**ECLIPSE IDE**

**Eclipse** is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in Java and its primary use is for developing Java applications, but it may also be used to develop applications in other programming languages via plug-ins, including Ada, ABAP, C, C++, C#, Clojure, COBOL, D, Erlang, Fortran, Groovy, Haskell, JavaScript, Julia, Lasso, Lua, NATURAL, Perl, PHP, Prolog, Python, R, Ruby (including Ruby on Rails framework), Rust, Scala, and Scheme. It can also be used to develop documents with LaTeX (via a TeXlipse plug-in) and packages for the software Mathematica. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others.

The initial codebase originated from IBM Visual Age. The Eclipse software development kit (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Since the introduction of the OSGi implementation (Equinox) in version 3 of Eclipse, plug-ins can be plugged-stopped dynamically and are termed (OSGI) bundles.

Eclipse software development kit (SDK) is free and open-source software, released under the terms of the Eclipse Public License, although it is incompatible with the GNU General Public License. It was one of the first IDEs to run under GNU Class path and it runs without problems under Iced Tea.

The Eclipse SDK includes the Eclipse Java development tools (JDT), offering an IDE with a built-in Java incremental compiler and a full model of the Java source files. This allows for advanced refactoring techniques and code analysis. The IDE also makes use of a workspace, in this case a set of metadata over a flat file space allowing external file modifications as long as the corresponding workspace resource is refreshed afterward.

Eclipse implements the graphical control elements of the Java toolkit called Standard Widget Toolkit (SWT), whereas most Java applications use the Java standard Abstract Window Toolkit (AWT) or Swing. Eclipse's user interface also uses an intermediate graphical user interface layer called JFace, which simplifies the construction of applications based on SWT. Eclipse was made to run on Wayland during a Google Summer of Code (GSoC) Project in 2014.

