Trabajo con el depurador. Perspectiva Debug de Eclipse

Depurar Aplicaciones

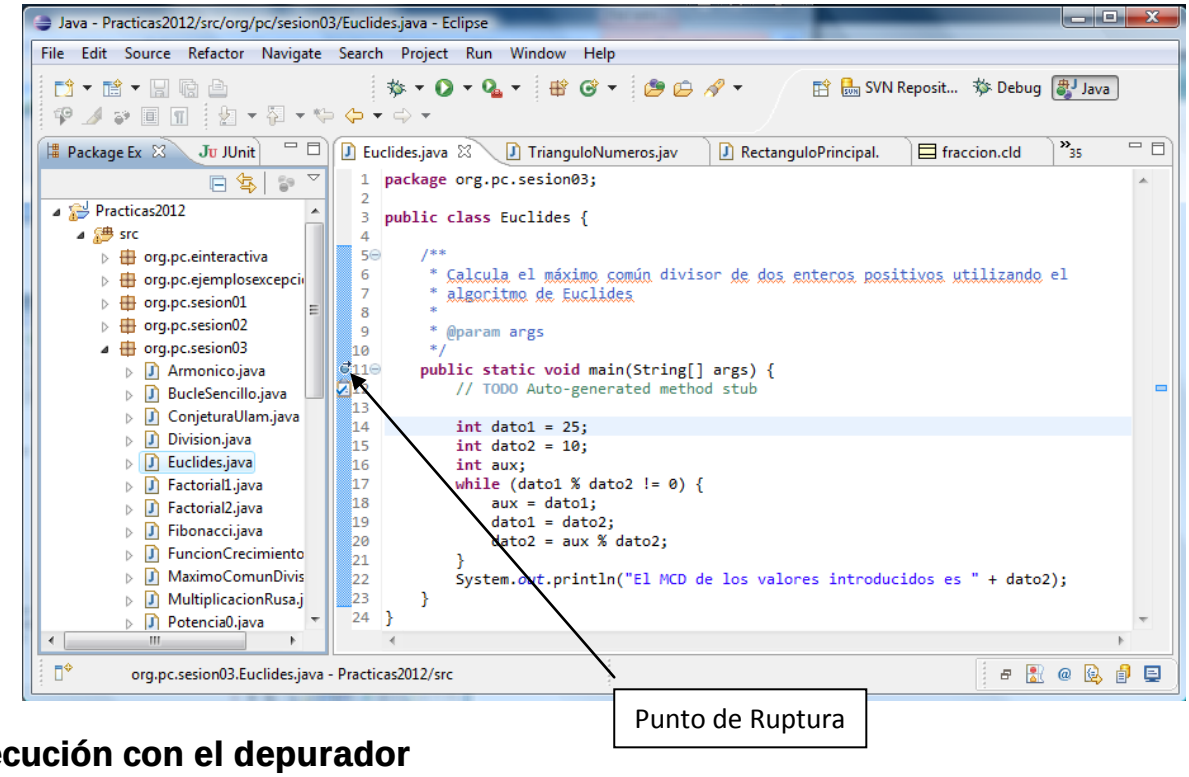
- ✓ La principal diferencia entre un simple editor y un buen entorno de desarrollo es que éste integre, o no, una buena herramienta visual para depurar los programas escritos. Eclipse incluye un depurador potente, sencillo y muy cómodo de utilizar.
- ✓ Cuando se lanza el proceso de depuración, siempre se realiza una compilación y construcción completa del código.
- ✓ Cuando el depurador entra en acción, de forma automática, se abre la **Perspectiva Depuración**, en la que se muestra toda la información relativa al programa que se está depurando.

Un depurador es una herramienta que permite seguir **el programa línea a línea y ver cómo cambian los valores de las variables** y expresiones que tengamos seleccionadas durante la ejecución de un programa. Permite, por tanto, *controlar y analizar* la ejecución del programa. Ayuda también a localizar diferentes tipos de errores. Se pueden realizar, entre otras, las siguientes tareas:

1. Establecer puntos de ruptura, interrupción o parada (**breakpoint**). La ejecución del programa se detiene en esos puntos, pudiendo continuar con una ejecución línea a línea, observar valores de las variables etc.
2. Hacer un seguimiento línea a línea.
3. Observar los cambios en los valores de ciertas variables.

