Setting Up

Servoy Developer 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 the open source Servoy projects, the following is required:

1. A Servoy Developer installation
2. The Servoy Server libraries
3. A separate Eclipse environment with the Eclipse plugin development tools installed
4. Configuration of the Eclipse installation

Servoy 5.2 branch instructions

These instructions are for trunk and any future branches, but are not relevant for the Servoy 5.2 branch. For instructions how to setup for Servoy 5.2 development, see Setting Up (Servoy 5.2 branch only)

In This Chapter

Quick Start for the Impatient

- Install Servoy Developer
- 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 (latest 7xx branch) from the repositories at https://github.com/Servoy/
    - servoy-osgi-target
    - servoy-eclipse
    - servoy-eclipse-tomcat
    - servoy-js-engine
    - servoy-extensions
    - servoy-client
    - servoy-mobile
- Download the Servoy Server libraries from http://build.servoy.com:8080/latest and place them in the 'libs_extra' directory of the eclipse_target project. Download only the files who's name postfix matches the value of the release number in the ClientVersion class in the checked-out source code
- Setup a Target Platform by:
    - Opening the servoy.target file, located in the 'eclipse_target' project
    - Make sure the entry for '${workspace_loc}/eclipse_target/libs_extra' display '(x of x included)', where x is equal to the number of libraries (.jar files) in the libs_extra directory. If not, force a refresh by selecting the entry, clicking **Edit** > **Finish** and then Control-S to save the editor
    - 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' (or 'Servoy Launch_mac.launch' when on OSX) file in the 'launch_files' directory of the 'com.servoy.eclipse.core' project and selecting **Debug as** > **Servoy Launch** (or **Servoy Launch_mac** when on OSX).

Installing Servoy Developer

An installation of Servoy Developer is required in order to be able to launch Servoy Developer from source code, because the source depends on the directory structure that is present in a Servoy Developer installation.

The installer for Servoy Developer can be downloaded at http://www.servoy.com/download. Running the installer will install Servoy Developer. For a more indepth description of installing Servoy Developer, see Download & Install Servoy Developer.

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 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.

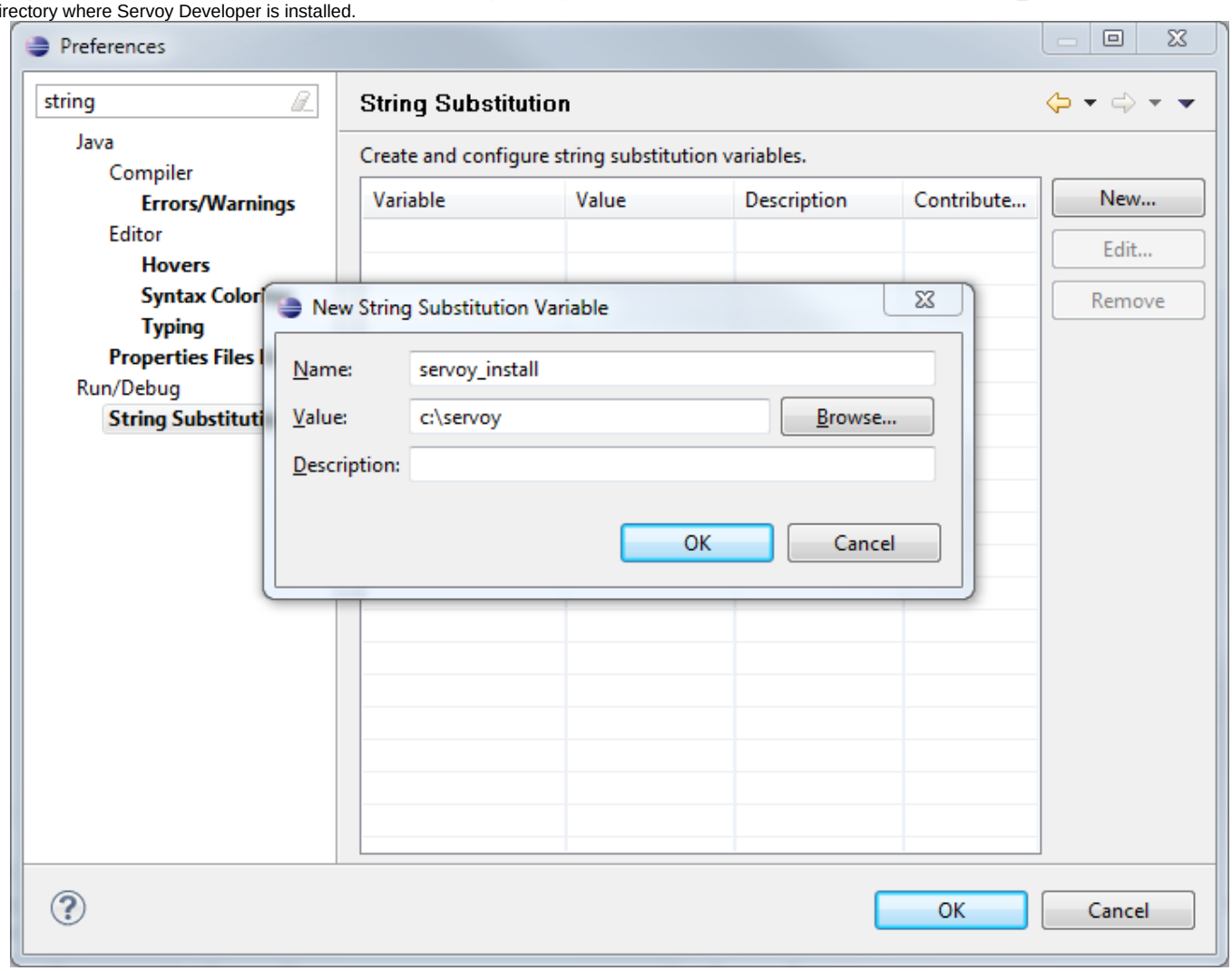
NOTE: if you want to run the Servoy mobile client, then you also need to install the Google plugin for Eclipse from https://developers.google.com/eclipse/docs/download

