Essential Eclipse – Self-Study Guide 1

By Gilbert Herschberger (http://www.mindspring.com/~gchii/)
Date Revised: 3 March 2013

Overview

Welcome to the first session of the *Essential EGL+Batch* course presented by `gchii.org` and prepared especially for our on-line students. This course is part of an education and training series.

A typical training session includes a presentation, a "live" demonstration and a hands-on lab for one or more exercises. For the first session, the presentation is *Introduction to Eclipse*. A demonstration illustrates how to decide on the location for a workspace, project and file and use drag-and-drop to rearrange views. During the lab, follow the instructions in *Exercise 1.1 – Introduction to Eclipse* to practice the skills from the presentation and demonstration.

Audience

If you are using Eclipse for the first time, this session is prepared especially for you. It features the EGL Development Tools (or EDT) as an example. EDT is a free-license and open-source product from Eclipse Foundation, contributed by IBM.

This session covers the basic concepts of Eclipse, such as workspace, perspective and view. These concepts are inherent to Eclipse and all Eclipse-based products.

Eclipse has many features. While many features are basic, this session is focused on those which you need immediately or are immediately perplexing.

Target Runtime Environment

Because we are focused on Eclipse by itself, a target runtime environment is beyond the scope of this exercise.

Target Development Environment

For this exercise, the target development environment includes:

- EGL Development Tools 0.8.2

Contents

Part 1 – Eclipse

In this section, we focus on Eclipse itself. To understand Eclipse, it is necessary to understand that Eclipse is a foundation for a software runtime environment, or platform. It is more than a development environment.

Definition: Platform

A platform is an abstract concept. It is anything that one can build upon. A software platform is software that one can build upon, or extend in ways that are limited only by your imagination.

Definition: Development Environment

A development environment is another abstract concept. It includes anything that one needs to develop software. It includes an operating system and administration tools. It includes an editor, compiler, debugger, profiler and other tools. Depending on the target runtime environment, it includes tools for demonstrating and testing software, such as emulators, web and application servers.

In a classic approach, you might assemble a development environment from your favorite tools, such as your favorite text editor and favorite compiler. To compile a program, you switch from your editor (in one window) to a command line (in another window) and enter a build command. Constantly switching from one window to another can become a distraction and lead to human error.

Definition: Interactive Development Environment (or IDE)

An interactive development environment enables a developer to more easily perform common tasks, such as write, compile, debug and run.

In a modern approach, a vendor assembles a development environment from their favorite tools, such as a custom text editor and, of course, a vendor-specific compiler. To compile a program, you press a button or select an option from a menu. You rarely leave the editor window.

Historically, this approach leads us to two serious problems. First, it is difficult to work with multiple operating systems. An IDE might work on only one operating system. When one developer is required to develop software targeted for two operating systems, it is difficult, if not impossible, to find an equally high-quality IDE for multiple operating systems. Constantly switching between operating systems is more of a challenge than simply switching between windows.

Second, it is difficult to work with multiple languages. When you want to work with a certain programming language, you buy an IDE. A programming language is tied to an IDE. When a programmer works with many programming languages, she switched from one IDE to another. An IDE for the COBOL programming language, for example, might not look or feel anything like an IDE for the Java programming language.

We have identified a need for a more universal IDE, one IDE that supports the development of multiple programming languages and multiple operating systems.

Universal Interactive Development Environment

Eclipse is part of a quest for a universal IDE. A developer can work on software for multiple operating systems and programming languages without leaving the comfort of Eclipse. It is language independent. It supports the development of multiple programming languages, such as C/C++, COBOL, EGL, Java and RPG. It supports the development of web applications, using languages such as Cascading Style Sheets (CSS), Hypertext Markup Language (HTML), JavaScript and Extensible Markup Language (XML).

Because it is based upon Java technologies, it is also more platform independent. It works equally as well, for example, on Linux, OS X and Windows.

Eclipse (www.eclipse.org)

In summary of its software architecture, the Java programming language is the primary language used to write Eclipse itself. Other languages are possible. The target runtime environment is Java SE plus OSGi. The free software license is the Eclipse Public License (EPL). The Eclipse Look and Feel Guidelines are strongly recommended when developers are working on Eclipse.

