Red Hat Developer Tools 2019.1

Using Eclipse

Installing Eclipse 4.10.0 and first steps with the application

Last Updated: 2019-03-25

Red Hat Developer Tools 2019.1 Using Eclipse

Installing Eclipse 4.10.0 and first steps with the application

Robin Owen
kowen@redhat.com

Legal Notice

Abstract

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

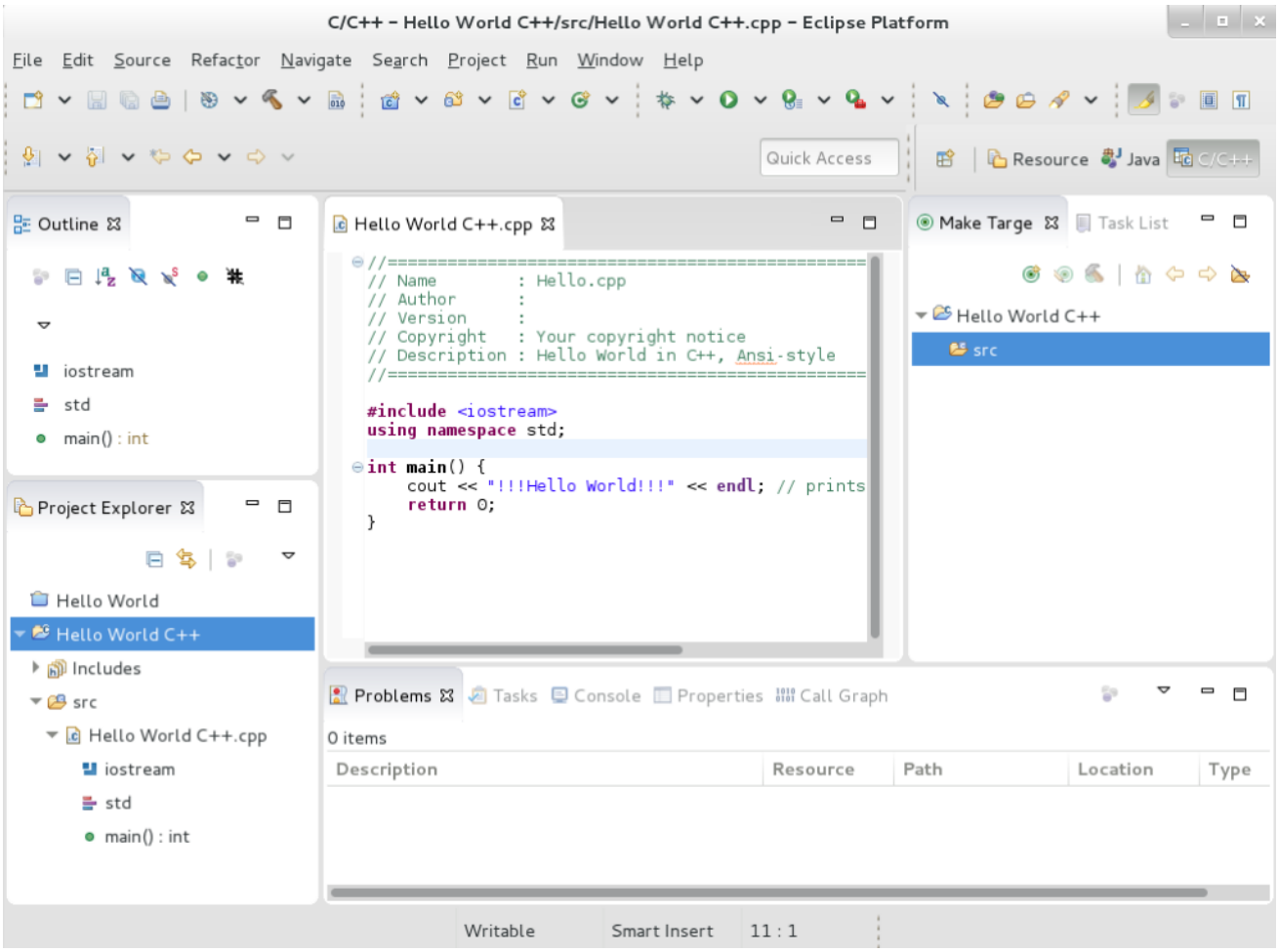
Table of Contents

CHAPTER 1. ECLIPSE 4.10.0

Red Hat Developer Tools, available for Red Hat Enterprise Linux 7, includes Eclipse 4.10.0, which is based on the Eclipse Foundation's 2018-12 release train. Note that the **rh-eclipse** collection 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 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 (such as **gcc** or **gdb**) and **Eclipse** offer two distinct approaches to programming.

For an overview on how 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-eclipse

Package	Description

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

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

1.1. ENABLING THE RED HAT DEVELOPER TOOLS REPOSITORIES

To install Eclipse, 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 **ComputeNode** variants.