Configuration

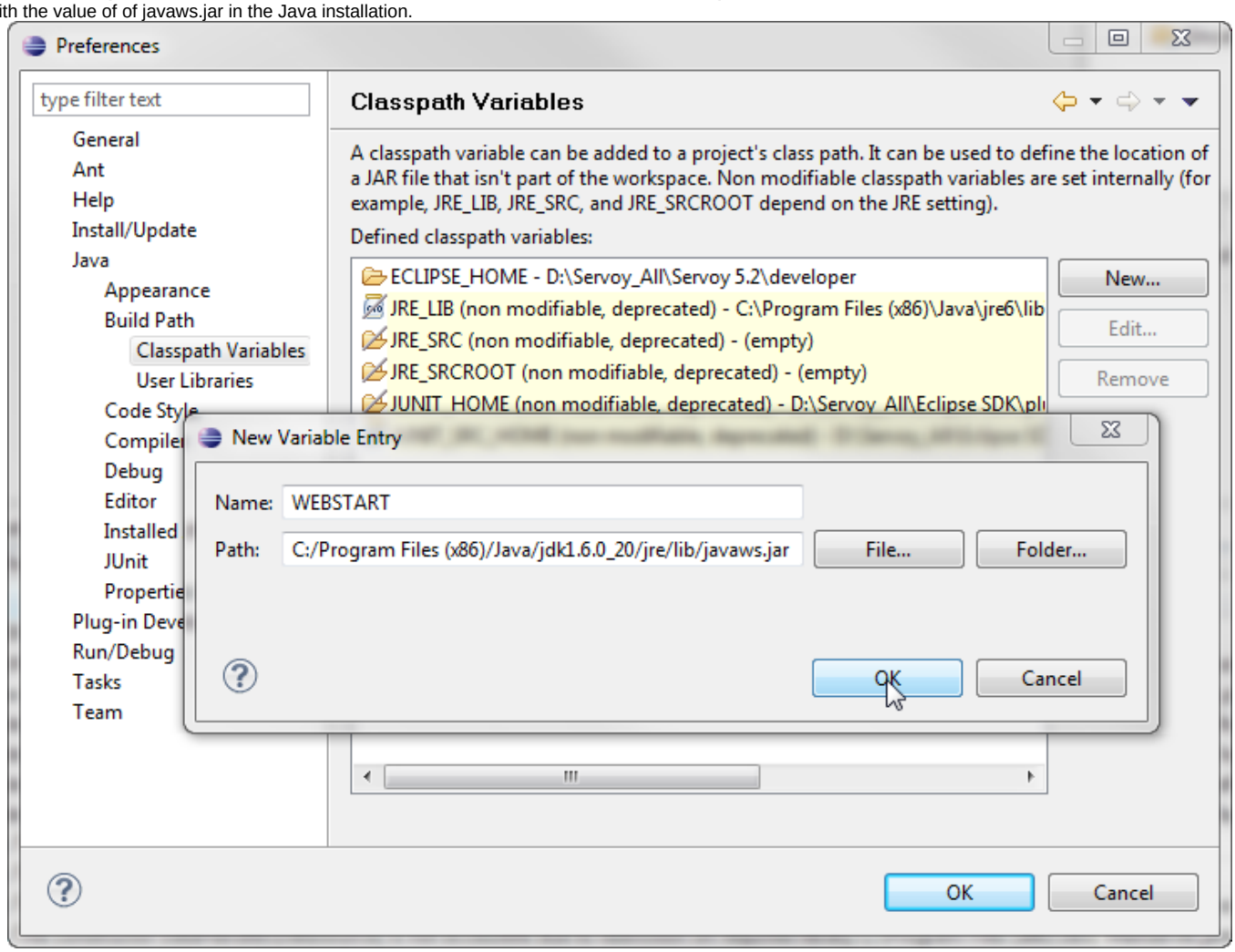
Start the downloaded and installed Eclipse installation and choose or create a workspace in which the development on the open source projects of Servoy will take place.

