

Apéndice: Depuración del código en Netbeans

Es muy probable que durante la ejecución de un programa aparezcan errores. Algunos errores de programación pueden ser evidentes y fáciles de solucionar, pero es posible que haya otros que no sepáis de dónde vienen. Para rastrear estos últimos, os recomendamos que utilicéis el depurador.

Depurar un programa consiste en analizar el código en busca de errores de programación (*bugs*). Los entornos de desarrollo suelen proporcionar facilidades para realizar dicha tarea. En el caso de **Netbeans** el **procedimiento de depuración** es muy sencillo:

Lo primero que debéis hacer es establecer un punto de control (*breakpoint*) en la sentencia del programa donde se desea que la ejecución se detenga para comenzar a depurar. Para ello, únicamente hay que hacer *clic* en el número de línea donde se encuentra dicha instrucción (*line breakpoint*). La línea, en el ejemplo la número 23 (figura 1), se resaltará en rosa.

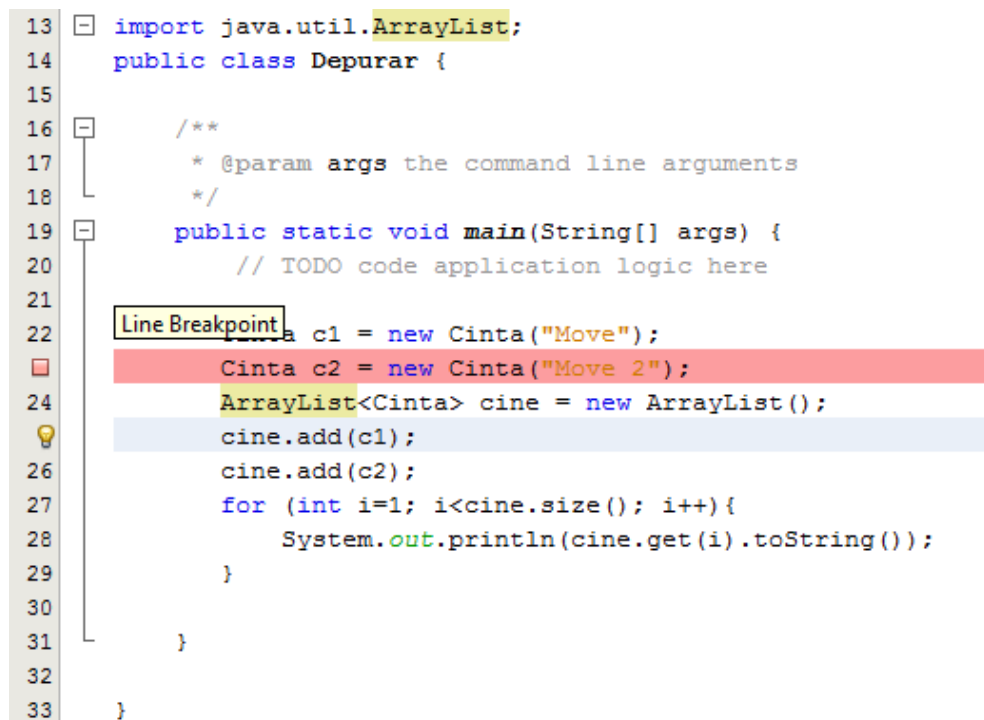


Figura 1: Line Breakpoint.

Después, pulsar **Control+F5** o la correspondiente opción del menú **Debug** para comenzar la depuración (figura 2).

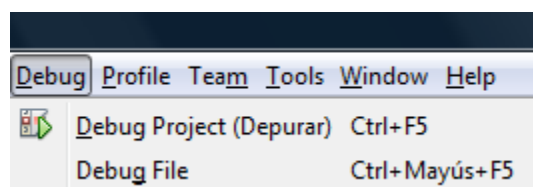


Figura 2: Menú Debug.

Una vez que el flujo de control del programa llegue a un *breakpoint*, la ejecución se pausará para que podáis seguirla y la línea de código correspondiente se coloreará en verde. Además, aparecerá una barra de herramientas de depuración (arriba en la figura 3, después del *play*) y dos paneles de depuración (abajo en la figura 3): variables y *breakpoints*.

