Pedro Vigara Haro

Tarea 2. Debugger

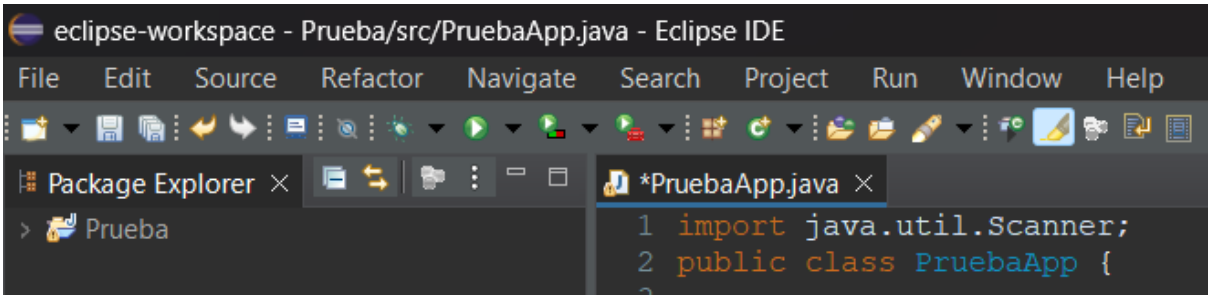
Depurar un programa (Debug) es una solución cuando no entendemos dónde está el error del programa que estamos creando, recorriendo paso por paso cada línea del código, viendo el valor de las variables del programa. Dependiendo del IDE que utilicemos, tendrá una manera de depurar u otra.

En **Eclipse**, primero deberemos crear el programa que queremos depurar. En mi caso, he creado un programa simple que pide el valor de dos números y realiza operaciones básicas con esos números.

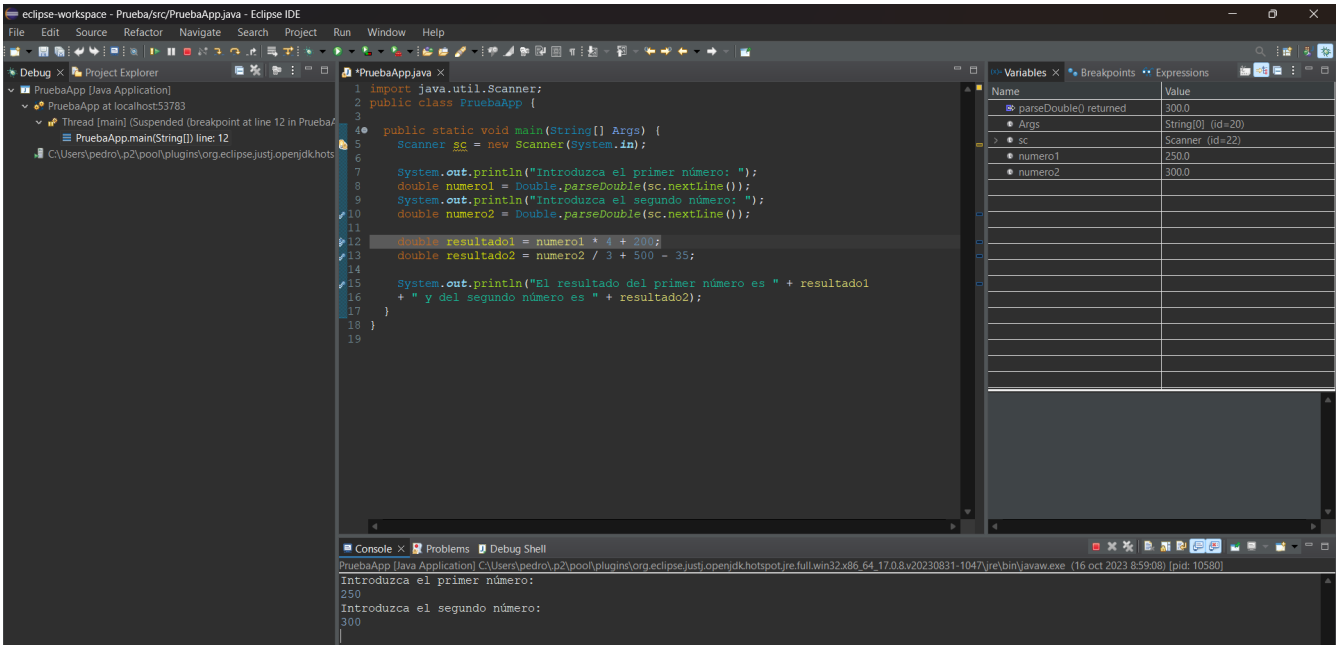
```
1  import java.util.Scanner;
2  public class PruebaApp {
3
4●   public static void main(String[] Args) {
5      Scanner sc = new Scanner(System.in);
6
7      System.out.println("Introduzca el primer número: ");
8      double numero1 = Double.parseDouble(sc.nextLine());
9      System.out.println("Introduzca el segundo número: ");
10     double numero2 = Double.parseDouble(sc.nextLine());
11
12     double resultado1 = numero1 * 4 + 200;
13     double resultado2 = numero2 / 3 + 500 - 35;
14
15     System.out.println("El resultado del primer número es " + resultado1
16     + " y del segundo número es " + resultado2);
17   }
18 }
```

Antes de depurar un programa, tenemos que establecer diferentes “breakpoints”, que irán deteniendo nuestra depuración a nuestro antojo. Para ello, tenemos que hacer doble click en los números que representan nuestra línea de código. Una vez hecho, nos aparecerán unos puntitos azules, como se puede ver en la imagen.

