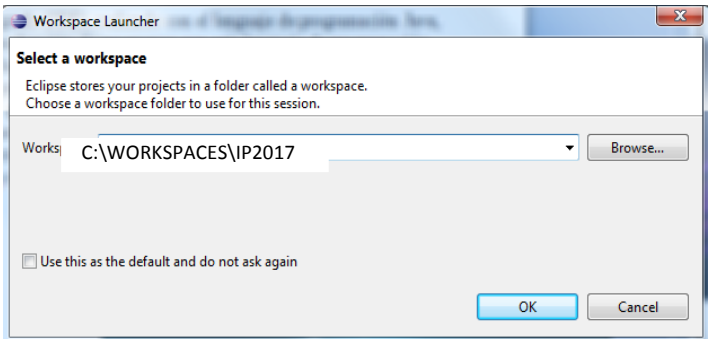
SESIÓN 1: Programación y ejecución de aplicaciones en Java

El entorno de desarrollo Eclipse Mars

Eclipse es un entorno de desarrollo integrado (IDE), realizado con el lenguaje de programación Java, que fue originalmente diseñado para la realización de programas con este lenguaje de programación. Con el tiempo, su uso y popularidad se han ido extendiendo entre los programadores profesionales. En la actualidad permite desarrollar programas en varios lenguajes de programación.

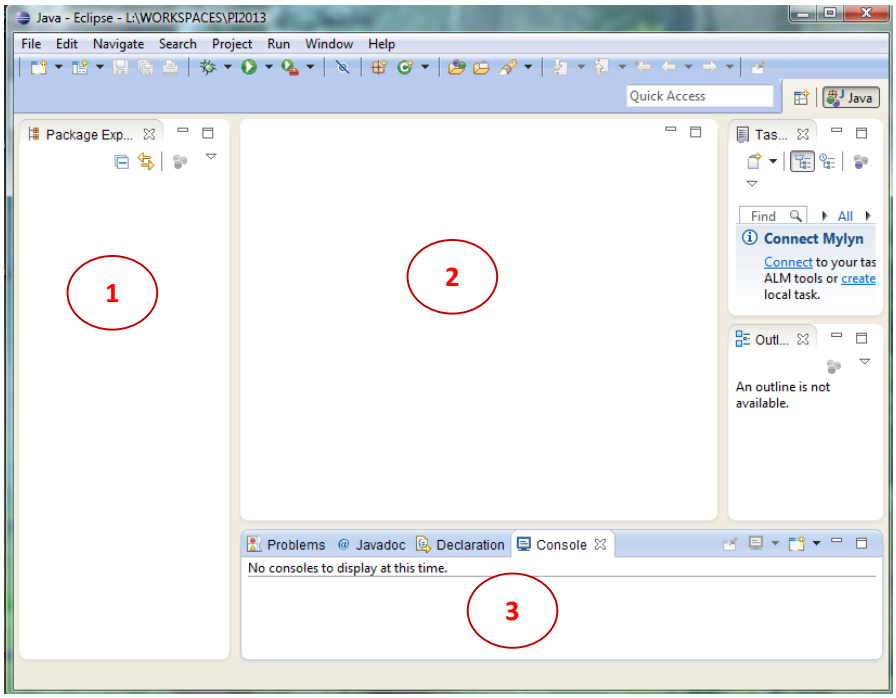
Cuando arrancamos Eclipse, lo primero que aparece es una pequeña pantalla de inicio (lo que se conoce en inglés como *splash screen*) que desaparece al cabo de un ratito y, de inmediato, aparece una ventana preguntándonos por el directorio o carpeta donde queremos ubicar nuestro espacio de trabajo (***workspace***). Dicho **workspace** contendrá los **proyectos** que utilicemos en Eclipse.

(a) *Splash screen* que aparece al arrancar *Eclipse*.



(b) Selección del directorio para ubicar el espacio de trabajo (*workspace*).

A continuación se mostrará la siguiente perspectiva:



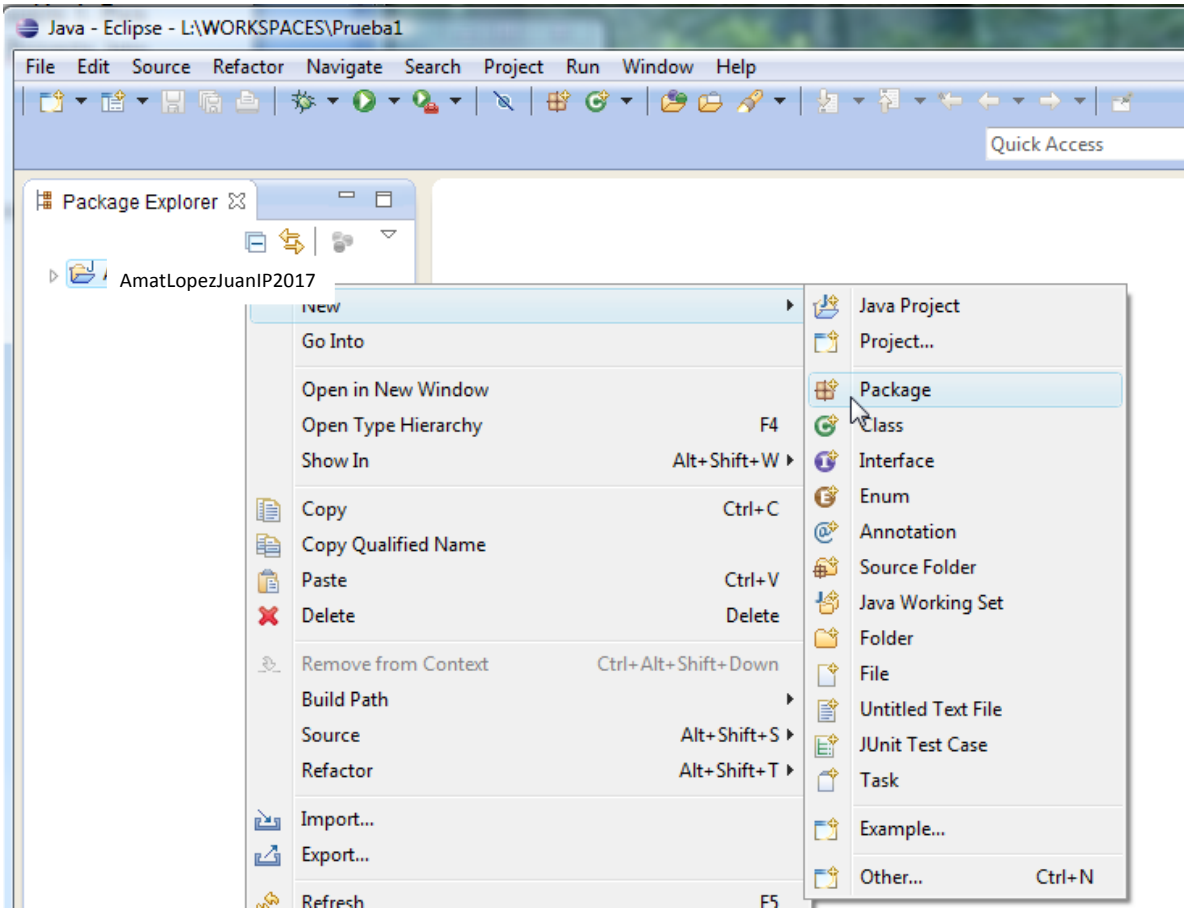
a. **Área #1**: Explorador de Paquetes. Siguiendo una estructura arborescente (forma de árbol), en esta área se muestran todos los proyectos, paquetes y clases que se almacenan en el *workspace* seleccionado.
b. **Área #2**: Editor de código. Zona de trabajo donde escribiremos el código fuente.
c. **Área #3**: Esta área está compuesta por múltiples "pestañas", entre las que destacamos la pestaña **Problems**, que mostrará aquellos mensajes de error en tiempo de compilación. Y, por otra parte, está la pestaña **Console**, que será donde se muestren los mensajes de error o bien los mensajes a los que da lugar nuestro programa. Otras pestañas se utilizarán más adelante.

