Laboratorio de Informática
Ingeniería de Sistemas
Escuela Colombiana de Ingeniería Julio Garavito

Inducción al Tutorial de Eclipse

Este tutorial trata de mostrar las ventajas de utilizar Eclipse para programar Java. Sirve como una guía detallada para usuarios novatos. No es necesario un conocimiento avanzado o específico de Java para poder seguir este tutorial. En unos pocos minutos, será capaz de dominar las técnicas principales de Eclipse, lo que acelerará la programación e incrementará la productividad.

El único prerrequisito que debe cumplirse antes de comenzar este tutorial es tener instalada cualquier versión de Eclipse (igual o superior a la 2.0), junto con una máquina virtual de Java. De hecho, este tutorial está optimizado para la versión 2.1.2 de Eclipse, pero es también válido para Eclipse 3.0. Para obtener información acerca de estos elementos, visite la sección de "Enlaces".

Última actualización: 17 de octubre de 2004. Añadida una lista de teclas rápidas de Eclipse en la sección de "Enlaces".

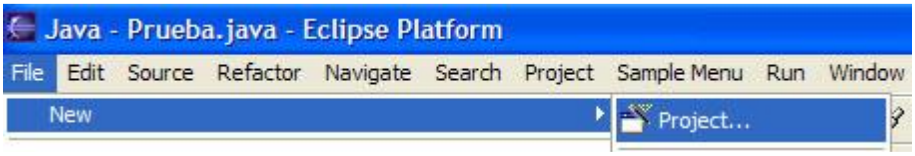
Tabla de contenido

Parte 1: Entorno de proyectos Eclipse

"La ciencia es conocimiento organizado. La sabiduría es la propia vida organizada." Immanuel Kant.

En el entorno de desarrollo Eclipse todo archivo se almacena dentro de un proyecto. Esto quiere decir que todo documento, carpeta, archivo de código fuente (.java) y código compilado (.class) tiene que estar contenido dentro de un proyecto. Así pues, el primer paso antes de usar Eclipse para programar en Java es comprender la estructura de proyectos de Eclipse.

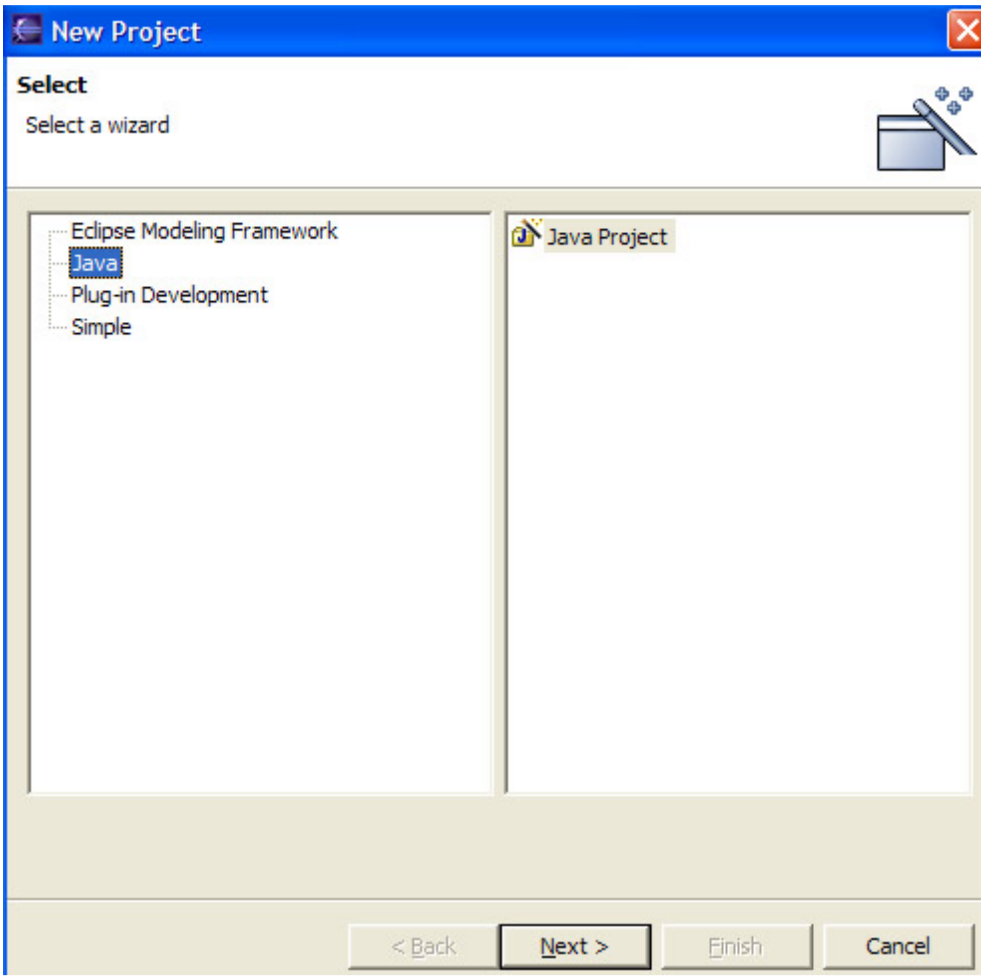
Es necesario crear un nuevo proyecto no sólo para desarrollar un nuevo programa de Java, sino para editar también archivos ya existentes (como, por ejemplo, un programa ".java" almacenado en un diskette). Para crear un nuevo proyecto, seleccione en la línea de menús principal "File > New > Project...". También es posible seleccionar "New > Project..." haciendo clic derecho en cualquier parte una vista de Eclipse (como, por ejemplo, el Package Explorer o el Resource Navigator).



Hay varios tipos de proyectos de Eclipse que pueden ser creados:

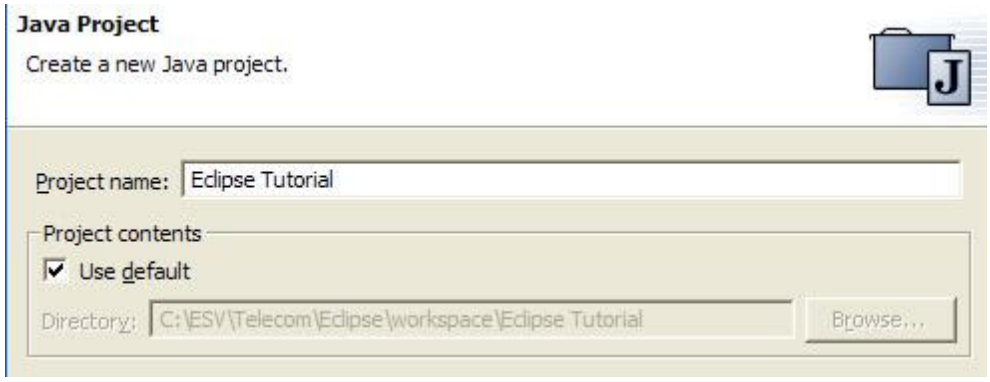
- Para crear o editar programas Java, un "Java Project" debería ser creado. Nótese que dentro de un proyecto de Java también puede almacenarse toda la información relacionada con el proyecto (la cual no tiene por qué reducirse estrictamente a código fuente, sino que también puede contener documentación y otros archivos relacionados).
- Los "Simple Project" sólo deberían crearse para almacenar documentos y otros archivos, pero no código Java que se desee compilar. Por ello, siempre que se creen archivos ".java" sería recomendable crear un "Java Project".
- Los "Plug-in Development Project" se usan para añadir nuevos módulos y funciones al entorno Eclipse. Estos proyectos sólo son necesarios si se es un desarrollador de Eclipse.
- Los proyectos de "EMF" se utilizan para crear modelos de análisis y diseño.

Puesto que este tutorial se centra en utilizar Eclipse para programar en Java, escogeremos crear un nuevo "Java Project" seleccionando "Java > Java Project" y haciendo clic en el botón "Next >" del asistente de creación.

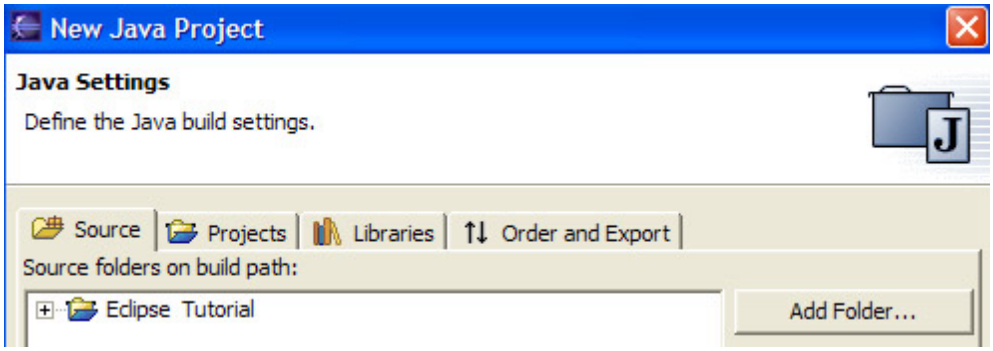


Tras esto, debe especificarse un nuevo nombre para el proyecto. Los proyectos profesionales se nombran utilizando nombres de dominio dados la vuelta para evitar posibles conflictos de nombrado. Sin embargo, para un proyecto de uso interno cualquier nombre descriptivo será suficiente. En este ejemplo, se ha llamado al nuevo proyecto "Eclipse Tutorial".

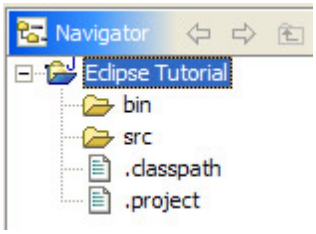
Si se usa el directorio por defecto, los archivos del proyecto se almacenarán en el directorio "[DirectorioDeInstalación]\workspace\[NombreProyecto]". También es posible especificar un directorio diferente en el que guardar dichos contenidos.



Pese a que en este punto es ya posible terminar la creación del nuevo proyecto haciendo clic en el botón "Finish", es recomendable pulsar de nuevo el botón "Next >" para definir explícitamente de esta manera una carpeta fuente para los archivos ".java" desde el principio. Las carpetas fuente ("Source Folders") almacenan los archivos de código fuente de Java (.java), de manera que Eclipse sepa dónde encontrarlos y pueda realizar la compilación automática de los mismos cada vez que un archivo sea guardado.

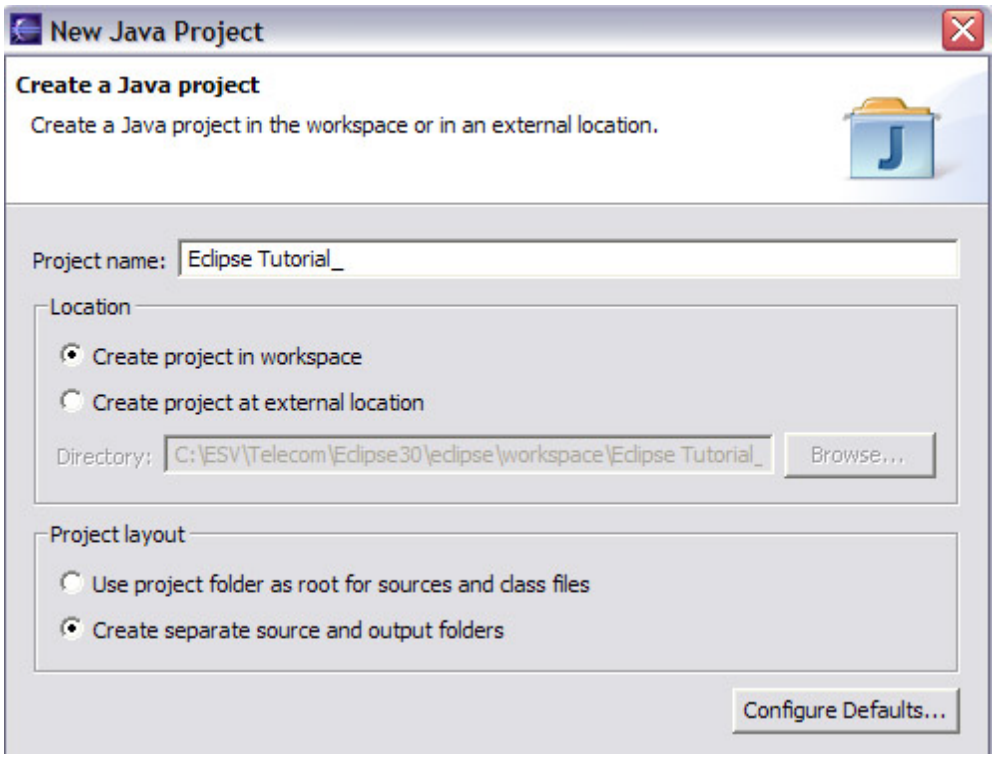


Para crear dicha carpeta fuente seleccione la pestaña de "Source" y pulse el botón "Add Folder...". Seleccione el proyecto recientemente creado y pulse el botón "Create New Folder". Un buen nombre para esta carpeta fuente podría ser sencillamente "src". Tras especificar dicho nombre, pulse "Ok". A la pregunta de si desea actualizar la carpeta de salida de archivos compilados, debería responder afirmativamente. De este modo, los archivos ".class" que resulten de la compilación de los ".java" almacenados en la carpeta fuente irán a parar a la carpeta "\bin". Pulse el botón "Finish" para terminar el proceso de creación del nuevo proyecto. Si se nos pregunta si deseamos pasar a la perspectiva de Java, deberíamos responder afirmativamente.

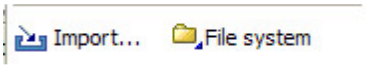


El proyecto recién creado debería parecerse a este, tal y como lo muestra la vista "Navigator".

En Eclipse 3.0 la creación de proyectos es un tanto diferente a Eclipse 2.x. Aún puede definirse una carpeta cualquiera para almacenar los proyectos. Pero ahora los pasos anteriormente expuestos acerca de la creación de una estructura adecuada de carpetas fuente y destino pueden omitirse, ya que en la nueva versión de Eclipse basta con marcar la opción "Create separate source and output folders" para construir de forma automática la estructura de archivos deseada.

