

Centro Formación Profesional Nro 35

Técnicas de programación, Lenguaje JAVA

Debugging

A la hora de encontrar y resolver problemas en nuestros programas es crucial que seamos hábiles usando el **Debugger**.

ejecutarla línea por línea para ir viendo cómo se modifican el valor de las variables y las acciones que realiza nuestro programa.

Es una herramienta simple integrada en cualquier **Características**

IDE que nos permite **definir un *breakpoint* (punto de parada) y ejecutar nuestro programa**

en modo debug (depuración). Al hacer esto, cuando el programa se detenga, podremos ir a paso a paso por nuestro programa y si llegamos a una línea que sea una función/procedimiento,

- Deben **retornar un valor**.
- Pueden **recibir parámetros**, aunque no es obligatorio es recomendado.

Centro Formación Profesional Nro 35

Técnicas de programación, Lenguaje JAVA

Pasos

1. Agregar uno o más *breakpoints*.
2. Iniciar el programa en modo *Debug*.
3. Ejecutar nuestro programa línea por línea.

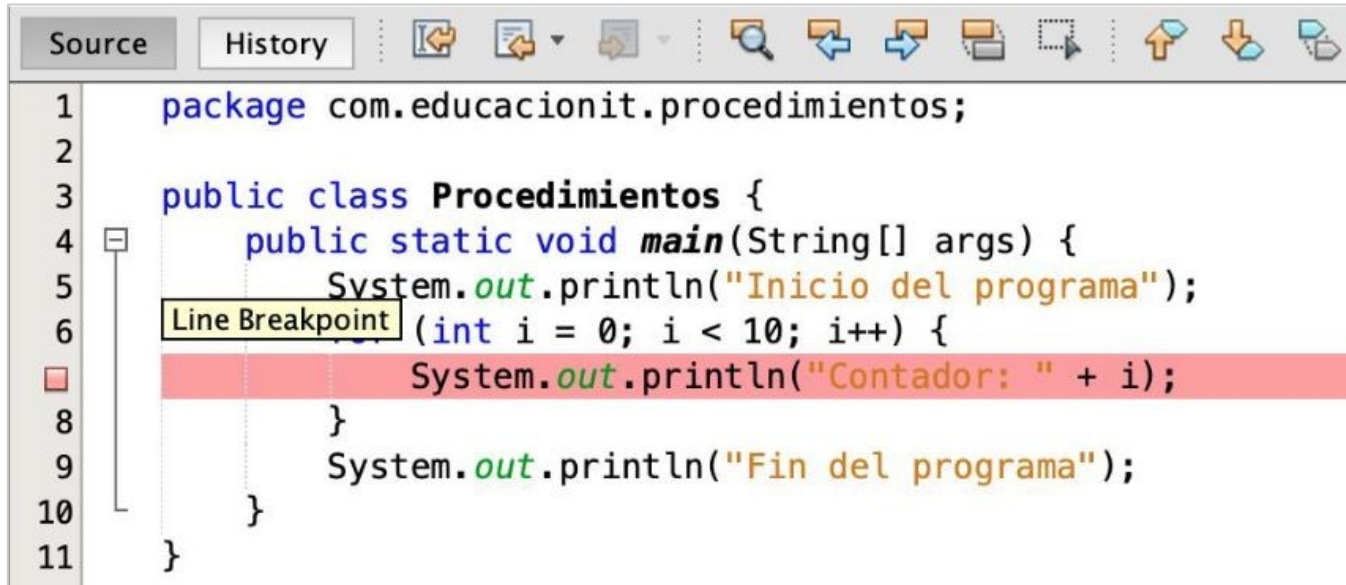
Paso 1: Agregar *Breakpoint*

En el lado izquierdo del editor de código de la siguiente pantalla, vemos el número de cada línea. Al hacer clic sobre el número podemos indicar que deseamos agregar un *breakpoint* en la línea correspondiente.



Centro Formación Profesional Nro 35

Técnicas de programación, Lenguaje JAVA



The image shows a screenshot of a Java IDE window. The window has a title bar with 'Source' and 'History' tabs. Below the tabs is a toolbar with various icons for file operations and navigation. The main area displays Java code for a class named 'Procedimientos'. The code is as follows:

```
1 package com.educacionit.procedimientos;
2
3 public class Procedimientos {
4     public static void main(String[] args) {
5         System.out.println("Inicio del programa");
6         (int i = 0; i < 10; i++) {
7             System.out.println("Contador: " + i);
8         }
9         System.out.println("Fin del programa");
10    }
11 }
```

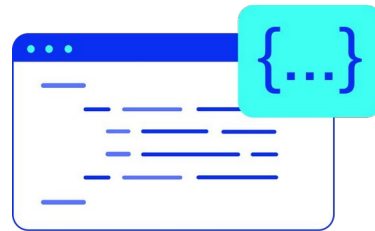
A line breakpoint is set on line 6, indicated by a small square icon in the left margin and a yellow box labeled 'Line Breakpoint' over the code. The line of code on line 6 is highlighted in red.

