


A Brief Overview of Building at Eclipse

Contents
1 Summary
2 Build Server
2.1 Hudson
3 Tycho Build - Current way
4 The Eclipse Platform Build
5 PDE Build - the old way
5.1 Ant
5.1.1 Plugin Builder plugin
6 Versioning
6.1 Information
6.2 Tooling
7 Multi-platform builds
7.1 Preferred way of doing multi-platform builds
7.2 Explanation of why it works
7.3 <i>What if you really want the traditional Delta Pack zip file?</i>
7.4 Old way of using the "DeltaPack"
7.4.1 What was the DeltaPack
7.4.2 Using the "DeltaPack"
7.4.3 When do you need it
7.4.4 How to get it
7.4.5 How to add it to your target platform

Summary

This page is about building Eclipse- and RCP-based products (including Eclipse itself). This document is focused on using **Tycho Build** (<https://www.eclipse.org/tycho/>) directly by Eclipse projects. Tycho is part of a **Common Build Infrastructure** (http://wiki.eclipse.org/Common_Build_Infrastructure).

Build Server

 **(/File:Warning2.png)** while the following paragraph is technically accurate, it is no longer the recommended way to do things. The webmasters eventually want to get rid of "shell accounts" and have everyone do things through Hudson.

Eclipse projects can use the Eclipse Foundation's build server (build.eclipse.org) to run automated builds. If you're a committer with a shell account, you should already have access to the server. If not, your project lead should send a note to **the Eclipse Webmaster** (<mailto:webmaster@eclipse.org>). Once you have access to the build.eclipse.org server, you can move your build scripts to the 'shared' directory (i.e. /shared/[top-level project]/[project]/[component]). For example, the build scripts for the aforementioned IDE4EDU are run from the /shared/technology/soc/ide4edu/releeng directory. If this directory doesn't exist for your project, please ask the webmaster to create it for you.

Hudson

For more information, please see **Hudson** (**/Hudson**).

Tycho Build - Current way

Tycho (<https://www.eclipse.org/tycho/>) is The Way to headlessly build Eclipse- and RCP-based products.

The Eclipse Platform Build

See **The Eclipse Platform Build with Tycho and Maven**. (**/Platform-releeng/Platform_Build**)

PDE Build - the old way

PDE Build was The Way to headlessly build Eclipse- and RCP-based products. The **online documentation** (http://help.eclipse.org/ganymede/nav/4_2_0) provides a good foundation of knowledge, but does fall short in on some specifics, including the use of **Pack200** (<http://wiki.eclipse.org/Pack200>) and **JAR signing** (http://wiki.eclipse.org/JAR_Signing) using the Eclipse Foundation's certificate. This document addresses these issues. PDE can still be used to build your RCP applications.

The following guides are available:

- PDE Build from Markus Barchfeld's **Build and Test Automation for plug-ins and features** (<http://www.eclipse.org/articles/Article-PDE-Automation/automation.html>). It's getting a bit old (having been written for eclipse-3.0).
- **Eclipse Headless build with PDE Build** (<http://www.vogella.de/articles/EclipsePDEBuild/article.html>) - Tutorial by Lars Vogel based on Eclipse 3.4

Ant

Ant is heavily used in the PDE build infrastructure that ships with the Plug-in Development Environment. Ant is used to run the build, with adequate plug-points where in you can perform custom tasks such as instrumentation of code.

To get comfortable with ant:-

<http://ant.apache.org/manual> (<http://ant.apache.org/manual>) - If you are new to ant

Eclipse provides a very usable interface for writing ant files(yes, with auto-complete and the stuff) as well as run the ant files from within the IDE. But, for headless(without UI, from command line) builds, you'll need to use the antRunner application provided by eclipse. For more help, peek into the help documentation of eclipse that you are using.

Plugin Builder plugin



(/File:Warning2.png) The following information is fairly old and it is unknown if still relevant or accurate, but is left here in case anyone finds it helpful.

<http://www.pluginbuilder.org> (<http://www.pluginbuilder.org>) - Something to make your life even easier

Some old articles were written related to this mentioned in **bug 87151** (https://bugs.eclipse.org/bugs/show_bug.cgi?id=87151) and **bug 139481** (https://bugs.eclipse.org/bugs/show_bug.cgi?id=139481).

Versioning

Plugins (including features and fragments) must specify their own versions as well as the versions of their dependencies appropriately.

Information

See the eclipse team's **plugin-versioning guidelines** (<http://www.eclipse.org/equinox/documents/plugin-versioning.html>). You can also ask Pascal Rapicault or John Arthorne for their "Plug-in versioning" presentation.

Tooling

Pascal Rapicault has created an application (`org.eclipse.core.runtime.versionchecker.dependencyChecker`) to "specify the ranges on all [your] required plug-ins."

Multi-platform builds

Eclipse RCP applications can be cross-built for several platforms using any platform on which eclipse runs.

Previously, before the Mars release, you could install the RCP delta pack in your Target Platform (which was downloadable as a zip file, from the download page).

In the Mars release, that delta pack zip file is no longer available on the download page. The reason for removing it was that p2 can be used instead, so it saves complications in the Platform build, and does not take up redundant space making the same stuff available in multiple ways.

Preferred way of doing multi-platform builds

The preferred way of getting the platform specific artifacts is to just add them to the target platform. There is no need to look-up and download the "DeltaPack" if you follow these instructions.

This is works not only with the Mars release, but also previous releases, as well.