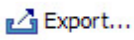


Como ya se mencionó anteriormente, para editar un elemento existente debería ser primero importado dentro de un proyecto de Eclipse. Esto puede hacerse desde el menú "File > Import..." o bien pulsando con el botón derecho en cualquier punto de la vista del "Package Explorer" o del "Resource Navigator". Para seleccionar un archivo o directorio hay que seleccionar "File system" en el submenú de importación. Entonces sólo habrá que recorrer los directorios marcando los archivos que se deseen importar, así como el proyecto y la carpeta destino.



Es importante tener en cuenta que todo elemento que se importa en Eclipse se duplica. Esto significa que borrar la copia que Eclipse esté manejando no borrará el archivo original, de manera que se seguirá teniendo una copia de reserva. Sin embargo, si se usa la opción de importar un proyecto de Eclipse ya existente, los contenidos de dicho proyecto serán duplicados. Así que hay que ser especialmente cuidadoso al borrar proyectos importados del entorno de trabajo de Eclipse, ya que es posible que otras copias de backup de dicho proyecto no existan.

Todo archivo creado mediante la utilización de Eclipse puede ser exportado como un archivo normal (seleccionando Export... > File System), como un archivo ".jar" e incluso como archivos comprimidos en ".zip". El proceso que hay que seguir es similar al proceso recientemente explicado para importar archivos, sólo que seleccionando ahora la opción "Export".



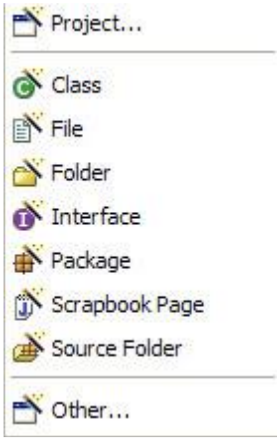
También es posible copiar, cortar y pegar archivos y carpetas desde Eclipse hasta los directorios del sistema operativo (por ejemplo, del explorador de Windows) y viceversa (seleccionando un archivo, pulsando CTRL+C, seleccionando la carpeta destino dentro de Eclipse y pulsando CTRL+V).

Parte 2: Crear Elementos de java

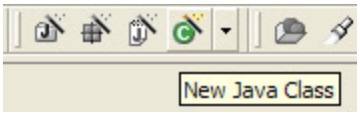
"Hasta el viaje más largo comienza con un solo paso." Proverbio chino.

Una vez que se ha creado un nuevo proyecto, es hora de aprender cómo crear los diversos elementos de Java. Para seguir los pasos que se exponen a continuación es necesario cambiar a la "Perspectiva Java" (si no se ha hecho ya) seleccionando "Window > Perspectives > Java". La perspectiva de Java contiene las vistas y editores más útiles a la hora de crear nuevos programas en Java.

Haciendo clic derecho en la carpeta fuente recientemente creada (por ejemplo, "src") dentro de la vista del Package Explorer, aparecerá un menú contextual. Seleccionando "New >" en dicho menú se mostrará una lista con los diversos elementos de Java que pueden ser creados.

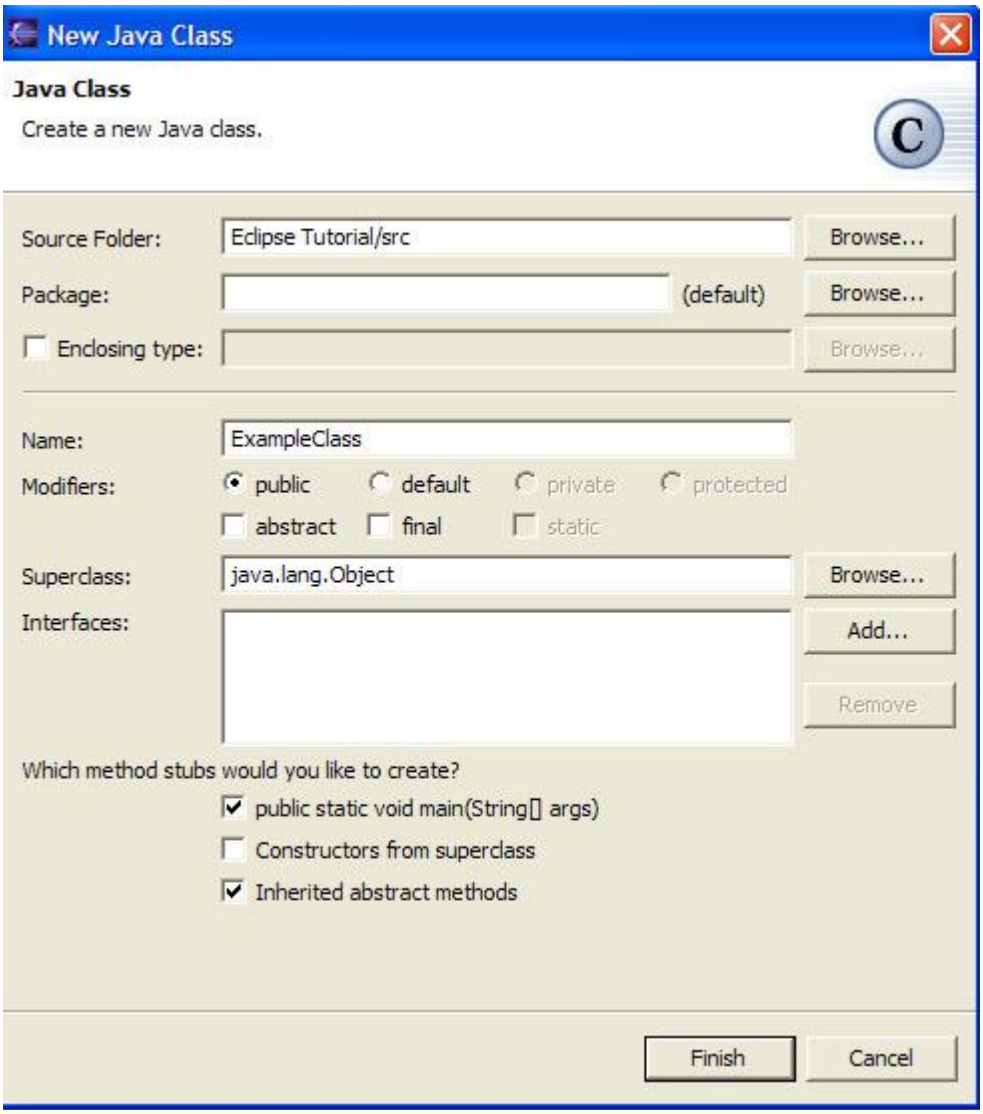


Algunos de dichos elementos pueden ser creados también realizando clic sobre los iconos del menú de la parte superior de la pantalla.



Java Class

Las clases de Java son los archivos ".java" que contienen el código fuente y que serán posteriormente compilados en archivos ".class". Estos archivos Java tienen que ser almacenados dentro de la carpeta fuente recientemente creada (por ejemplo, "src"). Pulse "New > Class" para abrir la ventana de creación de clases.



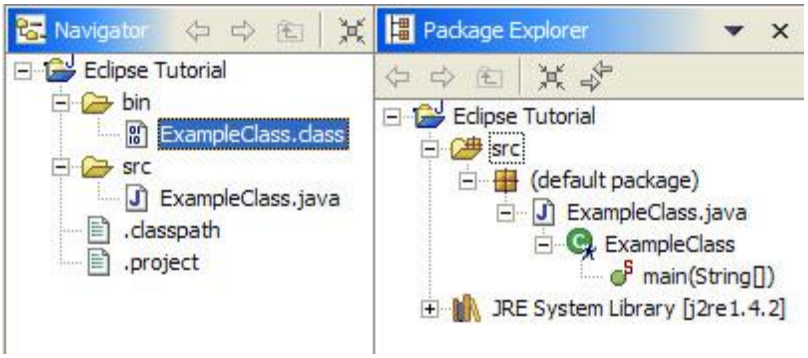
La carpeta fuente especificada debería ser la carpeta recientemente definida (src). Si no se especifica ningún paquete para contener las clases Java, se guardarán dentro de un paquete por defecto. El último campo obligatorio que debiera ser rellenado antes de proceder a la creación de la clase Java es el propio nombre de la clase. Las convenciones de Java sugieren que el nombre de una clase debería comenzar con mayúscula. En este tutorial se ha usado el nombre "ExampleClass".

También existen otros modificadores que pueden ser fácilmente añadidos a una clase desde el mismo momento de su creación. Sin embargo, estos elementos también podrían ser añadidos manualmente en otras fases más avanzadas del proceso de desarrollo. Si se pretende que la nueva clase extienda (herede de)

otra clase existente, se debería especificar la clase "padre" dentro del campo "Superclass". El botón "Browse..." es de gran utilidad a la hora de encontrar clases que sean posibles candidatas para ser extendidas. Aunque Java sólo soporta herencia única (sólo puede extenderse de una única clase) sí que es posible que una clase implemente más de una interfaz. Una vez más, el botón "Browse..." simplifica la tarea de seleccionar interfaces implementadas.

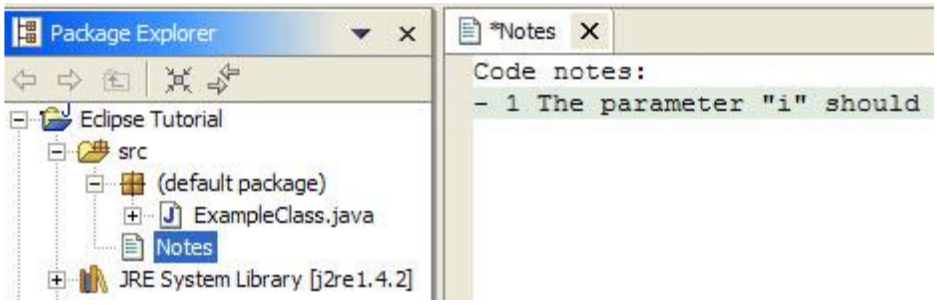
Si se desea que la nueva clase contenga un método "main" (es decir, el punto inicial de ejecución del programa), puede añadirse dicho método automáticamente sólo con marcar la casilla con la opción apropiada. También pueden implementarse de esta manera los constructores de la superclase y todos los métodos abstractos heredados. Esta última opción es muy útil si se desea instanciar la clase puesto que para esto todo método abstracto debería estar implementado.

Es interesante destacar que los archivos compilados ".class" sólo son visibles en la ventana "Navigator", abierta por defecto dentro de la perspectiva "Resource". Puesto que la perspectiva de Java no abre esa ventana por defecto, los ficheros .class no serán visibles en la vista del Package Explorer. Sin embargo, basta con escribir y guardar un archivo ".java" para que se cree un archivo ".class" resultante de compilar el archivo fuente anterior.



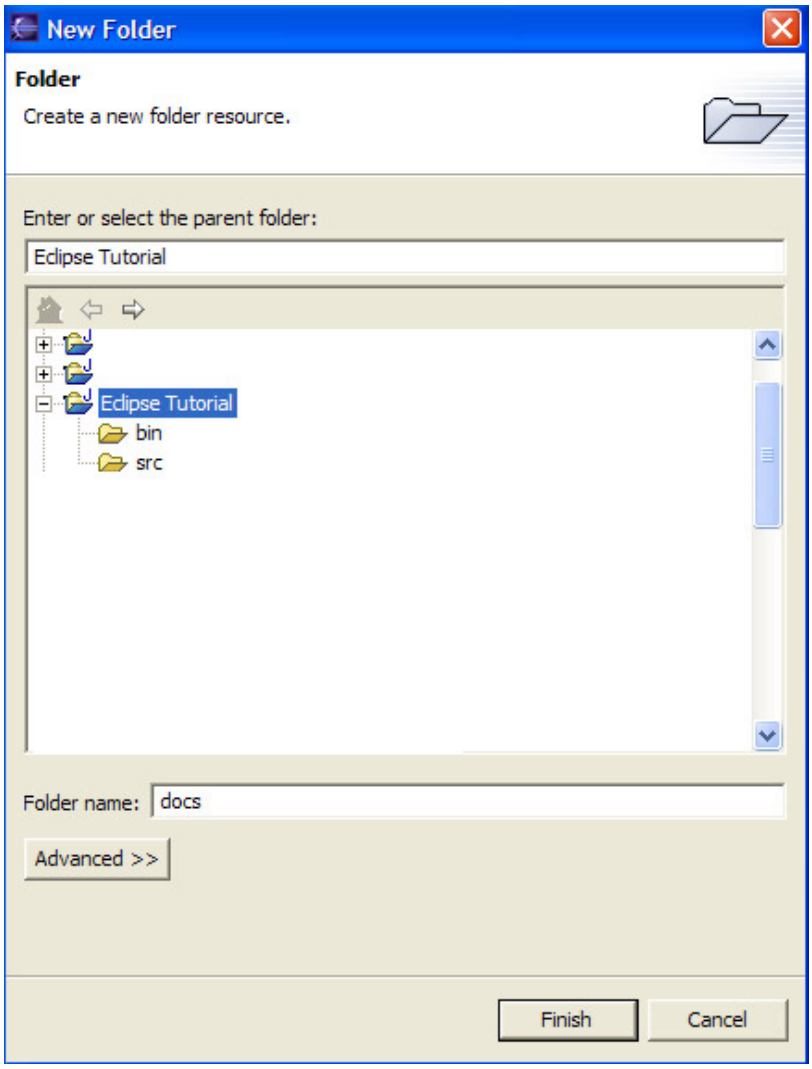
File

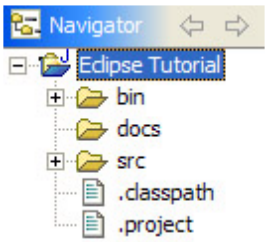
Los archivos que se crean a través de este menú suelen almacenar notas e información general. Otros archivos tales como archivos de configuración, archivos "build" de ant y similares también pueden crearse de esta manera. Crear un nuevo archivo es tan sencillo como pulsar "New > File", seleccionar el proyecto y carpeta adecuados en que se desea crear el archivo, dar nombre al nuevo archivo y pulsar el botón "Finish". Por defecto, archivos los archivos genéricos se abren en el editor de texto.