Eclipse.org is an open source project on the Internet. It is maintained by a large group of international volunteers using the Internet to collaborate. The latest version of Eclipse is available from an Internet web site. Add-ons and plug-ins are available from update sites on the Internet.

And something that should not be overlooked is the Eclipse policy that first- and third-party tools use the same standard for packaging and runtime. There is little technical distinction between a component developed by Eclipse and a component developed by individuals and vendors. The standard is OSGi.

Java Standard Edition (www.java.net)

Java technology is a software platform created by Sun Microsystems and owned by Oracle. Java Virtual Machine is an abstract machine (software) specification for real machine (hardware) independence. In other words, a Java program runs in an emulator. In turn, an emulator runs on Linux, OS X, Windows and more.

Standard Edition (SE) is the “desktop” edition, which runs on desktop and laptop hardware. This edition is free software license. At least part of Java technologies, OpenJDK, is open source.

OSGi (www.osgi.org)

OSGi promotes itself as “THE dynamic module system for Java”. It is a specification for a module system for Java, since Java does not have a module system of its own. It was created by a software industry consortium called OSGi Alliance.

Eclipse depends upon Equinox for loading modules. Equinox is the reference implementation of the OSGi specification. An Eclipse workbench, such as EGL Development Tools, has hundreds of OSGi modules.

Eclipse workbench

EGL Development

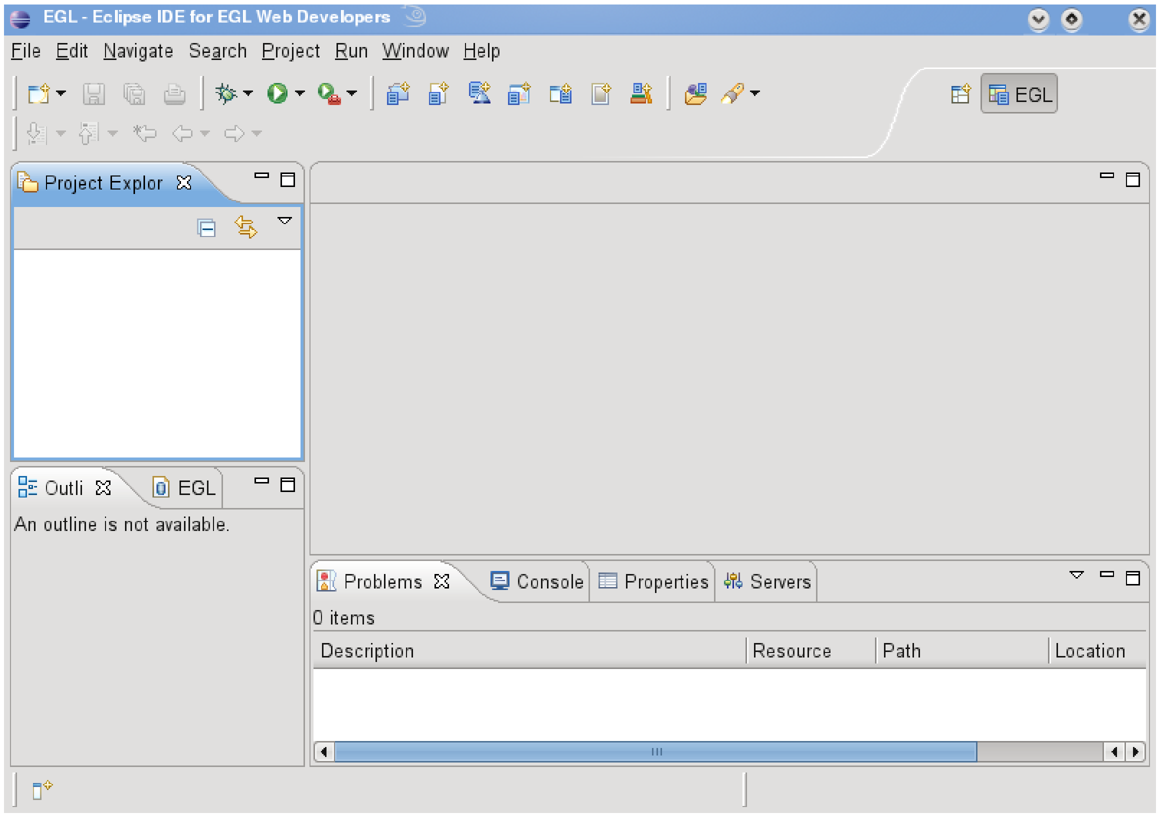
Enterprise Generation Language (or EGL) is a programming language from IBM. To create a development environment for EGL and related programming languages, EGL Development Tools project started with a copy of Eclipse and developed additional OSGi modules.

