Open Source - setting up development environment

Servoy Developer is a plugin for the Eclipse platform, so to develop the Servoy codebase an Eclipse SDK environment with Eclipse plug-in support is required.

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

- A Servoy Developer installation
- The Servoy Server libraries
- A separate Eclipse environment with the Eclipse plugin development tools installed
- Configuration of the Eclipse installation

Note: these instructions apply only to *master* branch. You can use this instructions as a guideline for other branches. For specific branch settings, please refer to the related branch documentation.

In This Chapter

Overview

- Install latest minor version of Java 8.
- Install Servoy Developer
- Install an Eclipse distribution that contains the plugin development support
- 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 (using either eGit eclipse plugin, git command line or your preferred git client) and import into your eclipse workspace the open source Servoy projects (master branch) from the following repositories at https://github.com/Servoy/
    - servoy-eclipse
    - servoy-eclipse-tomcat
    - servoy-js-engine
    - servoy-extensions
    - servoy-client
    - servoy-mobile
    - sablo (only the sablo project is needed)
- Download the Servoy Server libraries from https://build.servoy.com/latest/servoy_master/ and place them in the '**libs_extra**' directory of the *'* **launch_targets**' project. Download only the files who's name postfix matches the value of the release number from the **ClientVersion** class (checked-out as source code above).
- Setup a Target Platform:
    - open the **'eclipse.target**' file, located in the '**launch_targets**' project;
    - if you see some errors in the target editor for some target locations you might need to select those and click on the 'Update' button ('Reload' could help as well) - to make it see them correctly;
    - click the link 'Set as Target Platform' of the Target editor;
    - you might need to save the editor if it's marked as dirty.
- Make sources compile: get sablo mvn dependencies, ignore API baseline, run typescript compile.
- In order to add the documentation files, download the **servoy_update.zip** from https://build.servoy.com/latest/servoy_master/, unzip it and also unzip the '**com.servoy.eclipse.core**' plugin jar. Copy the xml files from the '**com/servoy/eclipse/core/doc**' folder to the same location of the '**com.servoy.eclipse.core**' project.
- 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).

Details

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 https://build.servoy.com/latest/servoy_master/servoy_installer.jar. Running the installer will install Servoy Developer. For a more in-depth 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 an Eclipse version with plugin development support is required.

The Eclipse.org download page (http://www.eclipse.org/downloads/) lists a wide range of prepackaged Eclipse distributions. See https://www.eclipse.org/downloads/packages/ for the overview of what each Eclipse distribution contains. An up to date (2019) version containing plugin support (Git support included with eGit) is Eclipse IDE for RCP and RAP Developers.

