# Red Hat Developer Tools 2018.3

## Using Eclipse

Installing Eclipse 4.8.0 and first steps with the application

Last Updated: 2018-07-19

# Red Hat Developer Tools 2018.3 Using Eclipse

Installing Eclipse 4.8.0 and first steps with the application

Supriya Takkhi
sbharadw@redhat.com

## Legal Notice



## Abstract

Information for users installing and starting to use Red Hat Developer Tools.

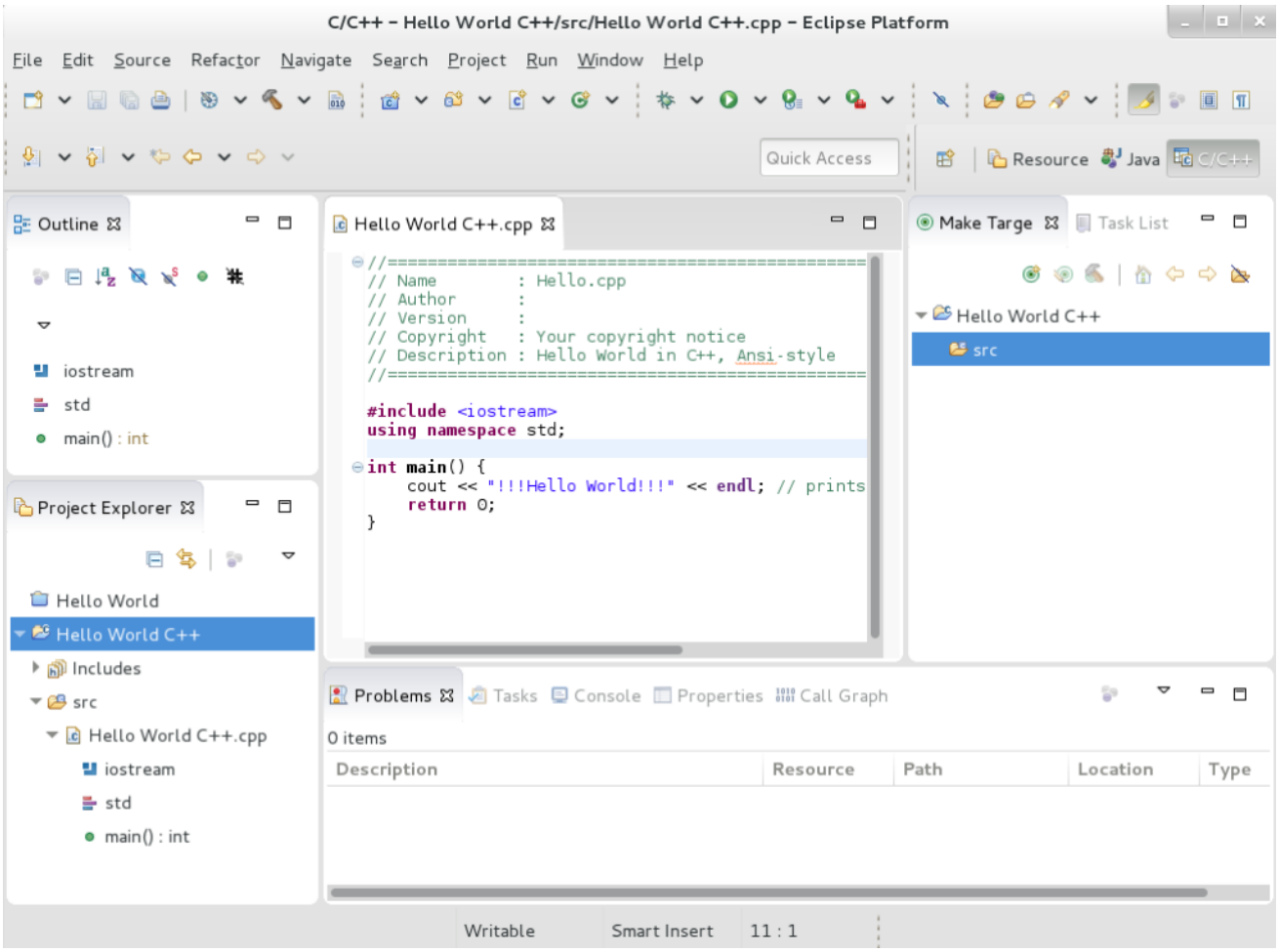
# Table of Contents

# CHAPTER 1. ECLIPSE 4.8.0

Red Hat Developer Tools, available for Red Hat Enterprise Linux 7, includes Eclipse 4.8.0, which is based on the Eclipse Foundation's Oxygen release train. Note that **rh-eclipse48** requires the **rh-maven35** collection.