These products support EGL development.

- EGL Development Tools (EDT) is an Eclipse workbench. It is free software license. It is open-source. This development environment is incomplete; it does not support all features of EGL.
- EGL Community Edition (or EGL-CE) is a Rational workbench. It is free software license. While it is not yet open source, IBM has contributed features of EGL-CE to the EGL Development Tools (EDT) project. This development environment is also incomplete; it does not support all of features of EGL.
- Rational Business Developer (RBD) is a Rational workbench. It is paid software license. It includes a complete implementation of EGL.
- RDi SOA is (or was) a suite of Rational products. It is paid software license. It includes Rational Business Developer.
- Rational Developer for z with EGL is a suite of Rational products. It is paid software license. It also includes Rational Business Developer.

Workbench

Eclipse displays a lot of information in a single window.


EGL - Eclipse IDE for EGL Web Developers
File Edit Navigate Search Project Run Window Help
EGL
Project Explor
Outli
EGL
Problems
Console
Properties
Servers

An outline is not available.

0 items

Description	Resource	Path	Location

The following are displayed in the main window.

- Title bar. Across the top, the standard title bar is displayed following the look of your operating system.
- Menu. Below the title bar, options on a pull-down menu are context-sensitive.
- Tool bar. Below the menu, icons represent actions.
- Views. Between the tool bar and status bar, there is a general-purpose area for editors and views.
- Status bar. Across the bottom, indicators and messages display the current status.

EGL Development Tools (or EDT)

In this course, we feature EGL Development Tools v0.8.2. The EGL Development Tools project started with Eclipse Indigo (or v3.6). It includes free license software from a catalog of OSGi-compatible modules.

EDT can be installed as a full copy of Eclipse, from a ZIP-compatible archive. Or it can be installed as a plug-in to Eclipse.

Updates to EDT are installed from an Internet update site. The official update site is configured during installation, when possible.

Other add-ons and plug-ins can be installed with EGL-CE. A plug-in must be compatible with Eclipse Indigo (3.6) or higher.

More EGL Development Tools

As you may remember, this development environment is incomplete; it does not support all of features of EGL. It supports generation to

- HTML/JavaScript, and
- Java.

It is possible, for example, to generate an existing EGL text user interface (TUI) application to Java; EDT provides support for creating a "Batch" project.

It supports a Java EE application server runtime.