Folder

Las carpetas se utilizan para almacenar y organizar archivos. Como ya se ha mencionado anteriormente, una carpeta normal no es exactamente lo mismo que una carpeta fuente. Una práctica recomendada de programación es crear explícitamente una carpeta fuente donde almacenar los archivos .java con el código fuente (src), una carpeta de salida donde guardar el código compilado (bin) y una carpeta para guardar toda la documentación relacionada (docs). Para crear una nueva carpeta basta con especificar el nombre de la nueva carpeta y la carpeta que la contiene.





Interface

Las interfaces son casos particulares de las clases de Java, que carecen de implementación y que se espera que otras clases implementen. Usualmente funcionan como una especie de contrato, indicando lo que la clase implementada debería hacer, mientras que los detalles de más bajo nivel

corresponderían al implementador. El procedimiento de creación de interfaces es muy similar al procedimiento de creación de nuevas clases ya detallado. Aunque una interfaz no puede implementar ninguna interfaz, sí que puede extender otra interfaz mediante una relación de herencia.

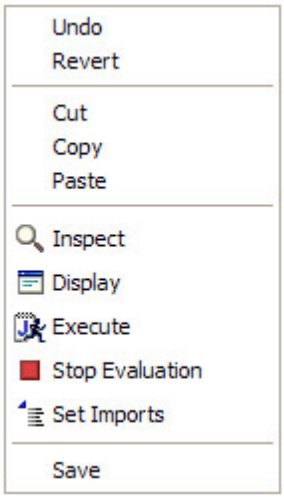
Package

Los paquetes se declaran para almacenar y organizar los archivos de Java. El nombre de un paquete consta usualmente de varias partes separadas por puntos. Cada una de estas partes será un directorio nuevo dentro del sistema de archivos. Las clases que se creen dentro de un paquete determinado en Eclipse llevarán añadida automáticamente la declaración "package" en su código fuente.

Scrapbook Page

Estas "hojas de sucio" son una forma sencilla de probar fragmentos de código antes de añadirlos al programa final. Sólo hay que crear una "Scrapbook Page" dentro de la carpeta deseada y escribir el código dentro de ella. No hace falta meter el código dentro de un método main para ejecutarlo.

En Eclipse 3.0 las "scrapbook pages" no se muestran directamente en el menú contextual. Para crear una nueva hoja de sucio seleccione "New > Other > Java > Java Run/Debug > Scrapbook Page". Otra diferencia respecto de las versiones 2.x es que en Eclipse 3.0 sí que están habilitadas las funciones de auto completar dentro de las scrapbook pages.



Antes de intentar ejecutar el código añadido es necesario importar las clases usadas. Para ello basta con realizar clic derecho sobre cualquier parte de la hoja y seleccionar "Set Imports" del menú contextual. Ahí es donde hay que especificar los tipos y paquetes que es necesario añadir.

Para ejecutar el código recién creado es necesario seleccionarlo pinchando con el botón izquierdo del ratón y arrastrando hasta tener todo el código seleccionado. Luego hay que pulsar el botón derecho del ratón sobre este código seleccionado y ejecutar la opción "Execute" del menú contextual. La salida estándar de dicho proceso se mostrará dentro de la vista "Console", y otros mensajes de error se mostrarán dentro de la misma hoja de sucio.

Una vez que se ha completado la prueba habría que pulsar el botón "Stop Evaluation" del menú contextual.

Puesto que el editor de estas hojas de prueba no proporciona tantas funciones de ayuda a la programación como el editor de código Java, una práctica recomendada es escribir el código original en el editor de Java y luego pegarlo en esta hoja de pruebas.

Source Folder

Como ya se vio, las carpetas fuente son un tipo especial de carpetas destinadas a almacenar los archivos fuentes de Java (es decir, los de extensión ".java"). Estos archivos de código serán automáticamente compilados en archivos ".class". Puesto que todo proyecto de Java debería tener una carpeta fuente, es una práctica recomendada organizar esto desde el primer momento, como se explicó en la primera parte de este tutorial.

Parte 3: Funciones Útiles de Programación

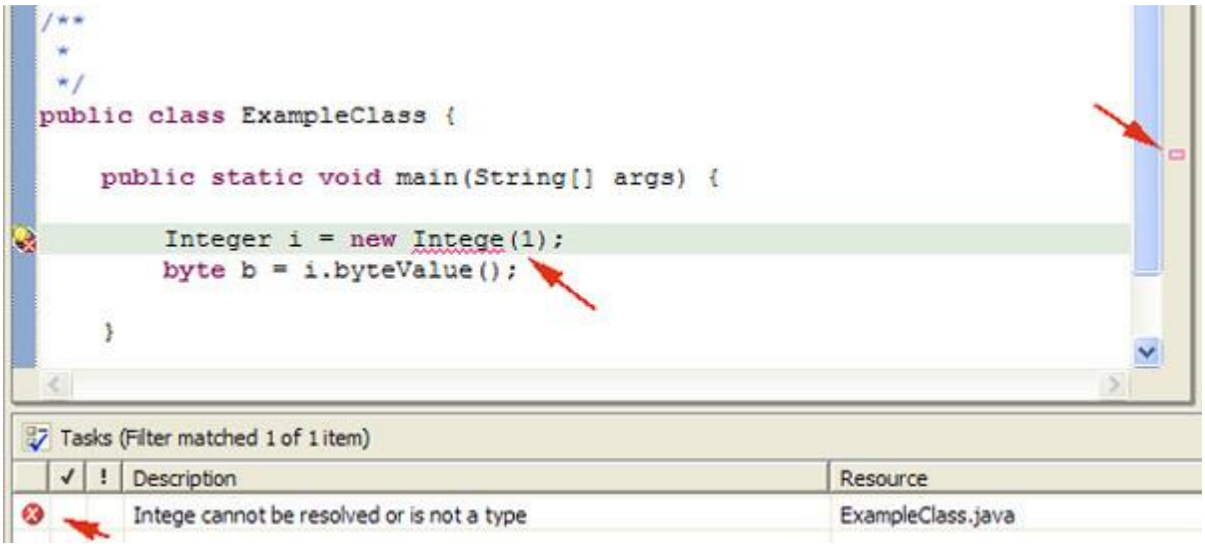
"Cualquier tecnología suficientemente avanzada es indistinguible de la magia." Arthur C. Clarke.

Hasta ahora se han expuesto conceptos básicos del funcionamiento general de Eclipse. Es hora de presentar las funciones de ayuda a la programación de Eclipse. Es en esta parte donde se dará cuenta de cómo usar Eclipse para programar en Java ahorra gran cantidad de tiempo y esfuerzo. Mientras que las partes anteriores de este tutorial eran necesarias, esta será la más interesante.

Compilar y Detectar Errores

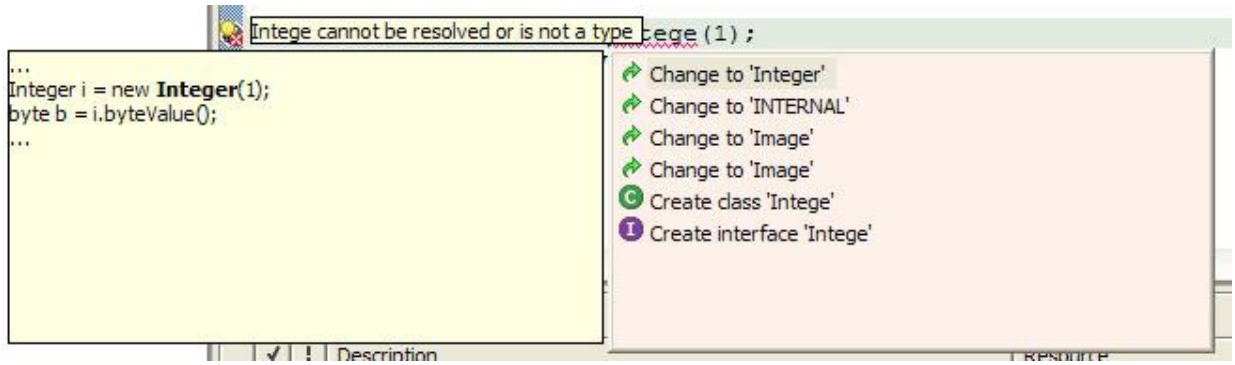
Es importante tener en cuenta que en Eclipse los errores de compilación se muestran en tiempo real subrayando el fragmento de código adecuado con una línea roja. Y además el entorno automáticamente compila los archivos salvados. Así pues, no será necesario pasar por el tedioso y lento proceso de compilar - observar los errores - corregir los errores.

Los errores pueden encontrarse fácilmente porque se muestran además como marcas rojas en el margen derecho del editor de código Java. También los errores y advertencias presentes en archivos ya guardados se muestran dentro de la vista de tareas (Tasks View), como se detallará posteriormente. Haciendo click en cualquiera de los dos tipos de marcadores de error llevará automáticamente hasta la línea en que el error está presente. Las advertencias (warnings) se muestran de la misma manera, pero con marcas amarillas.



Icono de Bombilla = Autocorregir

Hemos visto como Eclipse detecta y marca todo error y advertencia de compilación. Eclipse habitualmente permite autocorregir los posibles errores haciendo clic en el icono de bombilla presente en el margen izquierdo del editor de código. Así pues, aparecerá una ventana mostrando todas las opciones. Seleccionar una opción mediante los cursores del teclado o dejar el punto del ratón sobre dicha opción abrirá una nueva ventana mostrando detalladamente las modificaciones de código que la autocorrección efectuaría. Basta con pulsar la opción seleccionada (o pulsar ENTER) para hacer que Eclipse lleve a cabo la corrección automatizada.



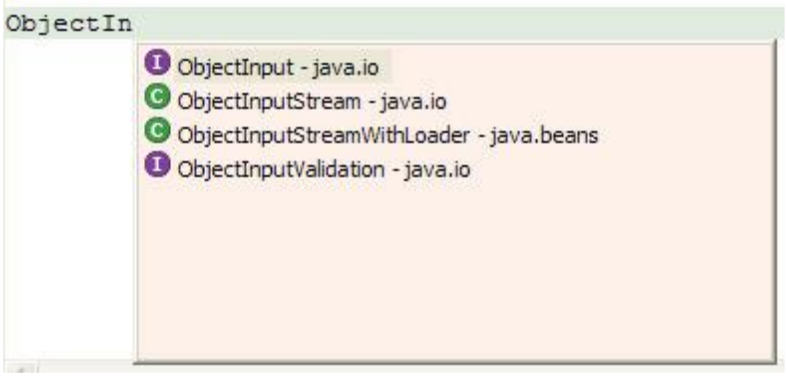
CTRL + Espacio = Autocompletar

Una vez que conozca la útil función de autocompletar la usará continuamente. A través de los siguientes ejemplos prácticos aprenderá cuales son las situaciones más comunes en que esta función de ayuda a la programación resultará muy útil.

- **Nombres de Clases**

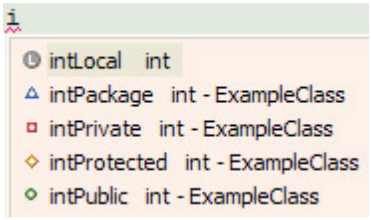
Crear referencias a otras clases dentro de la clase actual es una tarea de programación habitual. Sin embargo, algunas clases de Java tienen nombres muy largos que son difíciles de recordar. Además, es necesario añadir declaraciones de importación para poder resolver dichas referencias a clases a la hora de compilar.

Usar "CTRL + Espacio" tras escribir los primeros caracteres del nombre de una clase Java mostrará las posibles alternativas. Puede seleccionar cualquiera de ellas simplemente realizando clic izquierdo del ratón. Nótese que la sentencia de importación correspondiente se añadirá de forma automática. Las clases se marcan con una "C" verde mientras que las interfaces se marcan con una "I" morada. El paquete al que pertenece la clase se muestra también, permitiendo de este modo evitar posibles confusiones.



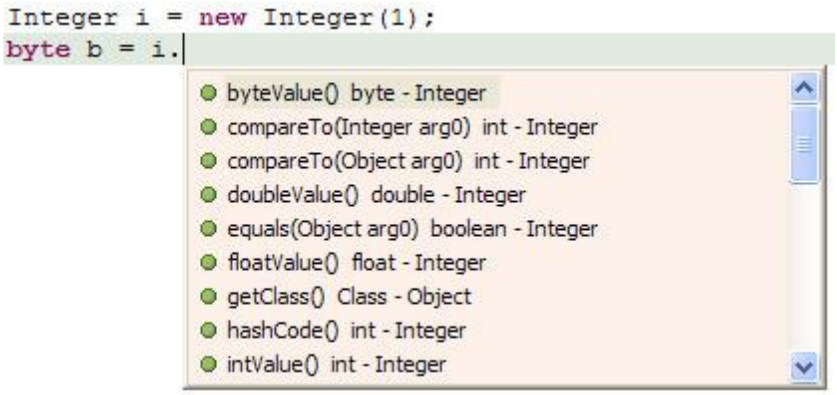
- **Atributos y Variables Locales**

Cuando se define una clase es normal dar nombres inventados a sus atributos y a las variables internas de los métodos. Pero en ocasiones resulta difícil recordar el nombre exacto. Tras escribir los primeros caracteres del atributo o de la variable local, pulsar "CTRL + Espacio" mostrará las posibles alternativas. Este proceso es muy similar al de autocompletar el nombre de las clases recientemente expuesto. Las variables locales se marcan con el icono de una "L" gris, mientras que los atributos se marcan con un icono que puede variar según la visibilidad del atributo.

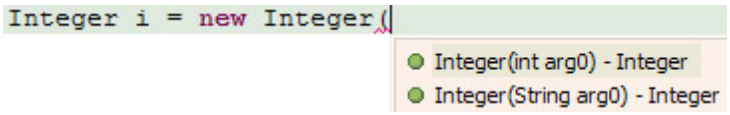


- **Métodos y Constructores**

Una vez que se ha creado un objeto Java pueden invocarse los métodos correspondientes a su clase. Sin embargo, es bastante habitual olvidar el nombre de un método en concreto, o incluso los tipos de sus parámetros y su orden. Este problema puede solucionarse fácilmente pulsando "CTRL + Espacio" tras escribir el nombre del objeto seguido de un punto, lo cual mostrará una ventana con las posibles alternativas. Pulsar sobre la alternativa escogida añadirá la signatura del método al objeto.