**Eclipse** is a powerful development environment that provides tools for each phase of the development process. It integrates a variety of disparate tools into a unified environment to create a rich development experience, provides a fully configurable user interface, and features a pluggable architecture that allows for an extension in a variety of ways. For instance, the **Valgrind** plug-in allows programmers to perform memory profiling, that is otherwise performed on the command line through the **Eclipse** user interface.

**Figure 1.1. Sample Eclipse Session**




**Eclipse** provides a graphical development environment alternative to traditional interaction with command line tools and it is a welcome alternative for developers who do not want to use the command line interface. The traditional, mostly command line-based Linux tools suite such as **gcc** or **gdb** and **Eclipse** offer two distinct approaches to programming.

For an overview to develop applications for Red Hat JBoss Middleware or for support for OpenShift Tools, see Red Hat Developer Studio.

**Table 1.1. Eclipse Components Included in rh-eclipse48**

| Package | Description |
|---|---|

| Package | Description |
| --- | --- |
| **rh-eclipse48-eclipse-cdt** | The C/C++ Development Tooling (**CDT**) that provides features and plug-ins for development in these two languages. |
| **rh-eclipse48-eclipse-changelog** | The ChangeLog plug-in that allows you to create and maintain changelog files. |
| **rh-eclipse48-eclipse-dtp** | Eclipse Data Tools Platform Project |
| **rh-eclipse48-eclipse-egit** | EGit, a team provider for **Eclipse** that provides features and plug-ins for interaction with Git repositories. |
| **rh-eclipse48-eclipse-emf** | The **Eclipse** Modeling Framework (**EMF**) that allows you to build applications based on a structured data model. |
| **rh-eclipse48-eclipse-epp-logging** | The Eclipse error reporting tool. |
| **rh-eclipse48-eclipse-gcov** | The GCov plug-in that integrates the **GCov** test coverage program with **Eclipse**. |
| **rh-eclipse48-eclipse-gef** | The Graphical Editing Framework (**GEF**) that allows you to create a rich graphical editor from an existing application model. |
| **rh-eclipse48-eclipse-gprof** | The Gprof plug-in that integrates the **Gprof** performance analysis utility with **Eclipse**. |
| **rh-eclipse48-eclipse-jdt** | The **Eclipse** Java development tools (**JDT**) plug-in. |
| **rh-eclipse48-eclipse-jgit** | JGit, a Java implementation of the **Git** revision control system. |
| **rh-eclipse48-eclipse-manpage** | The Man Page plug-in that allows you to view manual pages in **Eclipse**. |
| **rh-eclipse48-eclipse-mpc** | The **Eclipse** Marketplace Client. |
| **rh-eclipse48-eclipse-mylyn** | Mylyn, a task management system for **Eclipse**. |
| **rh-eclipse48-eclipse-oprofile** | The OProfile plug-in that integrates **OProfile** with **Eclipse**. |

| Package | Description |
| --- | --- |
| **rh-eclipse48-eclipse-pde** | The Plugin Development Environment for developing **Eclipse** plugins. |
| **rh-eclipse48-eclipse-perf** | The Perf plug-in that integrates the **perf** tool with **Eclipse**. |
| **rh-eclipse48-eclipse-ptp** | A subset of the PTP project that provides support for synchronized projects. |
| **rh-eclipse48-eclipse-pydev** | A full featured Python **IDE** for **Eclipse**. |
| **rh-eclipse48-eclipse-recommenders** | Code Recommenders |
| **rh-eclipse48-eclipse-remote** | The Remote Services plug-in that provides an extensible remote-services framework. |
| **rh-eclipse48-eclipse-rpm-editor** | The **Eclipse** Spec File Editor that allows you to maintain **RPM** spec files. |
| **rh-eclipse48-eclipse-rse** | The Remote System Explorer (**RSE**) framework that allows you to work with remote systems from **Eclipse**. |
| **rh-eclipse48-eclipse-systemtap** | The SystemTap plug-in that integrates **SystemTap** with **Eclipse**. |
| **rh-eclipse48-eclipse-subclipse** | Subclipse, a team provider for Eclipse that provides features and plug-ins for interaction with Subversion repositories. |
| **rh-eclipse48-eclipse-valgrind** | The Valgrind plug-in that integrates **Valgrind** with **Eclipse**. |
| **rh-eclipse48-eclipse-webtools** | The **Eclipse** Webtools plug-ins. |