- J2EE Preview
- Apache Tomcat 6.0

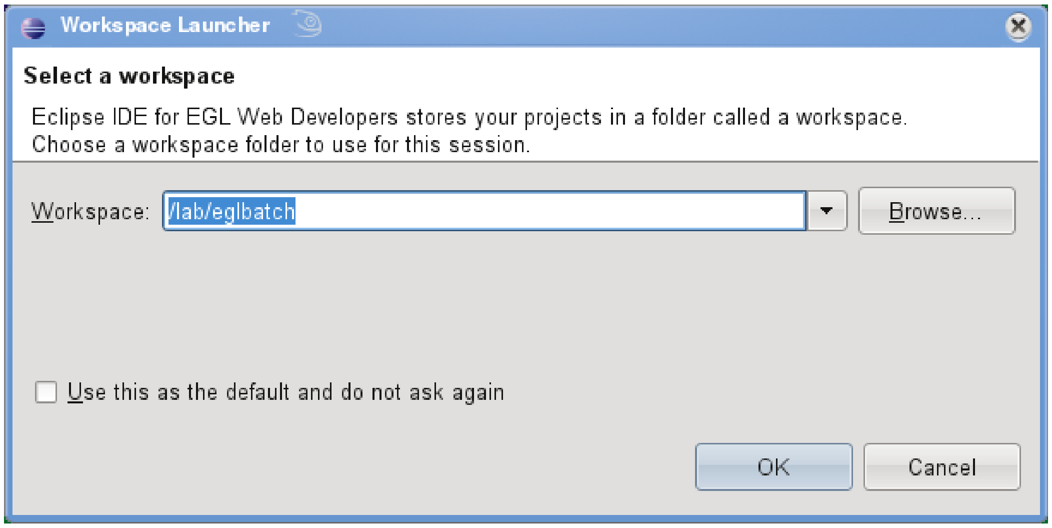
Note: It might be possible to use IBM WebSphere Application Server Community Edition, within the limitations of the tools. If you must use WebSphere Application Server or WebSphere Application Server Express, a paid license for Rational Business Developer is your only choice. It is forbidden by the EGL Development Tools software license.

It supports any JDBC-compatible database server.

- Derby
- IBM DB2
- MySQL
- SQL Server

Workspace

When you start Eclipse, the Workspace Launcher dialog is displayed.



Select a workspace

Eclipse IDE for EGL Web Developers stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: /lab/eglbatch

Definition: Workspace

What is a workspace? What is a good location for a workspace?

A workspace is an abstract concept. First, it is literally file storage on your local file system. You choose a top-level folder. For best results, use as short a name as possible for your top-level directory. Within a workspace, the full path to a file can be unusually long and complicated. If the path to your workspace is also long, Eclipse must wait longer for operations on the file system.

For best performance and safety, use a permanent and local drive. The workspace includes a top-level folder and all of its files and subfolders.

***Caution:* Do not store your workspace on a removable or network drive.** If you were to store your workspace on a network drive, for example, and network traffic is slightly delayed, your workspace can become corrupted and you lose your work.

Second, many things are associated with a workspace. A workspace is a collection of preferences, capabilities and projects. Your personal preferences and preferences for a project are separate settings for each workspace.

Finally, within a workspace, development effort is organized by project. A workspace enables you to work with a collection of projects. Project files are stored in the workspace.

Preferences

Eclipse enables you to fully customize your workbench. Preferences for your workbench are stored in a workspace. When creating another workspace, you can start with factory default settings or settings from an existing workspace. There are specific instructions on how to clone a workspace.

As a general rule, settings are organized by feature. When there are preferences for each feature and so many features, there can be many preferences. When using the Eclipse workbench, it can be perplexing to understand how to explore preferences and change them. In *Exercise 1.1*, we explorer a few preferences.

Capability

A capability is a collection of features and other components. With capabilities, individual features are grouped together so that they may be activated (and deactivated) as a group. For example, the following are capabilities.

- C/C++ Developer
- COBOL Developer
- Java Developer
- Java EE Developer
- Web Developer
- EGL Developer

When you are working exclusively on an EGL project, you may not want to see features for a C/C++ and COBOL project. With capabilities, you can hide unused (and unwanted) features.

Project

Eclipse supports many different kinds of projects for different target runtime environments. Each projects become a subfolder in a workspace. A project has folders and files. You use a project to create and modify files.

Eclipse supports a long list of import and export formats. To copy files from your operating system to a project, use import. To copy files from a project to your operating system, use export.

Caution: While you could switch to another tool, importing and exporting files from your workspace is the Eclipse way of doing things.

Use the Archive File (ZIP) format to back-up, restore or share a project. Use source control management to share a project within a development team.