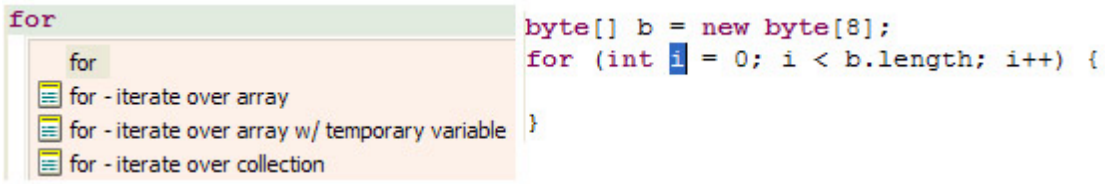
También es posible autocompletar la signatura de los constructores pulsando "CTRL + Espacio" tras escribir (o autocompletar) el nombre de la clase seguido de un signo de apertura de paréntesis, "(".



Escribir las primeras letras del modificador de un método tal como "public" o "private" y pulsar "CTRL + Espacio" le permitirá crear automáticamente una plantilla del método. Pulsar el tabulador permite saltar de un campo de la plantilla a otro, de manera que se pueda completar el tipo de retorno, el nombre del método y sus parámetros.

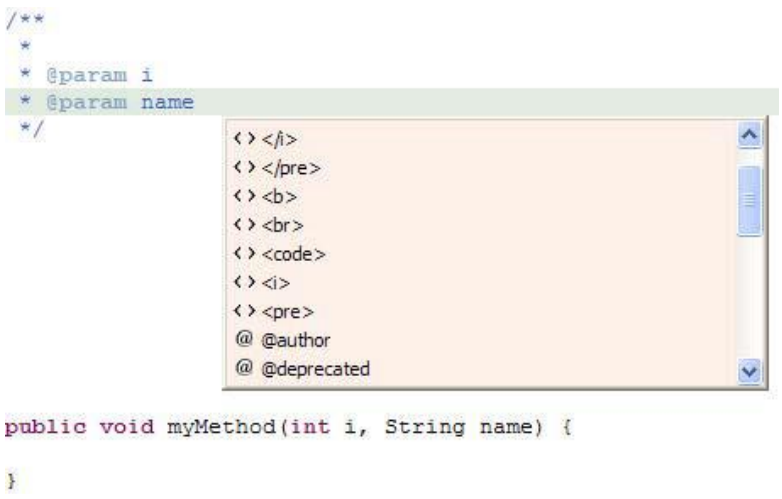
- **Bucles**

Los bucles suelen estar presentes en todos los programas. Aunque crear un nuevo bucle puede no ser una tarea muy compleja, Eclipse proporciona algunas funciones de auto completado que pueden acelerar considerablemente el proceso. Basta con escribir "do", "while" o "for" y pulsar "CTRL + Espacio" para mostrar las posibles opciones. Si el bucle ha sido creado con el propósito de iterar sobre un array de elementos, seleccionar esta opción intentará autocompletar incluso el nombre del array.



- **Etiquetas de Javadoc**

Mientras que los comentarios internos del programador se indican con una "//", los comentarios de Javadoc se inician con un "/**". Tras crear un método, añadir "/** + ENTER" sobre la signatura del método autocompletará información de Javadoc tal como "@param [nombreParámetro] [comentario]", "@return [descripciónDatosDevueltos]" y "@throws [tipoExcepción] [comentario]". Pulsar "CTRL + Espacio" dentro de un bloque "/** ... */" mostrará toda la lista de etiquetas Javadoc posibles.



Menú "Source"

Mediante un clic derecho en el editor de código se mostrará un menú contextual. Las funciones más importantes de su submenú "Source >" son las siguientes:

- **Comment and Uncomment**

Seleccionando esta opción se puede comentar o quitar la marca de comentario del código. Esto es especialmente útil usado con bloques de código, puesto que permite no ejecutar temporalmente partes del código sin llegar a eliminarlas. Las teclas rápidas asociadas son "CTRL + /" para añadir comentarios y "CTRL + \" para eliminar dichas marcas de comentario del bloque de código seleccionado.

En Eclipse 3.0 estos comandos han sido sustituidos por "Toggle Comment". Seleccionando este nuevo comando del menú "Source" es posible cambiar el estado del código seleccionado, pasándolo de comentario a código o viceversa. También la nueva opción "Add Block Comment" marcará el código como comentario rodeándolo con los símbolos "/*" y "*/". Mientras que seleccionar un bloque de código comprendido entre esos símbolos de comentario hará que en aparezca la opción "Remove Block Comment", la cual eliminaría los símbolos de comentario.

- **Format**

La función de formateado automático de código automáticamente indenta el código de la forma adecuada, además de llevar a cabo otras funciones de representación. Es una forma rápida de conseguir tener un código ordenado y comprensible. Las opciones del formateador de código pueden adaptarse a las

preferencias personales usando "Window > Preferences > Java > Code Formatter". Las teclas rápidas asociadas son "CTRL + Mayúsculas + F".

Nótese que las funciones de indentación de código permiten identificar rápidamente qué fragmentos de código son afectados por una condición o bucle determinados, por ejemplo. Pero también es útil situar el cursor tras un paréntesis o llave, porque de esta forma se marcará el paréntesis asociado que abre o cierra el bloque con un cuadrado gris. Así pues, se verá de un vistazo qué código se encuentra realmente comprendido entre los dos paréntesis.

```
public static void main(String[] args) {

}
```

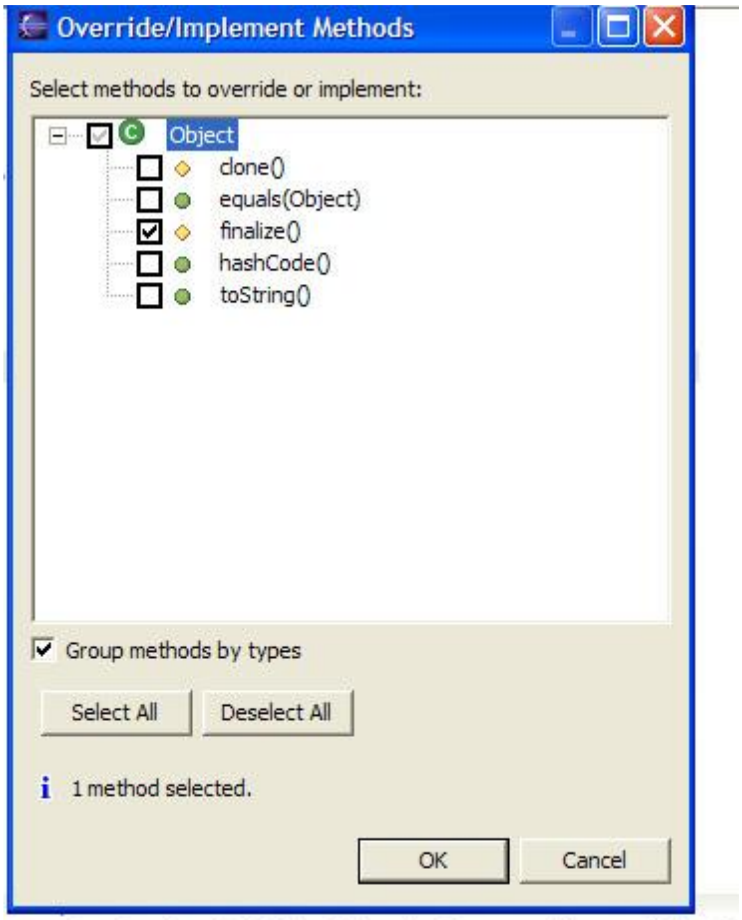
Eclipse 3.0 permite especificar opciones avanzadas de formateo de código. La página de preferencias que permite configurar el formateo de código se ha trasladado a "Window > Preferences > Java > Code Style > Code Formatter". Pese a que las opciones se pueden configurar de acuerdo con las preferencias personales, ya vienen configuradas por defecto para cumplir las convenciones de Java.

- **Organise and Add Imports**

Aunque las sentencias de importación adecuadas se muestran siempre cuando se usan las funciones de autocompletar código para completar el nombre de una clase Java, nuevas sentencias de importación pueden añadirse en cualquier momento usando la función "Add Import". Pulsar "Organise Imports" eliminará automáticamente todas las declaraciones de importación no utilizadas, incrementando la eficiencia del código. El método abreviado del teclado es "CTRL + Mayúsculas + O".

- **Override and Implement Methods**

Seleccionando esta opción de sombrear o implementar métodos abrirá una ventana de menú en la cual se podrán marcar los métodos de la superclase cuya signatura se desea que se añada.



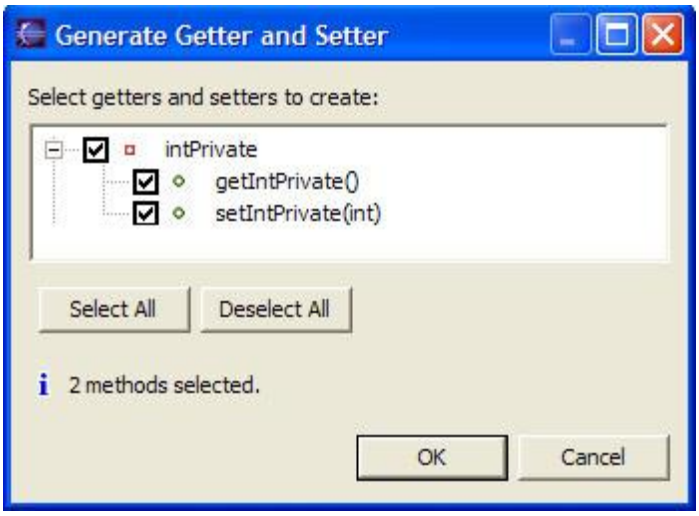

```
protected void finalize() throws Throwable {
    // TODO Auto-generated method stub
    super.finalize();
    socket.close();   }
```

La opción "Add Constructors from Superclass" permitirá especializar todos los constructores usados.

- **Generate Getter and Setter**

El lenguaje Java permite especificar diferentes niveles de visibilidad de atributos. Sin embargo, en programación orientada a objetos, los atributos internos de una clase deberían ser siempre privados. Esto quiere decir que no debería poder realizarse ningún tipo de modificación directa del atributo de una clase desde otra clase externa. A causa de esto, la única manera de acceder a un atributo privado para leer su valor o bien para darle un nuevo valor sería utilizando un método accesor o modificador que sea público. Seleccionando "Generate Getter and Setter" una ventana mostrando los posibles métodos que podrían

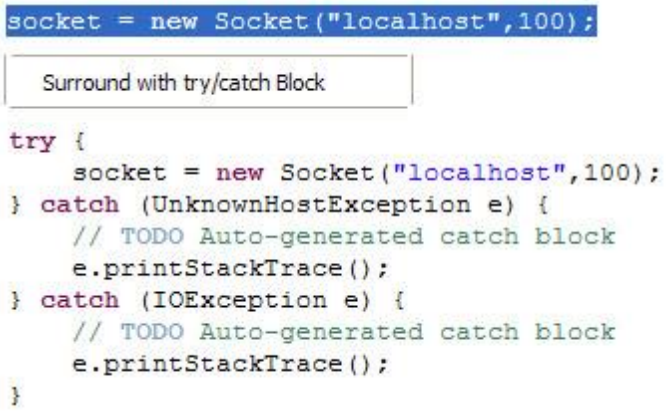
crearse de acuerdo con los atributos definidos aparecerá. Entonces los métodos necesarios podrían crearse simplemente seleccionándolos y pulsando "Ok".



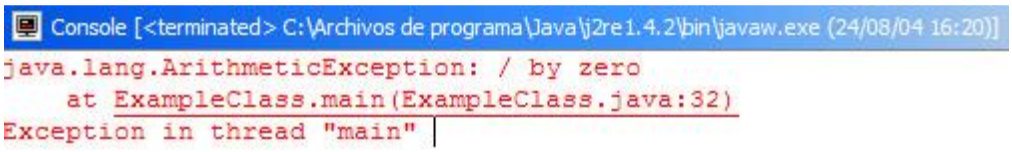

```
/**
 * @return
 */
public int getIntPrivate() {
    return intPrivate;
}

/**
 * @param i
 */
public void setIntPrivate(int i) {
    intPrivate = i;
}
```

- **Surround with try/catch block**

Para utilizar esta opción es necesario tener seleccionado de antemano un fragmento de código pinchando con el botón izquierdo del ratón (o pulsando Mayúsculas) y arrastrando. Activar esta opción creará un bloque "try" alrededor del código seleccionado. Tras este bloque se añadirán automáticamente los bloques "catch" adecuados, los cuales atraparán toda posible excepción que el código rodeado pueda lanzar. Por defecto se añade una sentencia de traza dentro de esos bloques "catch" de manera que sea posible identificar inmediatamente dónde se lanzó la excepción.



Nótese también que, cuando una excepción no es atrapada, aparecerá como texto en rojo (de la salida de error estándar) en la vista "Console". Pulsar la línea de código en que se muestra en qué línea tuvo lugar la excepción llevará directamente a ese punto del programa en el editor de código.

