**How to get OpenXML4J**

Version 0.1 – Julien Chable (Wygwam)

Table of Contents

[2. Introduction 2](#_Toc170528211)

[3. Get binaries 2](#_Toc170528212)

[4. Get the sources and package OpenXML4J with TortoiseSVN and Ant 2](#_Toc170528213)

[Step 1: Install software 2](#_Toc170528214)

[TortoiseSVN 2](#_Toc170528215)

[Ant 2](#_Toc170528216)

[Step 2: Get the sources 3](#_Toc170528217)

[Step 3: Compile the sources and package OpenXML4J with Ant 4](#_Toc170528218)

[Step 4: Open project in Eclipse and configuration (optional) 6](#_Toc170528219)

[5. Get and compile the sources with Eclipse 9](#_Toc170528220)

[Step 1: Install software 9](#_Toc170528221)

[Step 2: Get OpenXML4J sources and compile 9](#_Toc170528222)

[Step 3: Project configuration 12](#_Toc170528223)

# Introduction

To use OpenXML4J in your project you need the binaries. You can either download the binaries directly at sourceforge.net or use tools to get the latest version of the source and compile it by yourself.

# Get binaries

To get the binaries, go to sourceforge.net and download the latest version at: <https://sourceforge.net/project/showfiles.php?group_id=175435>

# Get the sources and package OpenXML4J with TortoiseSVN and Ant

We use SVN for the control version system. In order to get the sources (latest, old build, ...) you have to use a tool that use the SVN protocol to extract files on the SVN server.

## Step 1: Install software

Here are some information about tools that you can use to build OpenXML4J library.

### TortoiseSVN

One of the most common SVN tool is TortoiseSVN. It’s a very reliable tool and a Free software (under GPL licence), so you can get and use it freely. To download TortoiseSVN go to: <http://tortoisesvn.tigris.org/>.

TortoiseSVN is a shell based tool so all the features are placed in the shell of your menu in the explorer.

To install the software just follow the onscreen instructions.

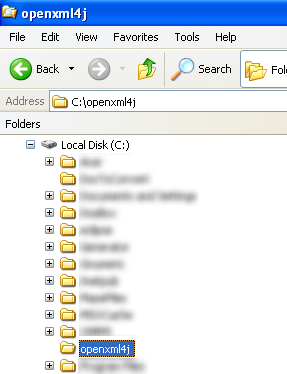
Caution : to install and use TortoiseSVN under Windows Vista, please use version 1.3 or later.

### Ant

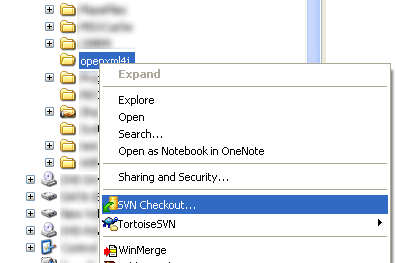
Download (latest version) and configure Ant as it’s described at <http://ant.apache.org/manual/install.html>.

## Step 2: Get the sources

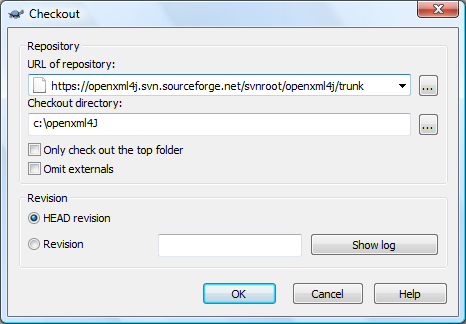
1. Create a folder where you want to store the openxml4j sources. For example c:\openxml4j :



1. Once, you’ve created your repository folder – and install successfully TortoiseSVN –, right click on the folder and select ‘SVN Checkout’ :