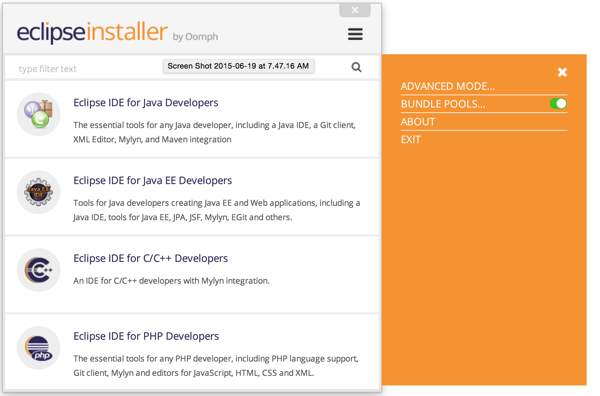
**MAIN FEATURES OF ECLIPSE IDE:**

**Following are some main features of Eclipse IDE:**

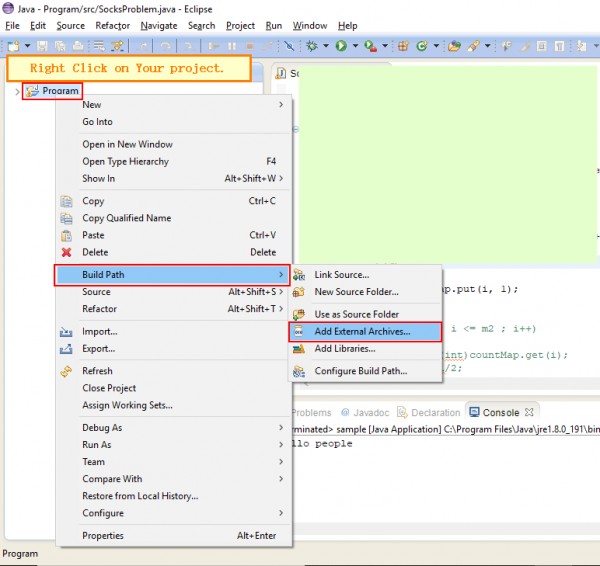
**1. The Eclipse Installer (by Oomph)**

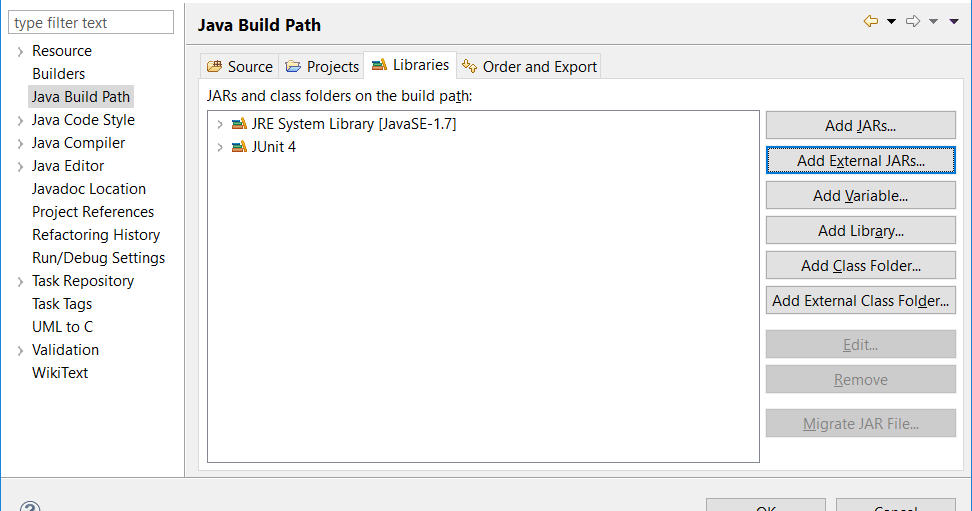
Eclipse offers a completely new way to download and work with the IDE. Instead of downloading each package individually, the Eclipse installer by Oomph allows you to choose the installation you wish, and it will provision it for you.

You can choose \_bundle pool\_, to share Eclipse plugins between installations. This means that if you install another package, all the common bits will be shared. Finally, once Eclipse is started, you can walk through the Eclipse Welcome Questionnaire to set common Eclipse preferences to your liking.

