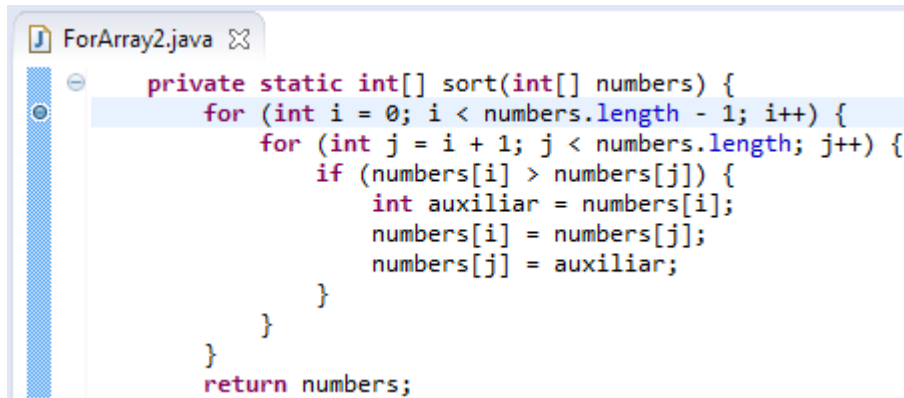


I / Depuración con Eclipse

Lo primero es colocar un **punto de interrupción** (*breakpoint*), en una línea de código de nuestro programa. Doble click en el margen izquierdo de la ventana donde vemos el código para crear el *breakpoint*.

Usaremos el código siguiente para probar la herramienta de depuración:



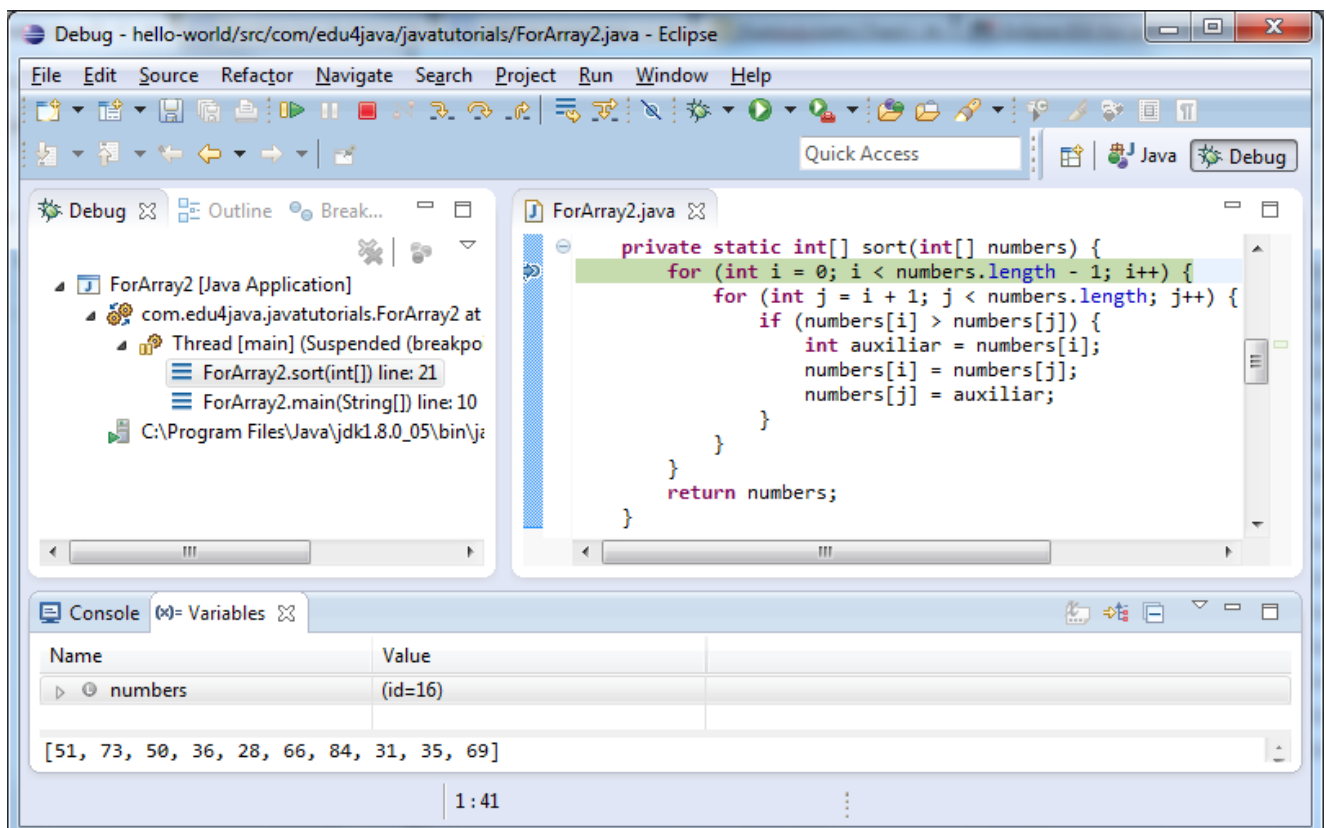
```

private static int[] sort(int[] numbers) {
    for (int i = 0; i < numbers.length - 1; i++) {
        for (int j = i + 1; j < numbers.length; j++) {
            if (numbers[i] > numbers[j]) {
                int auxiliar = numbers[i];
                numbers[i] = numbers[j];
                numbers[j] = auxiliar;
            }
        }
    }
    return numbers;
}
  