Centro Formación Profesional Nro 35

Técnicas de programación, Lenguaje JAVA

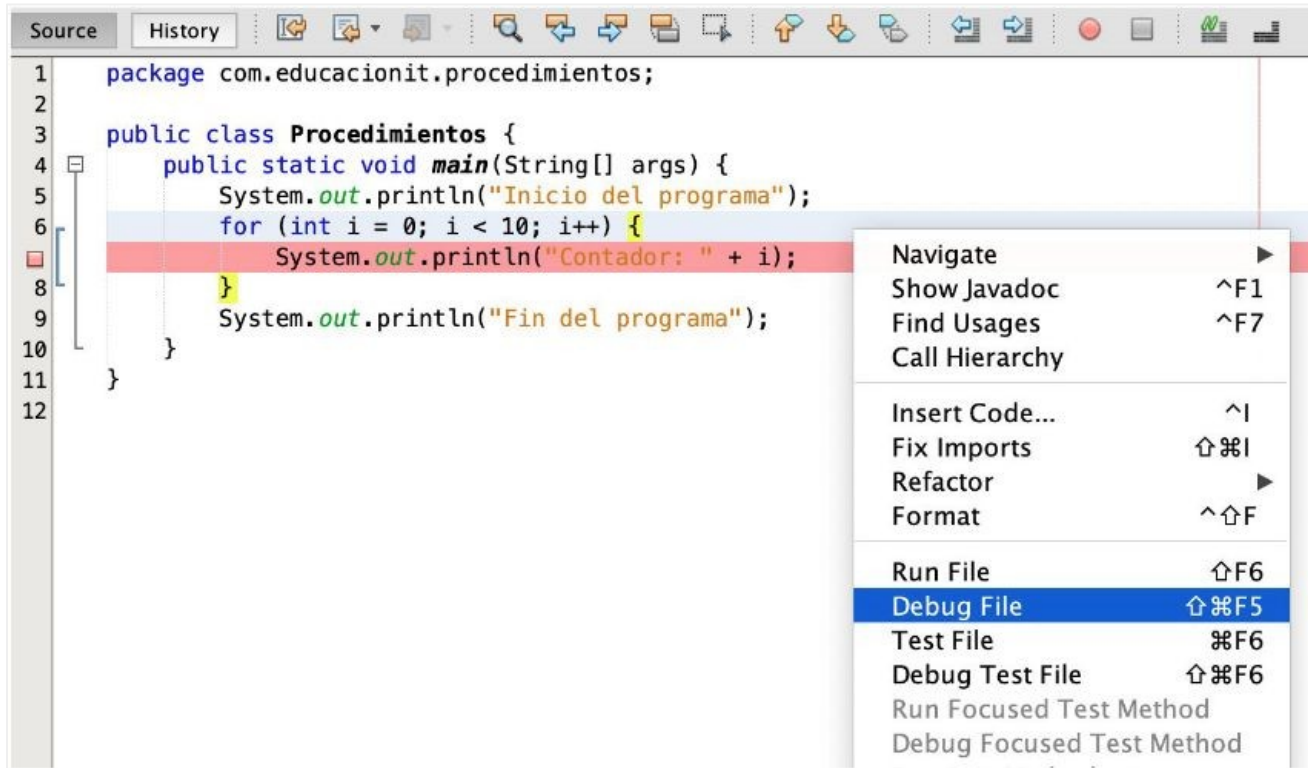
Paso 2: Modo Debug

Al apretar el botón derecho para desplegar el **menú de opciones**, como se muestra en la referencia de la siguiente diapositiva, vemos una que se llama “**Debug File**”, la cual sirve para ejecutar el programa en **modo debug** (o sea, frenando el programa cuando se llega a una línea que tenga un *breakpoint*).



Centro Formación Profesional Nro 35

Técnicas de programación, Lenguaje JAVA



Centro Formación Profesional Nro 35

Técnicas de programación, Lenguaje JAVA

Variables

Al ejecutar el programa en modo *debug*, veremos algunos tabs, por ej. “Variables”, con los cuales podremos ver el valor actual de cada variable.

Si este tab no estuviera presente podemos ir a **“Window → Debugging → Variables”** (esto varía entre los distintos IDE).