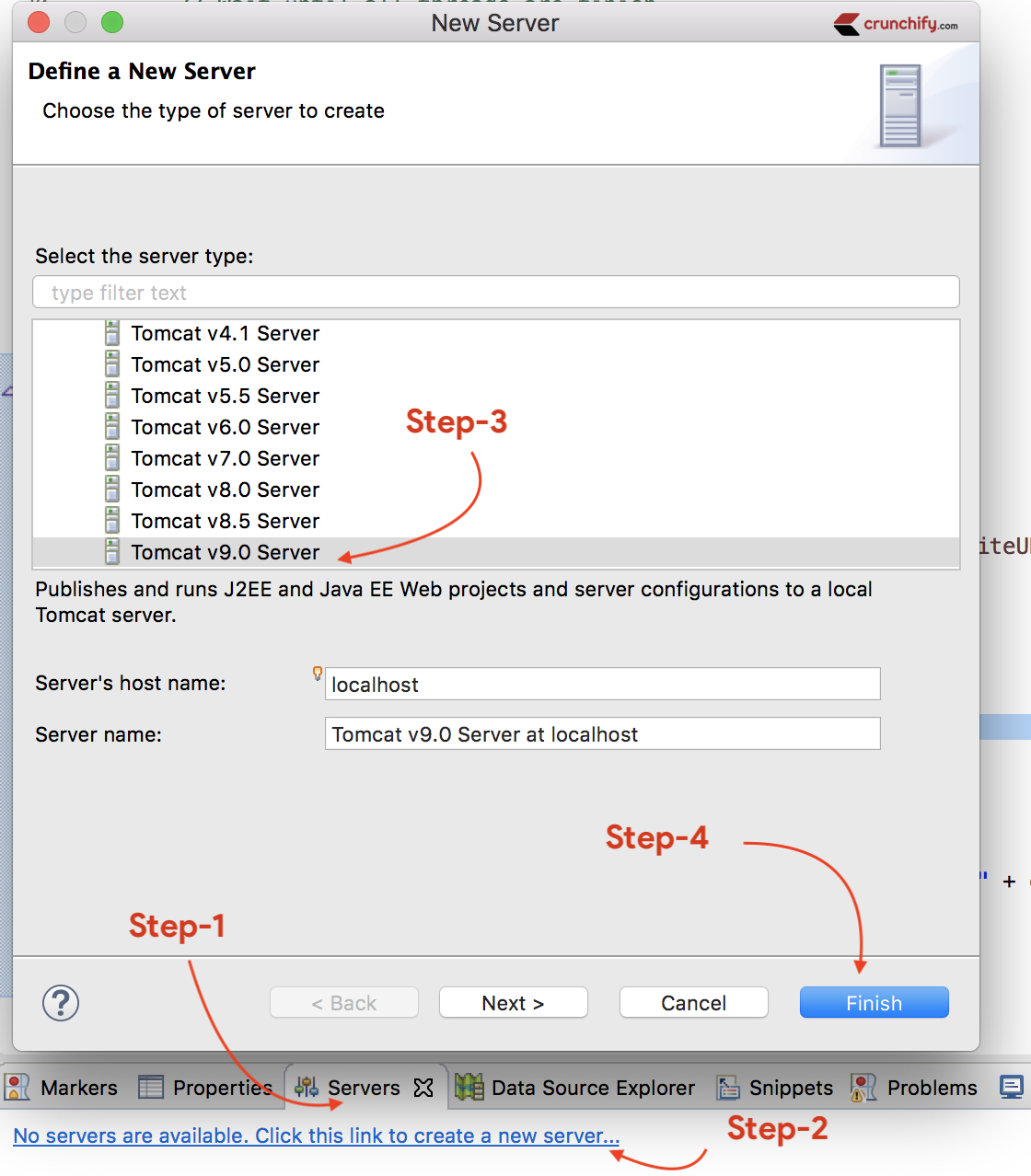


**2. Methods to import External Jar files in Eclipse:**

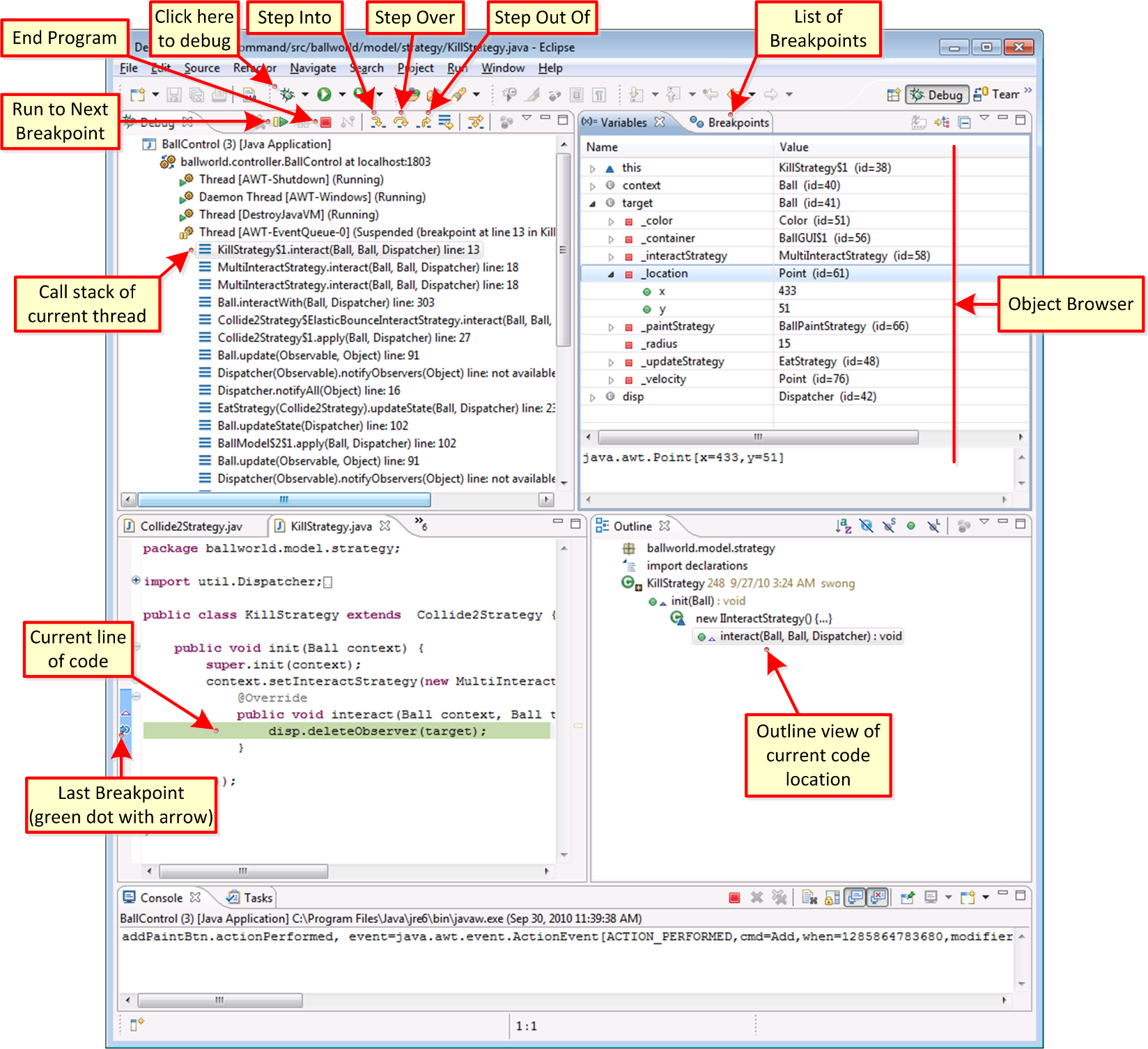
****

****

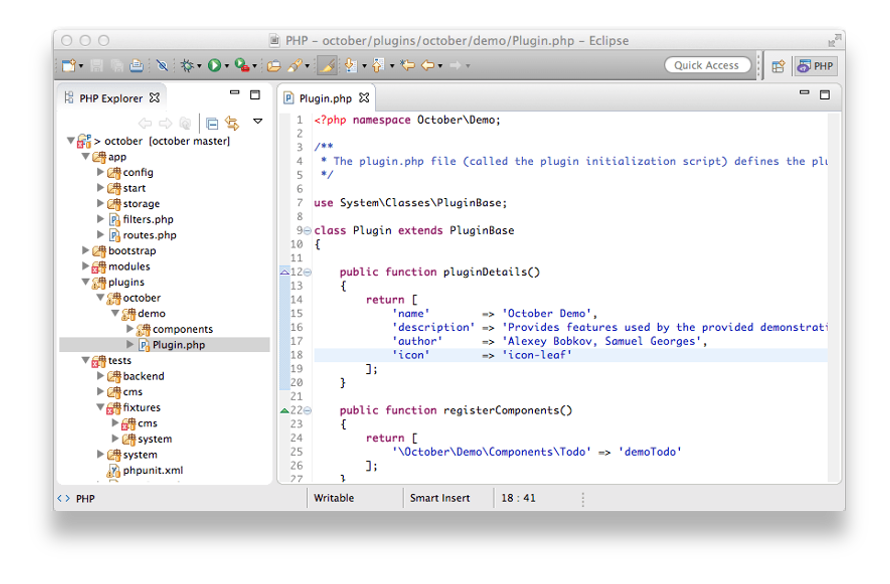
**3. Method to configure Server in Eclipse:**

****

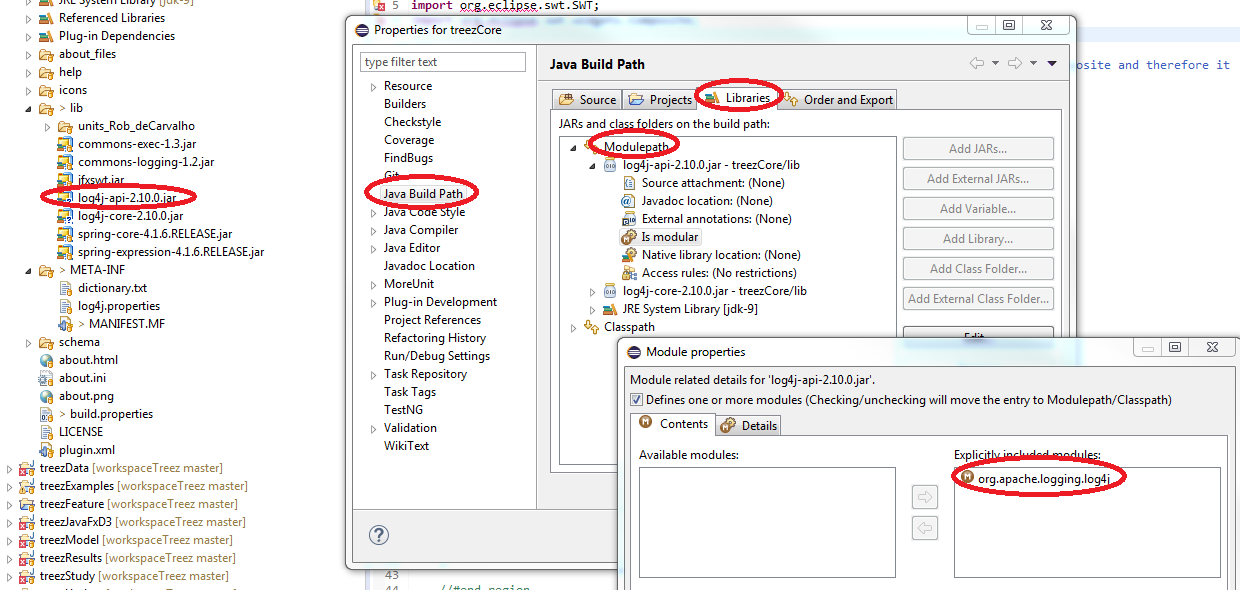
**4. Debug Process in Eclipse:**

****

**5. PHP in Eclipse:**

****

**6. Process in configure and use Apache Log4j:**

****

Log4j mainly has 3 components:

* Logger:

It is class, we can able to log the message, it has different methods to put the log message.