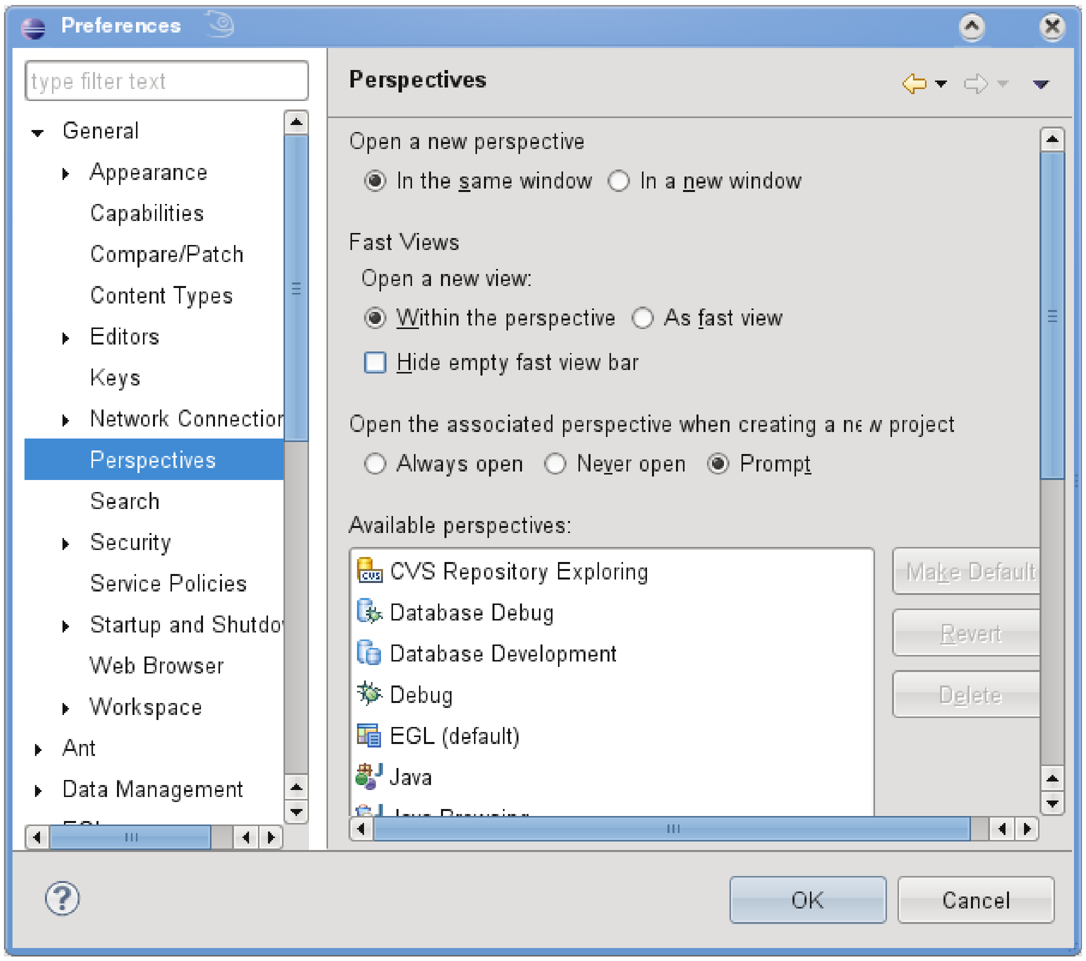
While Eclipse supports many different kinds of projects, building a project is a nearly universal way to produce artifacts.

1. Building a C/C++ project invokes a C/C++ compiler;
2. Building a Java project invokes the Java compiler; and,
3. Building an EGL project invokes the EGL generator and Java compiler.

As you have seen, a workspace is a fundamental part of working with Eclipse. Next, let's take another look at perspectives.

Part 2 – Perspective

A perspective is an abstract concept. It is a setting in your workspace. You can save and delete perspectives. A list of perspectives is displayed in the General | Perspectives folder in the Preferences dialog box.