## 1.1. ENABLING THE RED HAT DEVELOPER TOOLS REPOSITORIES

To install Eclipse you must first enable the Red Hat Developer Tools repositories.

- Enable the **rhel-7-server-devtools-rpms** repository on Red Hat Enterprise Linux Server or enable the **rhel-7-workstation-devtools-rpms** repository on Red Hat Enterprise Linux Workstation
- Enable the **rhel-server-rhscl-7-rpms** repository on Red Hat Enterprise Linux Server or enable the **rhel-workstation-rhscl-7-rpms** repository on Red Hat Enterprise Linux Workstation

**NOTE**

Red Hat Developer Tools is not supported on the **Client** or the **ComputeNode** variant.

To enable the Red Hat Developer Tools repositories:

1. Run the following commands as the root user:

```
# subscription-manager repos --enable rhel-7-server-devtools-rpms
# subscription-manager repos --enable rhel-server-rhscl-7-rpms
```

**NOTE**

In the above command, depending on the variant of Red Hat Enterprise Linux that you are using, **Server** or **Workstation**, the commands will be:

- For the **Workstation** variant: **subscription-manager repos --enable rhel-7-workstation-devtools-rpms**
- For the **Server** variant: **subscription-manager repos --enable rhel-7-server-devtools-rpms**

For developers, we recommend using Red Hat Enterprise Linux Server for access to the widest range of development tools.

For more information on registering and attaching subscriptions, see Using and Configuring Red Hat Subscription Management.

### 1.1.1. Enabling the Red Hat Developer Tools debuginfo Repositories

Red Hat Developer Tools also provides the **debuginfo** packages for all architecture-dependent RPMs included in the repositories. These packages are useful for core file analysis and for debugging of Eclipse itself. To enable the Red Hat Developer Tools debuginfo repositories:

1. Run the following commands as the root user:

```
$ subscription-manager repos --enable rhel-7-server-devtools-debug-
rpms
$ subscription-manager repos --enable rhel-server-rhscl-7-debug-rpms
```

For details about installing, understanding, and using the **debuginfo** packages, refer to Debugging a Running Application.

## 1.2. INSTALLING ECLIPSE

The **Eclipse** development environment is provided as a collection of **RPM** packages.

1. To install Eclipse, run the following commands as the **root** user:

```
yum install rh-eclipse48
```

For a list of available components, see Table 1.1, “Eclipse Components Included in rh-eclipse48”.

**NOTE**

**rh-eclipse48** fully supports C, C++, and Java development, but does not provide support for the Fortran programming language.

## 1.3. USING ECLIPSE

To start **rh-eclipse48**:

1. Click **Applications** > **Programming** > **Red Hat Eclipse**, or type the following at a shell prompt:

```
scl enable rh-eclipse48 eclipse
```

2. During its startup, **Eclipse** prompts you to select a *workspace* that is a directory in which you want to store your projects. You can either use **~/workspace/**, which is the default option, or click **Browse** to browse your file system and select a custom directory. Additionally, you can select the **Use this as the default and do not ask again** check box to prevent **Eclipse** from displaying this dialog box the next time you run this development environment.

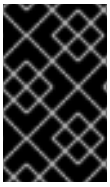
3. Click **OK** to confirm the selection and proceed with the startup.

### 1.3.1. Using the Red Hat Developer Toolset Toolchain

To use **rh-eclipse48** with support for the **GNU Compiler Collection** and **binutils** from Red Hat Developer Toolset:

1. Ensure that **devtoolset-Developer Toolset 7-toolchain** is installed and run the application as described in Section 1.3, “Using Eclipse”. The **rh-eclipse48** Collection uses the Red Hat Developer Toolset toolchain by default.