* Appender:

It is used to prediction where log messages are being stored like file, console and so on.

* Layout:

It is basically used to show on what pattern it will be stored like PatternLayout, HtmlLayout, SimpleLayout, XmlLayout, TTCCLayer

There are 5 methods in Logger Class:

* debug()
* info()
* warn()
* error()
* fatal()

**7. Mac Layout**

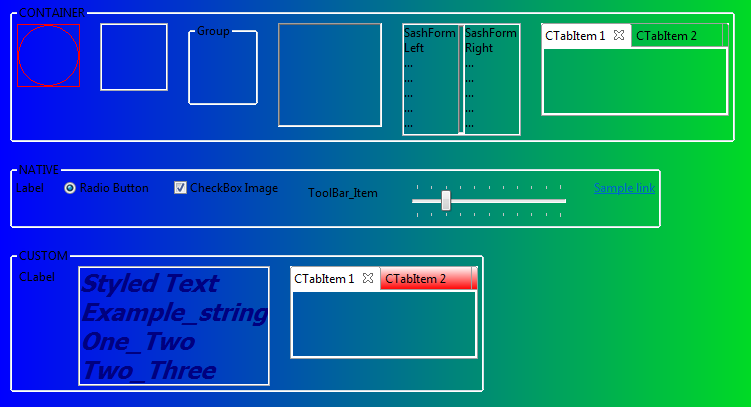
While 10 years is a long time, some Eclipse feature requests last longer than that. Since before YouTube had its first video, the Eclipse community has been requesting a proper Mac Application for Eclipse.

Eclipse team finally delivered, shipping the MacOS version as a proper Mac application. This means that Eclipse can be unzipped and dragged directly in your Application folder. With this change, installing Eclipse becomes much easier on Mac, and features like “open with” now work as expected.



**8. Platform Improvements**

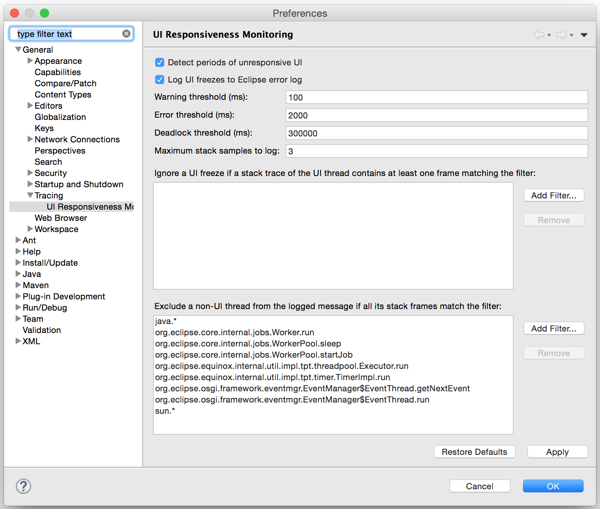
The Eclipse platform is the core of all Eclipse based applications and plugins. Improvements to the platform benefit us all. There are a number of improvements to the core platform this year, including an improved Dark Theme, crisper icons and proper transparency support in SWT.



**9. UI Monitoring**

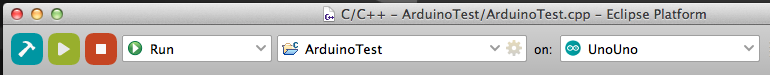
From Mac’s spinning beach ball of death to Window’s unresponsive UI, user interface lags take developers out of the flow. We’ve all tried to fix the problem by clicking the mouse a little harder or hammering on the esc key; but as we all know, that doesn’t work. The only real fix is to understand what long-running task is blocking the UI thread.

With Eclipse Mars, debugging this just got a lot easier as Eclipse now ships with a UI Responsiveness Monitoring Tool. The tool can be configured to report instances of UI unresponsiveness.