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-eclipse
```

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

NOTE

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

1.3. USING ECLIPSE

To start **rh-eclipse**:

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

```
$ scl enable rh-eclipse 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-eclipse** with support for the **GNU Compiler Collection** and **binutils** from Red Hat Developer Toolset:

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

For detailed instructions on how to install the **devtoolset-7-toolchain** package on your system, see the *Red Hat Developer Toolset User Guide* Section 1.5. Installing Red Hat Developer Toolset.

IMPORTANT

Discard all previous build results for projects previously built with the Red Hat Enterprise Linux version of the **GNU Compiler Collection**. 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-eclipse** 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.

See Also

- Chapter 2, *Changes Since Developer Tools 2018.4* 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 2018.4

Red Hat Developer Tools is distributed with Eclipse 4.10.0 and other plug-ins from the 2018-12 release train which provide a number of bug fixes and feature enhancements over that distributed in Red Hat Developer Tools 2018.4. The following is an abridged 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-eclipse`** collection depends on the **`rh-maven35`** collection. RHEL 7.6 is recommended for the best possible experience.

Significant Package Updates

- **`eclipse`** 4.9.0 → 4.10.0: Eclipse Platform JDT/PDE plug-ins were updated to 4.10.0. The highlights include:
  - Continued improvements to theming and styling.
  - Dropped support for GTK2 as an SWT backend.
  - Support for developing applications targeting Java 11.
  - Code-mining features for the Java editor.
  - Better support for modular projects using the Java Platform Module System.

  For details, see the upstream release notes at https://www.eclipse.org/eclipse/news/4.10/.
- **`eclipse-cdt`** 9.5.3 → 9.6.0: The CDT plug-ins have been updated to 9.6.0 to bring improvements to refactoring and process attachment for debugging purposes.
- **`eclipse-m2e-core`** 1.9.1 → 1.10.0: Eclipse Maven integration was updated to 1.10.0 to fix issues with the Java Platform Module System.
  For details, see the upstream release notes at https://projects.eclipse.org/projects/technology.m2e/releases/1.10.0/bugs.
- **`eclipse-webtools`** 3.11.0 → 3.12.0: The Webtools plug-ins were updated 3.12.0 to add support for JPA 2.2.
  For details, see the upstream release notes at https://projects.eclipse.org/projects/webtools/releases/3.12/bugs.
- **`eclipse-pydev`** 6.5.0 → 7.0.3: The Pydev plug-in was updated to 7.0.3 to bring support for Python 3.7 and drop support for Python 2.4.
  For details, see the upstream release notes: http://www.pydev.org/history_pydev.html.
- **`eclipse-egit`**/**`eclipse-jgit`** 5.1.1 → 5.2.0: The Git integration plug-ins were updated to 5.2.0 to add usability and performance improvements and improve support for GitFlow.
  For details, see the upstream release notes at https://projects.eclipse.org/projects/technology.jgit/releases/5.2.0 and https://projects.eclipse.org/projects/technology.egit/releases/5.2.0.

CHAPTER 3. KNOWN ISSUES IN ECLIPSE 4.10.0

This section details the known issues in Eclipse 4.10.0.

3.1. DEBUGINFO CONFLICTS

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

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

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

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

3.2. CONFLICT BETWEEN `RH-ECLIPSE-ECLIPSE-DEBUGINFO` AND `RH-ECLIPSE-ECLIPSE-CDT-DEBUGINFO` PACKAGES

There is a conflict between the **`rh-eclipse-eclipse-debuginfo`** and **`rh-eclipse-eclipse-cdt-debuginfo`** packages.

Consequence: Only the Eclipse Platform or CDT may be debugged at once.

Workaround: Debug the Eclipse Platform or CDT by removing the conflicting package and installing the necessary **`*-debuginfo`** package.

3.3. 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-eclipse-scldevel`** package (note that this package is not installed by default).

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

3.4. PYDEV USERS MAY EXPERIENCE ISSUES WITH THE `PIP` INTEGRATION

Modules that are pre-installed by the Python SCL cannot be uninstalled; attempts to do so will result in permission denied errors.

3.5. PYDEV SHORTCUT TO MANAGE PYTHON MODULES DOES NOT WORK

Using the **Ctrl+2** shortcut and selecting **`pip`** does not function properly.

Consequence: Attempts to use this shortcut result in an **`SWTException`** in the workspace log.

Workaround: Users can continue to manage their Python modules using the **PyDev** > **Interpreters** preference.

3.6. 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-eclipse-svnkit-cli # Command line support for SVNKit
```

Now, use the **`jsvn`** command anywhere you would normally use the **`svn`** command:

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

3.7. TYCHO CONFLICTS

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

Consequence: The installation of **`rh-eclipse-tycho`** package may fail when the **`rh-eclipse48-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-eclipse48-tycho`** package using the **`yum remove rh-eclipse48-tycho`** command before attempting to install the **`rh-eclipse-tycho`** package.

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

3.8. EGIT AND JGIT ED25519 SSH KEYS DO NOT WORK

A feature in EGit/JGit to support SSH keys using the ED25519 algorithm does not function properly.

Workaround: Users can continue using their existing, non-ED25519 SSH keys by ensuring that the **SSH Client** option is set to **`JSch`** in **Team** > **Git Preferences**.