For detailed instructions on how to install the **devtoolset-Developer Toolset 7-toolchain** package on your system, see Installing Red Hat Developer Toolset.



**IMPORTANT**

If you are working on a project that you previously built with the Red Hat Enterprise Linux version of the **GNU Compiler Collection**, make sure that you discard all previous build results. To do so, open the project in **Eclipse** and select **Project** > **Clean** from the menu.

### 1.3.2. Using the Red Hat Enterprise Linux Toolchain

To use **rh-eclipse48** with support for the toolchain distributed with Red Hat Enterprise Linux, change the configuration of the project to use absolute paths to the Red Hat Enterprise Linux system versions of **gcc**, **g++**, and **as**.

To configure **Eclipse** to explicitly use the Red Hat Enterprise Linux system versions of the tools for the current project:

1. In the C/C++ perspective, click **Project** > **Properties** to open the project properties.

2. In the menu on the left-hand side of the dialog box, click **C/C++ Build** > **Settings**.

3. Click the **Tool Settings** tab.

4. If you are working on a C project:

a. Select **GCC C Compiler** or **Cross GCC Compiler** and change the value of the **Command** field to:

```
/usr/bin/gcc
```

b. Select **GCC C Linker** or **Cross GCC Linker** and change the value of the **Command** field to:

```
/usr/bin/gcc
```

c. Select **GCC Assembler** or **Cross GCC Assembler** and change the value of the **Command** field to:

```
/usr/bin/as
```

5. If you are working on a C++ project:

a. Select **GCC C Compiler* or *Cross G Compiler** and change the value of the **Command** field to:

```
/usr/bin/g++
```

b. Select **GCC C Compiler** or **Cross GCC Compiler** and change the value of the **Command** field to:

```
/usr/bin/gcc
```

c. Select **GCC C Linker* or *Cross G Linker** and change the value of the **Command** field to:

```
/usr/bin/g++
```

d. Select **GCC Assembler** or **Cross GCC Assembler** and change the value of the **Command** field to:

```
/usr/bin/as
```

6. Click **OK** to save the configuration changes.

## 1.4. ADDITIONAL RESOURCES

A detailed description of **Eclipse** and all its features is beyond the scope of this document. For more information, see the following resources.

### Installed Documentation

- **Eclipse** includes a built-in Help system that provides extensive documentation for each integrated feature and tool. This greatly decreases the initial time investment required for new developers to become fluent in its use. The use of this Help section is detailed in the *Red Hat Enterprise Linux Developer Guide* linked below.

### See Also

- Chapter 2, *Changes Since Developer Tools 2018.2* provides a list of selected features and improvements over the **Eclipse** development environment included in the previous release of Developer Tools.
- The Red Hat Developer Toolset chapter in the User Guide provides an overview of Red Hat Developer Toolset and more information on how to install it on your system.
- The GNU Compiler Collection (GCC) chapter in the User Guide provides information on how to compile programs written in C, C++, and Fortran on the command line.

# CHAPTER 2. CHANGES SINCE DEVELOPER TOOLS 2018.2

Red Hat Developer Tools is distributed with Eclipse 4.7.3a and other plugins from the Oxygen release train, which provide a number of bug fixes and feature enhancements over the Red Hat Developer Tools 2018.1 version. Following is a comprehensive list of new features and compatibility changes in this release. For details on how to use these new features, see the built-in Eclipse documentation.

**Dependencies**

The rh-eclipse48 collection depends on the rh-maven35 collection. RHEL 7.5 is recommended for the best possible experience.

**Significant Package Updates**

- eclipse 4.7.3a > 4.8.0: Eclipse Platform was updated to 4.8.0. The highlights include:
  - This major update brings many user experience and user interface enhancements, including improved support for dark theme, improved accessibility support and improved HiDPI support.
  - The browser view now uses WebKit2 for increased stability.
  - Full Java 9 and Java 10 support in the Java Development Tools, improved code formatter and support for debug launch **prototypes**.
  - Support for OSGi capability headers in both the Equinox runtime and in the Plug-in Development Environment.