1. Enter the URL of openxml4j repository (<https://openxml4j.svn.sourceforge.net/svnroot/openxml4j/trunk>) :

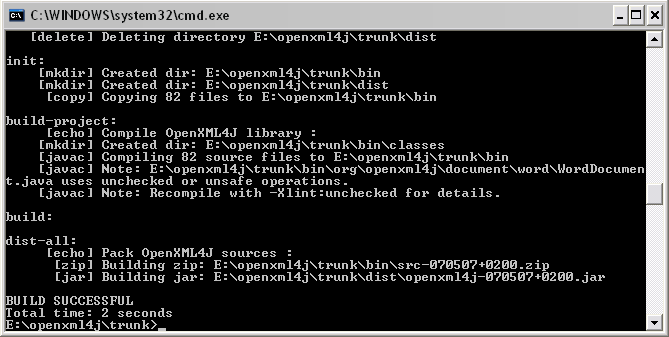


1. Click ‘OK’ and TortoiseSVN will download the openxml4j source repository.
2. For more detail about the use of TortoiseSVN, please refer to the following page : <http://tortoisesvn.net/support>

## Step 3: Compile the sources and package OpenXML4J with Ant

Once you’ve got the sources, you’ll be able to compile and package OpenXML4J with Ant. We’ll use the *build.xml* script located at the repository trunk root.

1. Make sure you have properly configured Ant (!).
2. Open a command shell and go into the OpenXML4J directory, you should find a *build.xml* script there.
3. Depending on how you want build, enter the following command :
   1. Only compile sources (no JAR packaging) : *ant –f build.xml build*
   2. Compile and package the library into a JAR file: *ant –f build.xml dist* (dist is optional since it’s the default target).
   3. Compile and package the library (with sources include) into a JAR file: *ant –f build.xml dist-all*
4. The console should be display a message like this :



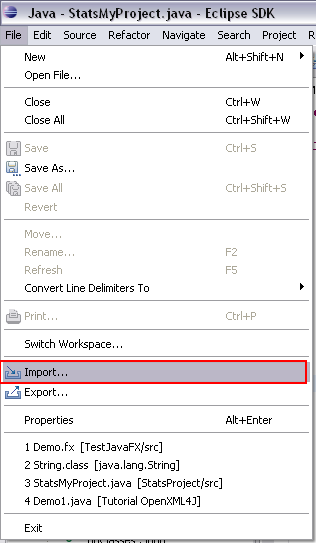
Here is a summary of the available Ant tasks :

|  |  |
| --- | --- |
| Task name | Description |
| build | Compile sources |
| dist | Compile sources and package binaries only |
| dist-all | Compile sources + tests, package binaries with a source zip archive inside |
| javadoc | Build the javadoc documentation |

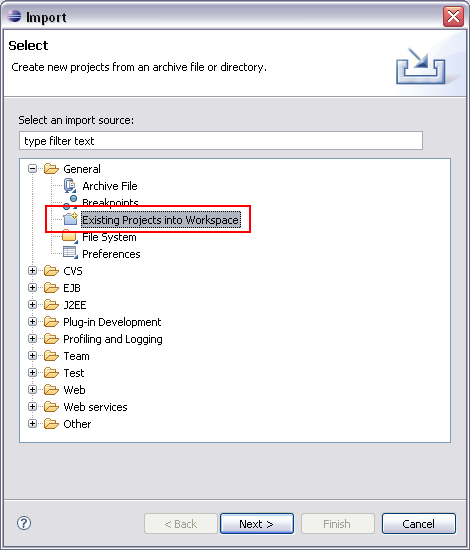
## Step 4: Open project in Eclipse and configuration (optional)

If you want or have to work with the OpenXML4J sources, you can open the Eclipse project file located at the *src* directory root. Here is the procedure to import the project into your workspace:

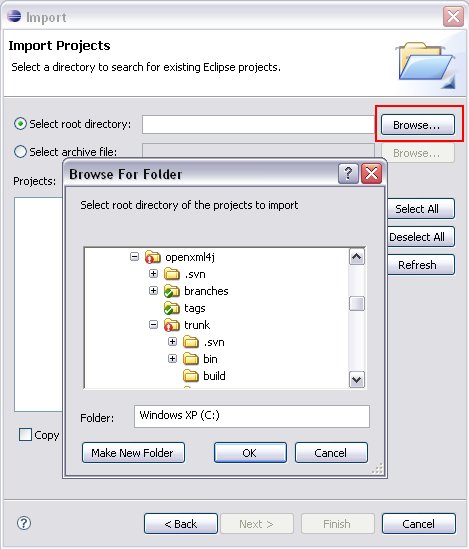
1. Select *File* 🡪 *Import* :