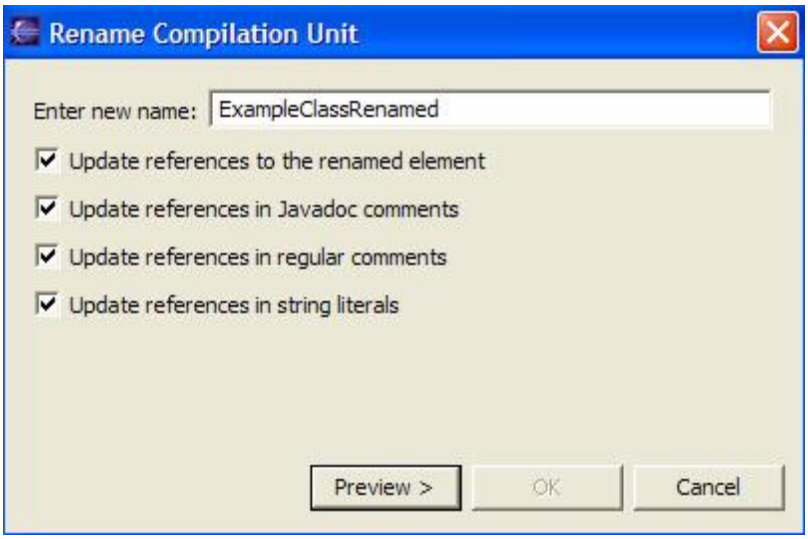


Refactor Menu

Haciendo clic derecho en el editor de código mostrará el menú contextual. A continuación se muestran las funciones más interesantes del sub menú "Refactor >".

- **Rename**

Para invocar la función de renombrado hay que tener previamente seleccionado un elemento. Marcar la opción de "update references" actualizará toda referencia al nuevo elemento renombrado. Usando esta opción de "Refactor > Rename..." es como deberían renombrarse todos los elementos incluyendo los archivos ".java". Puesto que así se actualizan todas las referencias no aparecerán problemas a la hora de compilar. Al renombrar determinados elementos será posible actualizar también referencias en comentario de Javadoc, comentarios normales y cadenas de caracteres entrecomilladas, lo cual también puede resultar bastante útil. La opción de "Preview" permite asegurarse de que no habrá ningún tipo de error durante la operación de renombrado.

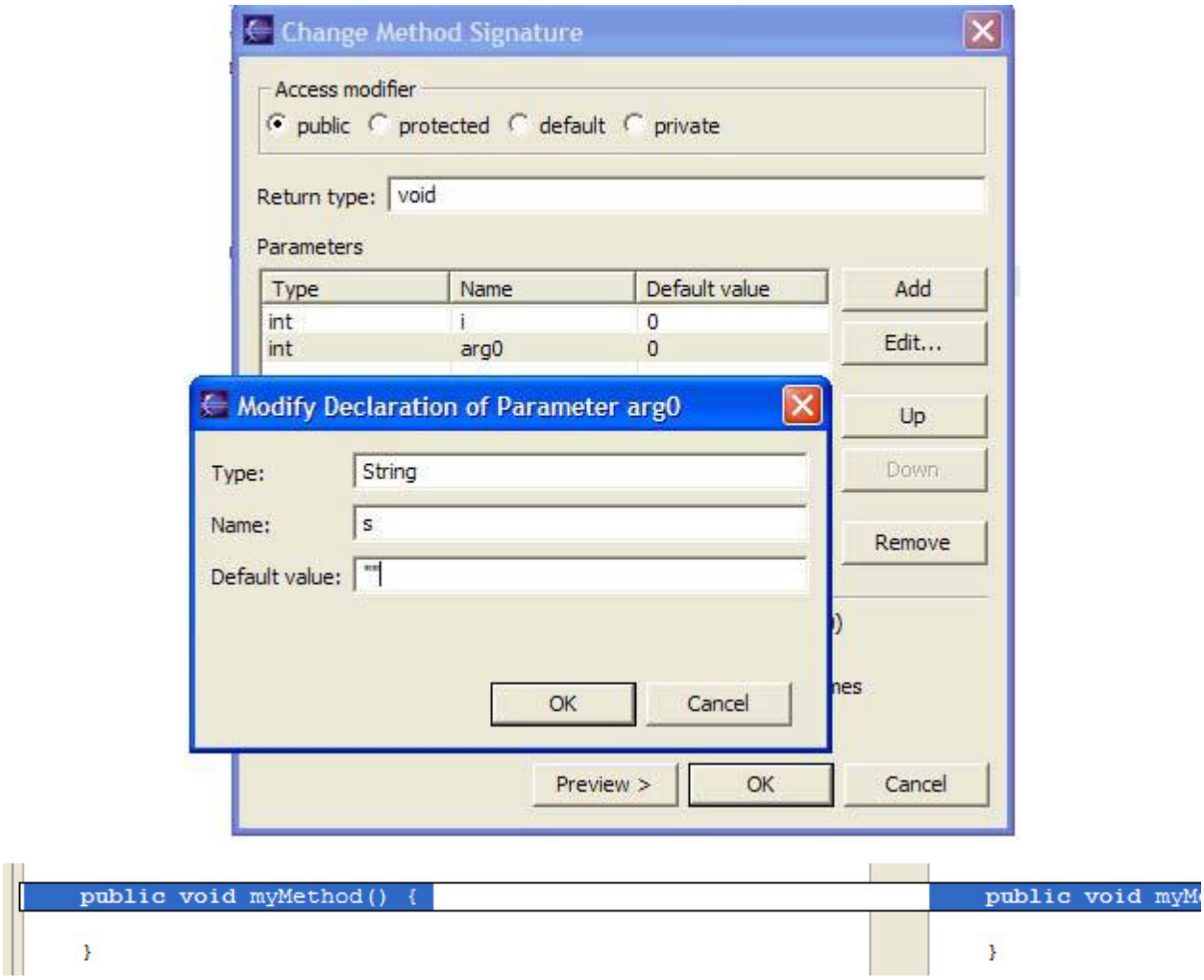


- **Move**

Antes de seleccionar "Refactor > Move...", el archivo fuente o elemento que se desea mover deberá haber sido seleccionado. Entonces será sólo necesario seleccionar el destino de manera que se lleve a cabo la operación de mover. Esta es la forma correcta de mover archivos ya que evita problemas futuros con referencias y rutas de compilación.

- **Change Method Signature**

Para modificar la signatura de un método es posible usar esta opción en lugar de hacerlo manualmente. Sólo hay que colocar el cursor dentro del método cuya signatura se desea cambiar. Esta es una forma rápida de cambiar la visibilidad, el tipo de retonro, los parámetros y su orden. Los nuevos parámetros se añaden pulsando el botón "Add" y se modifican pulsando el botón "Edit".

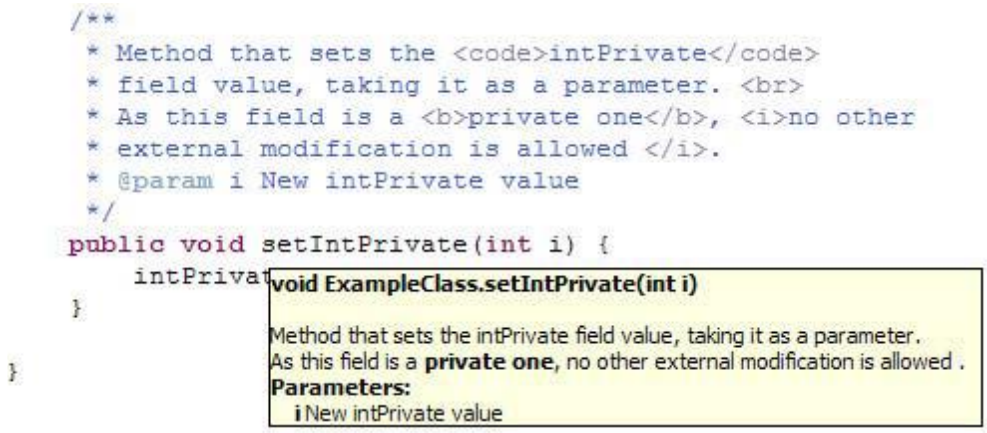


- **Pull Up and Push Down**

Si la clase actual extiende o es extendida por otra clase, puede ser interesante mover algunos elementos a la superclase (pull up) o a la subclase (push down) respectivamente. Seleccionar el elemento y la opción adecuada llevará a cabo esta operación de forma automatizada.

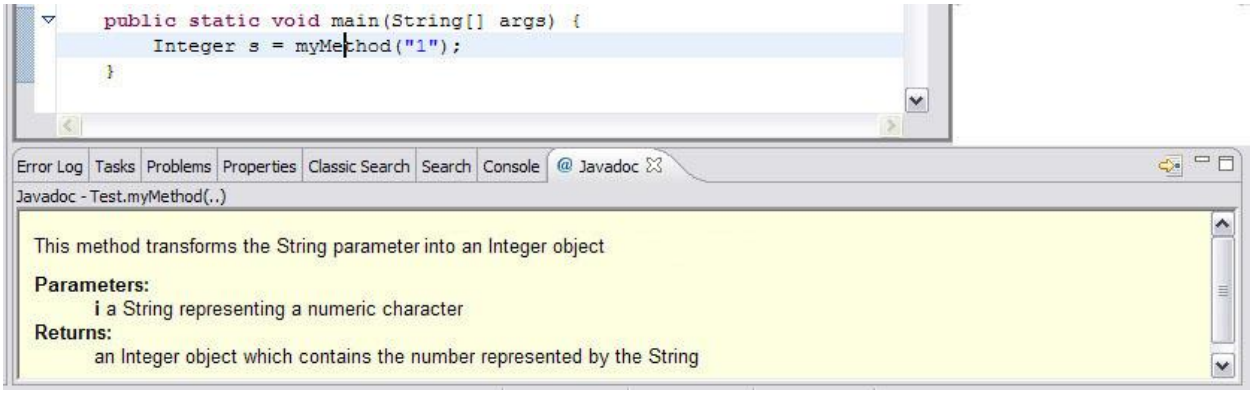
Consultar la Documentación

La documentación Javadoc del código que se esté actualmente programando puede ser consultada en tiempo real simplemente colocando el cursor o el puntero del ratón sobre el elemento elegido. Para expandir la ventana con esta documentación basta con pulsar la tecla de función F2.



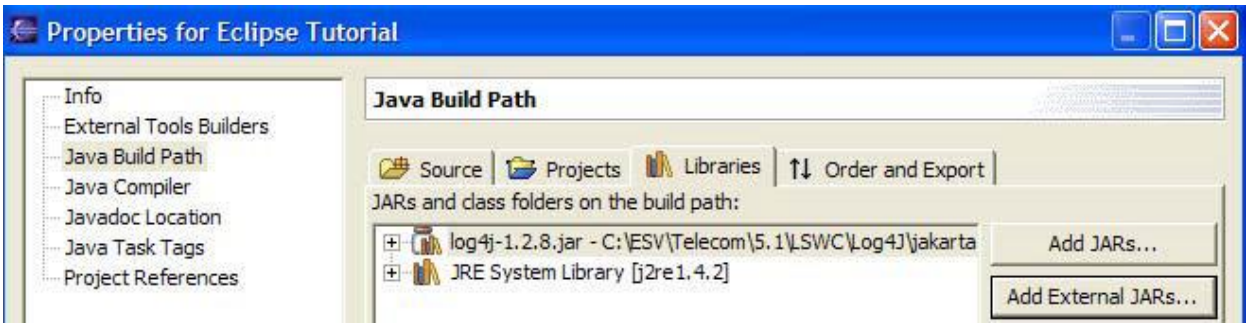
La documentación Javadoc externa (como por ejemplo, la documentación oficial de las clases de Java) puede consultarse modificando dentro de las preferencias del JRE instalado ("Window > Preferences > Installed JRE") con "Edit" la ubicación del Javadoc. De esta manera, dejando el cursor sobre el nombre de la clase y pulsando "Mayúsculas + F2" se abrirá la documentación por el punto adecuado.

Eclipse 3.0 dispone de una nueva vista de Javadoc ("Window > Show View... > Java > Javadoc"). Dicha vista muestra la documentación Javadoc asociada al elemento sobre el que está situado el cursor.



Importar Archivos JAR

En ocasiones puede ser necesario importar algunos archivos Jar no incluidos por defecto en el JRE estándar para que el proyecto pueda compilar. Basta con pulsar el botón derecho del ratón sobre la carpeta adecuada, elegir "Properties > Java Build Path", seleccionar la pestaña "Libraries", pulsar el botón "Add External Jars" y seleccionar el archivo ".jar" o ".zip". El nuevo Jar añadido será visible en la ventana Package Explorer como un pequeño frasco.


Properties for Eclipse Tutorial
Info
External Tools Builders
Java Build Path
Java Compiler
Javadoc Location
Java Task Tags
Project References
Java Build Path
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
log4j-1.2.8.jar - C:\ESV\Telecom\5.1\LSWC\Log4J\jakarta
JRE System Library [j2re1.4.2]
Add JARs...
Add External JARs...

Parte 4: Vistas de Eclipse

"Hay dos clases de conocimiento. Podemos conocer un tema por nosotros mismos, o bien conocer dónde encontrar información al respecto." Samuel Johnson.

La interfaz de usuario de Eclipse consta de dos tipos de elementos: vistas y editores. Mientras que los editores normalmente permiten realizar una tarea completa, las vistas proporcionan funciones de apoyo. En este punto del tutorial ya debería tener bastante práctica usando el editor de código cuyas funciones principales se detallaron en la parte 3. Ahora, las vistas más interesantes de Eclipse se explicarán con detalle, junto a algunos consejos de cómo navegar a través de los editores.

Perspectivas

Una perspectiva de Eclipse es una agrupación de vistas y editores de manera que den apoyo a una actividad completa del proceso de desarrollo software. Sin embargo, es posible crear perspectivas propias añadiendo nuevas vistas y cambiando su distribución en la pantalla. Las perspectivas pueden seleccionarse haciendo clic en los iconos de perspectiva del lateral izquierdo o eligiendo "Window > Open Perspective" del menú. Las perspectivas son:

Resource: esta perspectiva está estrechamente relacionada con el sistema de archivos puesto que representa la localización física de los recursos almacenados dentro de los proyectos

Java: esta perspectiva se centra en tareas de programación, mostrando paquetes, clases, métodos y atributos en sus vistas asociadas.

Plug-in development: la perspectiva de desarrollo de plug-in permite a los desarrolladores añadir nuevos módulos de Eclipse.

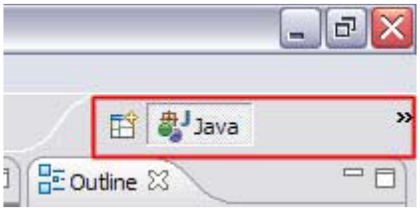
Install/Update: permite gestión de la configuración. Muestra los componentes instalados, así como sus versiones y conflictos.