```

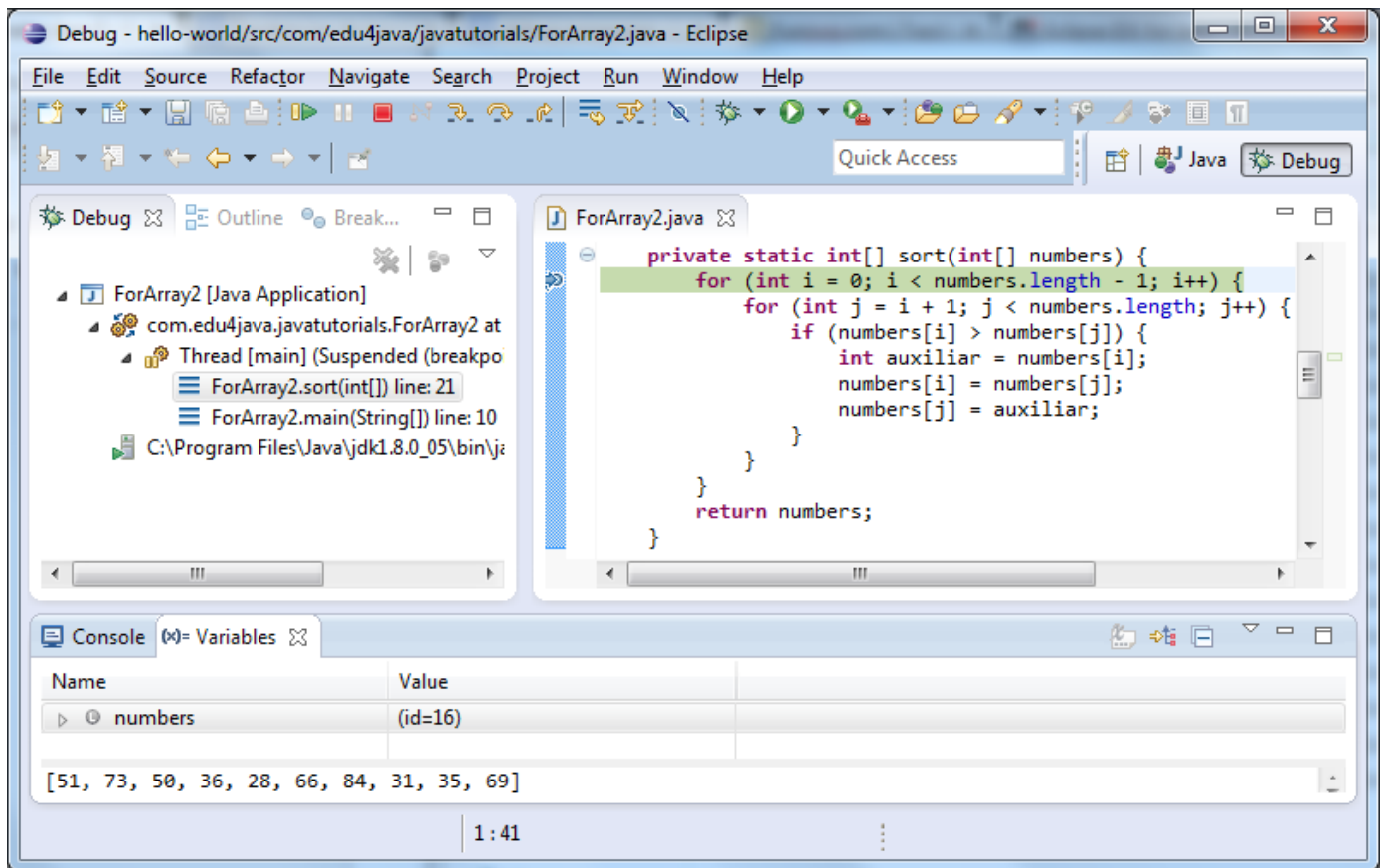
Lo siguiente es iniciar nuestro programa con la opción depurar (debug). Nuestro programa se ejecutará normalmente hasta encontrar la línea donde colocamos el punto de interrupción. Eclipse detendrá la ejecución y solicitará permiso para cambiar a la perspectiva Debug. Hacer click en "Remember my decision" y "yes". Entonces aparecerá la perspectiva debug con nuestra línea marcada, mostrando información sobre el estado de la ejecución.