Preferences

type filter text

- General
  - Appearance
  - Capabilities
  - Compare/Patch
  - Content Types
  - Editors
  - Keys
  - Network Connectior
  - Perspectives
  - Search
  - Security
  - Service Policies
  - Startup and Shutdo
  - Web Browser
  - Workspace
- Ant
- Data Management

Perspectives

Open a new perspective

[✓ In the same window] In a new window

Fast Views

Open a new view:

[✓ Within the perspective] As fast view

Hide empty fast view bar

Open the associated perspective when creating a ne w project

Always open Never open [✓ Prompt]

Available perspectives:

- CVS Repository Exploring
- Database Debug
- Database Development
- Debug
- EGL (default)
- Java

Definition: Perspective

A perspective is task-oriented. It is distracting and unwanted for irrelevant information to be displayed at the same time. The following illustrate some common programming tasks.

- Designing. When designing a form or page, a database or table, a developer is focused on design-time tasks.
- Developing. When writing lines of Java source code, for example, a developer is focused on programming language tasks.
- Debugging. When looking for one character in a program that is causing a intermittent bug, a developer is focused on debugging tasks.
- Testing. When running unit tests, stress tests, performance tests, a developer is focused on monitoring a test and test results.
- Checking files into source control management. As soon as a problem is resolved, a developer

is focused on committing changes to source control management.

A perspective is a collection of views. While Eclipse provides a long list of views, you are unlikely to want to see them all at the same time. A perspective determines which views are displayed.

A perspective is an arrangement of views. When a view is displayed, a perspective determines where is is displayed, how wide and how tall.

Resize

Eclipse provides features to resize a view. With a mouse, you can drag the edge of a view to resize it. Its new location is saved in your workspace. Even if you were to leave the workspace and return, the view has the same size.

Rearrange

Eclipse provides features to rearrange views. With a mouse, you can drag the tab for a view and drop it in a new location. Even if you were to leave the workspace and return, the arrangement of views is preserved.

Perspective Icons

There are two essential icons for working with perspectives.

- Open Perspective ( ) icon. Select a perspective to open.
- Switch Perspective ( ) icon. Switch quickly between open perspectives.

Explore Perspectives

To get a better idea of what a perspective is, take a quick look at some the perspectives in your Rational workbench.

- CVS Repository Exploring
- Database Development
- Debug
- EGL
- Java
- Team Synchronizing
- Web

Reset Perspective

It is possible to rearrange a perspective to the point that we can no longer get our work done. Fortunately, there is a feature to reset a perspective back to its factory settings. The feature is called Reset Perspective.

As you have seen, a perspective is a fundamental part of working with Eclipse. After considering workspace and perspective, our last fundamental concept is a view.

Part 3 – View

We have mentioned the view from the beginning of our discussion about Eclipse. In this part, we focus on views and define them more formally.

Definition: View

A view is an abstract concept.

- It is always specialized. It does exactly one thing.
- It provides the user interface. It responds to the keyboard and mouse.
- It is part of the presentation layer. In Model/View/Controller separation of concerns, a view is responsible for displaying data from a model.
- A view may be read-only, displaying data. A view may be an editor, displaying and modifying data.
- All views are integrated. To display a consistent view of the workspace, Eclipse is event driven. A view listens for events that might affect what it currently displays. A view might send an event in response to the keyboard or mouse.

View icons

The following are essential icons for a view.

