## Eclipse

Eclipse es una comunidad de código abierto cuyo objetivo es desarrollar una plataforma para el desarrollo del software. Esta comunidad recibe el nombre formal de Eclipse Foundation, o Fundación Eclipse.

La plataforma desarrollada por la Fundación Eclipse se llama también Eclipse, formalmente, Eclipse Plattform o Plataforma Eclipse. Está desarrollada en el lenguaje Java y su código se distribuye bajo una licencia propia, la EPL, Eclipse Public License, muy similar a las licencias de la fundación Apache.

Eclipse proporciona una plataforma de software para la ejecución de aplicaciones basadas en una arquitectura de plugins y features, que son agrupaciones de plugins.

Eclipse se conoce sobre todo como un IDE, Entorno de Desarrollo Integrado, para la plataforma Java, pero la plataforma ha ido creciendo soportando multitud de lenguajes y multitud de nuevas características.

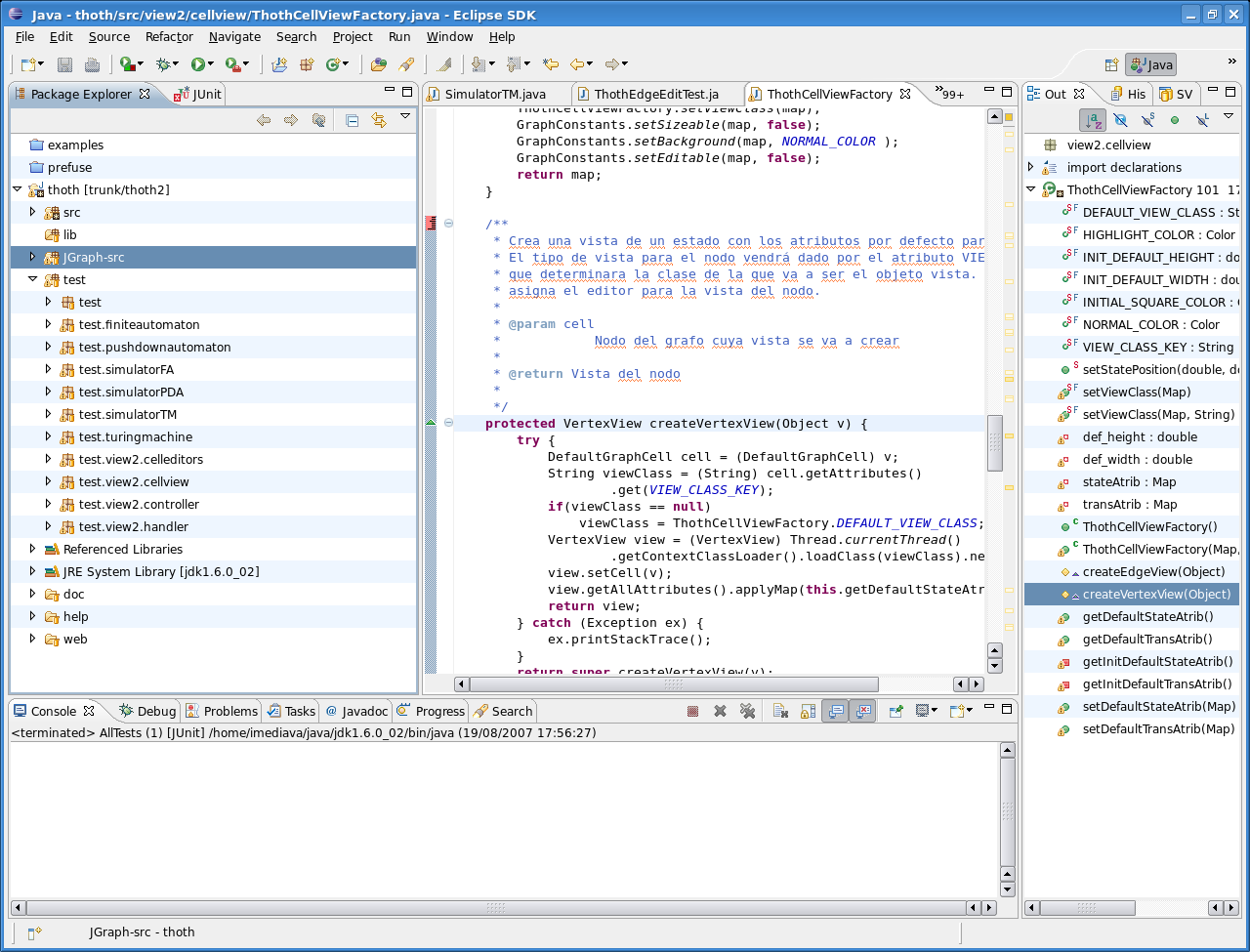


Ilustración 6.3: Entorno de desarrollo Eclipse

La distribución de Eclipse por defecto, denominada formalmente Eclipse SDK, solo incluye el soporte para lenguaje Java, pero existen ya distribuciones que incluyen soporte para otros lenguajes y otras tecnologías.