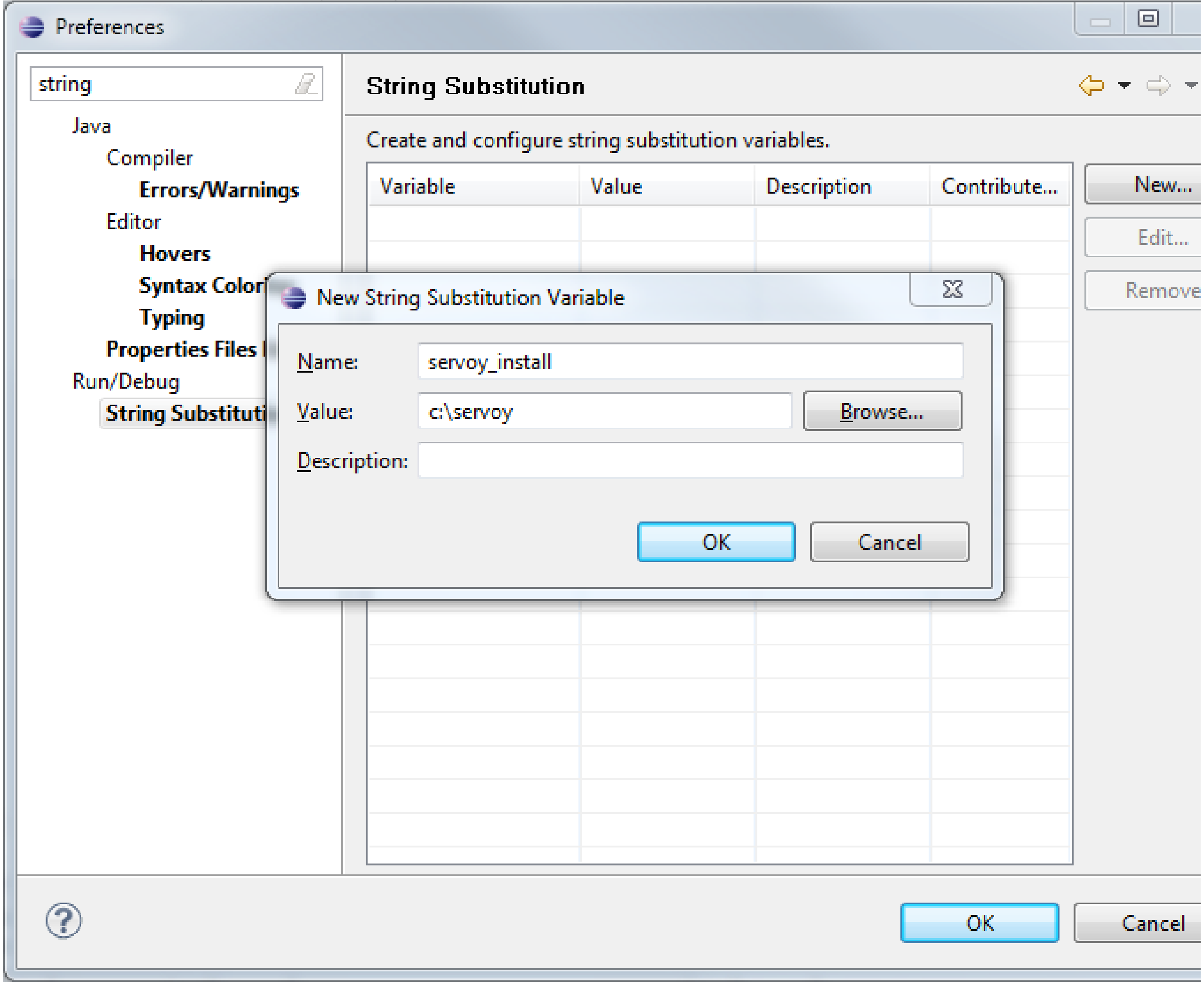
NOTE: if you want to run the Servoy mobile client, then you also need to install the GWT Eclipse plugin.

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:

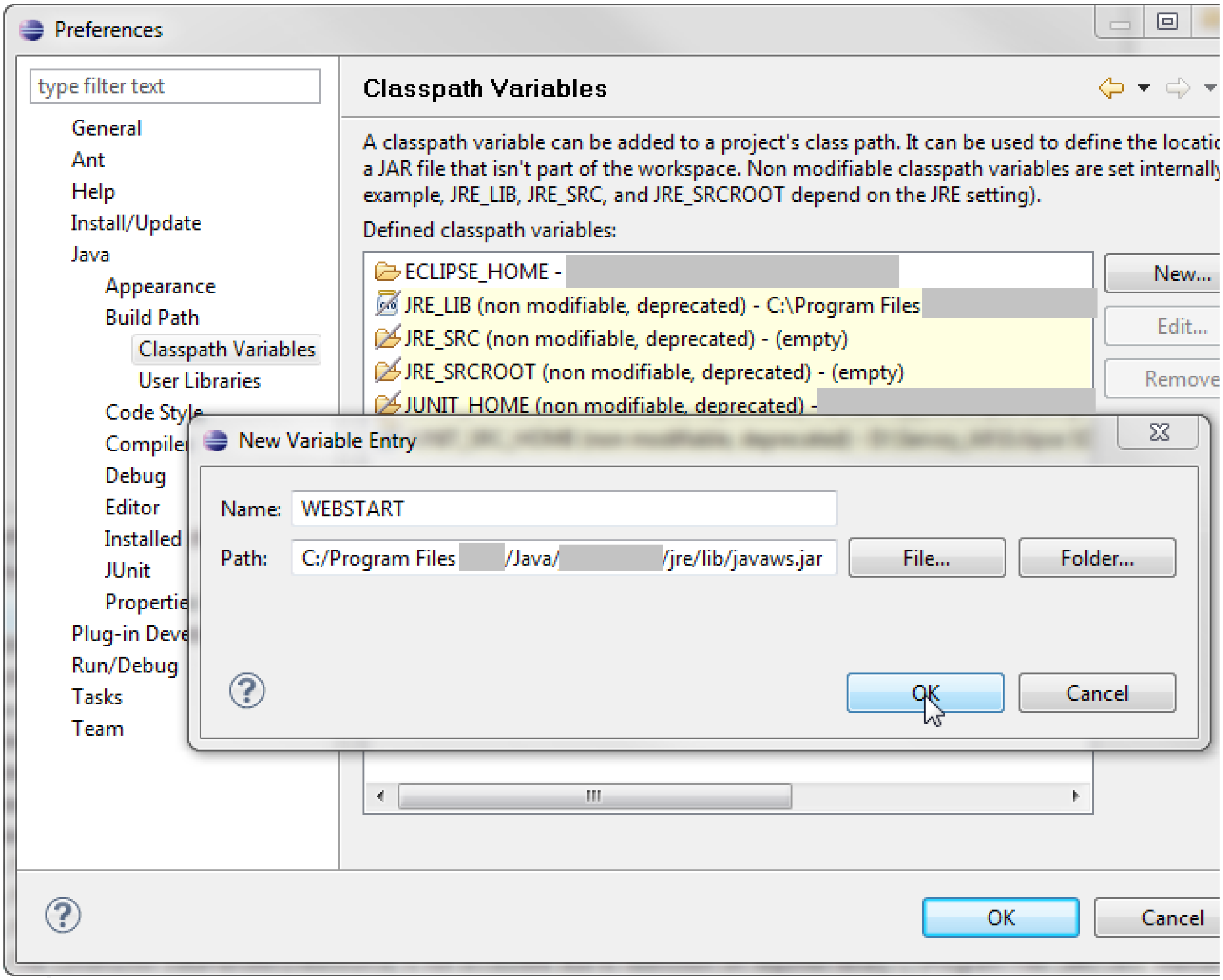
Setup a variable: Go to **Window** > **Preferences** > **Run/Debug** > **String Substitution** and create a new variable named 'servoy_install' with the value of the directory where Servoy Developer is installed.



Create and configure string substitution variables.

Name: servoy_install
Value: c:\servoy
Description:

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.



A classpath variable can be added to a project's class path. It can be used to define the locatic
a JAR file that isn't part of the workspace. Non modifiable classpath variables are set internally
example, JRE_LIB, JRE_SRC, and JRE_SRCROOT depend on the JRE setting).

Defined classpath variables:

- ECLIPSE_HOME - [redacted]
- JRE_LIB (non modifiable, deprecated) - C:\Program Files [redacted]
- JRE_SRC (non modifiable, deprecated) - (empty)
- JRE_SRCROOT (non modifiable, deprecated) - (empty)
- JUNIT_HOME (non modifiable, deprecated) - [redacted]

Name: WEBSTART

Path: C:/Program Files[redacted]/Java/[redacted]/jre/lib/javaws.jar

Setup a connection to the Servoy source:

- Install the "eGit" eclipse plugin (not needed if you have a recent Eclipse RCP already)
- Open the GIT Perspective: **Window** > **Open Perspective** > **Other** > **GIT**
- Use the GIT Repositories view to clone the open source Servoy projects (master branch) from the repositories at https://github.com/Servoy/
  - servoy-eclipse
  - servoy-eclipse-tomcat
  - servoy-js-engine
  - servoy-extensions
  - servoy-client
  - servoy-mobile
  - sablo (only the sablo project is needed)
- Import projects into eclipse workspace: uncheck the parent (root) folder; for *sablo* do not import the examples.