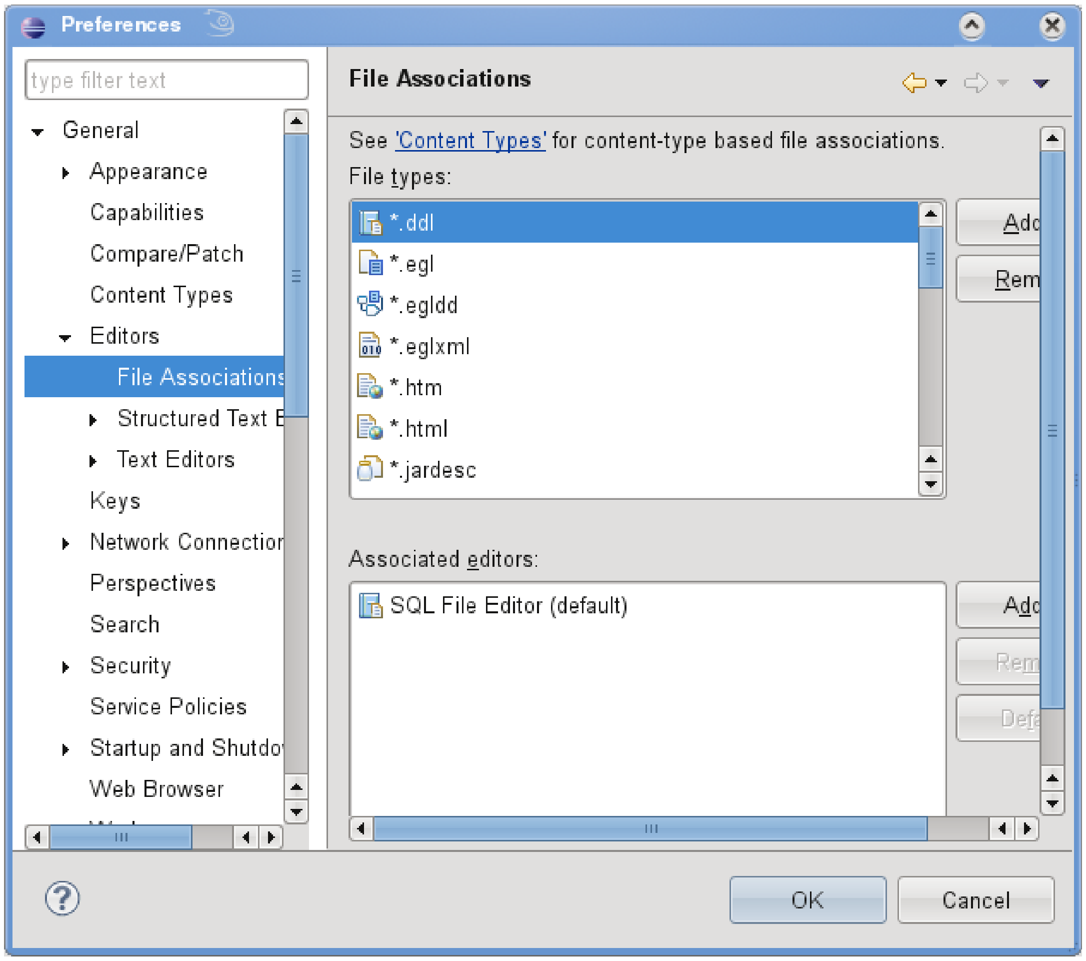
- Minimize (🗕) icon. Reduce to the size of an icon.
- Restore (🗗) icon. Switch a view to its normal size.
- Maximize (🗖) icon. Increase to the size of the general area, if possible.
- Save (💾) icon. Save recent changes to disk.
- Close (✖) icon. Remove a view or editor.

Editors

An editor (usually) enables you to modify the content of a file. Eclipse includes many editors. An editor is specialized. Different editors are used for different types of files. When selected automatically, an editor is selected based upon a file extension. You can manually select an editor for a file, using the Open With option.

File Associations

The association between a file extension and an editor is part of the settings for your workspace. You can add, change and remove an association.



File Associations

See 'Content Types' for content-type based file associations.

File types:

*.ddl
*.egl
*.egldd
*.eglxml
*.htm
*.html
*.jardesc

Associated editors:

SQL File Editor (default)

Explore Views

To get a better idea of what a view is, take a quick look at some the perspectives in your Eclipse workbench.

- Debug
- Outline
- Problems
- Properties
- Servers
- Tasks
- Team
- Variables
- Work Items

Summary

Thanks for your participation in the first session in this series. As we have seen, workspace, perspective and view are the fundamental concepts of Eclipse. We have introduced EGL Development Tools as an Eclipse-based product. Our next step is to start the hands-on lab.

After the lab, we continue with *Essential EGL*, our next session in this series.

Lab

Follow the instructions in *Exercise 1.1 – Introduction to Eclipse* in this series. Like the presentation, the exercise is divided into three parts.

1. Workspace
2. Perspective
3. View