Perspectiva Debug en Eclipse

Eclipse tiene varias formas de mostrarnos nuestros proyectos. Estas formas son llamadas perspectivas. Aunque no lo hayamos notado, hemos estado trabajando hasta ahora en la perspectiva Java.



Cuando cambiamos a la perspectiva Debug, la primera impresión es confusa ya que aparecen nuevas vistas (ventanas) y cambian de posición con respecto a la perspectiva Java. Dado que usaremos mucho la perspectiva Debug, aconsejo reordenar las pestañas. Arrastrando y soltando las pestañas, podremos modificar su posición.



En la esquina derecha arriba, podemos ver la barra de perspectivas con la perspectiva Debug seleccionada. Si queremos volver a la perspectiva Java, basta con hacer click en Java.

Las nuevas vistas de la perspectiva debug son Debug, Variables y Breakpoint. Además, la vista de código, muestra marcado con verde, la línea donde está detenida la ejecución.

Vista Variables

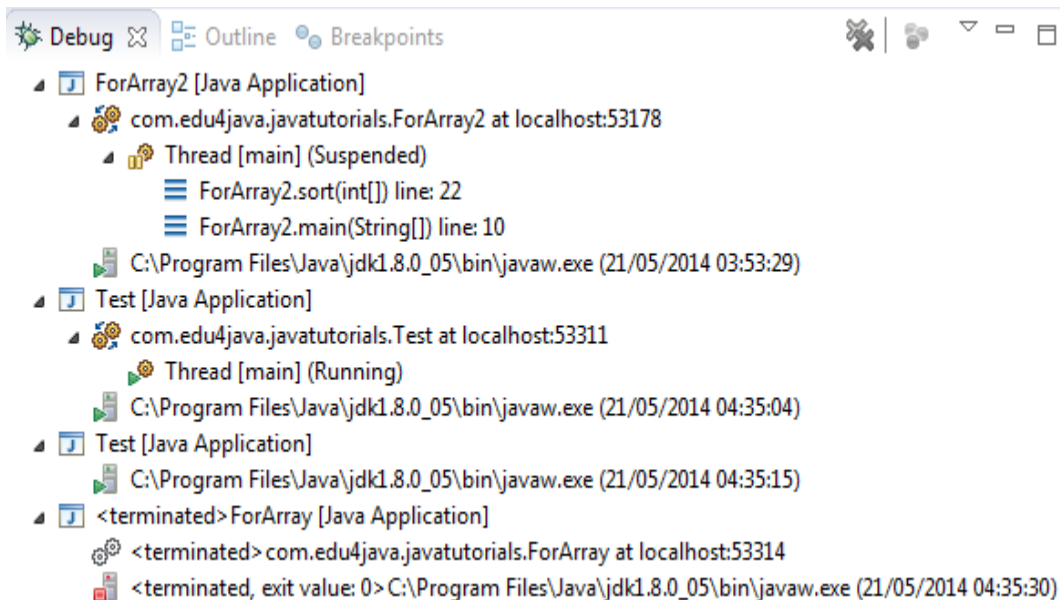
En esta vista veremos una línea por cada variable, campo o parámetro. Si hacemos click en cada uno, veremos su contenido.

Vista Breakpoints

Aquí podemos ver la lista de todos los puntos de interrupción que tenemos. Podemos habilitar o deshabilitar un breakpoint con el check a la izquierda de cada breakpoint.

Vista Debug

La vista debug nos muestra la lista de aplicaciones que se están ejecutando y algunas que terminaron su ejecución.



En la imagen, podemos ver que los tres primeros programas están ejecutándose y el último terminado. Los dos primeros, se ejecutan en modo debug y se puede ver el hilo de ejecución (thread). El tercero no está en debug por lo que no se ve el hilo.

El primero tiene la ejecución suspendida en la línea 22, del método sort, de la clase ForArray2. También vemos que está detenido en el método main, línea 10 de la misma clase. Esto es lo que se conoce como pila de ejecución.