Eclipse reúne características muy interesantes tanto para las empresas y los particulares. El ser software de código abierto, el ser gratuito, la organización de la comunidad y el soporte de grandes empresas suponen una garantía de soporte para la plataforma tanto en el presente como en el futuro, y lo convierten en una opción muy competitiva.

### Componentes principales de la plataforma Eclipse

#### Eclipse Plattform (Plataforma Eclipse)

Eclipse Plattform es el nombre del proyecto bajo el que se desarrolla el núcleo de la plataforma Eclipse y se divide en varios proyectos separados. Proporciona componentes básicos para la plataforma y para el IDE que son los siguientes:

* Core: El núcleo de la plataforma que proporciona soporte para el manejo de recursos y el entorno de ejecución.
* Ant: Integración de la plataforma con Ant, el sistema de construcción de proyectos del software preferido para la comunidad de desarrollo Java.
* Debug: Entorno genérico para la depuración.
* Team & Compare: Framework genérico para trabajo en equipo y control de versiones y para la comparación de recursos.
* CVS (Control Version System): Integración con el sistema de control de versiones CVS.
* Search: Framework de búsqueda.
* Release Engineering: Componente de gestión de publicaciones de versiones o releases
* SWT (Standard Widget Toolkit): Toolkit gráfico sobre el que se desarrolla la interfaz de la plataforma.
* User Assistance: Framework de ayuda al usuario.
* User Interface: Módulo para la interfaz de usuario.
* Update: Módulo de actualización e instalación de complementos.
* WebDAV:Integración con WebDAV.

#### SWT (Standard Widget Tookit)

SWT es un toolkit gráfico que se utiliza en la interfaz gráfica de Eclipse. A pesar de estar desarrollado en Java, Eclipse no utiliza el toolkit gráfico por defecto de Java, que es Swing.

SWT utiliza librerías nativas de cada plataforma soportada, por lo que la apariencia de Eclipse se adapta a cada uno de los sistemas operativos soportados y mejora la velocidad de ejecución.

Los toolkits gráficos soportados actualmente son GTK+ y Motif, para sistemas Linux, HP-UX, AIX y Solaris, Windows, y Windows CE para sistemas Microsoft Windows, Photon, para sistemas QNX y Carbon para Mac OSX.

SWT es un proyecto muy importante dentro de la plataforma, pues se ha convertido en una alternativa muy válida para el desarrollo de aplicaciones de escritorio en Java. Se distribuye como una librería independiente, con implementación para cada uno de los proyectos.

En muchos proyectos, SWT se elige preferiblemente a Swing, debido a que las máquinas virtuales libres no tienen una implementación funcional de Swing, mientras que SWT es de código abierto.

#### JDT (Java Development Tools)

JDT es la base del IDE para desarrollo en Java. Actualmente, Eclipse se considera uno de los mejores entornos de desarrollo para Java y además gratuito y de código abierto.

JDT incluye todas las características que facilitan el desarrollo en este lenguaje.

Algunas de las características más importantes que comprenden son:

1. Un compilador incremental para Java, una característica única que permite que el código sea compilado mientras se escribe. El compilador incremental hace innecesario el proceso de compilación y permite detectar todos los errores y warnings en tiempo de compilación mientras se escribe el código.
2. Asistentes de código para autocompletar, para solucionar los problemas automáticamente y generación de código para métodos de establecimiento y obtención, métodos delegados, constructores, bloques try catch, etc.
3. Un modelo en forma de árbol para los elementos Java y un API para acceder a él, incluyendo todos los elementos que se encuentran en el classpath.
4. Una base de datos indexada con todos los elementos para realizar búsquedas, para asistentes de código y selección de código.
5. Sistema avanzado de Refactorización que incluye renombrado de elementos, extracción de métodos, cambio de signatura, migración de versión de las librerías y muchos otros.
6. Asistentes de creación de clases y paquetes.
7. Creación automática y comprobación de comentarios Javadoc.
8. Manejo de los imports inteligente. Ordenación y gestión automática.
9. Formateo automático del código. Ayuda en línea de documentación Javadoc.
10. Sistema de Vías de construcción que permite especificar librerías externas, archivos compilados .class, bibliotecas, variables, etc. Permite navegar el contenido de las librerías y acceder a su código fuente si establecemos su código fuente.
11. Vista jerárquicas de herencia con descubrimiento de clases heredadas o que heredan.
12. Sistema de ejecución y depuración. El sistema de depuración permite la ejecución en diferentes máquinas virtuales, cambiar código en caliente para inyectarlo sin reiniciar y el análisis de expresiones y variables.