Debug: relacionada con la tarea de depuración. Se centra en los procesos ejecutados, puntos de ruptura, variables, salida, etc.

Java Browsing: esta perspectiva permite ojear rápidamente código, proyectos, paquetes y jerarquías.

En Eclipse 3.0 los iconos de perspectiva se han trasladado a la esquina superior derecha. También hay ahora un botón etiquetado como "Open a Perspective" que permite acceder rápidamente a otras perspectivas. Otro cambio es que la

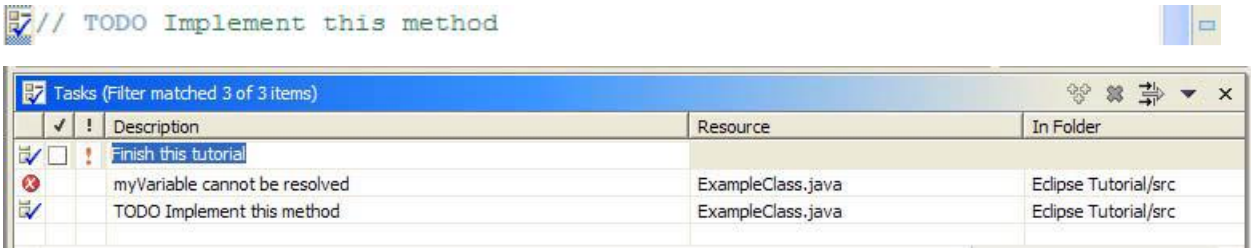
perspectiva "Install/Update" ha sido eliminada, y puede accederse a sus funciones seleccionando "Help > Software Updates".



Tareas

La vista de tareas ("Tasks View") permite una rápida gestión de tareas pendientes. Seleccionando "Window > Show View > Tasks" se muestra esta vista. Pueden añadirse nuevas tareas haciendo clic en el botón "Add task". Nótese que la prioridad de la tarea y su estado también pueden modificarse sin más que hacer clic en dichos campos. También los errores y las advertencias de los archivos con código guardados se muestran en esta vista. Haciendo clic en la descripción de un error llevará hasta el punto exacto del código en que se encuentra dicho error.

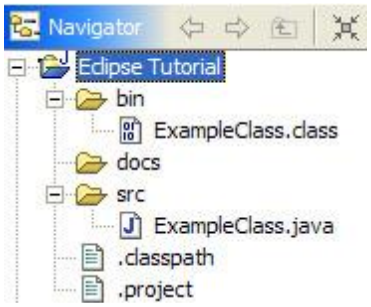
Añadir "TODO [descripción]" a un archivo Java añadirá una nueva tarea "por hacer" a la vista de tareas. Una vez más, hacer clic sobre su descripción conducirá hasta el punto exacto en que la etiqueta "TODO" se añadió. Dentro del editor de código Java las etiquetas de "TODO" pueden encontrarse rápidamente ya que se muestran como pequeñas marcas azules en el margen derecho. Hacer clic sobre estas marcas llevará directamente a la línea etiquetada. Nótese que varios procesos automatizados insertan estas etiquetas "TODO" para asegurarse de que el código autogenerado es revisado y comentado.



Haciendo clic derecho en cualquier punto de la vista de tareas se mostrará un menú contextual que permitirá realizar de forma rápida cualquier actividad relacionada con la gestión de las tareas definidas.

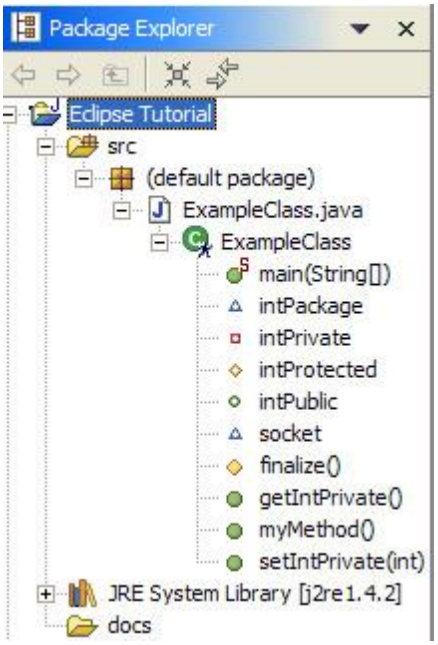
Navigator

La ventana del navegador de recursos permite echar un vistazo a la estructura de archivos de los proyectos definidos. Nótese que esta vista es la única que muestra la carpeta de salida ("bin") así como los archivos Java compilados (".class").



Package Explorer

La vista del explorador de paquetes muestra la estructura lógica de paquetes y clases Java almacenados en los distintos proyectos. las carpetas fuente (que deben almacenar los archivos fuente ".java") se muestran aquí decoradas con el icono de un paquete contenido. Los archivos Java también pueden ser expandidos de modo que muestren sus métodos y atributos internos al pulsar el botón "+".



- **Working Set**

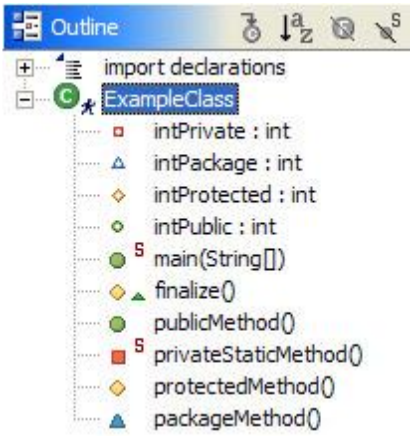
Un conjunto de trabajo es un grupo de elementos que se muestran en las distintas vistas de eclipse. Estos conjuntos de trabajo se usan como filtros que permiten separar claramente los diferentes proyectos en que se está trabajando. Esto puede resultar muy útil cuando se está trabajando simultáneamente en varios proyectos no directamente relacionados entre sí. Organizar los distintos proyectos en grupos de trabajo acelerará el proceso de buscar los elementos deseados y reducirá la confusión visual.

Para definir un conjunto de trabajo, basta con pulsar en el icono de menú del Package Explorer (el icono de un triángulo invertido) y seleccionar "Select Working Set". Aquí se permitirá nombrar un nuevo conjunto de trabajo, así como seleccionar sus recursos relacionados y editar o quitar otros conjuntos de trabajo existentes. Todos los conjuntos de trabajo disponibles se muestran directamente la próxima vez que se pulse el icono triangular de menú.

Es importante tener en cuenta que crear un nuevo proyecto cuando un conjunto de trabajo está siendo usado hará que el nuevo proyecto no se muestre dentro de las vistas de Ecipse. Para poder ver el proyecto recién creado, será necesario editar el conjunto de trabajo actual ("Menú de la vista > Select Working Set > Edit" o directamente "Edit Current Working Set") y seleccionar el nuevo proyecto para que se muestre.

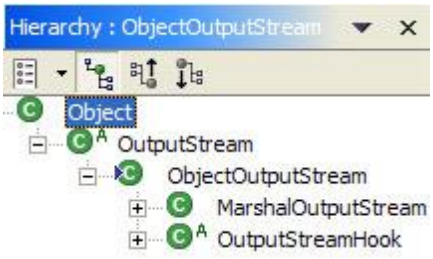
Outline View

La vista de resumen es una forma rápida de ver qué métodos i atributos se encuentran definidos dentro de una clase de Java. Los iconos asociados proporcionan información adicional de acuerdo con la visibilidad del atributo o método en cuestión. Y sólo con hacer clic en cualquiera de estos iconos conducirá a la línea de código exacta en que dicho atributo o método está definido. La vista de resumen es una herramienta esencial para entender y navegar archivos Java voluminosos.



Hierarchy View

La vista de jerarquía muestra las relaciones de herencia presentes entre distintos elementos de Java. Haciendo clic derecho en el nombre de una clase Java en el editor de código y seleccionando "Open Type Hierarchy" abrirá esta vista de jerarquía. La tecla rápida asociada es "F4"



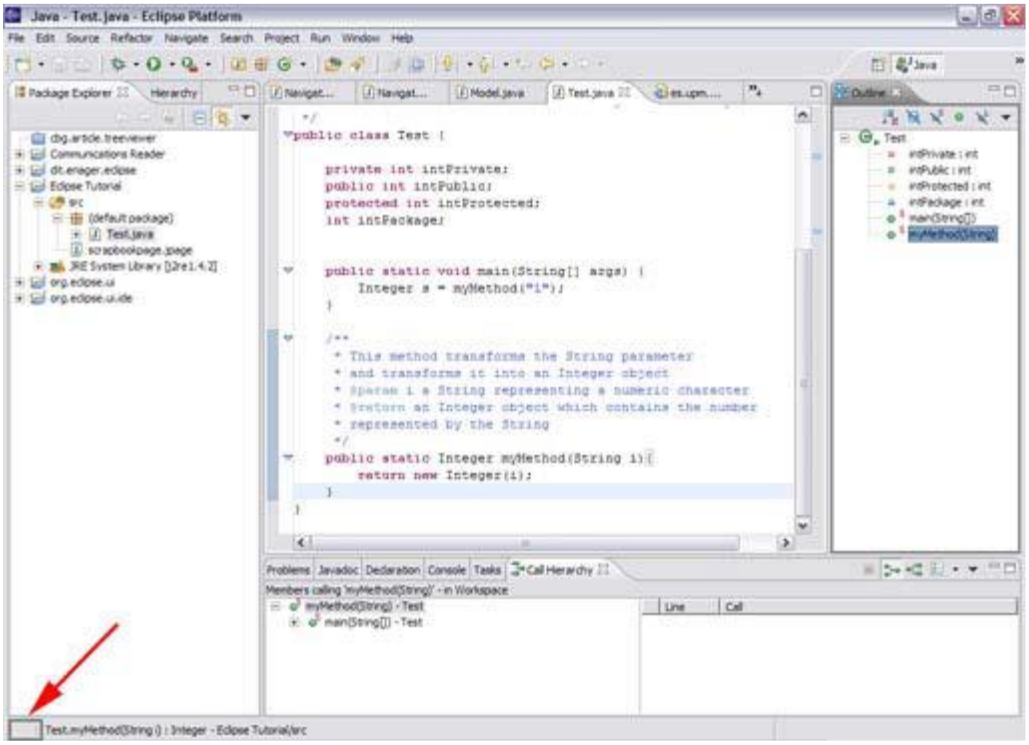
En Eclipse 3.0 se ha añadido la opción "Open Call Hierarchy" al menú contextual del editor de código. Tras seleccionar un método, al hacer clic en esta opción se abrirá una vista que mostrará dónde es usado dicho método. Las teclas rápidas asociadas son "CTRL + ALT + H".

Fast Views

Arrastrar una vista hasta el margen izquierdo (hasta que aparezca un icono de carpetas apiladas) convierte esta vista en una "vista rápida". Pulsar el icono de la vista rápida hará que dicha vista se muestre, mientras que volver a pulsarlo (o pulsar en cualquier otro punto de la pantalla) hará que se oculte. Mediante un clic derecho en el icono de la vista rápida y seleccionando "Fast View" restaurará la vista a su posición original.

El área por defecto en que se apilan las vistas rápidas ha sido cambiada en Eclipse 3.0. Ahora es un pequeño rectángulo situado en la esquina inferior

izquierda de la pantalla. Así pues, las vistas rápidas se crean ahora arrastrando la vista dentro del rectángulo hasta que aparece un icono de una flecha dentro de un cuadrado. No obstante, la zona en que almacenar las vistas rápidas puede cambiarse de sitio colocando el cursor sobre ella hasta que se transforma en un cursor con cuatro flechas, arrastrando y depositando la zona en el lugar deseado

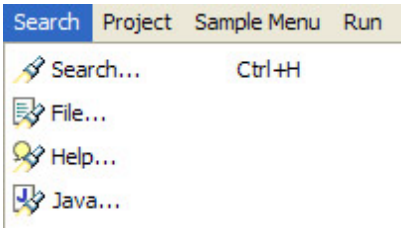


Search View

Para realizar una búsqueda dentro de Eclipse, el menú "Search" de la barra superior de menús debería ser seleccionado. También se pueden lanzar búsquedas pulsando el icono de linterna.

Hay varios tipos de búsquedas dentro de Eclipse.

La búsqueda de archivos "File Search" es una búsqueda textual que puede ser ejecutada sobre archivos de todo tipo. Es equivalente a una búsqueda tradicional.



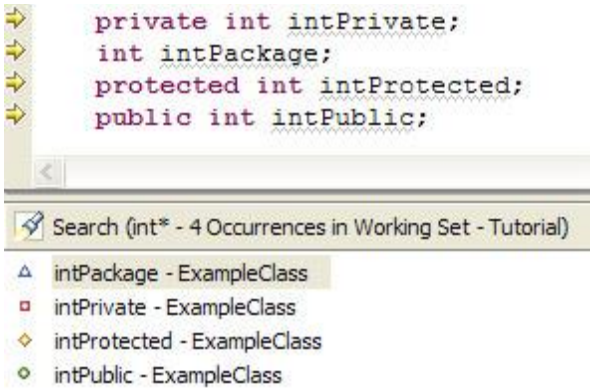
La búsqueda de ayuda "Help Search" efectúa búsquedas dentro de la ayuda de Eclipse.

La búsqueda de Java "Java Search" es similar a la búsqueda de archivos, pero proporciona funciones

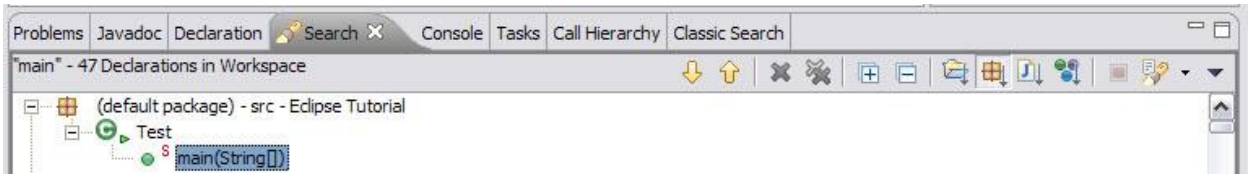
adicionales para buscar en archivos Java. Así pues, permite buscar explícitamente por tipos, métodos, paquetes, constructores y campos, usando restricciones de búsqueda adicionales (como, por ejemplo, buscar sólo el punto del código en que se declararon los elementos coincidentes).