- Open Window/Preferences.
- Find PDE/Target Platform
- Select your (active) target platform
- Click Edit
- Click Add
- Select "Software Site"
- Click Next
- In "Work With" type: **<http://download.eclipse.org/eclipse/updates/4.3>** (<http://download.eclipse.org/eclipse/updates/4.3>) (replace 4.3 with your current version)
- Check "Eclipse RCP Target Components"
- Check "Equinox Target Components"
- Uncheck "Include required software"
- Check "Include all environments"
- Press Finish
- Press Finish
- Press OK

Open your product file and select the "Export" option. You will see that the "Export for multiple platforms" checkbox is available.

Explanation of why it works

The key to doing multi-platform builds is to have the equinox executable feature in your target (but, none of your features should "include" it directly). The "p2 name" for this feature (IU) is `org.eclipse.equinox.executable.feature.group` and it's human readable name is `Eclipse Platform Launcher Executables`. It may be useful to also have the `rcp` configuration feature in your target. Its p2 IU is named `org.eclipse.rcp.configuration.feature.group` and its human readable name is `Eclipse Product Configuration`. They are both available in the Eclipse Project's software repository, and the executable feature is in the `../releases/mars` repository, but the `rcp` config feature is not (**bug 470911** (https://bugs.eclipse.org/bugs/show_bug.cgi?id=470911)). It is the executable feature that enables the PDE wizard to enable "Export for Multiple Platforms".

Of course, in addition to those key features, you need all the other prerequisite features, bundles, and platform specific fragments in your target. That's why, in the steps above, large "categories" of prerequisites are given. As your build gets more refined, it is typically better to select only exactly what you need, but this takes some knowledge and experience to know that those are.

The community of previous delta pack users are welcome to add to this wiki any tips and tricks they learn ... and provide concrete working examples. I have opened **bug 470913** (https://bugs.eclipse.org/bugs/show_bug.cgi?id=470913) to track making improvements to these instructions in the future.

What if you really want the traditional Delta Pack zip file?

It is easy to make your own. There are a number of blogs, stackoverflow answers, and similar sources of information devoted to the topic. There is even an **example in Eclipse Help about p2 tasks** (http://help.eclipse.org/luna/index.jsp?topic=%2Forg.eclipse.platform.doc.isv%2Fguide%2Fp2_repositorytasks.htm) that shows how to make one.

Plus, there is an **ant script in our Git repository** (<http://git.eclipse.org/c/platform/eclipse.platform.releng.aggregator.git/plain/scripts/createdeltapack.xml>) that will create one simply by executing the script in your IDE. (That one is very similar to the one we used to create as part of our download page, in the past). You'll need to make your own copy of that file, and edit it, putting in your own values for some of the variables (or, overriding them via the command line. Of course, you only need to make it one, to use as many product builds for which it is suitable.

One advantage of creating your own delta pack, is that there are a number of ways to create one, things to include, or not, so you can pretty much pick what you need, instead of being locked in to the exact one that was previously provided on the download page.

Old way of using the "DeltaPack"

Note: The download is no longer available as of 4.5.0. Older releases still provide that download.

What was the DeltaPack

The DeltaPack is a distribution from Eclipse that contains platform specific files for all supported platforms and widget sets. It contains:

- SWT Platform Layers
Various plugins to handle connection to the native OS widgets.
- Various launchers (exe, cmd, etc..)
Files to launch Eclipse from the command line
- Other platform specific plugins
Hooks into secure storage, native networking and native Eclipse filesystem hooks

In total there are some 70 plugins to support the complete range of supported operating systems (around a dozen).

Using the "DeltaPack"

The DeltaPack is used to get platform specific artifacts so that you can export your RCP product for multiple platforms.

When do you need it

You need it if you want to export your RCP application to multiple targets (a combination of OS/ARCH/WIDGETSET: e.g. linux/ppc64/gtk). This is typically done from the product configuration file.

How to get it

You can get it from: <http://download.eclipse.org/eclipse/downloads/> (<http://download.eclipse.org/eclipse/downloads/>)

Click on any of the builds and look for DeltaPack. Please note that [there are discussions on removing the DeltaPack](https://bugs.eclipse.org/bugs/show_bug.cgi?id=419246) and getting it the preferred way.

How to add it to your target platform

- Download the DeltaPack as described above
- Unzip it to a location .e.g. "C:\eclipse-4.4M7-delta-pack"
- Open Window/Preferences.
- Select PDE/Target Platform
- Select your (active) target platform
- Click Edit
- Click Add
- Select "Directory"
- Click Next
- In "Location" type: "C:\eclipse-4.4M7-delta-pack\eclipse"
- Press Next
- Press Finish
- Press Finish
- Press OK

Open your product file and select the "Export" option. You will see that the "Export for multiple platforms" checkbox is available.

This page was last modified 11:56, 1 August 2016 by [David Williams](#) ([/index.php?title=User:David_williams.acm.org&action=edit&redlink=1](#)). Based on work by [Jens übler](#) ([/index.php?title=User:Kuebler.aquintos.com&action=edit&redlink=1](#)), [Aurelien Pupier](#) ([/index.php?title=User:Aurelien.pupier.bonitasoft.com&action=edit&redlink=1](#)) and [Wim Jongman](#) ([/User:Wim.jongman.remainssoftware.com](#)) and others ([/index.php?title=A_Brief_Overview_of_Building_at_Eclipse&action=credits](#)).