**10. C/C++ Launching**

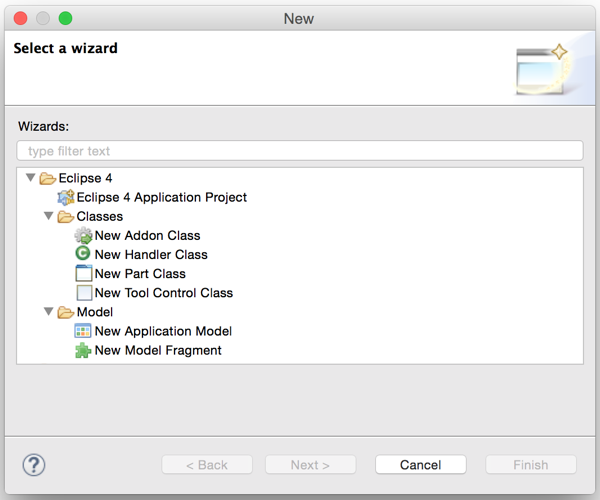
The developers behind the C/C++ tooling in Eclipse (CDT) have brought a number of new enhancements this year. To make it easier to launch your applications and choose your target device, a new launch bar was added. As well as the launch bar, the CDT also added support for direct Docker deployment. This means that you can deploy your C/C++ applications from Eclipse directly to a docker container.



**11. e4 Tools**

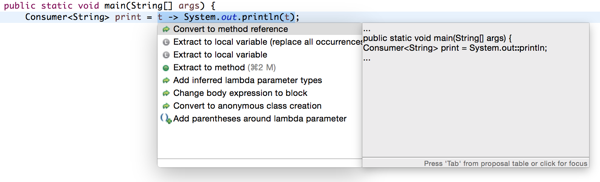
In 2012 Eclipse launched a new application model called e4. The application can be designed through a series of UI models, styled using CSS and dependencies are managed via DI.

This year, the Eclipse team has released the long awaited tooling for e4 as part of the simultaneous release. The Plug-in Development Environment (PDE) has a series of wizards and tools to help you create an Eclipse 4 application and editors to help you configure your models.



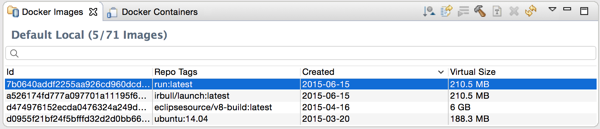
**12. JDT Improvements**

Eclipse Luna (2014) shipped with Java 8 support and this year the team expanded on that support. There are quick fixes for creating method references from lambda expressions; quick-fixes for inferring lambda parameter; several new templates have been added for things like try / finally and lock acquisition; and much improved static flow analysis.



**13. Docker Tools**

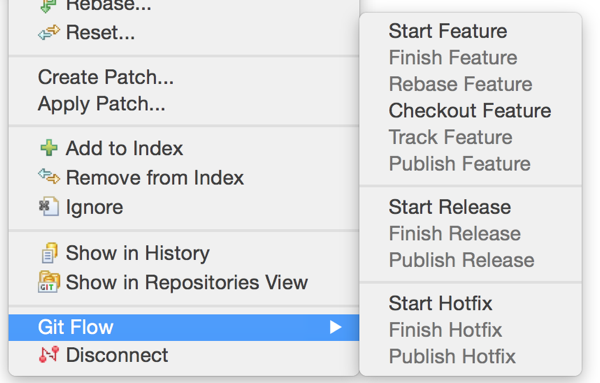
Throughout my day I work on several different projects. I work with Tomcat servers, CouchDB, C/C++ builds, node.js and many other technologies. Docker has been a life-saver when it comes to managing all these stacks. There is a new Docker perspective that you can use to connect to a Docker daemon. There are views to manage your images and containers and even a log inspector.



**14. Git Flow**

Eclipse has shipped with Git support since 2011. While the tool has improved significantly over the years, many developers would still drop to the command line for custom tasks. One such example is the use of Git Flow. Git Flow is a branching model that places meaning on different branches. Each feature is developed in its own branch, and merged into a standard ‘develop’ branch. To perform a release, the ‘develop’ branch is merged into master. There are tools to help simplify this process.

With Eclipse Mars, there is now support for Git Flow directly from Eclipse.



**15. Automatic Error Reporting**

Instead of simply logging errors to the console, Eclipse can now report errors directly to the Eclipse committers; putting the information in the hands of those who can use it.

When errors are detected, you will be prompted if you would like to send the details to eclipse.org. Committers will be able to create bug reports from the data, see similar incidents and help triage issues as they come in. Finally, if an issue has already been fixed, Eclipse will notify you and inform you of the version that contains the fix.