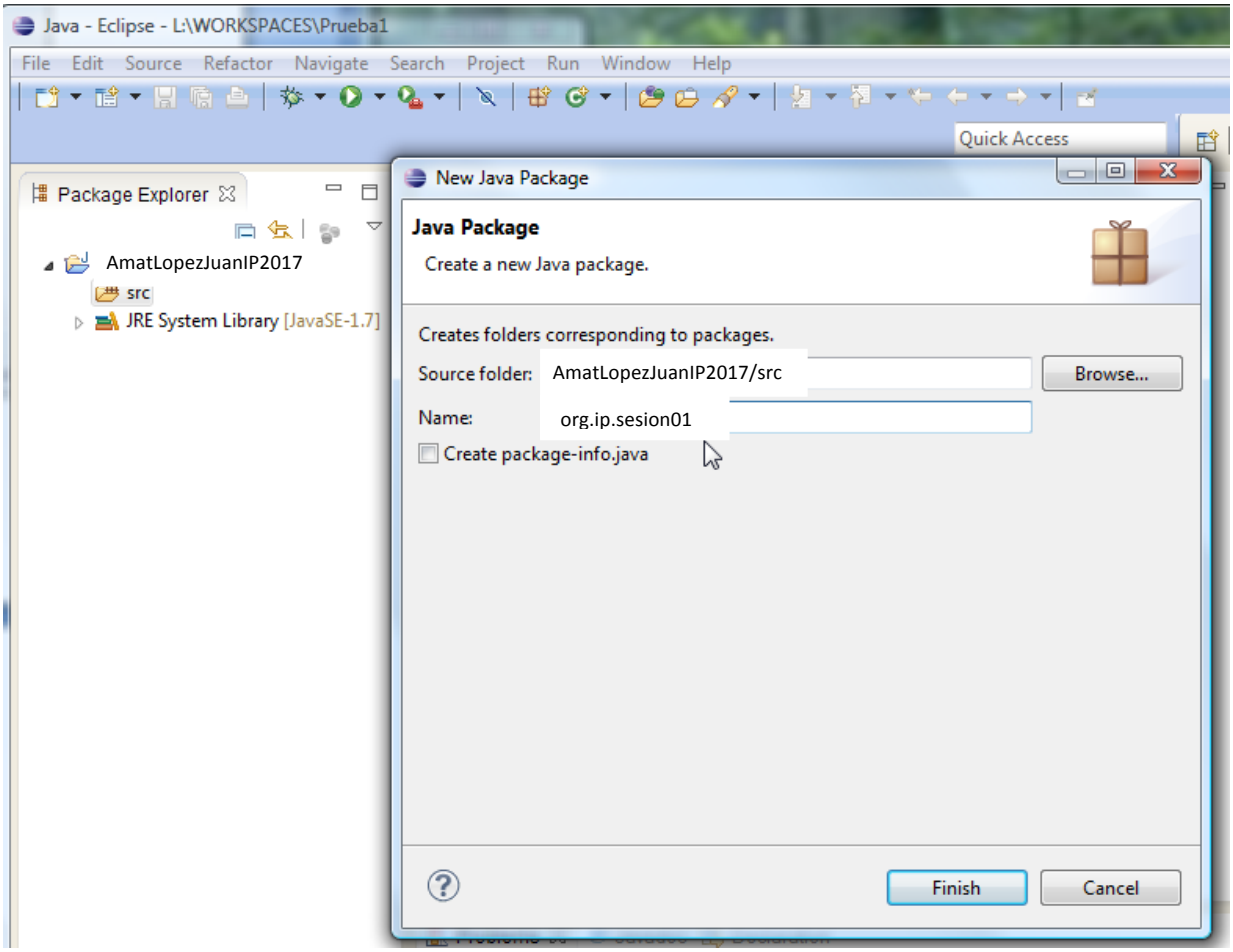
En el entorno de desarrollo Eclipse **todo archivo se almacena dentro de un proyecto (*project*)**. Esto quiere decir que todo documento, carpeta, archivo de código fuente (.java) y código compilado (.class) tiene que estar contenido dentro de un proyecto. Así pues, el primer paso antes de usar Eclipse para programar en Java es comprender la estructura de proyectos de Eclipse.
Para **crear un nuevo proyecto** Java nos situaremos en el área del explorador de paquetes y seleccionamos botón derecho, **New > Java Project** y nos guiaremos por el asistente.

Podemos también **importar un proyecto** ya existente. Para ello seleccionaremos igualmente el botón derecho del ratón, **Import > General > Existing Projects into Workspace > Select archive file** e indicamos donde se encuentra.

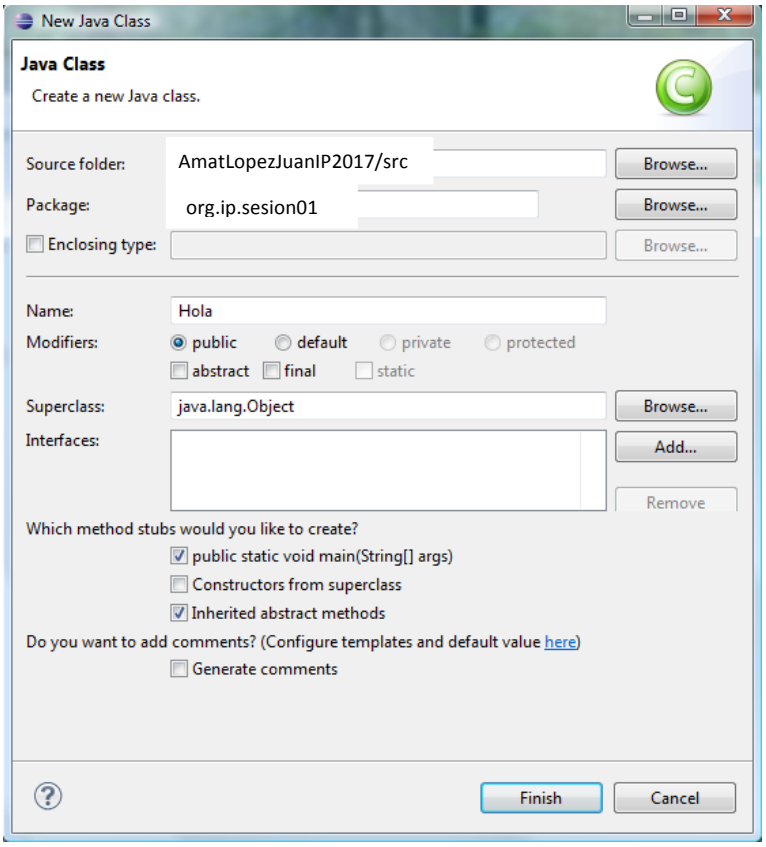
Creación (programación) de una aplicación Java