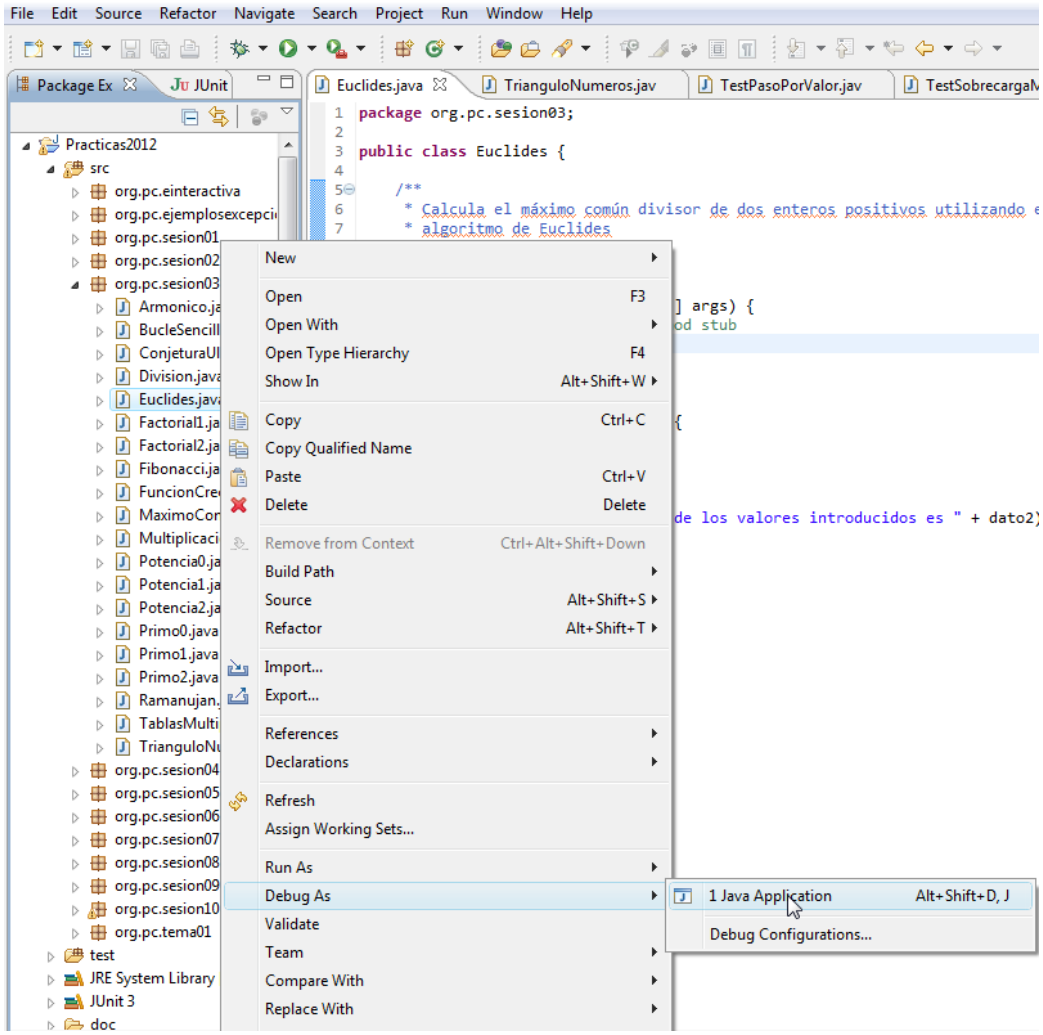
Colocar un Breakpoint

Antes de iniciar una sesión de depuración pondremos al menos un *breakpoint*. Establecer un *breakpoint* es tan sencillo como hacer doble clic en el margen izquierdo del Editor del código, a la altura de la línea sobre la que se quiere detener la ejecución. El *breakpoint* creado quedará identificado por un punto azul sobre la línea.