Output

Además, vemos un tab llamado “Output”, que nos va mostrando **las cosas que fue escribiendo nuestro programa a la fuente de salida**, en este caso la consola.

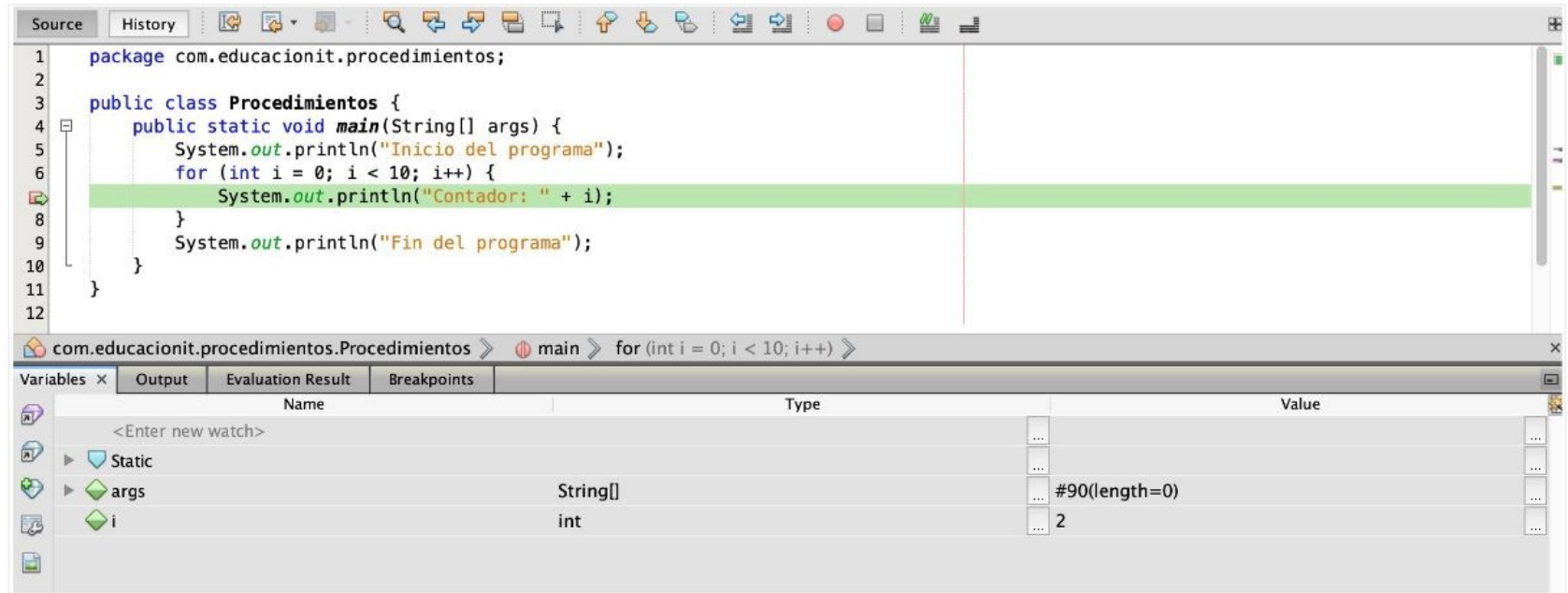
Si este tab no estuviera presente podemos ir a **“Window → Output”** (esto varía entre los distintos IDE).



Centro Formación Profesional Nro 35

Técnicas de programación, Lenguaje JAVA

Variables



The screenshot displays an IDE window with a Java source file. The code defines a package, a class, and a main method with a loop. The current execution point is at line 6, where the variable `i` is being incremented. Below the code editor, the 'Variables' tab is active, showing a table of variables in scope.

```
1 package com.educacionit.procedimientos;
2
3 public class Procedimientos {
4     public static void main(String[] args) {
5         System.out.println("Inicio del programa");
6         for (int i = 0; i < 10; i++) {
7             System.out.println("Contador: " + i);
8         }
9         System.out.println("Fin del programa");
10    }
11 }
12
```