1.- En el proyecto **creamos un paquete** (**Package**) con el nombre indicado en la sesión **org.ip.sesion01**
Botón derecho sobre el nombre del proyecto, y seleccionamos **New > Package**

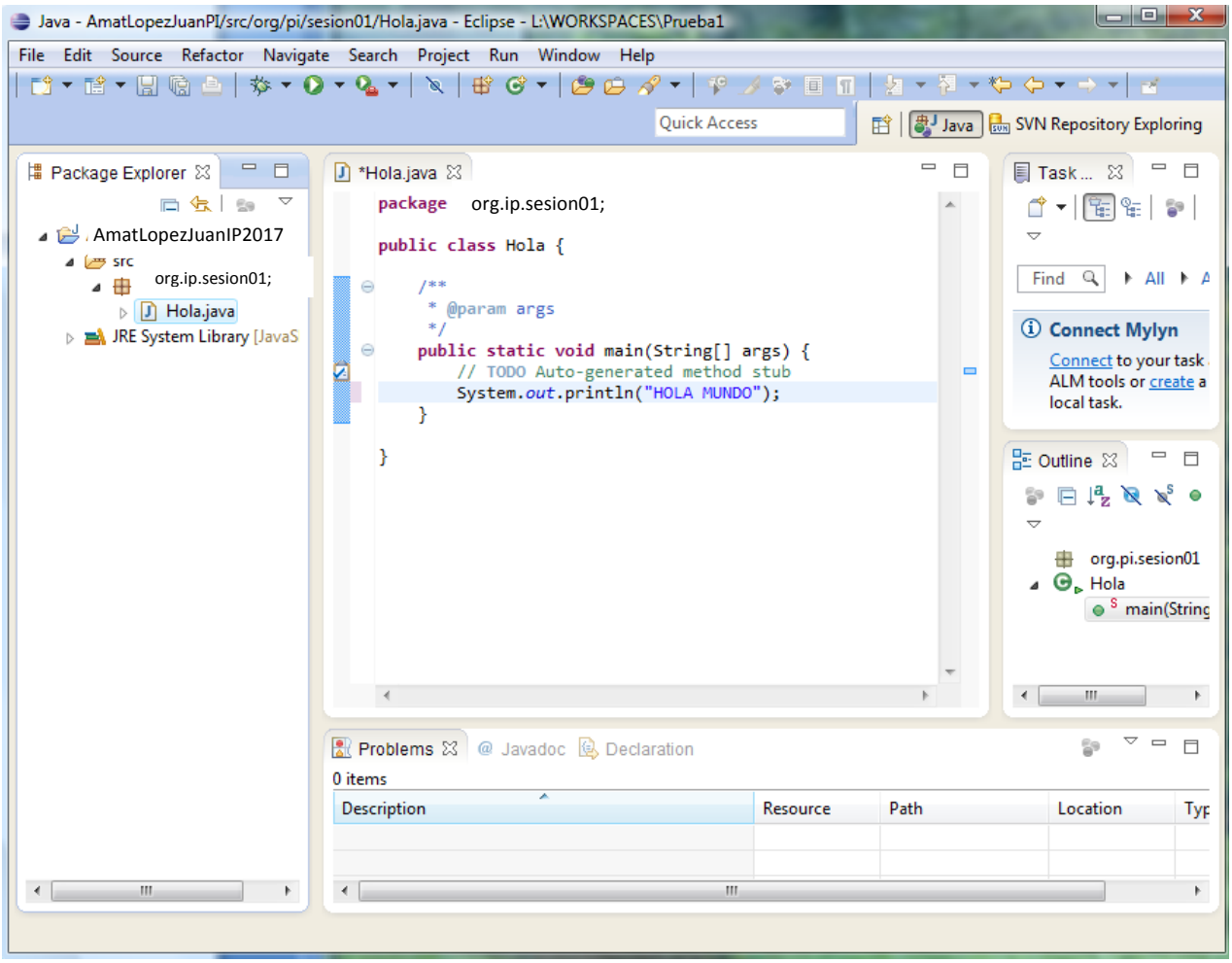




2.- Dentro del paquete crearemos las clases java. Sobre el nombre del paquete, pulsamos botón derecho y seleccionamos **New > Class**. A continuación damos el nombre a la clase.