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 Developer 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 of javaws.jar in the Java installation.



Setup a connection to the Servoy source:

- Install the "egit" eclipse plugin
- Open the GIT Perspective: **Window** > **Open Perspective** > **Other** > **GIT**
- Use the GIT Repositories view to clone the open source Servoy projects (latest 7xx branch) from the repositories at https://github.com/Servoy/
    - servoy-osgi-target
    - servoy-eclipse
    - servoy-eclipse-tomcat
    - servoy-js-engine
    - servoy-extensions
    - servoy-client
    - servoy-mobile

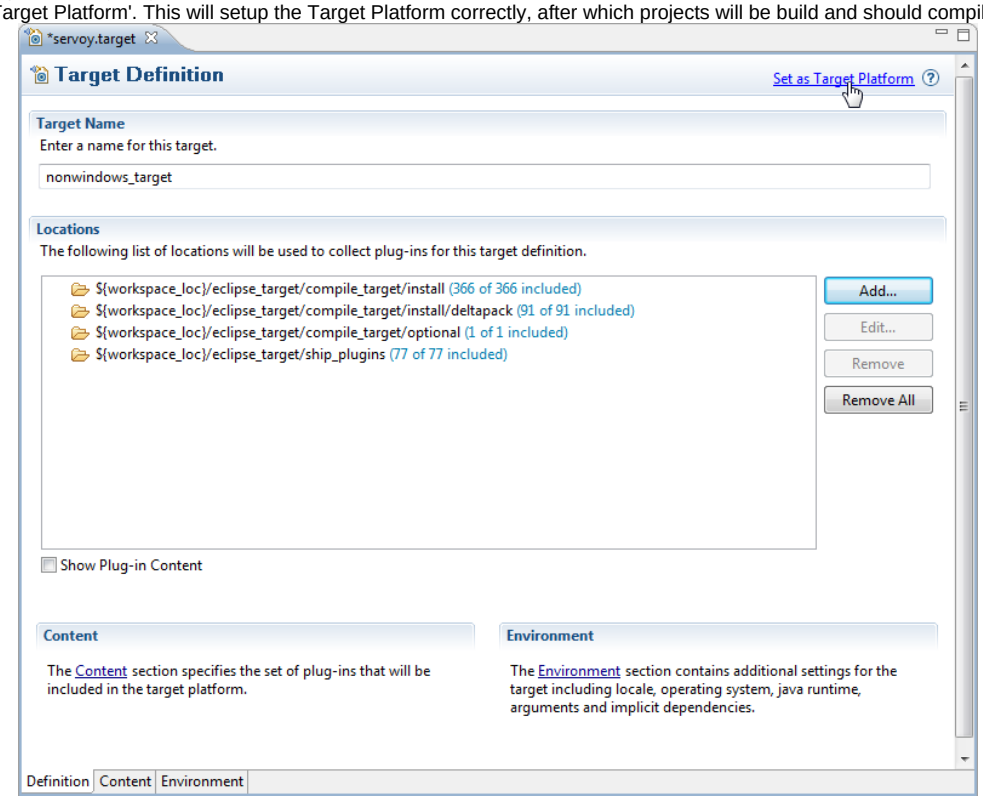
> Make sure to enable the checkbox "check out projects when clone is complete" in the GIT clone wizard

