chem-bla-ics

Making Bioclipse Development easier: the New Manager Wizard

Egon Willighagen

Published August 13, 2009

Citation

Willighagen, E. (2009). Making Bioclipse Development easier: the New Manager Wizard. In *chem-bla-ics*. chem-bla-ics. https://doi.org/10.59350/g2qh1-n7q51

Keywords

Eclipse, Bioclipse

Abstract

Today, Jonathan, Carl, Arvid and I made writing managers for Bioclipse a bit easier. Plug-in development Eclipse in itself is already tricky to learn, and the use of Spring by the Bioclipse managers is not helping. And because very soon two new people will be starting with writing a new manager rather soon, we thought it was time to lower the activation barrier a bit.

chem-bla-ics

Today, Jonathan, Carl, Arvid and I made writing managers for Bioclipse a bit easier. Plug-in development Eclipse in itself is already tricky to learn, and the use of Spring by the Bioclipse managers is not helping. And because very soon two new people will be starting with writing a new manager rather soon, we thought it was time to lower the activation barrier a bit.

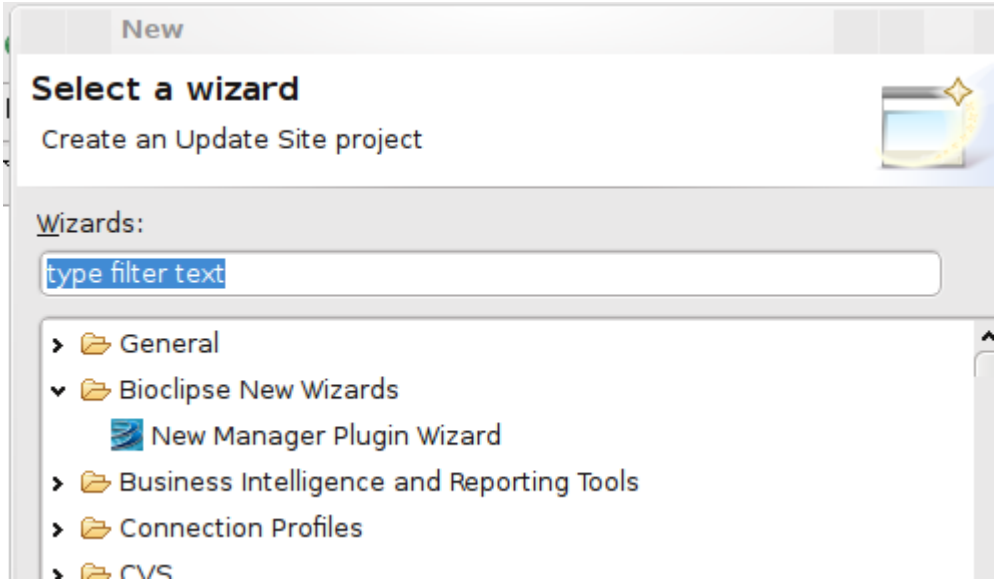
The basic file structure of an Bioclipse manager looks like:

```
net.bioclipse.foo/
|--META-INF
|   |--MANIFEST.MF
|   `-- spring
|       `-- context.xml
|-- plugin.xml
|-- .classpath
|-- .project
|-- build.properties
`-- src
    `-- net
        `-- bioclipse
            `-- foo
                |-- Activator.java
                `-- business
                    |-- FooManager.java
                    |-- FooManagerFactory.java
                    |-- IFooManager.java
                    |-- IJavaFooManager.java
                    `-- IJavaScriptFooManager.java
```