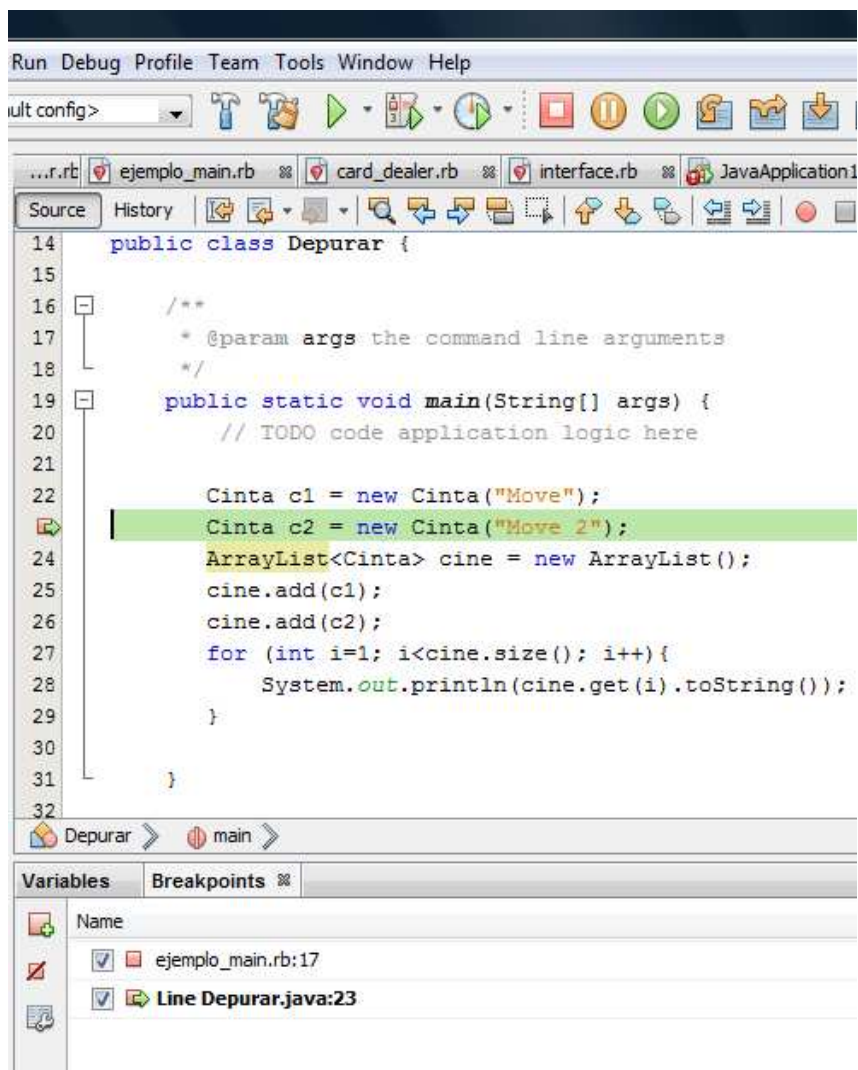


Figura 3: Herramientas de depuración.

Ahora se puede trabajar de dos modos diferentes: pulsando **F7** o **F8**.

- Si se pulsa F7, el control entrará en el método invocado en la instrucción actual (en verde). En nuestro ejemplo se ejecutaría el constructor de la clase Cinta, parándose en la primera línea de dicho método.
- Si se pulsa F8, el control salta a la siguiente instrucción del programa. En nuestro ejemplo se ejecutaría el constructor (línea 23 del código) y se pararía en la línea 24. Usaréis, por tanto, esta opción cuando estéis seguros de que el *bug* no está ni se deriva del método invocado.

Si se han definido varios *breakpoints* en el fichero o proyecto, podéis usar la opción **F5** que continuará ejecutando instrucciones hasta el siguiente *breakpoint*, donde se pausará de nuevo la ejecución. Si se pulsa F5 continuará la ejecución hasta el siguiente punto de control. No tiene sentido en nuestro ejemplo, pues solo hemos definido un *breakpoint*.