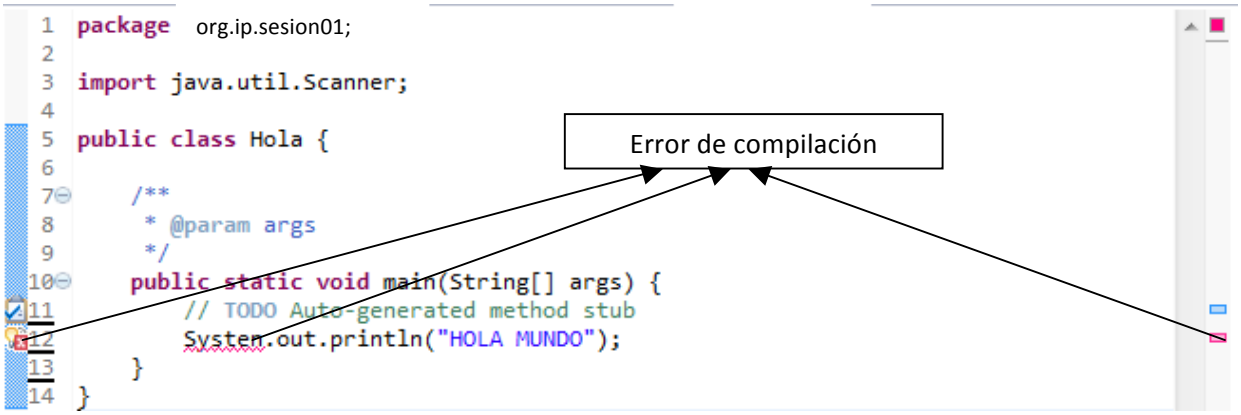


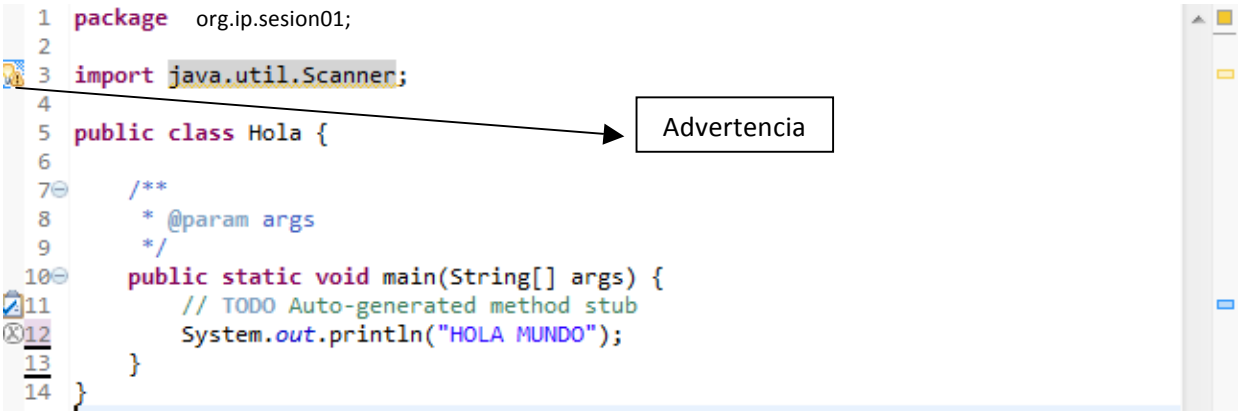
3.- Ya podemos escribir el **código** de nuestro programa usando el Editor de Eclipse:



Compilar y detectar errores

La compilación es una tarea que se lanza automáticamente al guardar los cambios realizados en el código. Por esta razón, no se va a controlar manualmente. No será necesario pasar por el tedioso y lento proceso de compilar - observar los errores - corregir los errores.
En Eclipse los errores de compilación se muestran **en tiempo real** subrayando el fragmento de código adecuado con una línea roja. Los errores pueden encontrarse fácilmente porque se muestran además como marcas rojas en el margen derecho del editor de código Java. Las advertencias (*warnings*) se muestran de la misma manera, pero con marcas amarillas.