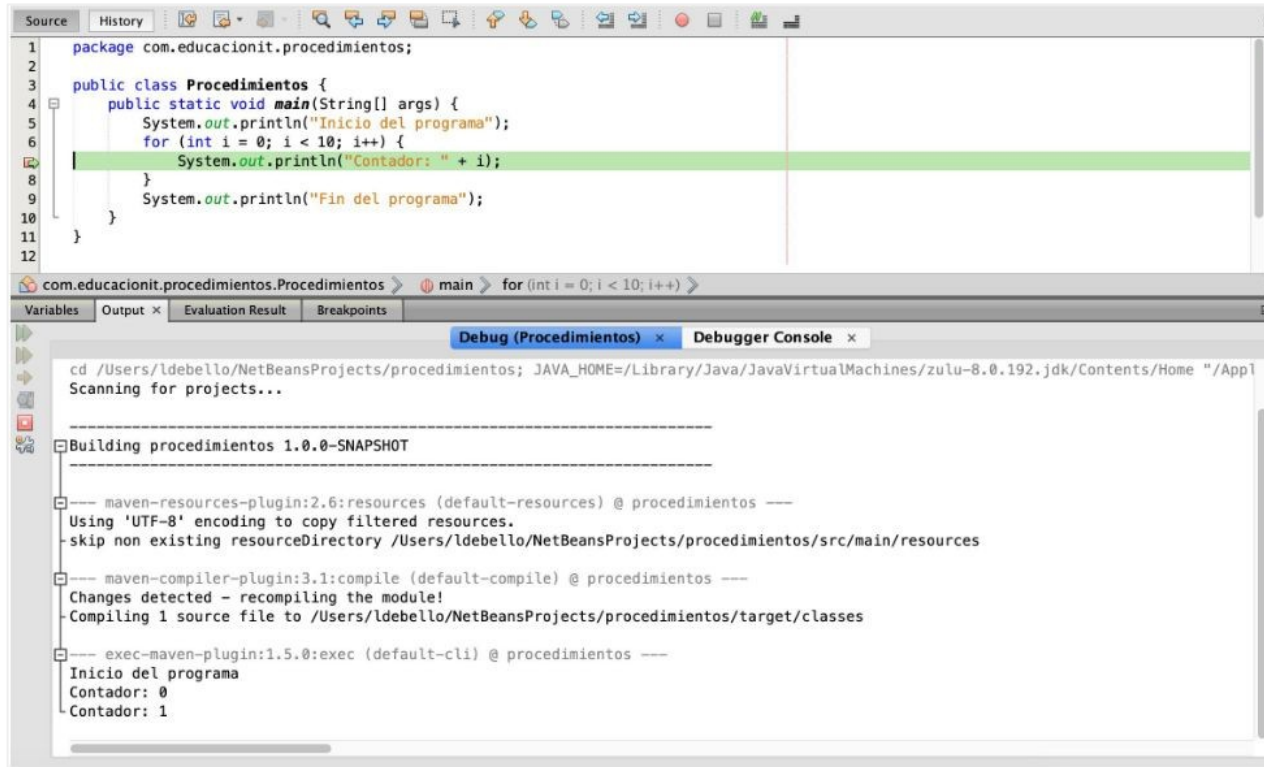
com.educacionit.procedimientos.Procedimientos > main > for (int i = 0; i < 10; i++) >

Name	Type	Value
<Enter new watch>		
Static		
args	String[]	#90(length=0)
i	int	2

Centro Formación Profesional Nro 35

Técnicas de programación, Lenguaje JAVA

Output



The screenshot displays an IDE window with a Java source file and its execution output. The source code defines a package `com.educacionit.procedimientos` and a class `Procedimientos` with a `main` method. The `main` method prints "Inicio del programa", then enters a loop from `i = 0` to `i < 10`, printing "Contador: " followed by the value of `i` for each iteration, and finally prints "Fin del programa".

The execution output, shown in the "Debugger Console" tab, details the build process and the program's execution. It shows the compilation of the source file and the resulting output:

```
cd /Users/ldebello/NetBeansProjects/procedimientos; JAVA_HOME=/Library/Java/JavaVirtualMachines/zulu-8.0.192.jdk/Contents/Home "/App1
Scanning for projects...

-----
Building procedimientos 1.0.0-SNAPSHOT
-----

--- maven-resources-plugin:2.6:resources (default-resources) @ procedimientos ---
Using 'UTF-8' encoding to copy filtered resources.
-skip non existing resourceDirectory /Users/ldebello/NetBeansProjects/procedimientos/src/main/resources

--- maven-compiler-plugin:3.1:compile (default-compile) @ procedimientos ---
Changes detected - recompiling the module!
-Compiling 1 source file to /Users/ldebello/NetBeansProjects/procedimientos/target/classes

--- exec-maven-plugin:1.5.0:exec (default-cli) @ procedimientos ---
Inicio del programa
Contador: 0
Contador: 1
```


Centro Formación Profesional Nro 35

Técnicas de programación, Lenguaje JAVA

Paso 3: Línea a línea

Depurar nos permite ejecutar nuestro programa **línea a línea**. Para ello, cada IDE utiliza algún botón o combinación de teclas para avanzar a la línea siguiente.

En NetBeans, esa tecla es **F8**. Podemos consultar esta tecla en “**Debug Step Over**”.

