as default.
 - Clicking the link 'Set as Target Platform' of the Target editor
- Create a Launch Configuration (and directly start a debug session) by right-clicking the 'Servoy Launch.launch' file in the 'target' directory of the 'com.servoy.eclipse.core' project and selecting **Debug as > Servoy Launch**

Before Getting Started: 32 Or 64 Bit Installation

The easiest setup will be when both Servoy Developer and Eclipse operate under either 32 or 64 bit.

Servoy Developer: When installing Servoy Developer, the installer will automatically detect if it's being operated under a 32 or 64 bit Java Virtual Machine and will install a 32 or 64 bit version of Servoy accordingly. By making a certain JVM the default JVM before launching the cross platform Servoy installer or launching the cross platform Servoy installer using a specific JVM, one can control if the installer will install a 32 or 64 bit version of Servoy.

Eclipse: When downloading Eclipse distributions, there are specific downloads available for 32 or 64 bit systems. Note that for Windows, only limited Eclipse distributions offer a download for 64 bit Windows.

When mixing 32 and 64 bit, more configuration is required. This will be discussed under [Configuration > Setup the Target Platform](#).

Installing Servoy Developer

The installer for Servoy Developer can be downloaded after login in at <http://www.servoy.com/download>.

Running the installer will install Servoy Developer. For a more indepth description of installing Servoy Developer, see [Installing Servoy Developer](#).

After installation, start Servoy Developer and install the SQL Explorer plugin and close Servoy developer again. See [SQL Explorer integration](#) for more information.

Installing Eclipse with the Eclipse Plugin Development Tools

The open source Servoy projects are all plugins for the Eclipse environment, so in order to develop on one of the projects a Eclipse version with Eclipse plugin development tools (PDE) installed is required.

The Eclipse.org download page (<http://www.eclipse.org/downloads/>) lists a wide range of prepackaged Eclipse distributions. See <http://www.eclipse.org/downloads/packages/compare-packages> for the overview of what each Eclipse distribution contains.

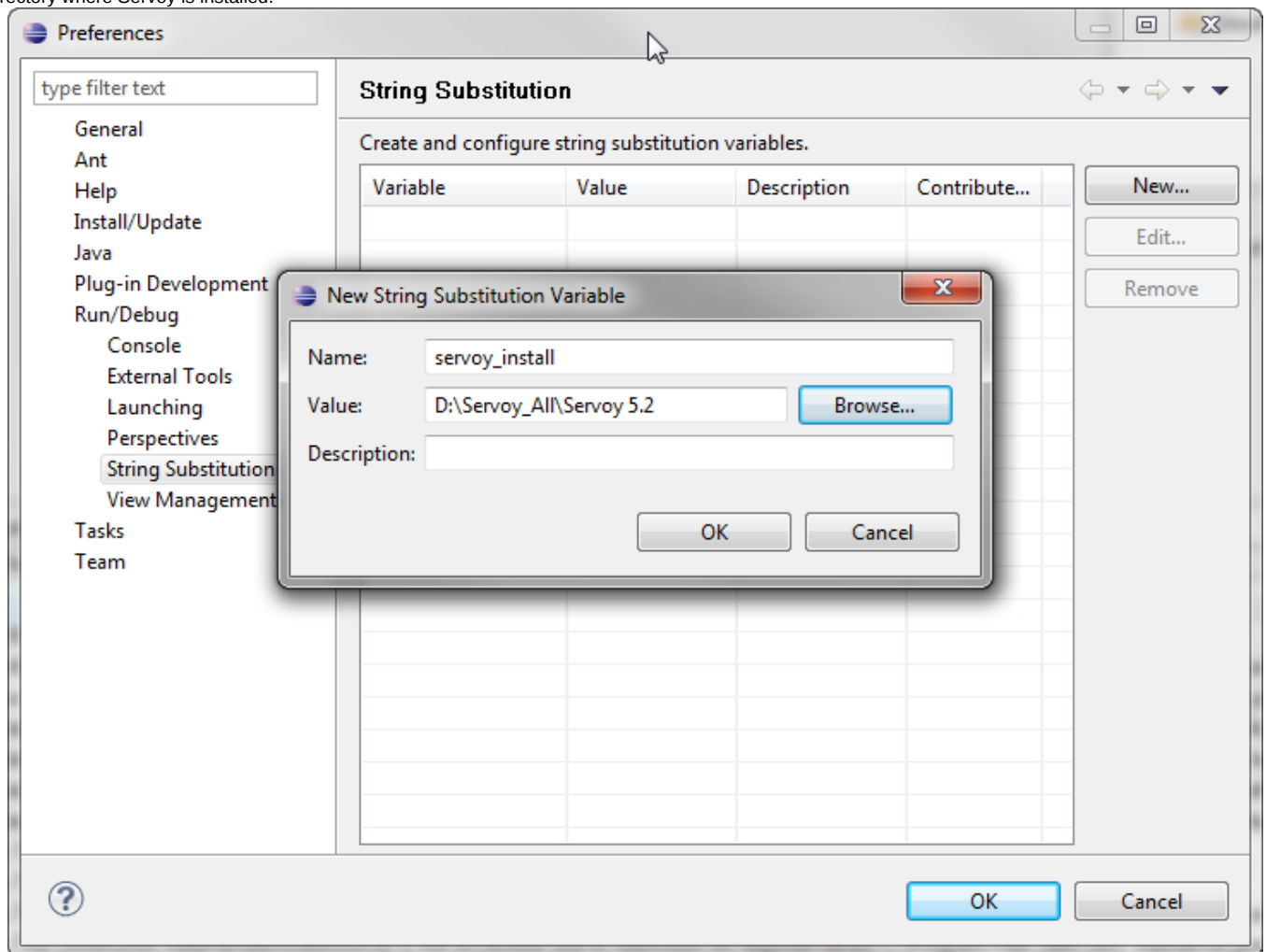
One of the offered distributions that includes PDE is the Eclipse Classic distribution. Although the plugin development tools can be installed in any Eclipse environment, the easiest is to download the 'Eclipse Classic' distribution and install it.