Icono de Bombilla = Autocorregir

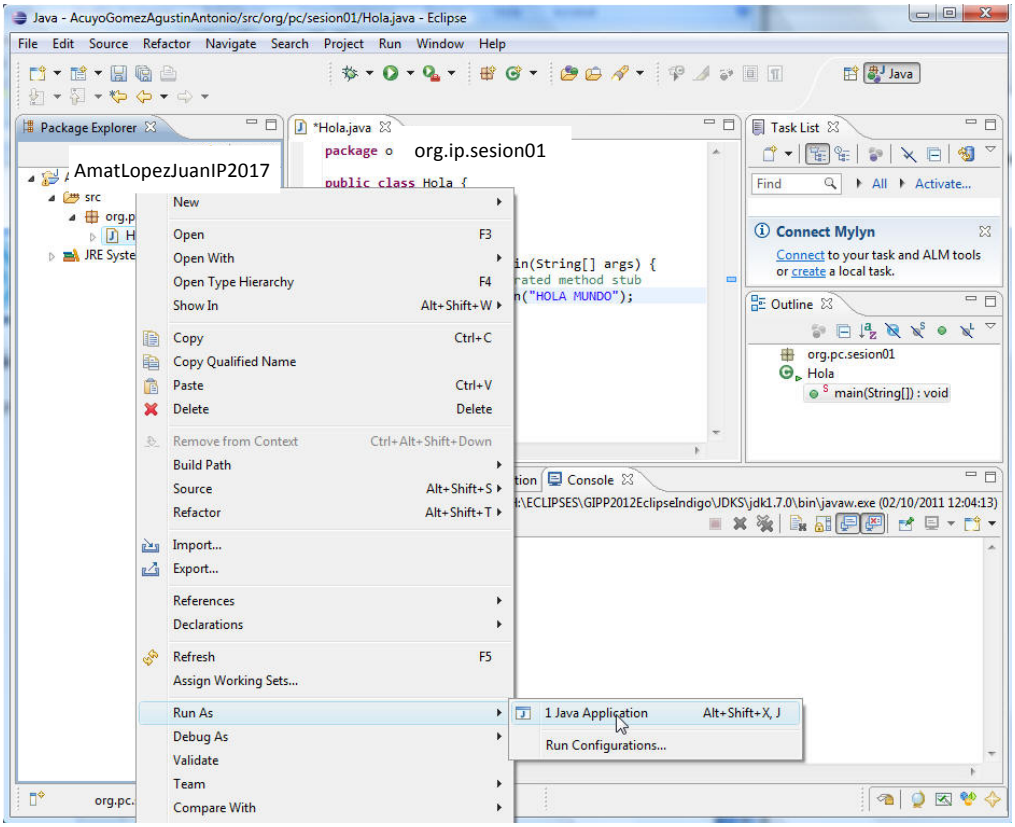
Hemos visto como Eclipse detecta y marca todo error y advertencia de compilación. Eclipse habitualmente permite autocorregir los posibles errores haciendo clic en el icono de bombilla presente en el margen izquierdo del editor de código. Así pues, aparecerá una ventana mostrando todas las opciones. Seleccionar 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

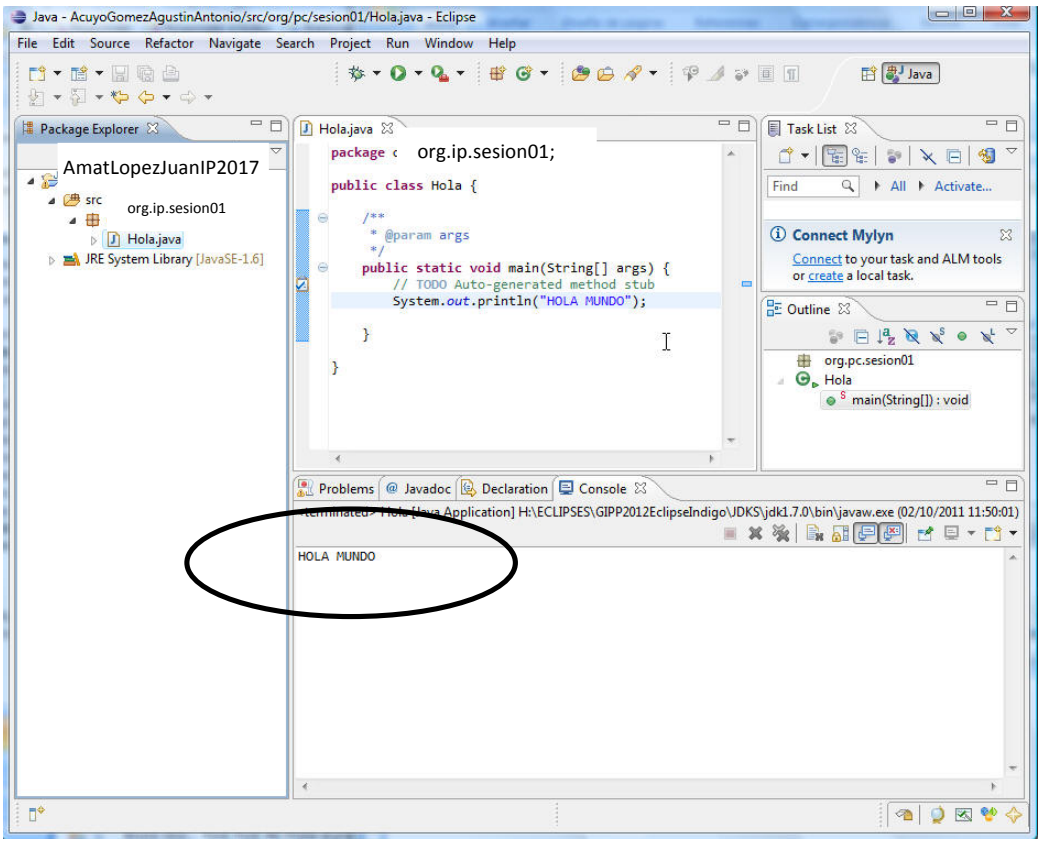
Una vez que conozcas la útil función de autocompletar la usarás continuamente. Se escribirán los primeros caracteres de una sentencia, clase, etc. y a continuación CTRL + Espacio y se autocompletará el resto.