For details, see the upstream release notes at https://www.eclipse.org/eclipse/news/4.8/.

- eclipse-cdt 9.4.3 > 9.5.0: C/C++ Development Tools was updated to 9.5.0. The highlights include:
  - Improved support for both the C+14 and C+17 standards.
  - Container build support is added to more project types, including CMake and Standard Make projects.

For details, see the upstream release notes at https://wiki.eclipse.org/CDT/User/NewIn95.

- eclipse-linuxtools 6.2.1 > 7.0.0: Linuxtools was updated to 7.0.0 including improved launching and debugging of processes within containers and an upgraded Docker client implementation for improved stability.
- eclipse-m2e-core 1.8.3 > 1.9.0: Eclipse Maven integration was updated to 1.9.0 including greatly enhanced source lookup and many other bugfixes.
- eclipse-webtools 3.9.4 > 3.10.0: The Webtools plug-ins were updated 3.10.0 including improved support for using Tomcat 9 with Java 9 and 10, initial support for JEE 8 and the removal of support for deprecated Axis 1-style web services.

# CHAPTER 3. KNOWN ISSUES IN ECLIPSE 4.8.0

This section details the Known Issues in Eclipse 4.8.0.

## 3.1. DEBUGINFO CONFLICTS

**Cause**: When a user attempts to install a **`debuginfo`** package for this collection, for example: **`rh-eclipse48-eclipse-cdt-debuginfo`**, the contents may conflict with the same packages from the earlier collections.

**Consequence**: The installation of the **`rh-eclipse48 debuginfo`** package may fail if the same **`debuginfo`** package from the rh-eclipse47 collection is installed.

**Workaround**: To use debuginfo for the rh-eclipse48 collection, **`debuginfo`** packages from the rh-eclipse47 collection should first be uninstalled using the command **`yum remove rh-eclipse47*debuginfo`**.

**Result**: The installation of the **`debuginfo`** packages from the rh-eclipse48 collection is successful.

## 3.2. CONFLICT BETWEEN THE RHSCL `RH-MAVEN33-SCLDEVEL` AND `RH-MAVEN35-SCLDEVEL` PACKAGES

There is a conflict between the RHSCL **`rh-maven33-scldevel`** and **`rh-maven35-scldevel`** packages. This affects the **`rh-eclipse48-scldevel`** package (note that this is not installed by default).

The **`rh-eclipse48-scldevel`** and **`rh-eclipse46-scldevel`** packages cannot be present simultaneously.

## 3.3. PYDEV USERS MAY EXPERIENCE ISSUES WITH THE 'PIP' INTEGRATION

Since SCL Pythons are readonly installations, users will always have to add the **--user** option to install modules with pip. And modules that are pre-installed by the Python SCL cannot be uninstalled; attempts to do so will result in permission denied errors.

## 3.4. INCOMPATIBILITIES BETWEEN ECLIPSE SUBCLIPSE AND BASE RHEL SUBVERSION

Working copies of Subversion repositories created with Eclipse Subclipse are incompatible with the base RHEL version of Subversion. Using the **svn** command on such working copies may result in the following error:

```
$ svn up
svn: E155021: This client is too old to work with the working copy
```

**Workaround**: Use the pure-java implementation of Subversion used by Eclipse Subclipse on the command line:

```
# yum install rh-eclipse48-svnkit-cli # Command line support for SVNKit
```

And now, use the **jsvn** command anywhere you would normally use the **svn** command:

```
$ jsvn up
Updating '.':
At revision 16476.
```

## 3.5. TYCHO CONFLICTS

**Cause**: The **rh-eclipse48-tycho** package conflicts with the same package from the earlier collections, for example: **rh-eclipse47-tycho**.

**Consequence**: The installation of **rh-eclipse48-tycho** package may fail when the **rh-eclipse47-tycho** package is already installed.

**Workaround**: Note that only users that want to build or re-build Eclipse or its plug-ins need tycho. If needed, uninstall the **rh-eclipse47-tycho** package using the **yum remove rh-eclipse47-tycho** command before attempting to install the **rh-eclipse48-tycho** package.

**Result**: The installation of the **rh-eclipse48-tycho** package is successful.