Es importante comprobar que la búsqueda se efectúa sobre los ficheros apropiados. Esto puede definirse usando el campo "scope". "Workspace" hace referencia al entorno de trabajo completo. "Selected Resources" son sólo los archivos seleccionados (es posible seleccionar más de un archivo haciendo clic izquierdo en ellos mientras se mantiene pulsada la tecla CTRL). "Working Set" es un conjunto de trabajo previamente definido.

Los resultados de la búsqueda aparecen en la vista "Search". También se subrayan en gris dentro del editor de código, con una flecha amarilla en el margen izquierdo y con una marca gris en el margen derecho. Haciendo clic en cualquiera de estos elementos seremos conducidos al punto en que la cadena buscada se encontró.



Los resultados de búsqueda se muestran como un árbol jerárquico en Eclipse 3.0.



Navegar por las Vistas y los Editores

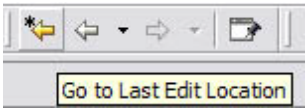
Hasta ahora hemos visto una introducción de cómo utilizar las vistas de Eclipse y cómo dichas vistas ayudan a manejar la información. Es hora de explicar algunas funciones de navegación adicionales que serán útiles para encontrar rápidamente la información deseada y que permitirán presentarla adecuadamente en los diversos editores y vistas.

Maximizar una Vista o Editor

Basta con hacer doble clic en el título de una ventana para maximizarla. Doble clic en el título de nuevo hará que las dimensiones y posición de la ventana sean restauradas a las que tenía originalmente. En Eclipse 3.0 se ha añadido "CTRL + M" como tecla rápida asociada a maximizar o restaurar la ventana del editor actual.

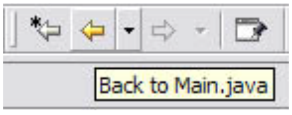
Ir al Último Cambio

El icono del menú representado como "una flecha con un asterisco" sirve para colocar el cursor en el último punto del código que fue modificado dentro del editor activo. Es habitual que tras cambiar algo de código (por ejemplo, tras escribir algunas instrucciones nuevas) movamos el cursor a otra línea para revisar otra parte del programa. Si deseáramos volver al punto en que añadimos el último cambio (que suele ser el lugar por el que "íbamos programando") tendríamos el problema solucionado con sólo pulsar este icono de "ir al último lugar editado". Las teclas rápidas asociadas son "CTRL + Q".

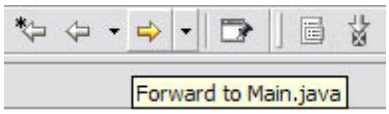


Acciones de Navegación de los Editores

Pero ¿qué haríamos si quisiéramos volver a un punto del programa en el que no introdujimos ningún cambio (es decir, en el que situamos el cursor pero en que no escribimos o borramos ningún carácter)? O, ¿y si quisiéramos regresar al lugar en que estuvimos justo antes de editar algo en otro lugar? Las flechas de navegación del menú resolverán estos problemas. Basta con pulsar la flecha de "navegar hacia atrás" para regresar a puntos previamente visitados del programa.



Y pulsando la flecha de "navegar hacia delante" recorreremos el historial de lugares visitados hacia los puntos más recientes.

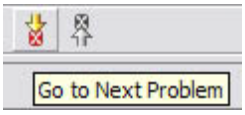


De hecho, estas útiles acciones funcionan de forma muy similar a como lo hacen los botones de "atrás" y "adelante" de un navegador web. La opción de "atrás" sólo se activa si existen puntos de programas que se visitaron previamente. Y la opción de "adelante" se activa tras haber pulsado el botón de "atrás". También hay que tener en cuenta que pulsando en el pequeño triángulo negro que se encuentra junto a las flechas de navegación desplegaremos un menú que muestra otros archivos (distintos del abierto en la ventana activa del editor) en los que se encuentran otros puntos visitados accesibles. Por cierto, para cambiar la ventana activa del editor a otras ventanas abiertas existe un método abreviado: "ALT + F6" (comando de "siguiente editor").

Las útiles teclas rápidas asociadas a estas acciones de navegación son "ALT + IZQUIERDA" para navegar hacia atrás y "ALT + DERECHA" para navegar hacia delante.

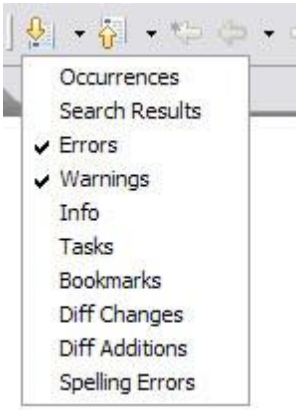
Revisar Problemas

Los botones de "Ir al siguiente/anterior problema" permiten recorrer uno tras otro los problemas pendientes que aparecen en el editor actual.



Aunque esta es una forma sistemática de revisar los problemas, es interesante recordar que también puede accederse directamente a los problemas, advertencias, resultados de búsqueda y tareas pendientes sin más que hacer clic en sus marcas asociadas que aparecen en el margen derecho del editor.

En Eclipse 3.0 estos botones de "ir a problema" se han sustituido por botones de "ir a anotación". Haciendo clic en el pequeño triángulo negro cercano a estas flechas de navegación por anotaciones se abrirá una lista editable con los tipos de anotaciones que serán recorridas. Así pues, el uso de estos botones no está ya limitado sólo a la comprobación de problemas ya que ahora se pueden realizar también otras tareas útiles tales como, por ejemplo, comprobar sistemáticamente todas las "tareas pendientes" ("TODO") sin más que seleccionar "Tasks" en dicha lista.


Occurrences
Search Results
Errors
Warnings
Info
Tasks
Bookmarks
Diff Changes
Diff Additions
Spelling Errors

Parte 5: Ejecutar y Depurar

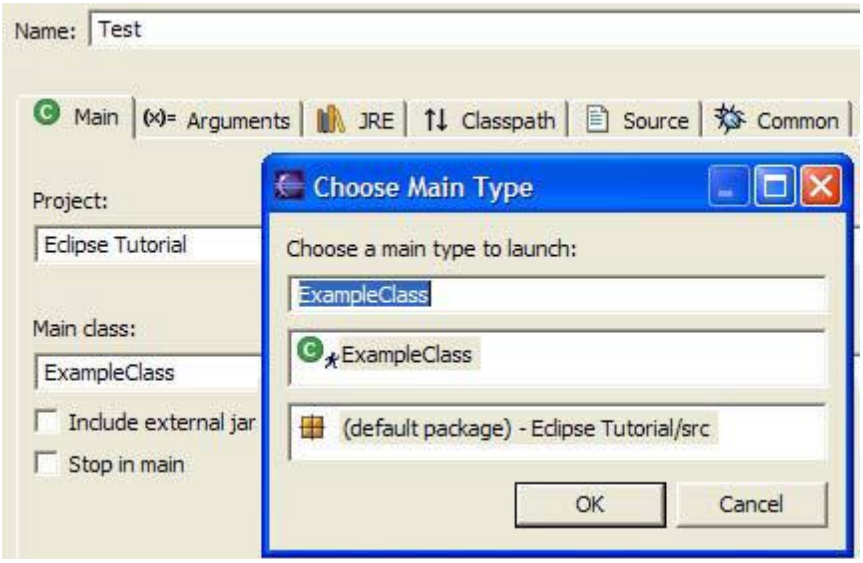
"En unos pocos minutos, un ordenador puede cometer un error tan grave que muchos hombres necesitarían muchos meses para igualarlo." Anónimo.

Cuando el programa de Java esté completado será hora de ejecutarlo y probarlo. Quizás aparezcan algunos "bugs" a la hora de ejecutarlo. Entonces será hora de depurar el programa. Afortunadamente, Eclipse proporciona ayuda a las tareas de ejecutar y depurar código.

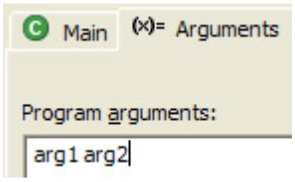
Ejecutar

Para ejecutar un programa dentro de Eclipse hay que seleccionar "Run > Run..." del menú principal. Dentro de "Configurations" se almacenan diferentes configuraciones de ejecución. Hay cuatro tipos de configuraciones de ejecución: Java Applet (para applets web), Java Application (para programas normales de Java), JUnit (casos de prueba) y Run-Time Workbench (otras instancias de Eclipse que permiten probar nuevos módulos de Eclipse).

Así pues, para ejecutar un programa de Java normal debería seleccionarse "Java Application" y pulsar el botón "New" para crear una nueva configuración. Dentro de la pestaña "Main" hay que dar nombre a la nueva configuración seleccionar el proyecto que contiene la clase con el método main y seleccionar dicha clase. El método "main" es el punto de ejecución de un programa Java, y se representa como un pequeño icono de un hombre corriendo al lado del icono de la clase.



Si se desea pasar argumentos al método main (en la forma de "String[] args"), no hay más que hacer clic en la solapa de "Arguments" y escribir esos argumentos separados por espacio dentro de la zona en blanco de "Program Arguments".



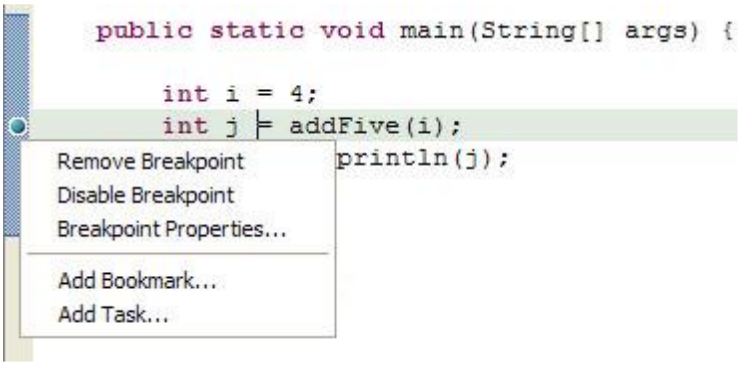
Finalmente, hacer clic en el botón "Run" lanzará la ejecución del programa seleccionado. Una forma más rápida de arrancar la ejecución del programa recientemente ejecutado es pulsar en el icono de un hombre corriendo que aparece en el menú principal. Pulsar la flecha próxima a este icono mostrará otras configuraciones de ejecución recientemente utilizadas.

En Eclipse 3.0 el icono anterior ha sido reemplazado por una flecha blanca dentro de un círculo verde.

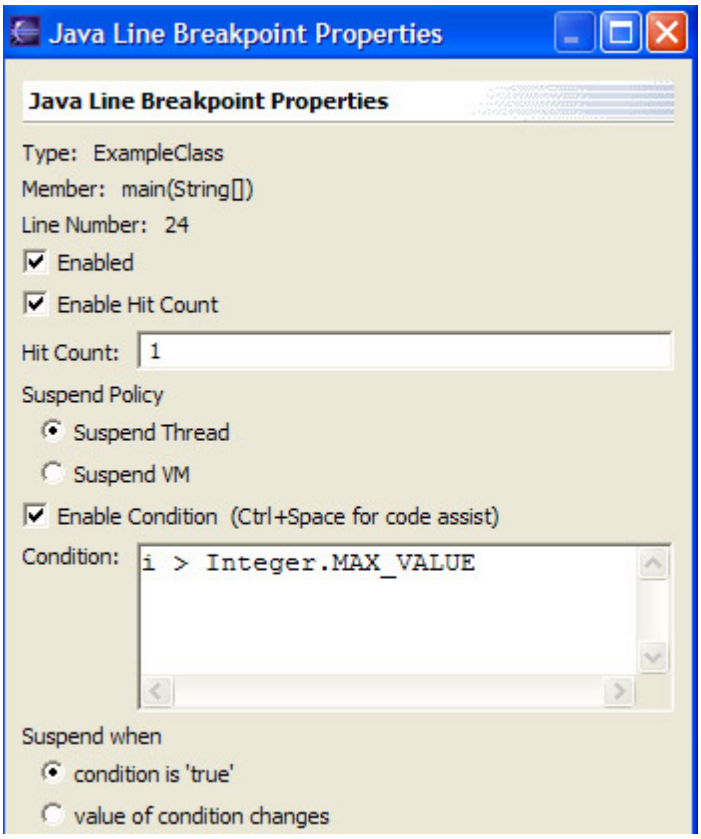
Depurar

Aunque Java no es tan difícil de depurar como otros lenguajes de programación, también es perfectamente posible que surjan complejos problemas de

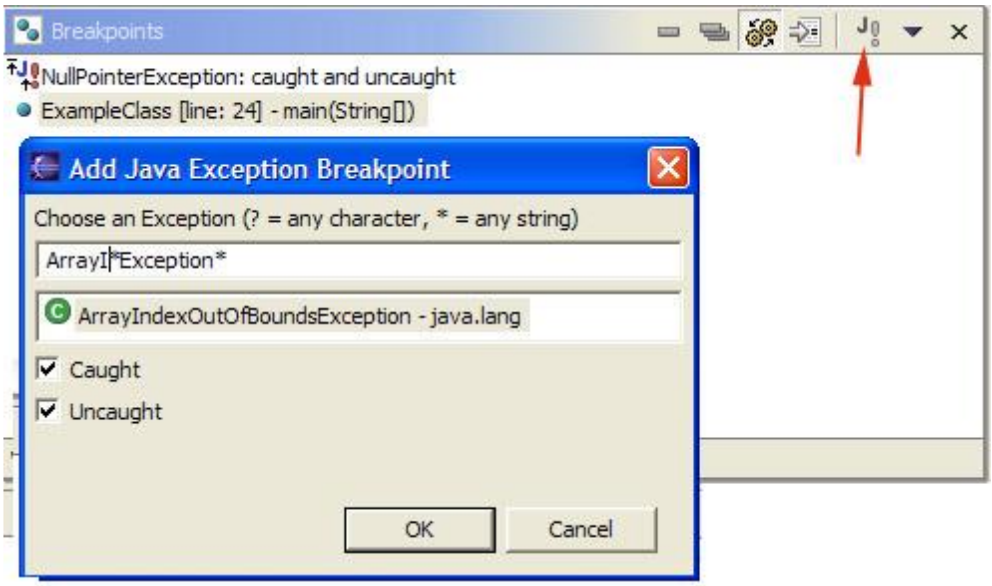
ejecución. Eclipse da apoyo completo a la tare de depuración a través de su perspectiva "Debug" ("Window > Open Perspective > Debug" o seleccionando el icono del "bicho" en el margen izquierdo). Dentro de esta perspectiva de depuración, haciendo clic en el margen izquierdo del editor de código aparecerá un menú contextual. Seleccionando "Add/Remove Breakpoint" añadirá o quitará un punto de ruptura, mientras que "Toggle Breakpoint" cambiará el estado de activación del punto de ruptura. Los puntos de ruptura marcan líneas en que la ejecución del programa se detendrá de manera que sea posible comprobar el valor de las variables en ese instante, identificando así posibles errores.