Ejecución de una aplicación Java

Sobre el nombre de la clase, pulsamos botón derecho y seleccionamos: **Run As > Java Application**

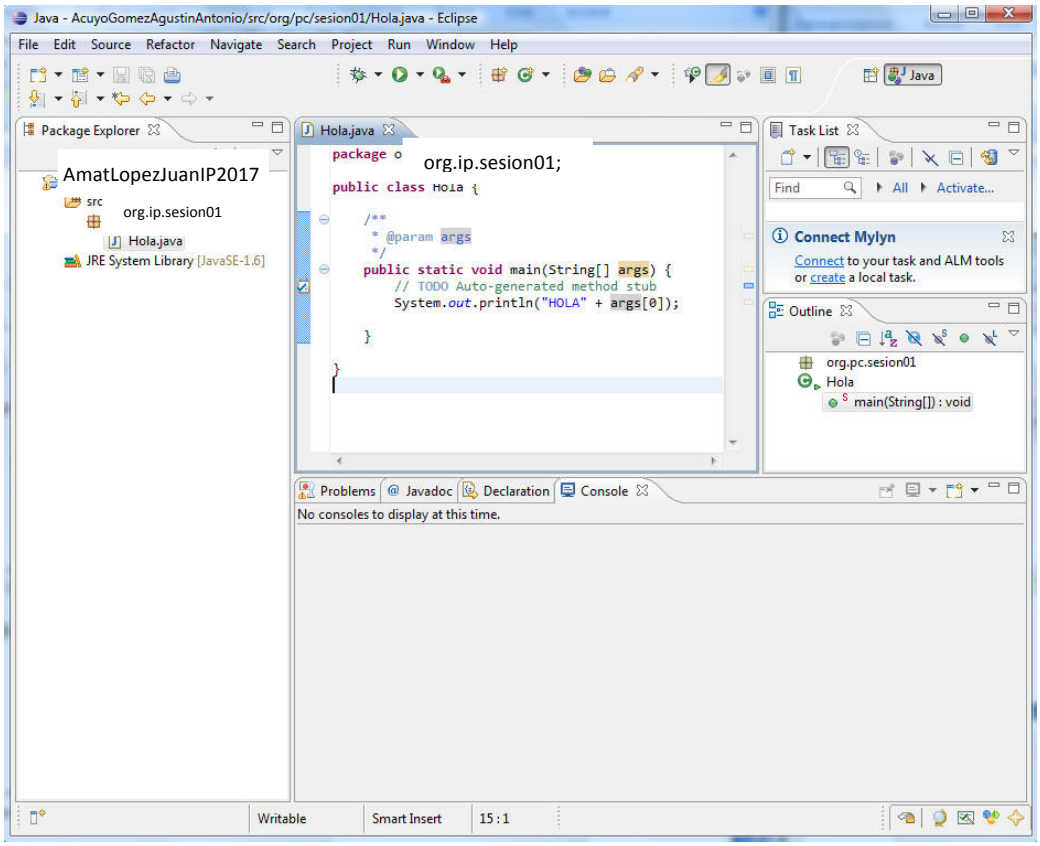


La salida que se genera al ejecutar el programa se muestra en la pestaña **Console** :