Configuration

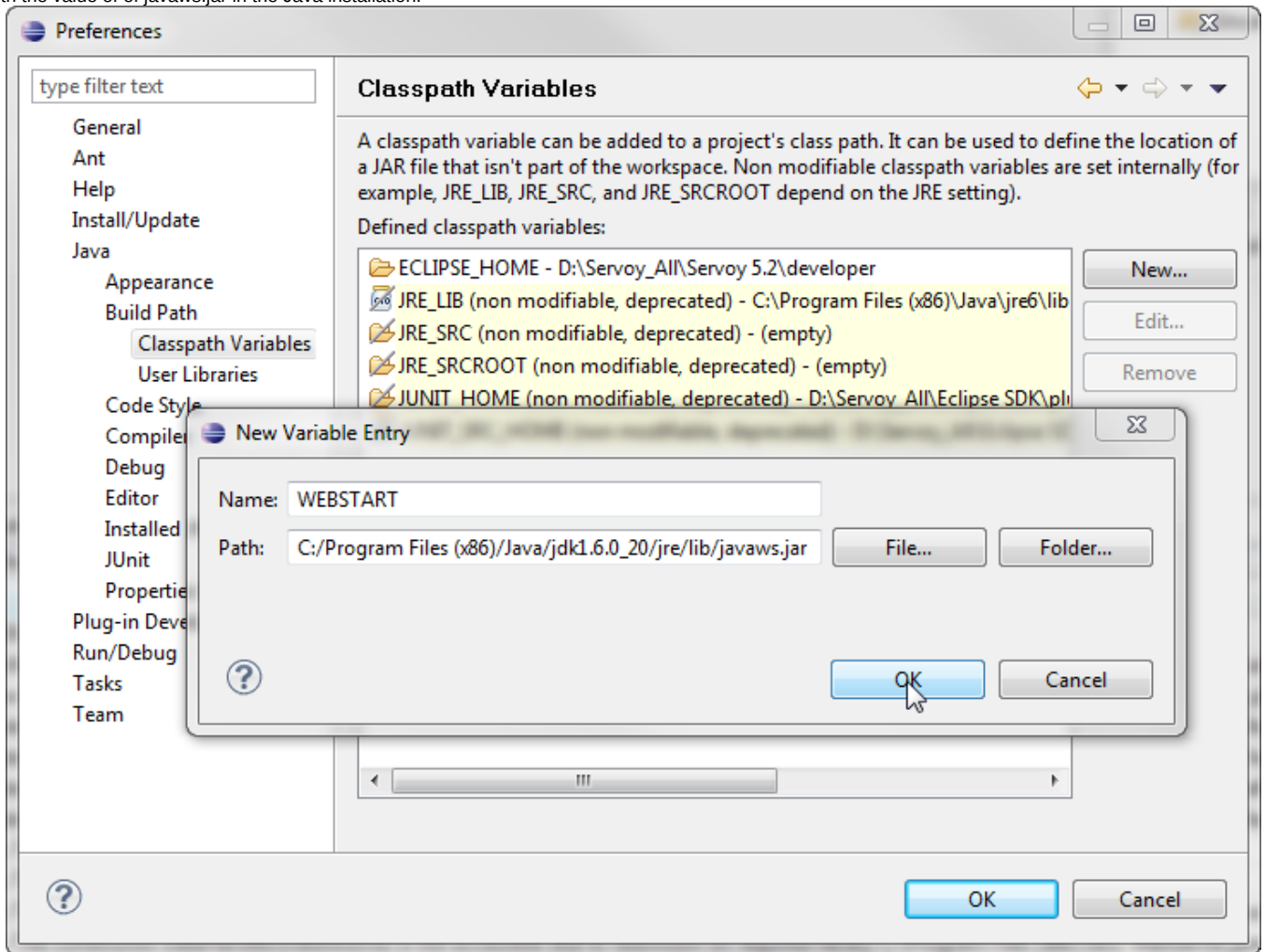
Start the downloaded and installed Eclipse installation and choose or create a workspace in which the development on the open source projects of Servoy will take place.