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 https://build.servoy.com/latest/servoy_master/. 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 '**lib s_extra**' directory of the '**eclipse_target**' project

For example, if in the ClientVersion class we have the following lines:

```
private static final int majorVersion = 2019;
private static final int middleVersion = 6;
private static final int minorVersion = 0;
private static final int releaseNumber = 3500;
private static final String versionPostfix = "rc";
```

then the following libraries need to be downloaded:

```
j2db_documentation-2019.6.0.3500_rc.jar
j2db_log4j-2019.6.0.3500_rc.jar
j2db_server-2019.6.0.3500_rc.jar
```

(...)

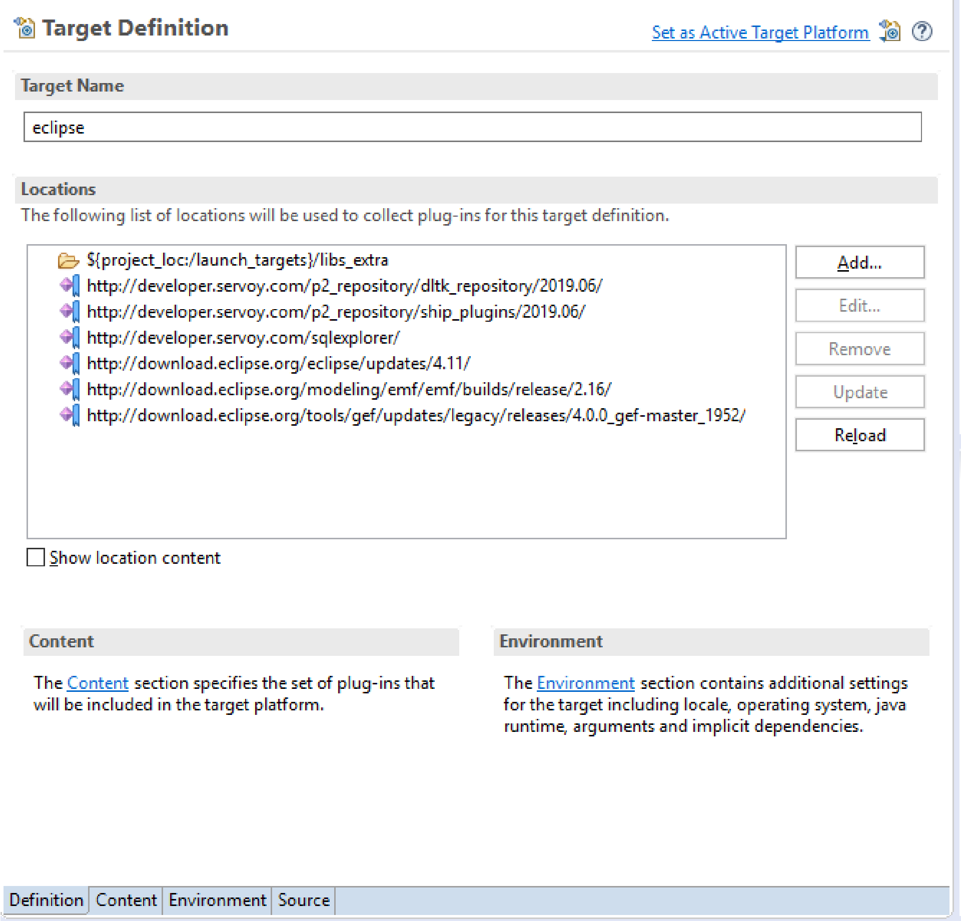
Download Servoy Server plugin for running mobile

Please download the latest version of the *mobile* and *mobileservice* plugins; these can also be downloaded from the Servoy build server, located at https://build.servoy.com/latest/servoy_master/. 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). Using the same example as above the following two libraries need to be downloaded:

mobile_2019.6.0.3500_rc.jar
mobileservice_2019.6.0.3500_rc.jar

Setup the Target Platform:

- Switch to the Java Perspective: **Window** > **Open Perspective** > **Java**
- Expand the '**launch_targets**' project in the Package Explorer
- Open (double-click) the '**eclipse.target**' file
- il you see some errors in the target editor for some target locations you might need to select those and click on the 'Update' button ('Reload' could help as well) - to make it see them correctly
- Click the '**Set as Target Platform**'. This will setup the Target Platform correctly; you will still have some compile errors probably due to sablo project's maven dependencies (handled below).
- you might need to save the editor if it's marked as dirty.