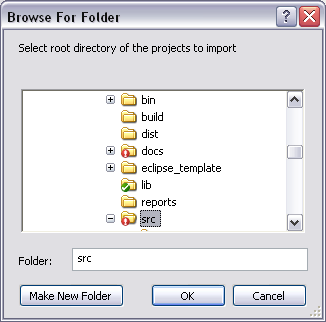


1. Then select *General* 🡪 *Existing Projects into Workspace* and click *Next*:

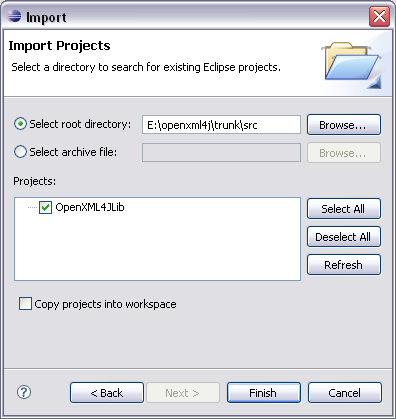


1. Select the openxml4j project source directory (‘src’ directory located where you’ve checkout the project) :

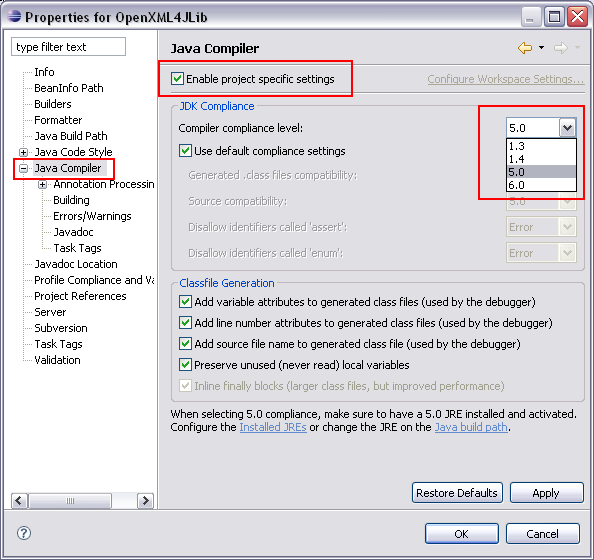




1. The OpenXML4JLib should appear in the projects list :



1. Then click Finish to add OpenXML4J library project to your current workspace.
2. Depending on your checkout directory location, you may have to modify the build path to correctly setup the libraries use by OpenXML4J. To do that, please follow this procedure :
   1. Right click on the project and select *Properties* (at the bottom),
   2. Select the Libraries tab, and edit all the depending external libraries to match the correct path:
      1. dom4j-1.6.1.jar
      2. jaxen-1.1.jar
      3. log4j-1.2.14.jar
      4. xmlunit1.0.jar
   3. Click *Ok*
3. If Eclipse can’t compile class files: if you’ve correctly set up the libraries paths, the other trouble you can encountered is that the Eclipse compiler compliance level is not correct. OpenXML4J is a Java 5 project (a 1.4 port may be done if several people really need it), to enable your compiler to match this version, you have to properly configure it in the Java compiler option:
   1. Right click on the project and select *Properties* (at the bottom),
   2. In the properties window, check Enable project specific settings and select at least a 5.0 compiler compliance level:



* 1. Click OK, normally Eclipse (if the Build Automatically option is enable) will compile the entire project and all have to be fine after this manipulation!

# Get and compile the sources with Eclipse

If you want to contribute to OpenXML4J project, this the way to configure your working environment.

## Step 1: Install software

First, you have to install Java SDK and Eclipse IDE which is the tools used by developers to make, build and test OpenXML4J :

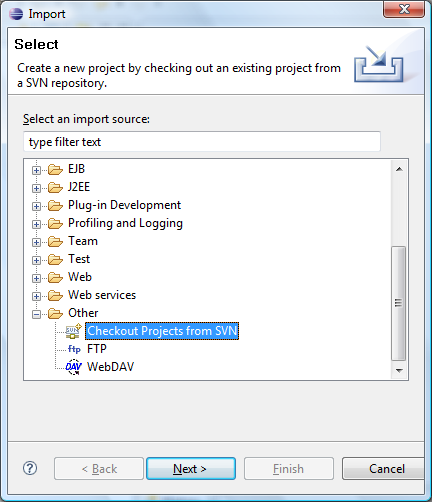
1. Install JDK prior to any other download (Java 5 or upper) : <http://java.sun.com/javase/downloads/index.jsp>
2. Go to the official Eclipse download section : <http://www.eclipse.org/downloads/> :
   1. Download the latest Eclipse version,
   2. Install Eclipse by just unzipping the archive on your hard drive,
   3. Launch Eclipse by executing the *eclipse.exe* launcher at the root of the folder you’ve just uncompressed
3. Once you’ve downloaded and install Eclipse, you have to download a SVN plugin in order to enable Eclipse to manage SVN repositories inside the IDE:
   1. Go to the following page : <http://subclipse.tigris.org/install.html>
   2. Follow the instructions to install SVN Eclipse plugin.