After startup, the following configuration needs to be done:

Setup a variable: Go to **Window > Preferences > Run/Debug > String Substitution** and create a new variable named 'servoy_install' with the value of the directory where Servoy is installed.



Create a classpath variable: Go to **Window > Preferences > Java > Build Path > Classpath Variables** and create a new variable named 'WEBSTART' with the value of javaws.jar in the Java installation.



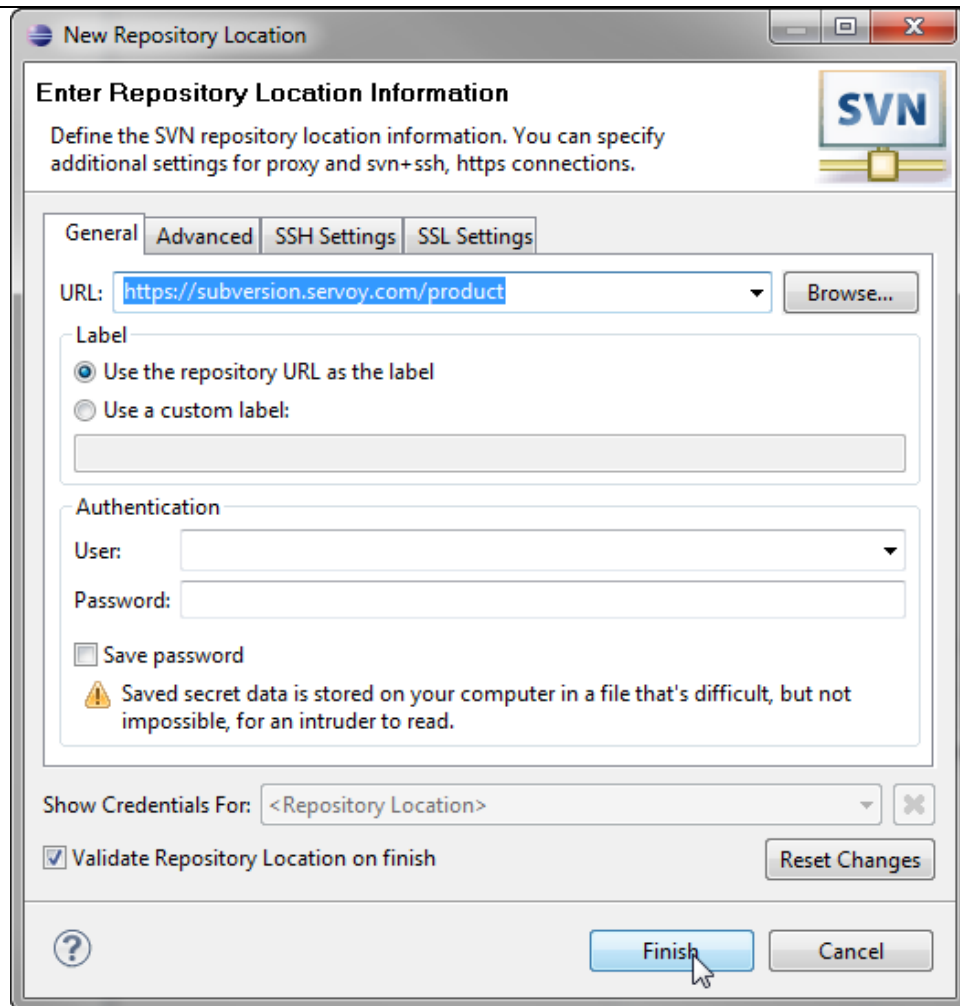
Setup a connection to the Servoy SVN:

- Open the SVN Perspective: **Window > Open Perspective > Other > SVN Repository Exploring**



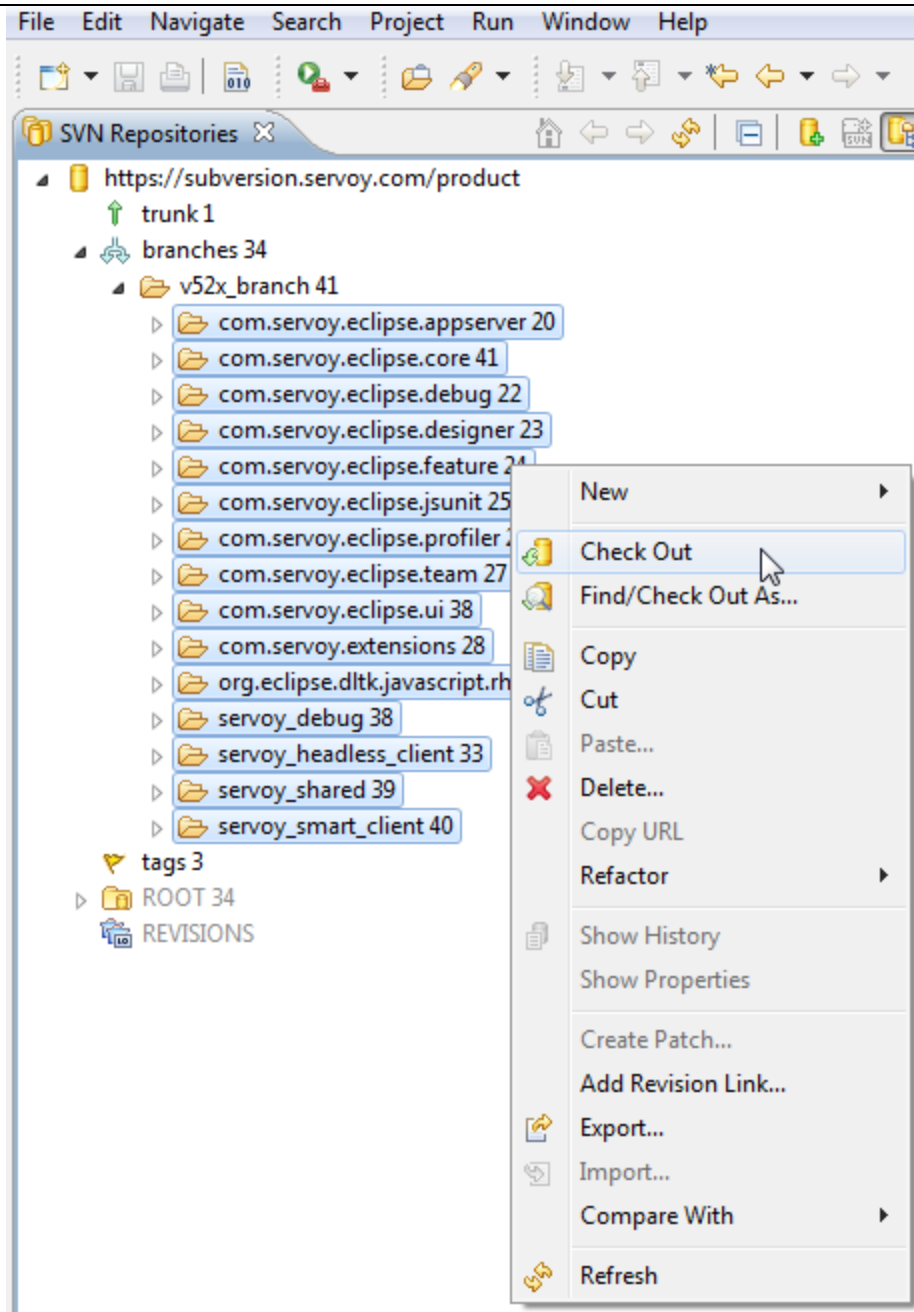
If the SVN Repository Exploring perspective is not available, most likely there is no SVN Team Provider installed. For more information on installing an SVN Team Provider see [SVN Team Provider](#)