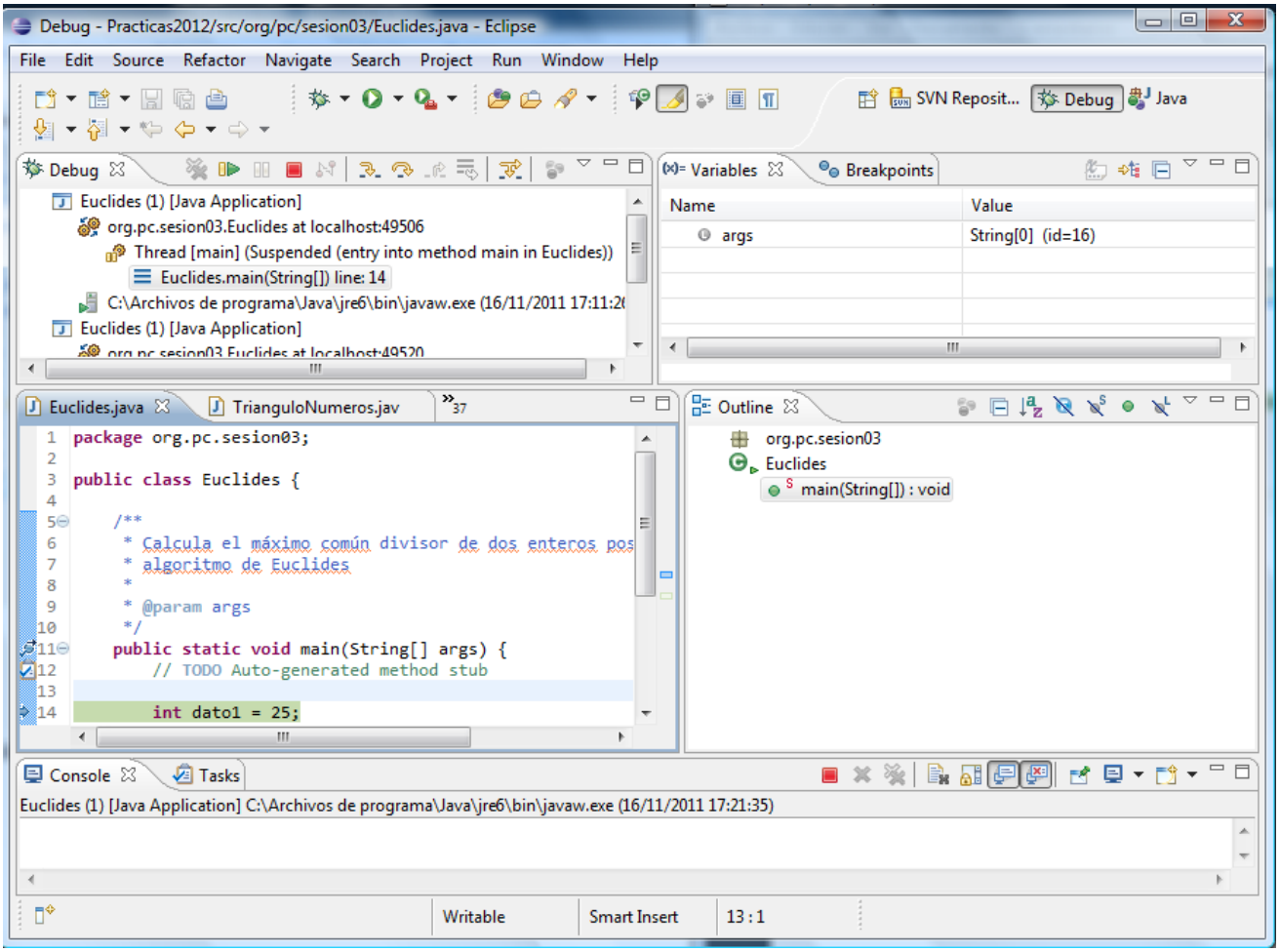


Ejecución con el depurador

Seleccionamos el programa que deseamos depurar, pulsamos el botón derecho y la opción Debug As, Java Application.



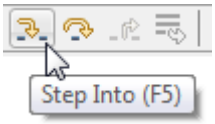
Se inicia una sesión de depuración abriéndose una nueva perspectiva, Debug.



La vista de inspección (a la derecha de la vista Debug), permite ver los valores de los diferentes elementos (variables, breakpoints, expresiones…) que intervienen en el programa, en un instante de ejecución determinado.
Las dos vistas más interesantes son la vista de inspección de variables, que muestra los valores que toman todas las variables (atributos, campos, etc.) cuyo ámbito alcanza a la línea que se está ejecutando en un momento dado y la lista de inspección de breakpoints.

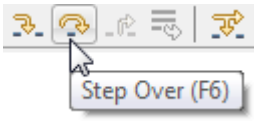
Ejecución paso a paso

Step Into (F5)



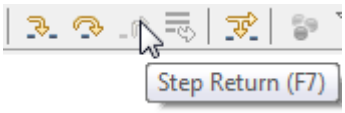
- ✓ Paso a paso por cada instrucción. Si la instrucción es una llamada a un método, la ejecución continua dentro del método llamado.

Step Over (F6)



- ✓ No entra en las llamadas a métodos.

Step Return (F7)

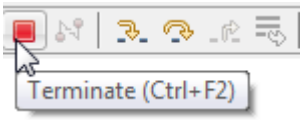


- ✓ Ejecuta hasta el final del método actual y vuelve. Para después de la llamada al método(o si encuentra un breakpoint).

Resumir ejecución hasta encontrar un Breakpoint

- ✓ Para no ir paso a paso y saltar la ejecución hasta el siguiente Breakpoint, o el final del programa.

Terminar la ejecución paso a paso



Evaluar expresiones