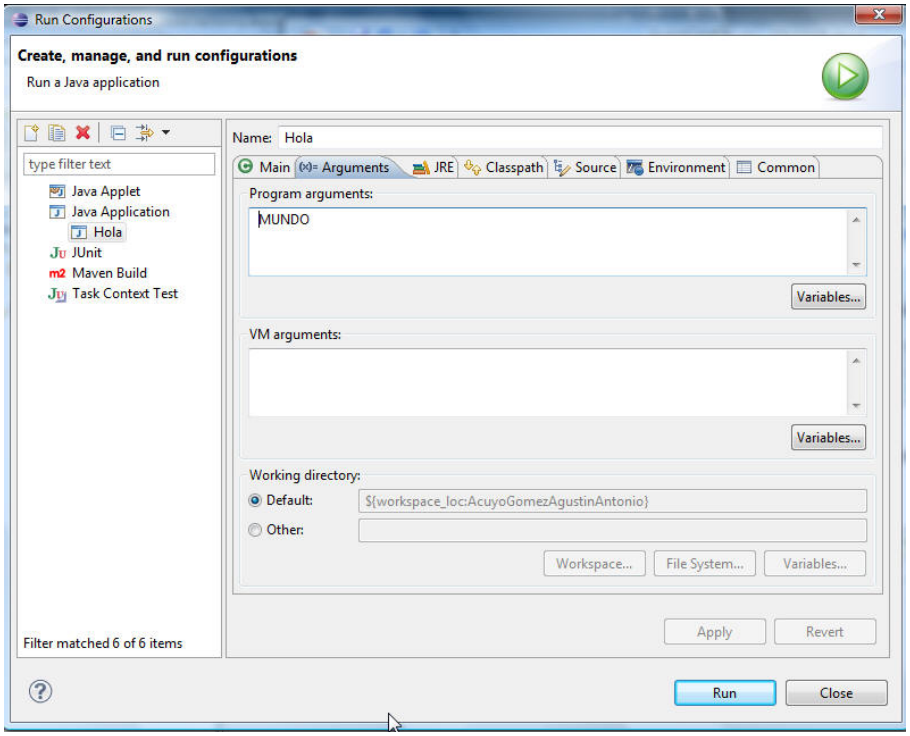


Ejecución con argumentos

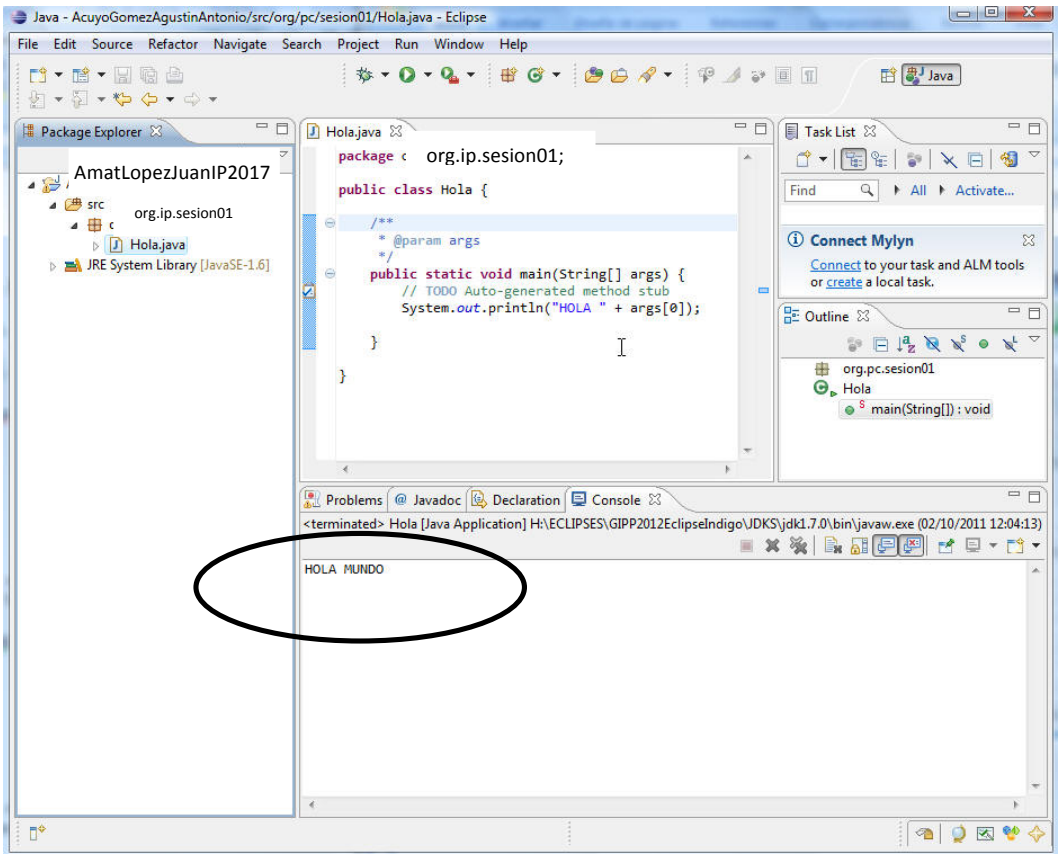
Los argumentos son valores de entrada que el programa necesita para su ejecución, y que serán almacenados en la variable **args** del método main(). Para introducir estos valores tenemos que volver a seleccionar el menú de ejecución sobre el nombre de la clase, y seleccionar: **Run As > Run Configurations**



Introducimos los argumentos necesarios en la ventana "Program Arguments" y ejecutamos (**Run**):

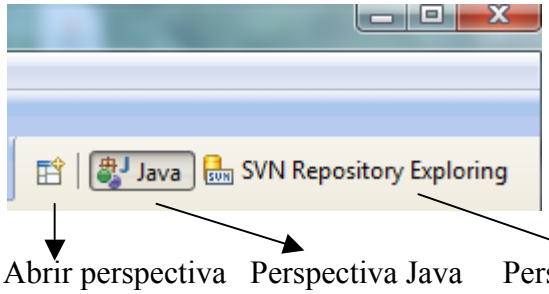


Salida generada:



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. Los editores normalmente permiten realizar una tarea completa, las vistas proporcionan funciones de apoyo o auxiliares tales como mostrar información, requerir datos, etc. Las perspectivas pueden seleccionarse haciendo clic en los iconos de perspectiva en la esquina superior derecha o eligiendo **Window > Open Perspective** del menú.



Perspectivas que usaremos:

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

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

SVN: esta perspectiva se centra en el trabajo con repositorios. Permitirá establecer conexiones con diferentes repositorios, acceder a los mismos, navegar por ellos etc.