- Add a new Repository Location: **File > New > Repository Location**
 - Specify the URL 'https://subversion.servoy.com/product' and click **Finish**



Checkout the Servoy projects:

- Expand the tree structure under the newly created Repository Location to the desired version to work on. Make sure to select the version of the projects that match the version of the previously installed Servoy Developer.
- Select all the projects
- Right-click one of the selected projects and select **Check Out** from the context menu



Setup the Target Platform: The Target Platform setting provides the Eclipse instance with information on the Servoy Developer installation, namely whether it's installed to run on a 32 or 64 bit Java virtual Machine (JVM). The checked out Servoy projects already contain the target setup files for different platform/JVM combinations:

1. Generic
2. Mac OSX
3. Mac OSX 64 bit
4. Windows 32 bit
5. Windows 64 bit
6. Linux
7. Linux 64 bit

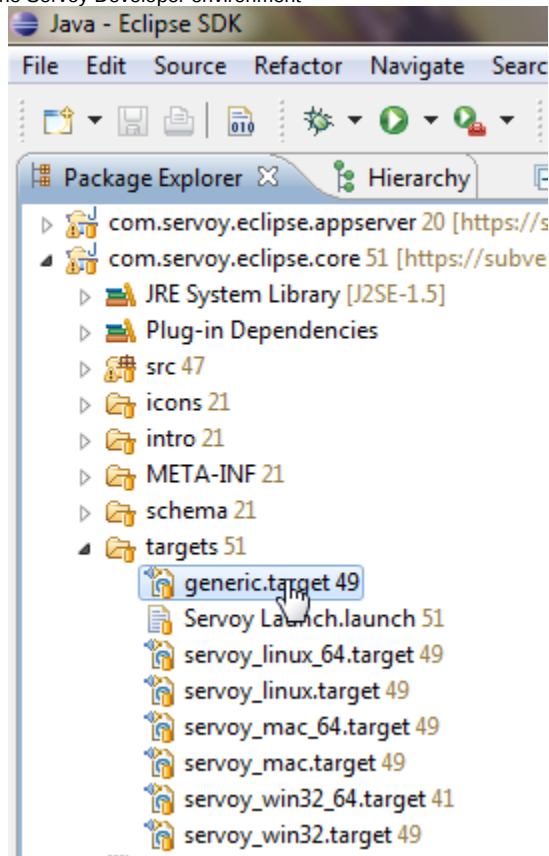


To check if Eclipse/Servoy Developer runs on a 32 or 64 bit JVM follow the following steps:

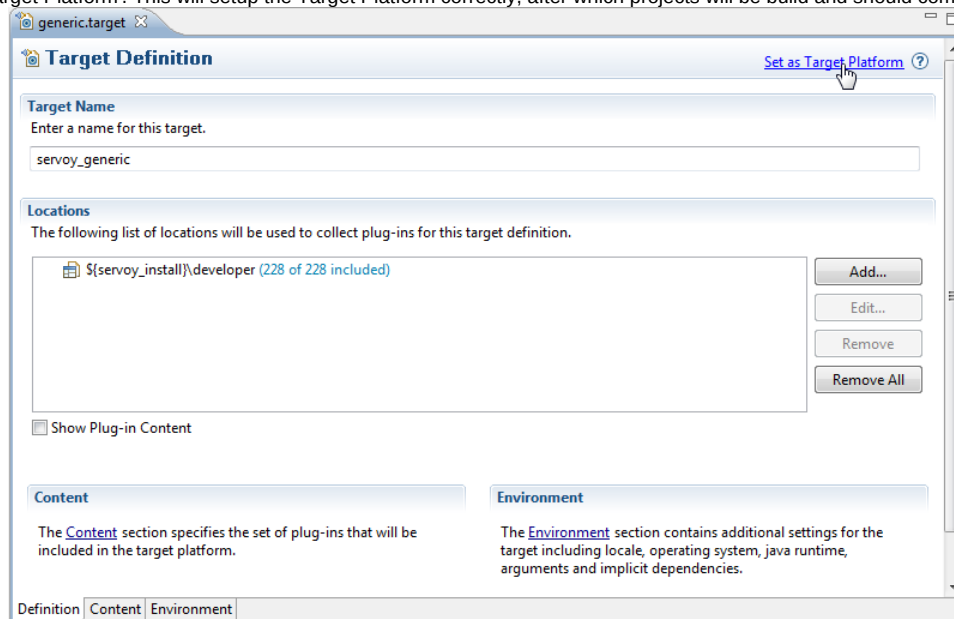
- Open Eclipse/Servoy Developer
- Go to **Help > About Eclipse/About Servoy Developer > Installation Details > Configuration** tab and look for the text '-arch'.
- If the text is followed by 'x86', Eclipse/Servoy Developer runs on 32 bit. If it is followed by 'x86_64', Eclipse/Servoy Developer runs on 64 bit

- Switch to the Java Perspective: **Window > Open Perspective > Java**
- Expand to `com.servoy.eclipse.core > target` in the **Package Explorer**
- Open (double-click) the correct .target file:
 - when Eclipse and Servoy Developer both run on either a 32 or 64 bit JVM, open the generic.target file

- otherwise, open the file that matches the Servoy Developer environment



- Click the 'Set as Target Platform'. This will setup the Target Platform correctly, after which projects will be build and should compile without error.



- When using a mixed setup (generic.target is NOT used), the following additional steps are required:
 - Add the JVM that is used by Servoy Developer to the Eclipse instance:
 - Open **Window > Preferences > Java > Installed JREs** and click **Add**
 - Select **Standard VM** and click **Next**
 - Select the directory of the JVM used by Servoy Developer as 'JRE home' and click **Finish**
 - Due to a bug in Eclipse: close the Preferences window
 - Open **Window > Preferences > Java > Installed JREs > Execution Environments**
 - Select the correct Execution Environment (the correct version of J2SE/JavaSE) and check the checkbox besides the name of the JRE under 'Compatible JREs' that was added in the previous step

Update Servoy Server libraries in the Servoy Developer installation: In order to also be able to start debug clients from Servoy Developer launched from the source code, it is required to use the correct version of several libraries of the Servoy Application Server.

Download the builds of the Servoy Server libraries from http://build.servoy.com:8080/latest/servoy_v52x_branch/ who's name postfix matched the value of the releaseNumber variable in the ClientVersion class of the checked-out source code and place them in {servoyInstall}/developer/plugins. Remove any other version of the same file if it exists.

Creating a Launch Configuration: Last step in the setup of the environment is the creation of a Launch Configuration. The checked out open source Servoy projects already contain a pre-configured Launch Configuration:

- Expand to com.servoy.eclipse.core > target in the **Package Explorer**
- Right-click the 'Servoy Launch.launch' file and select **Debug As > Servoy Launch**
The steps above will install the configuration contained in the 'Servoy Launch.launch' file as a launch configuration and launch a Servoy Developer as a debug session

When starting a debug session, Servoy Developer will launch with settings of the Launch Configuration. By default, for example the workspace that is opened is set to /workspace, relative to the Servoy Developer installation.

To edit the launch settings, open **Run > Debug Configurations > Eclipse Application > Servoy Launch** and edit the settings.

Ready: After completing the steps described above, the Eclipse installation is setup correctly to launch Servoy Developer in a debug session:

- **Run > Debug** (or F11)