## Step 2: Get OpenXML4J sources and compile

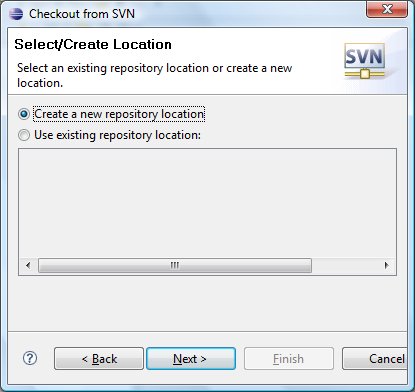
Note : Subclipse plugin must be install.

These are the steps to follow in order to get the sources from the SVN repository, to build and test OpenXML4J in Eclipse:

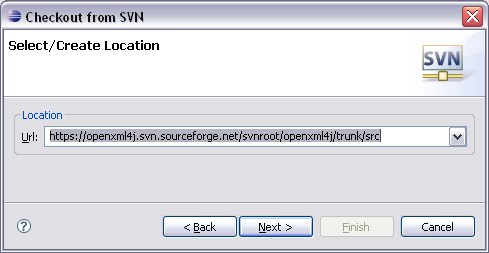
1. Import the project (*File* 🡪 *Import* 🡪 select ‘*Other’* or *‘SVN'Checkout Projects from SVN ...*’ in the tree):



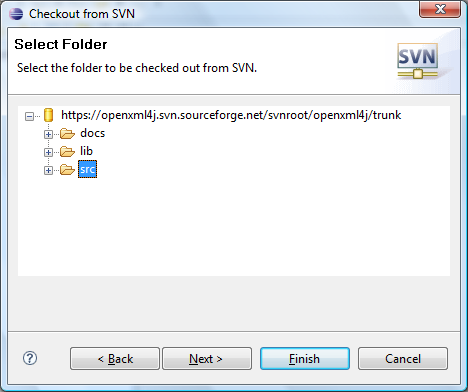
1. Depending of your SVN configuration :
   1. **If the OpenXML4J repository is not in the list**, select ‘Create a new repository location’, then click Next:



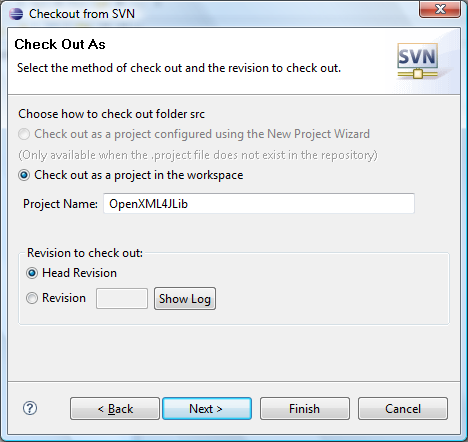
* + 1. Enter the following repository location [*https://openxml4j.svn.sourceforge.net/svnroot/openxml4j/trunk*](https://openxml4j.svn.sourceforge.net/svnroot/openxml4j/trunk), and click Next:



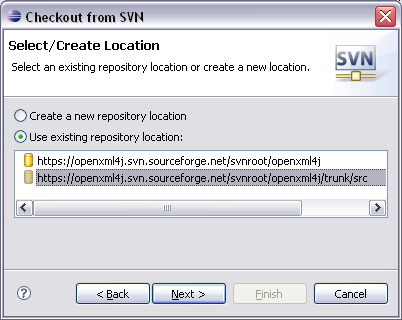
* + 1. After a while, Eclipse will display a window asking you to select the folder to checkout. Select the ‘src’ folder and click Next:



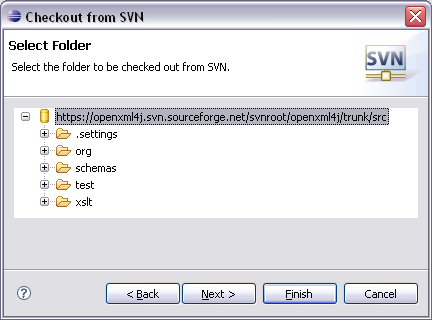
* + 1. Once Subclipse get the remote project info, make sure you select the following information in the next window before clicking Next (if you want to change your default workspace, the location of your checkout projects) or Finish:



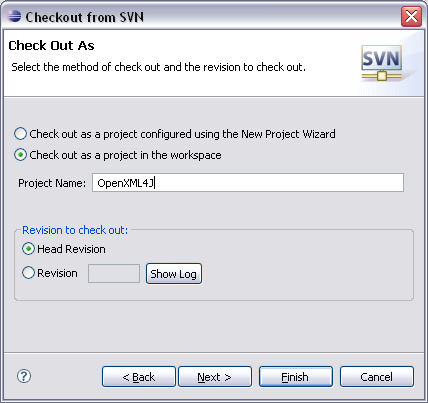
* 1. **If the OpenXML4J repository is already in the list**, select the repository and click Next.



* + 1. Select the *trunk/src* directory and click Next



* + 1. Select the ‘*Check out as project in the workspace’* option, and enter a project name. Take care to check the *Head Revision* option in the *Revision to check out* frame :  
       Note : If you want to check out the project and follow the *New Project Wizard* to configure the project, select the first option *Check out as a project configure using the New Project Wizard*.



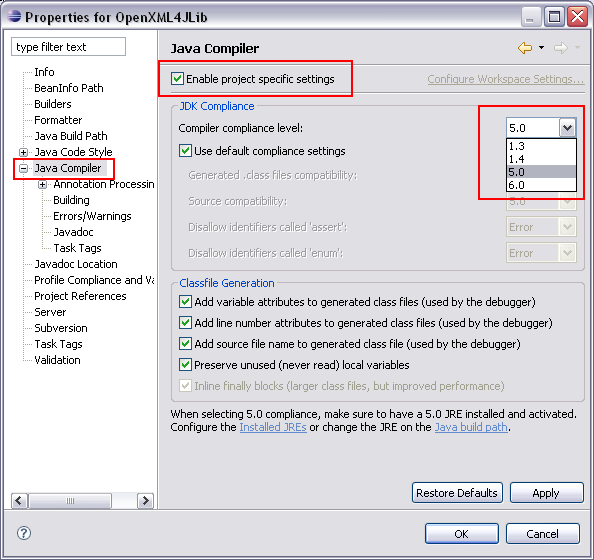
* + 1. Select your workspace (use default if you don’t know what it is), then click Finish or Next (if you want to configure your default workspace location).

## Step 3: Project configuration

Now, you have to properly configure the build path by adding the library dependencies to the project and resolve some potential path issues.

This step may be optional depending on how your environment is configure, however if you have a compilation or building issue then follow these few steps below:

1. Depending on your checkout directory location, you may have to modify the build path to correctly setup the libraries use by OpenXML4J. To do that, please follow this procedure :
   1. Right click on the project and select *Properties* (at the bottom),
   2. Select the Libraries tab, and edit all the depending external libraries to match the correct path:
      1. dom4j-1.6.1.jar
      2. jaxen-1.1.jar
      3. log4j-1.2.14.jar
      4. xmlunit1.0.jar
   3. Click *Ok*
2. If Eclipse can’t compile class files: if you’ve correctly set up the libraries paths, the other trouble you can encountered is that the Eclipse compiler compliance level is not correct. OpenXML4J is a Java 5 project (a 1.4 port may be done if several people really need it), to enable your compiler to match this version, you have to properly configure it in the Java compiler option:
   1. Right click on the project and select *Properties* (at the bottom),
   2. In the properties window, check Enable project specific settings and select at least a 5.0 compiler compliance level:



* 1. Click OK, normally Eclipse (if the Build Automatically option is enable) will compile the entire project and all have to be fine after this manipulation!
  2. If you still have problem, you can drop a post on the project forum or on the mailing list.