Target Name: eclipse

The following list of locations will be used to collect plug-ins for this target definition.

${project_loc:/launch_targets}/libs_extra
http://developer.servoy.com/p2_repository/dltk_repository/2019.06/
http://developer.servoy.com/p2_repository/ship_plugins/2019.06/
http://developer.servoy.com/sqlexplorer/
http://download.eclipse.org/eclipse/updates/4.11/
http://download.eclipse.org/modeling/emf/emf/builds/release/2.16/
http://download.eclipse.org/tools/gef/updates/legacy/releases/4.0.0_gef-master_1952/

☐ Show location content

The Content section specifies the set of plug-ins that will be included in the target platform.

The Environment section contains additional settings for the target including locale, operating system, java runtime, arguments and implicit dependencies.

In order to successfully run servoy developer from sources, a file must be created in the maven-settings-path (c:\Users\username\.m2\): toolchains.xml. This file must contain the JDK home folder (The JDK from the class patch variable **WEBSTART**).

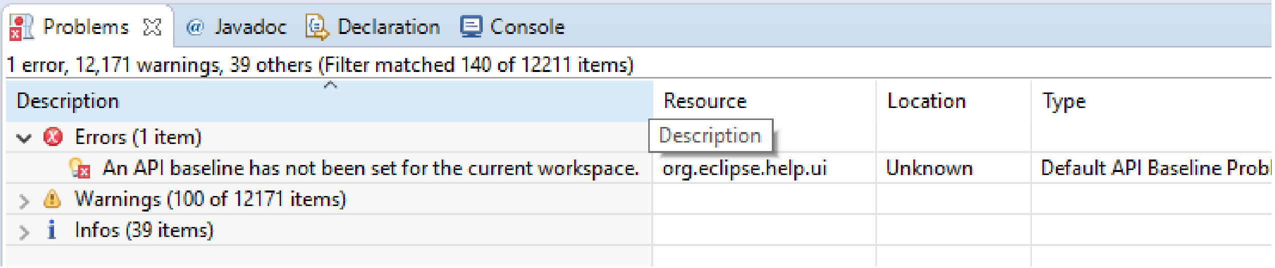
Example of toolchains.xml:

```
<toolchains>
  <!-- JDK toolchains -->
  <toolchain>
       <type>jdk</type>
       <provides>
       <id>JavaSE-1.8</id>
       </provides>
       <configuration>
       <jdkHome>c:/Program Files/Java/jdk1.8.0_201</jdkHome>
       </configuration>
  </toolchain>
</toolchains>
```

Get sablo project's maven dependencies

- run **mvn clean install -Ptarget** in servoy-eclipse project
- run '**mvn clean install**' in '**sablo**' project of 'sablo' repository from command line (or if you want to do it from eclipse, right click the pom.xml file from sablo project and run the same maven goals). Although eclipse does have maven integration that would handle all of that if sablo had the maven project nature, there are some problems with that because sablo is also an eclipse plugin - and if we set the maven nature as well some other things go wrong with the build structure. That is why this manual step is need.

NOTE: if an error message related to API baseline is displayed



1 error, 12,171 warnings, 39 others (Filter matched 140 of 12211 items)

Description	Resource	Location	Type
Errors (1 item)			
An API baseline has not been set for the current workspace.	org.eclipse.help.ui	Unknown	Default API Baseline Probl
Warnings (100 of 12171 items)			
Infos (39 items)			

the error must be ignored from the API Baseline preferences page.

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)

NOTE: when running Servoy Developer from sources it is need to compile typescript files for sablo and servoy-ngclient projects else "Form" objects will not be dislayed. From the command line this can be done using npm, by running the following commands (next to tsconfig.json file):

- npm install typescript --save-dev
- npm run build

ⓘ 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 from which the development on the Servoy open source projects is being done.
- Open the '**eclipse.target**' file in the root of the '**launch_targets**' 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.
- In the launcher configuration plugins tab you have to check the new plugins that are needed.

Note that a separate target file is already provided for eGit support in source-launched Servoy Developer.

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 https://build.servoy.com/latest/servoy_master/ to plugins folder of your Servoy installation (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