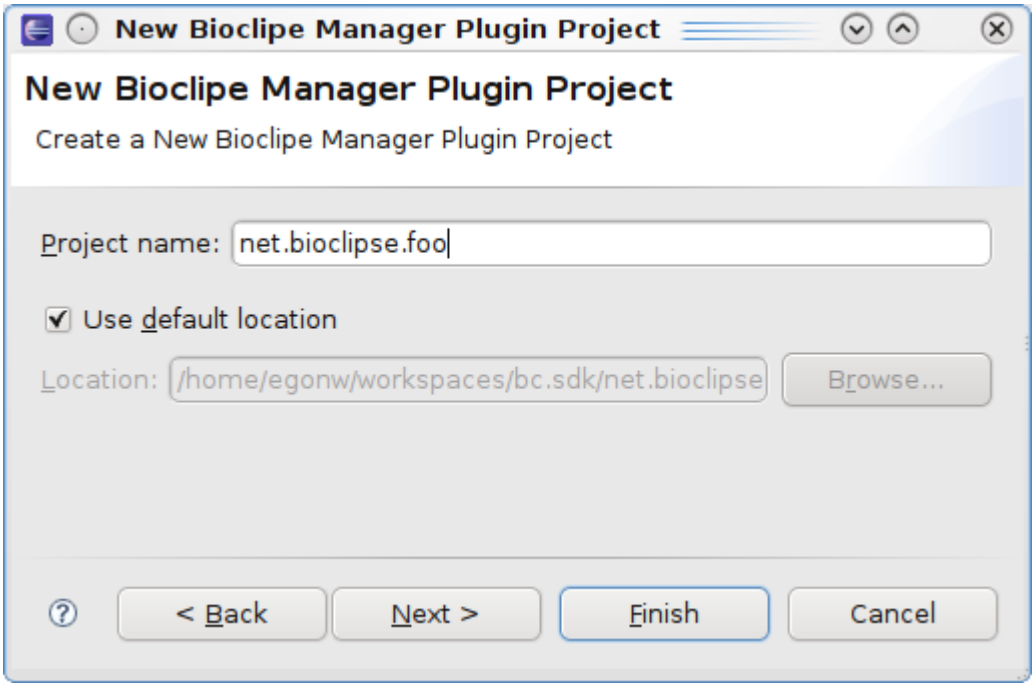
That is ***twelve*** files which need to be just right. I used to copy/paste from an earlier (simple) manager.

But we know and understand that setting up this framework is even more challenging if you have not done this at least 10 times before. So, today we implemented a *New Wizard* (source available from this Git repository: bioclipse.sdk).

chem-bla-ics

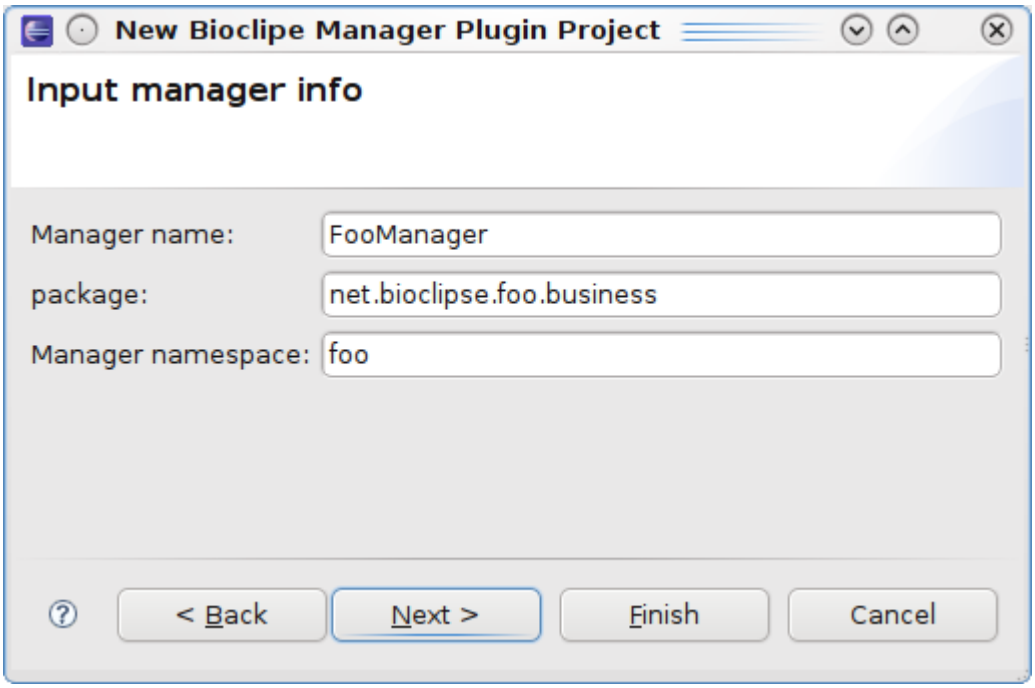


It just asks you a project name:



and a few other settings:

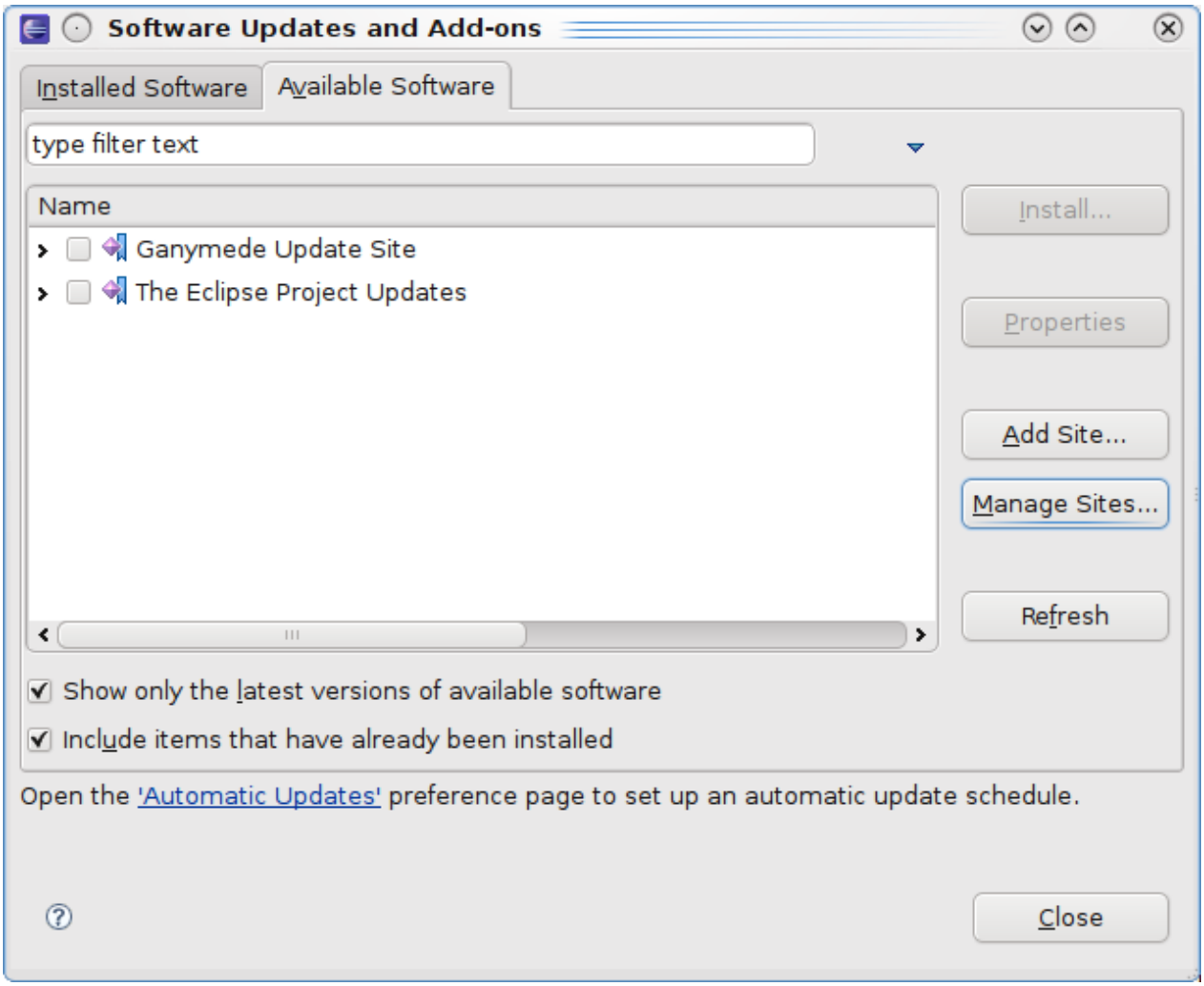
chem-bla-ics



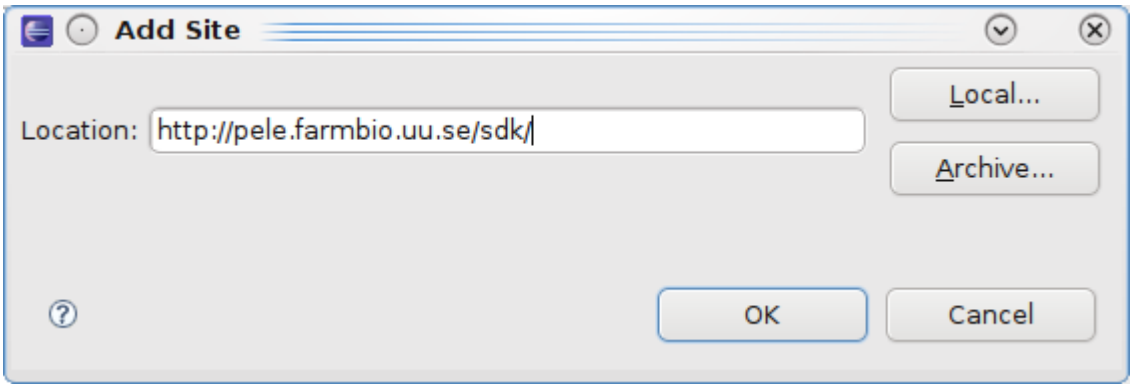
Installing the Bioclipse SDK

Installing this new plugin is fairly easy, and we have set up an *Update Site* at http://pele.farmbio.uu.se/sdk/. Just add this as Update site in Eclipse 3.4.x (which is still required for Bioclipse2). It depends on the JDT and PDE, which you will likely already have installed being part of the default Eclipse RCP release.

Go to the *Software Updates* in the *Help* menu:

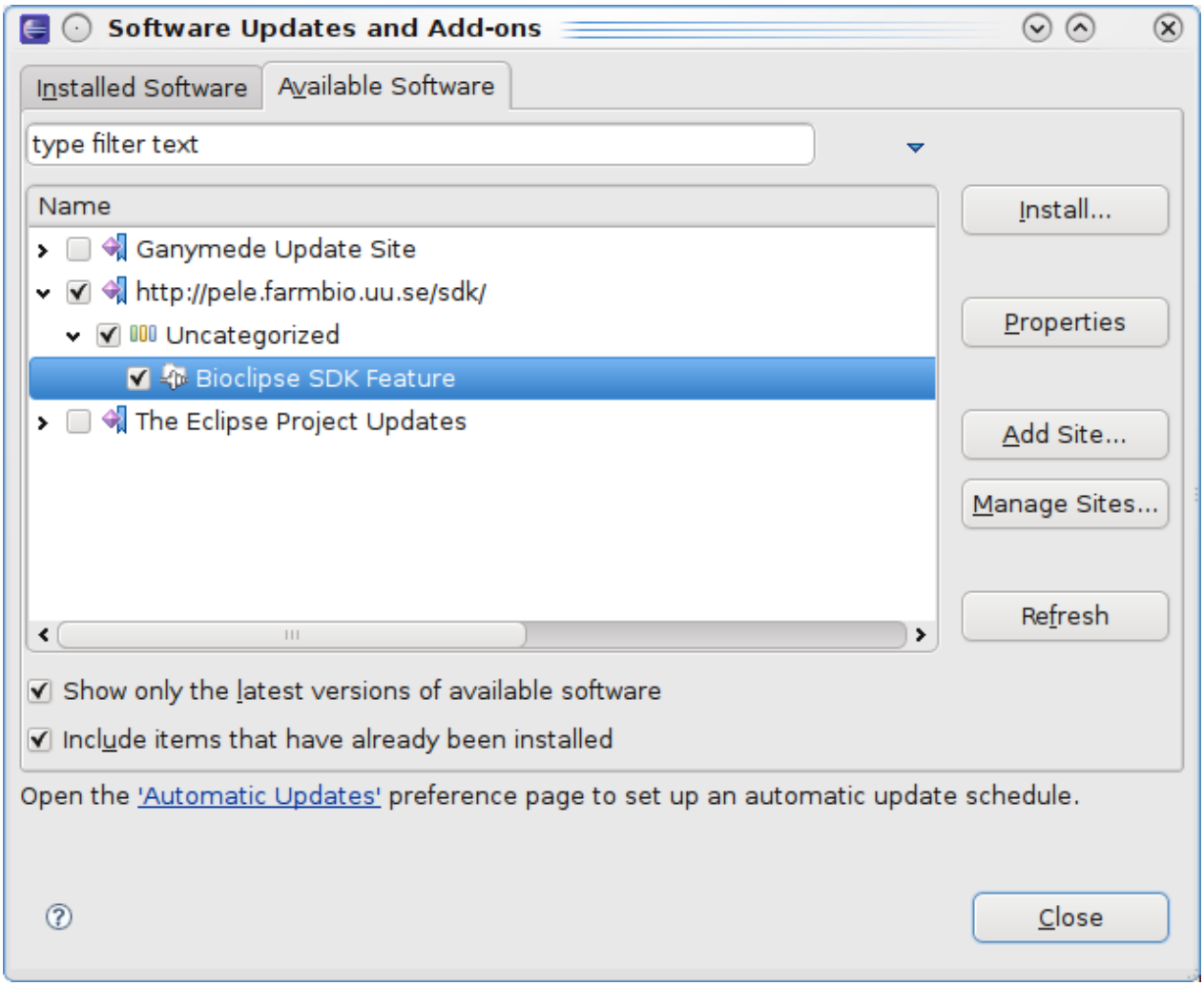


and pick *Add Site...*. Enter the aforementioned update site as shown here:

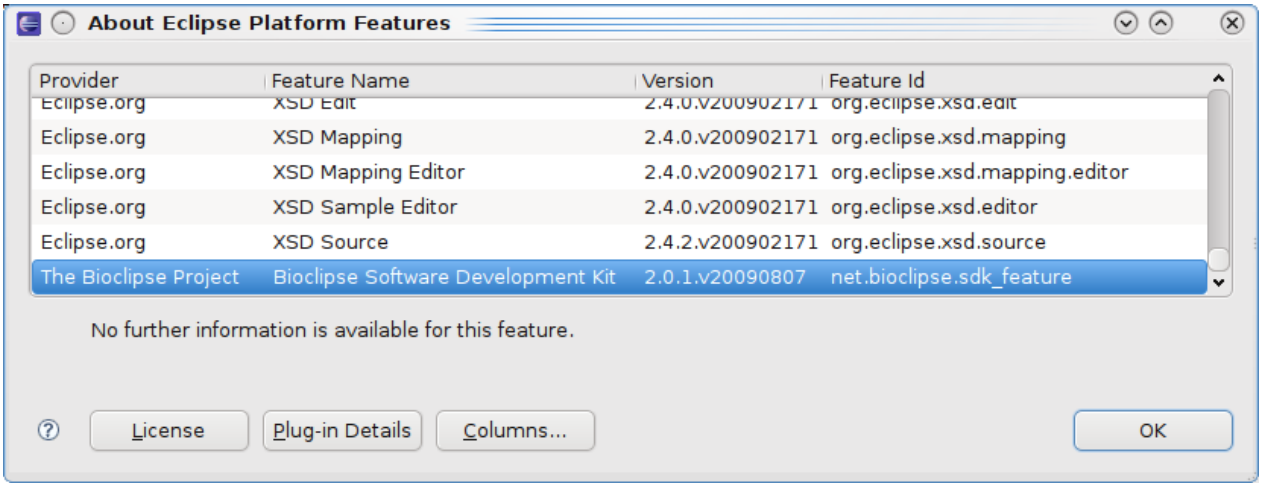


Then, select the Bioclipse plugin:

chem-bla-ics



After you hit *Install* and Eclipse install the fews tens of kBs of the plugin, the plugin should show up in your installation, like it did in mine:



Implementation Details

Writing the plugin was a challenge to me, and I am happy we were doing this in a hackaton. The Bioclipse-QSAR project already had a New Project wizard, but not for a new Plug-in Project. Some things are just slightly different then. For example, it turned out that creating a

chem-bla-ics

`.classpath` cannot be done in the regular way (it never showed up), and I had to dig up some internal code of the PDE. Actually, our current implementation is still using a few internal classes because of this:

```
IClasspathEntry[] entries = new IClasspathEntry[3];
String executionEnvironment = null;
ClasspathComputer.setComplianceOptions(
  project,
  ExecutionEnvironmentAnalyzer.getCompliance(executionEnvironment)
);
entries[0] = ClasspathComputer.createJREEntry(executionEnvironment);
entries[1] = ClasspathComputer.createContainerEntry();
IPath path = project.getProject().getFullPath().append("src/");
entries[2] = JavaCore.newSourceEntry(path);
```

Ideas are most welcome on how to clean up this code, and not make it use internal, non-exported classes. For the Java source files and even the `MANIFEST.MF` we are using templates, though I have seen this file being created programmatically too.

I'm sure we'll run in some needed plumbing here and there, but that's what update sites are for, not? *Release soon, release often* is an Open Source concept that works well in the Eclipse world.