**Workbench**

The term *Workbench* refers to the desktop development environment. The Workbench aims to achieve seamless tool integration and controlled openness by providing a common paradigm for the creation, management, and navigation of workspace resources.

Each Workbench window contains one or more perspectives.  Perspectives contain views and editors and control what appears in certain menus and tool bars.  More than one Workbench window can exist on the desktop at any given time.

# Perspectives

Each Workbench window contains one or more perspectives. A perspective defines the initial set and layout of views in the Workbench window. Within the window, each perspective shares the same set of editors. Each perspective provides a set of functionality aimed at accomplishing a specific type of task or works with specific types of resources. For example, the Java perspective combines views that you would commonly use while editing Java source files, while the Debug perspective contains the views that you would use while debugging Java programs. As you work in the Workbench, you will probably switch perspectives frequently.

Perspectives control what appears in certain menus and toolbars. They define visible *action sets*, which you can change to customize a perspective. You can save a perspective that you build in this manner, making your own custom perspective that you can open again later.

You can use the [[http://help.eclipse.org/helios/topic/org.eclipse.help/command_link.png](javascript:executeCommand(%22org.eclipse.ui.window.preferences(preferencePageId=org.eclipse.ui.preferencePages.Perspectives)%22))**General > Perspectives**](javascript:executeCommand(%22org.eclipse.ui.window.preferences(preferencePageId=org.eclipse.ui.preferencePages.Perspectives)%22)) preference page to open perspectives in the same window or in a new window.

# Editors

Most perspectives in the Workbench are comprised of an editor area and one or more views.

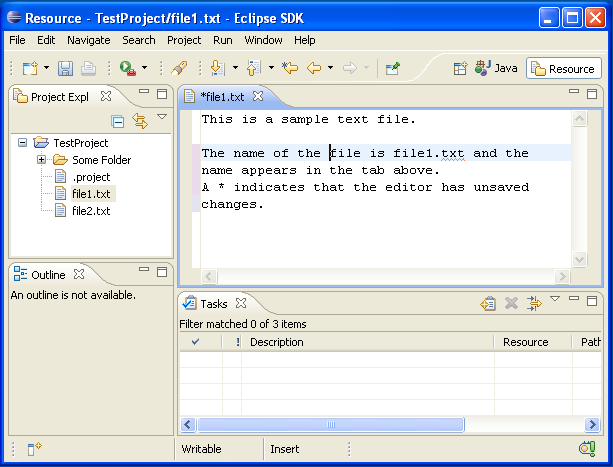
You can associate different editors with different types of files. For example, when you open a file for editing by double-clicking it in one of the navigation views, the associated editor opens in the Workbench. If there is no associated editor for a resource, the Workbench attempts to launch an *external editor* outside the Workbench. (On Windows, the Workbench will first attempt to launch the editor in place as an OLE document. This type of editor is referred to as an *embedded editor*. For example, if you have a .doc file in the Workbench and Microsoft Word is registered as the editor for .doc files in your operating system, then opening the file will launch Word as an OLE document within the Workbench editor area. The Workbench menu bar and toolbar will be updated with options for Microsoft Word.)

Any number of editors can be open at once, but only one can be active at a time. The main menu bar and toolbar for the Workbench window contain operations that are applicable to the active editor.

Tabs in the editor area indicate the names of resources that are currently open for editing. An asterisk (\*) indicates that an editor has unsaved changes.

By default, editors are stacked in the editor area, but you can choose to tile them in order to view source files simultaneously.

Here is an example of a text editor in the Workbench:



 The gray border at the left margin of the editor area may contain icons that flag errors, warnings, or problems detected by the system. Icons also appear if you have created bookmarks, added breakpoints for debugging, or recorded notes in the Tasks view. You can view details for any icons in the left margin of the editor by moving the mouse cursor over them.

# Views

Views support editors and provide alternative presentations as well as ways to navigate the information in your Workbench.  For example, the Project Explorer and other navigation views display projects and other resources that you are working with.

Views also have their own menus. To open the menu for a view, click the icon at the left end of the view's title bar. Some views also have their own toolbars. The actions represented by buttons on view toolbars only affect the items within that view.

A view might appear by itself, or stacked with other views in a tabbed notebook. You can change the layout of a perspective by opening and closing views and by docking them in different positions in the Workbench window.

Citar la página web con la documentación de eclipse 3.6