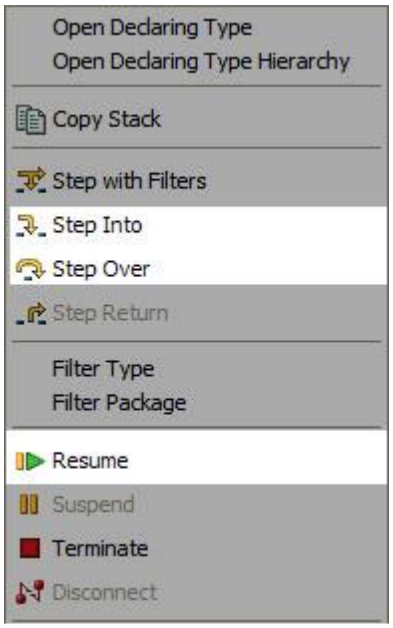
Haciendo clic derecho en un punto de ruptura y seleccionando "Breakpoint Properties..." permitirá especificar opciones avanzadas del punto de ruptura. "Hit Count" especifica que la ejecución del programa se detendrá cuando se pase por el punto de ruptura el número especificado de veces. Las condiciones de activiación detendrán la ejecución cuando la condición sea cierta o bien cuando el valor de la condición cambie. Especificar una variable como una condición de activación y seleccionar "suspend when value of condition changes" es una forma de "detener la ejecución en el punto de ruptura cuando dicha variable sea modificada".



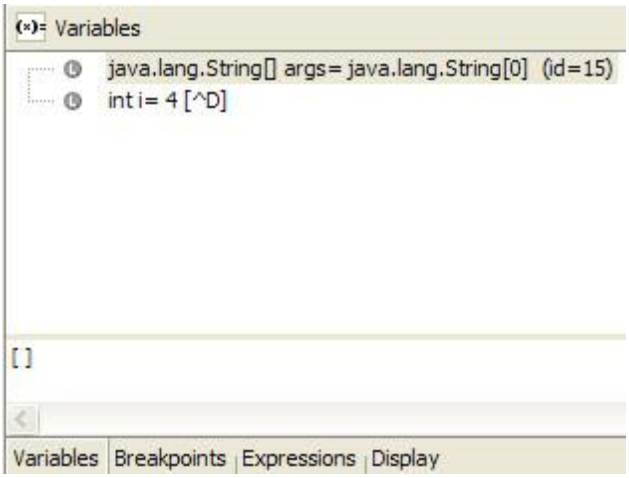
Las excepciones son uno de los síntomas más evidentes de errores de ejecución. Los "Java Exception Breakpoints" detienen la ejecución cuando salta una excepción del tipo seleccionado. ¡Estos puntos de ruptura se activan haciendo clic en el icono "J!" de la vista de "Breakpoints" o desde el menú principal "Run". La ejecución puede detenerse cuando la excepción sea capturada, no capturada o ambas. Añadir siempre los puntos de ruptura de excepciones Java de "ArrayIndexOutOfBoundsException" (lanzada cuando el índice de una matriz se sale de sus dimensiones) y "NullPointerException" (lanzada cuando se intenta acceder a una referencia que apunta a null) es una práctica de depuración recomendada.



Si se desea que el programa se detenga en los puntos de ruptura definidos deberá ser ejecutado en modo depuración ("Run > Debug..."). Tras detenerse en un punto de ruptura la ejecución del programa puede continuar de diversas maneras. Haciendo clic derecho en el editor de código dentro de la perspectiva de depuración aparecerá un menú contextual con estas opciones. "Run to line" reanuda la ejecución del programa hasta que se alcanza la línea en que está el cursor. "Step into selection" continuará la ejecución dentro del método seleccionado siempre y cuando el código fuente del método esté disponible. La ejecución también puede reanudarse mediante un clic derecho en la ventana de "Debug" y seleccionando las opciones adecuadas, o directamente pulsando los iconos de dicha ventana. "Step over" parará en la línea siguiente a la invocación de un método. "Resume" reanudará la ejecución normal del programa y sólo se interrumpirá en los puntos de ruptura si sus condiciones de activación se satisfacen.



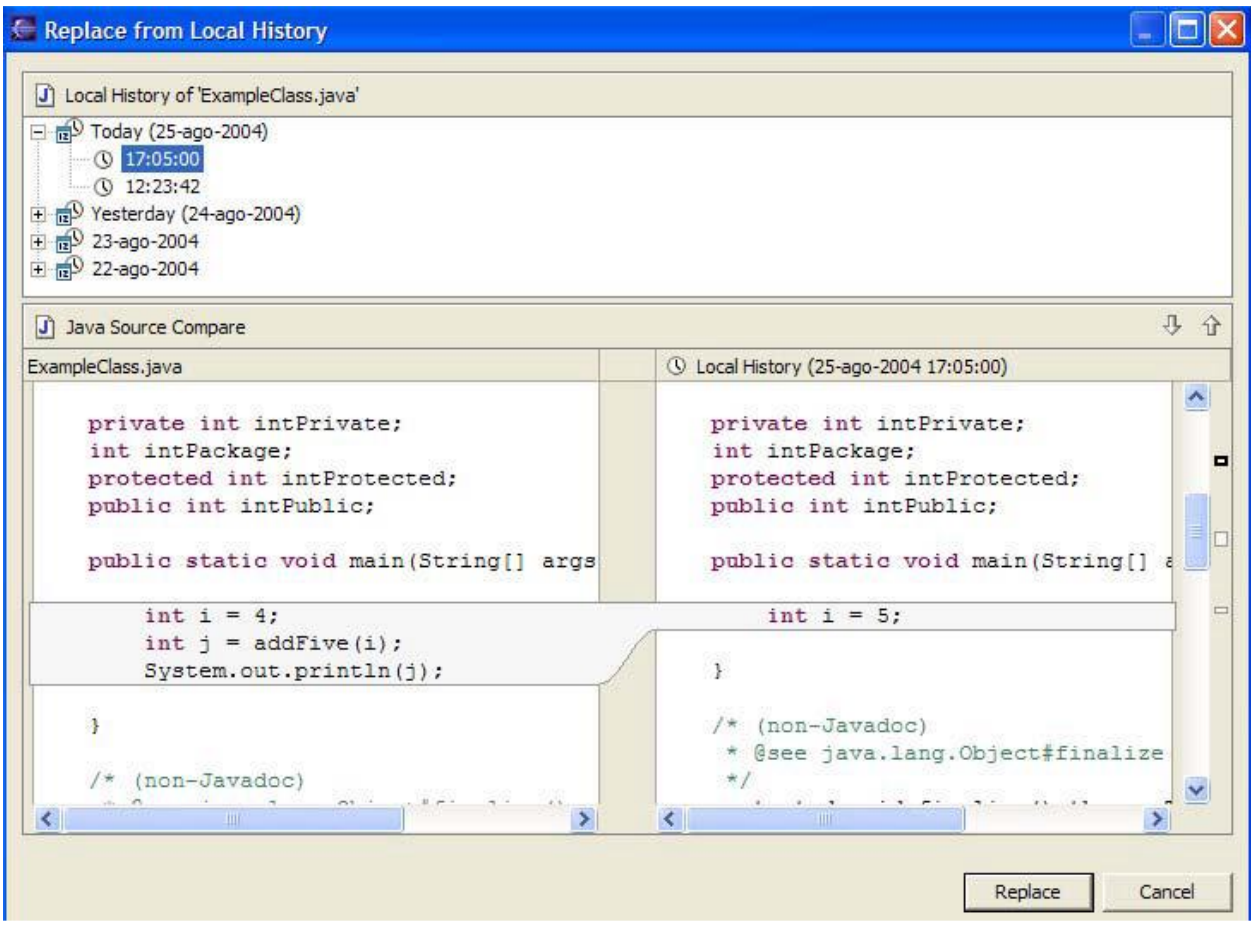
La vista "Variables" proporciona información verdaderamente útil ya que muestra los valores que tienen actualmente las variables cuando la ejecución se detiene en un punto de ruptura.



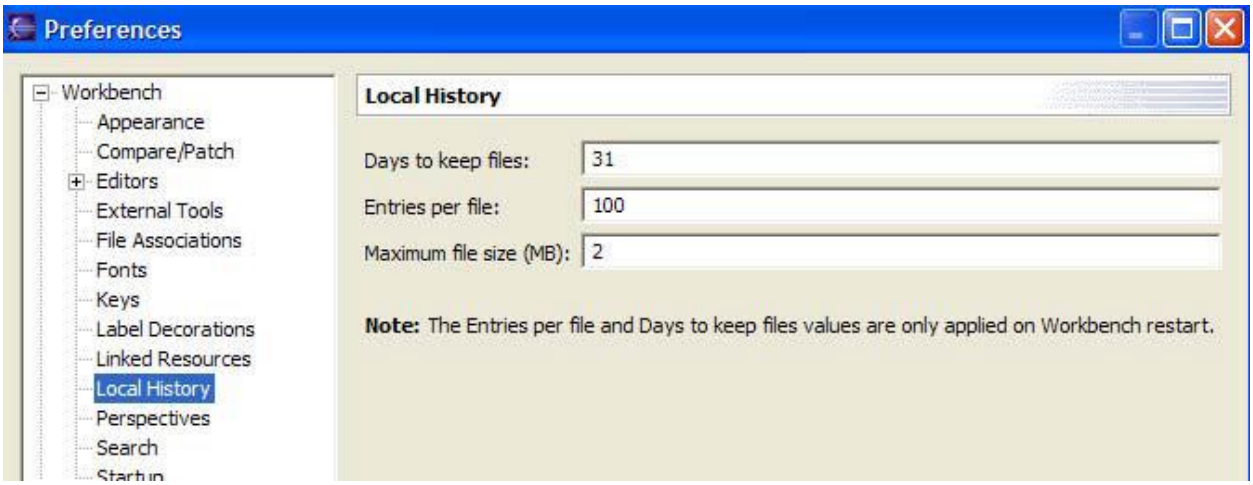
La vista de "Debug" también es útil para observar diferentes procesos que están siendo ejecutados simultáneamente, como, por ejemplo, hebras. Cuando el proceso de depuración ha terminado, los procesos mostrados en la ventana de depuración se muestran como "Finished" (pueden acabarse manualmente con "Clic derecho > Terminate"). La información de ejecuciones previas puede elminarse realizando clic derecho sobre ella y seleccionando "Terminate and Remove" o "Terminate All" más "Remove All Terminated".

Gestión de Cambios

No es extraño cambiar código del programa y darse cuenta después de que la nueva versión funciona incluso aún peor que la primera versión. Ese es el motivo de que los programadores tengamos que guardar diferentes versiones del programa cada vez que introducimos un nuevo cambio. Sin embargo, esta práctica inevitable es normalmente tediosa y consume mucho tiempo. Afortunadamente, Eclipse proporciona un potente sistema de gestión de cambios y de control de versiones. Haciendo clic derecho en un archivo Java dentro del Package Explorer y selecionando "Replace With > Local History" permitirá reemplazar la versión actual por una versión previamente guardada. La hora y fecha de modificación se muestran junto con dos ventanas que destacan las diferencias existentes entre ambas versiones.



Seleccionando "Window > Preferences > Workbench > Local History" permitirá seleccionar cuantas versiones, megas y días almacenar. Así pues, se puede obtener un buen equilibrio personalizado entre seguridad y eficiencia.



Enlaces

Para descargar la última versión del entorno de desarrollo Eclipse y para consultar otros tutoriales y documentos, visite el sitio web de eclipse: http://www.eclipse.org

Para descargar la máquina virtual Java J2SE JRE visite: http://www.java.sun.com

Descargar la Referencia completa de teclas rápidas de Eclipse 3. Este documento está diseñado para ser visto e impreso con la siguiente configuración de página: hoja A4, orientación apaisada (horizontal), 25,4 mm de márgenes izquierdo y derecho y 31,8 mm de márgenes superior e inferior. Tenga en cuenta que, puesto que se trata de un fichero RTF editable, puede cambiarlo para que sirva como referencia rápida para su configuración personalizada de teclado. Para contactar con el autor de esta web, Enrique Serrano, remita sus dudas y sugerencias a su dirección de e-mail: eclipsetutorial_)yahoo.es

Para visitar la página de proyectos de os4os donde puede acceder a noticias e información y contribuir a éste y otros proyectos de código abierto, visite: http://forge.os4os.org

El autor agradece a la Dirección General de Investigación del Ministerio de Educación y Ciencia (anteriormente MCyT) por su soporte en la realización de este trabajo (bajo referencia TIC2002-12426-E, proyecto ITEA-FAMILIES).

Gracias a todo el equipo de os4os por su apoyo.

Agradecimientos especiales a Susan Iwai y a todo el personal de Eclipse por su tiempo y por su ayuda con el proyecto.

Creado por Enrique Serrano Valle

Control de versiones	**Versión actual 1.0.1**	
Autor		
Fecha de creación		
Público objetivo	Monitores	
Tags	General, manuales	
Modificaciones		
Responsable	**Fecha**	**Cambio**
Nikolai Bermudez V	12/2/2020	Modificación de Formato
David Rivera	3/3/2021	Modificación de formato, correcciones gramaticales