Download Servoy Server libraries
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. The latest version of these libraries can be downloaded from the Servoy build server, located at http://build.servoy.com:8080/latest. Download all the libraries from the subdirectory for a specific branch or for trunk 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 the 'libs_extra' directory of the 'eclipse_target' project

Download Servoy Server plugin for running mobile
Please download the latest version of the mobile and mobileservice plugins; these can also be downloaded from the Servoy build server, located at http://build.servoy.com:8080/latest. Download the plugins from the subdirectory for trunk (currently) who's name postfix matches the value of the releaseNumber variable in the ClientVersion class. Copy these over to the the `{servoyInstall}/application_server/plugins` directory without the postfix (overwrite existing mobile and mobileservice plugins if needed).

Setup the Target Platform:

- Switch to the Java Perspective: **Window** > **Open Perspective** > **Java**
- Expand the 'eclipse_target' project in the Package Explorer
- Open (double-click) the 'servoy.target' file
- Click the 'Set as Target Platform'. This will setup the Target Platform correctly, after which projects will be build and should compile without error.



Creating a Launch Configuration: Last step in the setup of the environment is the creation of a Launch Configuration.

- Expand to com.servoy.eclipse.core > launch_files 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 on OSX, add the -XstartOnFirstThread argument, to the VM Arguments tab in the launch configuration)

> **Upgrade notification on startup**
>
> When starting Servoy Developer from source code, it is most likely that a notification is shown indicating that the Servoy Application Server version should be upgraded. This message can be canceled and ignored.

When starting a debug session, Servoy Developer will launch with settings of the Launch Configuration. By default the workspace for example 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)

Optional Additional Configuration

*Adding other features to the target: *If additional features are required in the Servoy Developer that is launched from source code, for example an SVN Team Provider plugin, an easy way to achieve this is by adding the feature(s) to the target Definition.

- Make sure the feature is installed in a Eclipse installation, doesn't have to be a Servoy Developer installation, can also the the Eclipse SDK from which the development on the Servoy open source projects is being done.
- Open the 'servoy.target' file in teh root of the 'eclipse_target' project
- Click **Add** > **Features** > **Next**
- Point the location to an installation directory of Eclipse. Note that when using a Servoy Developer instance in this case, the `{servoyInstall}/developer` is the directory to point to.
- Check the feature(s) to be included and press **Finish**.
- Click the 'Set as Target Platform' link again.

Running Mobile Client From Sources

- Install GWT plugin for Eclipse
- Run 'build_mobile_in_developer' ant task from 'build.xml' of 'servoy_mobile' project (this will create the template war file)
- Refresh Workspace for Eclipse file system to see the war file
- Copy(Overwrite) mobile_xyz*.jar and mobileservice_xyz.jar from http://build.servoy.com:8080/latest to plugins folder of your Servoy install (where variable servoy_install points to). Download only the files who's name postfix matches the value of the release number in the ClientVersion class in the checked-out source code.
- Run Servoy Developer
- From Servoy Developer, export mobile solution to folder: `{myEclipseWorkspace}/servoy_mobile/war`
- Refresh 'servoy_mobile' project from Eclipse workspace
- From context menu of 'servoy_mobile' project run: **Debug as** > **Web Application** (choose **index.html** as html file)
- Activate service solution in Servoy Developer
- In Eclipse, from Development Mode view, open the provided url in Chrome browser
- If GWT plugin for Chrome is not installed you will be prompted to install it; afterwards the mobile solution will run, breakpoints can be put in java classes of 'servoy_mobile' project