La pila de ejecución, se forma con las subsecuente llamadas de un método dentro de otro método. En la base, está el primer método "main" desde donde se llamó al método sort.

Botones para controlar la depuración

En la barra de botones principal, al lado de imprimir, vemos los botones para manipular la ejecución de la depuración de código.



Las funciones de estos botones, por orden, son:

1. **Resume (F8)**; continúa con la ejecución (hasta el próximo breakpoint).
2. **Suspend**; podemos detener la ejecución aunque no alcancemos un breakpoint (muy útil cuando entramos en un ciclo infinito).
3. **Stop**; detiene la depuración.
4. **Step Into (F5)**; se detiene en la primer línea del código del método que estamos ejecutando. Si no hay método, hace lo mismo que Step Over.
5. **Step Over (F6)**; pasa a la siguiente línea que vemos en la vista de código.
6. **Step Return (F7)**; vuelve a la línea siguiente del método que llamó al método que se está depurando actualmente. O lo que es lo mismo, sube un nivel en la pila de ejecución, que vemos en la vista Debug.

II / Depuración con NetBeans

La depuración (*debug*) permite examinar las posibilidades de observar las líneas que se van ejecutando, las variables en cada paso.

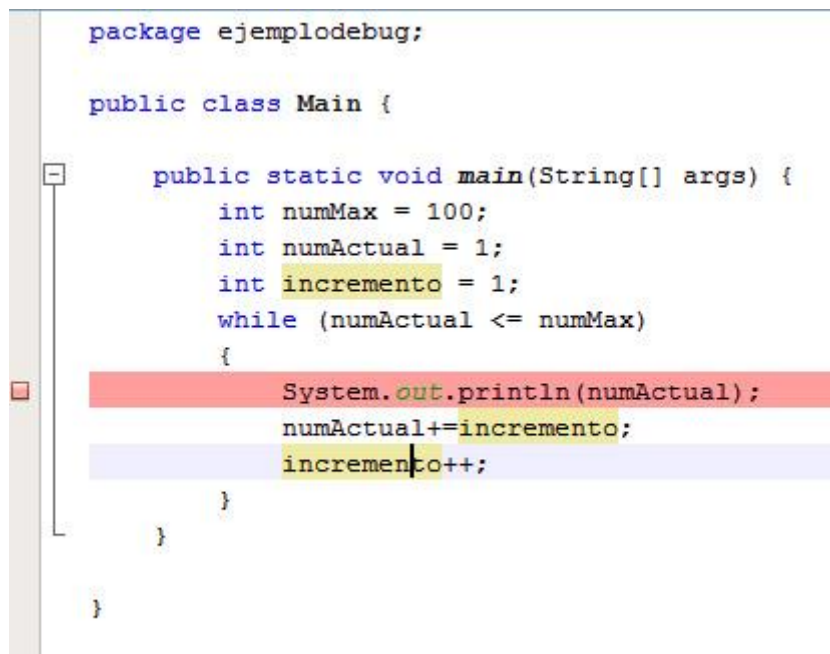


aplicaciones para buscar errores, ya que así como los valores que van tomando

Para realizar la depuración de un programa, se debe establecer en primer lugar un **punto de interrupción** donde debe pararse la ejecución de la aplicación. Esto se consigue con alguna de las siguientes acciones sobre la línea de código en la que se desee establecer el punto de interrupción:

- ▶ **Clic en el margen izquierdo** (en la línea de código donde comenzará la depuración)
- ▶ Menú contextual > **Breakpoint / Toggle Line Breakpoint**
- ▶ Pulsando la combinación de teclas: **Ctrl + F8**
- ▶ Menú Depurar > Toggle Line Breakpoint

Al realizar alguna de esas acciones, se marca en color rosado la línea que se ha convertido en un punto de interrupción, y se muestra un pequeño cuadrado en el margen izquierdo.



Una vez establecido al menos un punto de interrupción, se debe **ejecutar la aplicación en modo depuración**. Esto se puede llevar a cabo sobre el proyecto o sólo sobre el archivo actual:

Depurar archivo actual:

- ▶ Menú contextual > Debug *nombreArchivo*
- ▶ Menú Depurar > Debug *nombreArchivo*
- ▶ Pulsando la combinación de teclas: **Ctrl + Mayúsculas + F5**