En cualquier momento podéis situar el cursor sobre una variable del programa y, si la variable está activa (en ámbito), obtendréis su valor. En el ejemplo (figura 4), la variable *i* tiene valor 1 en ese preciso instante de la ejecución.

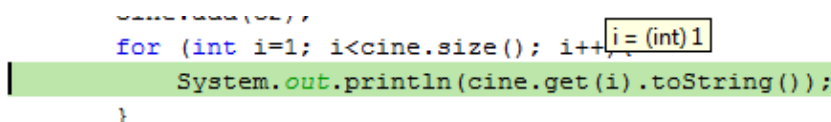


Figura 4: Ver el valor de una variable.

Adicionalmente, en el panel informativo de Variables (ver figura 3, abajo), podéis consultar el valor de cualquier variable activa en el contexto de ejecución actual. En el ejemplo (figura 5) es posible examinar que **i** tiene valor 1, y también el tipo de las variables, por ejemplo que **c1** es un objeto de la clase Cinta.

Name	Type	Value
m		>"m" is not a known variable in the current context.<
x		>"x" is not a known variable in the current context.<
<Enter new watch>		
Static		
args	String[]	#74(length=0)
c1	Cinta	#72
c2	Cinta	#75
cine	ArrayList<Cinta>	"size = 2"
i	int	1

Figura 5: Panel de Variables.

Si pulsamos el símbolo + que aparece junto a cada variable, aparecen más detalles sobre ésta (figura 6). Por ejemplo, para el *ArrayList* **cine** podemos conocer su tamaño (2) y cada uno de sus elementos (dos objetos de la clase Cinta). Y para un objeto de la clase Cinta podemos inspeccionar sus atributos (en este caso, nombre).

cine	ArrayList<Cinta>	"size = 2"
[0]	Cinta	#72
nombre	String	"Move"
[1]	Cinta	#75
nombre	String	"Move 2"

Figura 6: Detalles sobre la variable cine.

Si lo deseáis, podéis situaros sobre una variable y pulsando *New Watch* en el menú contextual que se despliega, podéis definir un centinela (*watch*) que permitirá escribir una expresión y consultar su valor en el contexto actual con dicha variable. Por ejemplo, en las figuras 7 y 8, una operación aritmética definida sobre el valor de **i**, o la llamada a un método del objeto **cine**. Los *watches* aparecen en el panel de variables, tal y como se observa en las figuras.

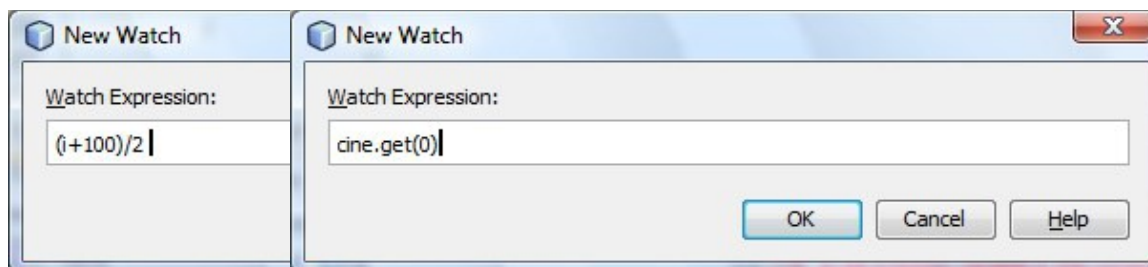


Figura 7: New Watch, expresión












Variables 		Breakpoints	
	Name	Type	Value
	<input checked="" type="checkbox"/>  x		 >"x" is not a known variable in the current context.<
	<input checked="" type="checkbox"/>  (i+100)/2	int	 50
	<input checked="" type="checkbox"/>  cine.get(0)	Cinta	 #72

Figura 8: New Watch. consulta

Finalmente, para detener la depuración y continuar con la ejecución normal usaremos Mayúscula+F5 o el botón correspondiente.