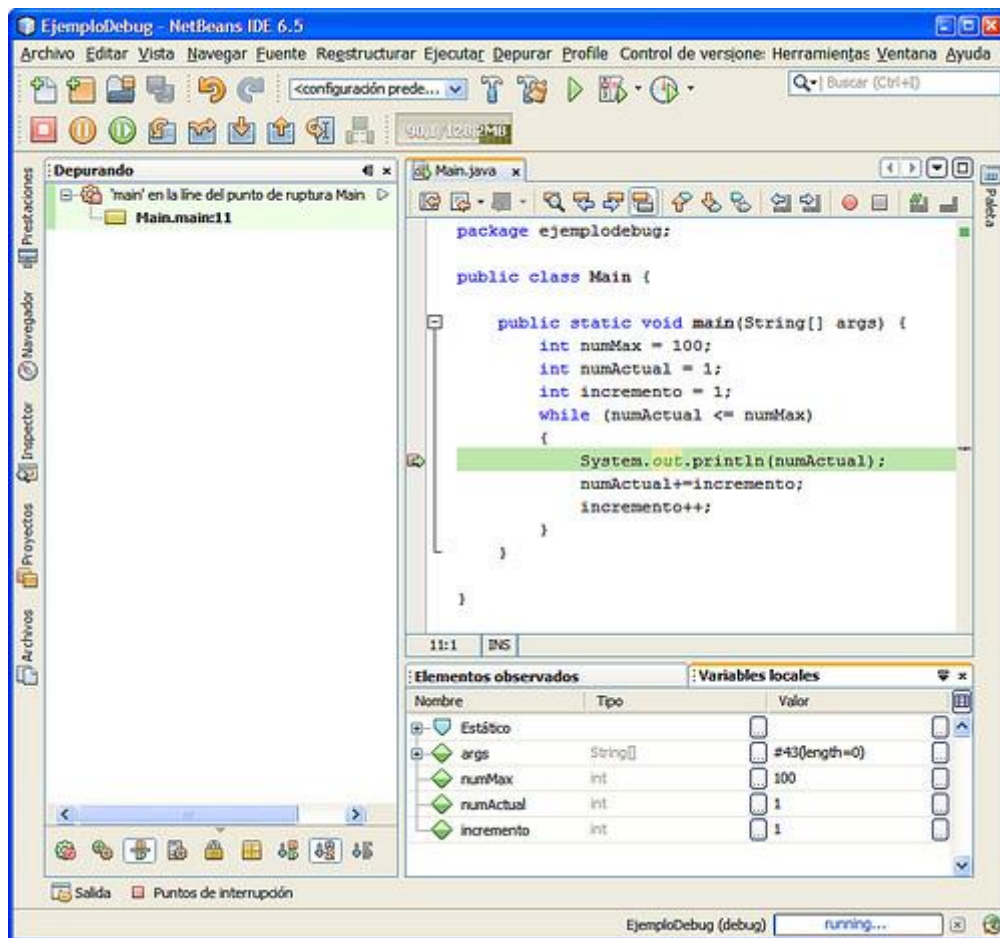
Depurar proyecto:

- ▶ Menú Depurar > Debug Main Project
- ▶ Icono "Debug Main Project"
- ▶ Pulsando la combinación de teclas: **Ctrl + F5**

Pulsa **Ctrl-F5** para comenzar el proyecto de depuración. El depurador ejecutará el programa hasta el primer punto de interrupción. Ahora puedes colocar el **ratón sobre las variables y las ventanas de información** aparecerán junto a ellas. Estas ventanas de información muestran el valor de la variable y el tipo. En la parte superior derecha de la ventana de NetBeans, se muestra el uso de memoria actual de tu programa.

Pasa a la siguiente línea de código presionando F7 o F8. La tecla **F7** hace que el depurador "entre en" (**step into**) el código, mientras que **F8** provoca que el código "pase a lo siguiente" (**step over**). Entrar en código muestra los entresijos de llamadas a funciones y profundizará en él cuando se realizan muchas llamadas de funciones anidadas. Por otro lado, pasar a través de él ignora el funcionamiento interno de las llamadas a funciones y sólo se enfoca en el valor que retorna. Si necesitas depurar una función llamada por el programa utiliza la opción de "entrar en" (step into), pero si deseas depurar el programa actual utiliza la opción de "pasar a lo siguiente" (step over).

Al llegar la ejecución de la aplicación al punto de interrupción establecido, se destaca en **color verde** la línea de código en la que se encuentre la **ejecución**.



En la parte inferior se muestra la **ventana de observación de las variables locales**, en la que se puede comprobar el valor que va tomando cada variable.

A partir del momento en que se para la ejecución del programa se puede continuar con la ejecución línea a línea utilizando la opción **"Continuar ejecución"**:

- ▶ Pulsando la tecla **F8**
- ▶ Menú "Depurar > Continuar ejecución"
- ▶ Icono "Continuar ejecución"