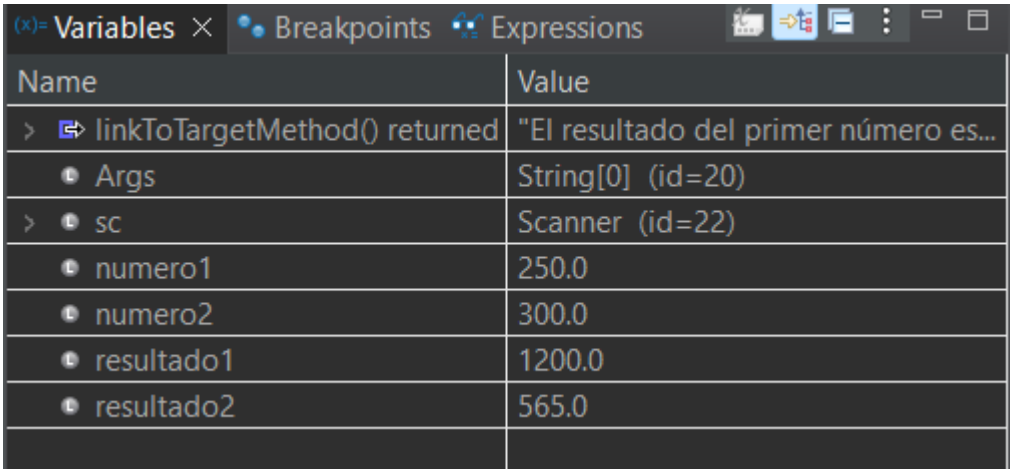
Para empezar a depurar el programa, tendremos que hacer click en el icono de un bicho en la parte superior de la interfaz de Eclipse.



Una vez hecho click, nos aparecerá un menú en la parte derecha de la pantalla. Es el menú debugger. Irá depurando nuestro programa línea por línea, parándose en los puntos que hemos establecido.



Una vez escritos los resultados para almacenarlos en las variables Numero1 y Numero2, nuestro programa se detendrá en el primer breakpoint establecido. En la parte superior derecha, podemos ver los valores de las variables. Para que nuestra depuración vaya avanzando a los siguientes puntos, debemos darle a la flecha amarilla cerca del botón donde hemos hecho click antes, llamado Step Over (También podemos darle a la tecla F6).



Aquí podremos ver que han aparecido los resultados de las variables que hemos escrito. Al ir paso por paso, antes no aparecían, pero como hemos ido avanzando de breakpoints, ya nos aparece con el resultado deseado.

Ya hemos terminado con nuestra depuración en Eclipse.

En **Visual Studio Code**, de igual manera, deberemos crear breakpoints en nuestro código, de la misma forma que hicimos en Eclipse.

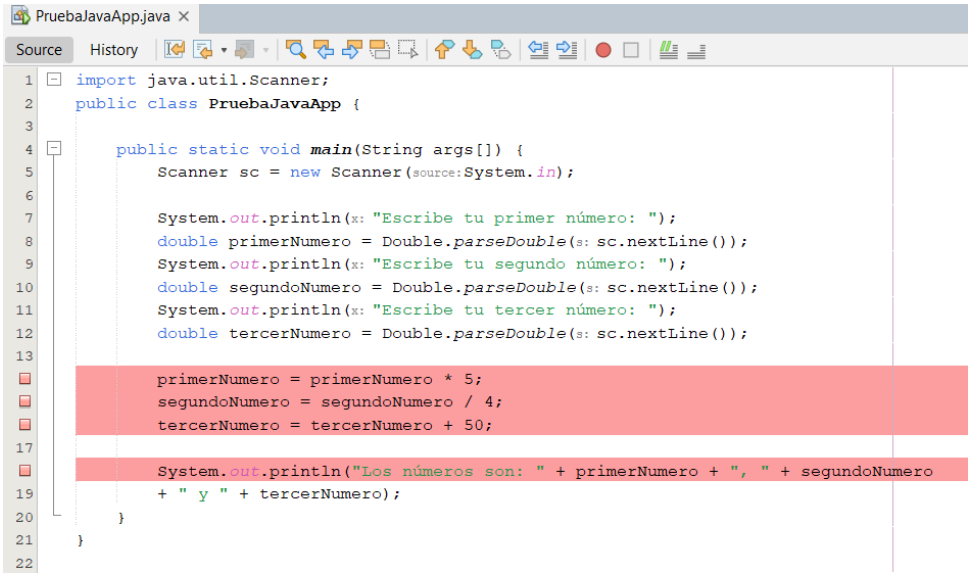
```
C: > Users > pedro > Downloads > PruebaApp.java > PruebaApp
1   import java.util.Scanner;
2   public class PruebaApp{
        Run | Debug
3       public static void main(String[] Args) {
4           Scanner sc = new Scanner(System.in);
5
6           System.out.println(x:"Escribe un número al azar: ");
7           double primerNumero = Double.parseDouble(sc.nextLine());
8           System.out.println(x:"Escribe otro número al azar: ");
9           double segundoNumero = Double.parseDouble(sc.nextLine());
10          System.out.println(x:"Escribe un último número al azar: ");
11          double tercerNumero = Double.parseDouble(sc.nextLine());
12
13          primerNumero = primerNumero * 2;
14          segundoNumero = segundoNumero + 50;
15          tercerNumero = tercerNumero / 2;
16
17          System.out.println(x:"Los números han quedado así");
18          System.out.print(primerNumero + ", " + segundoNumero + ", " + tercerNumero);
19      }
20  }
21
```

En este programa, creamos un código simple donde el usuario escribe tres números y el programa realiza 3 cálculos simples con esos números.

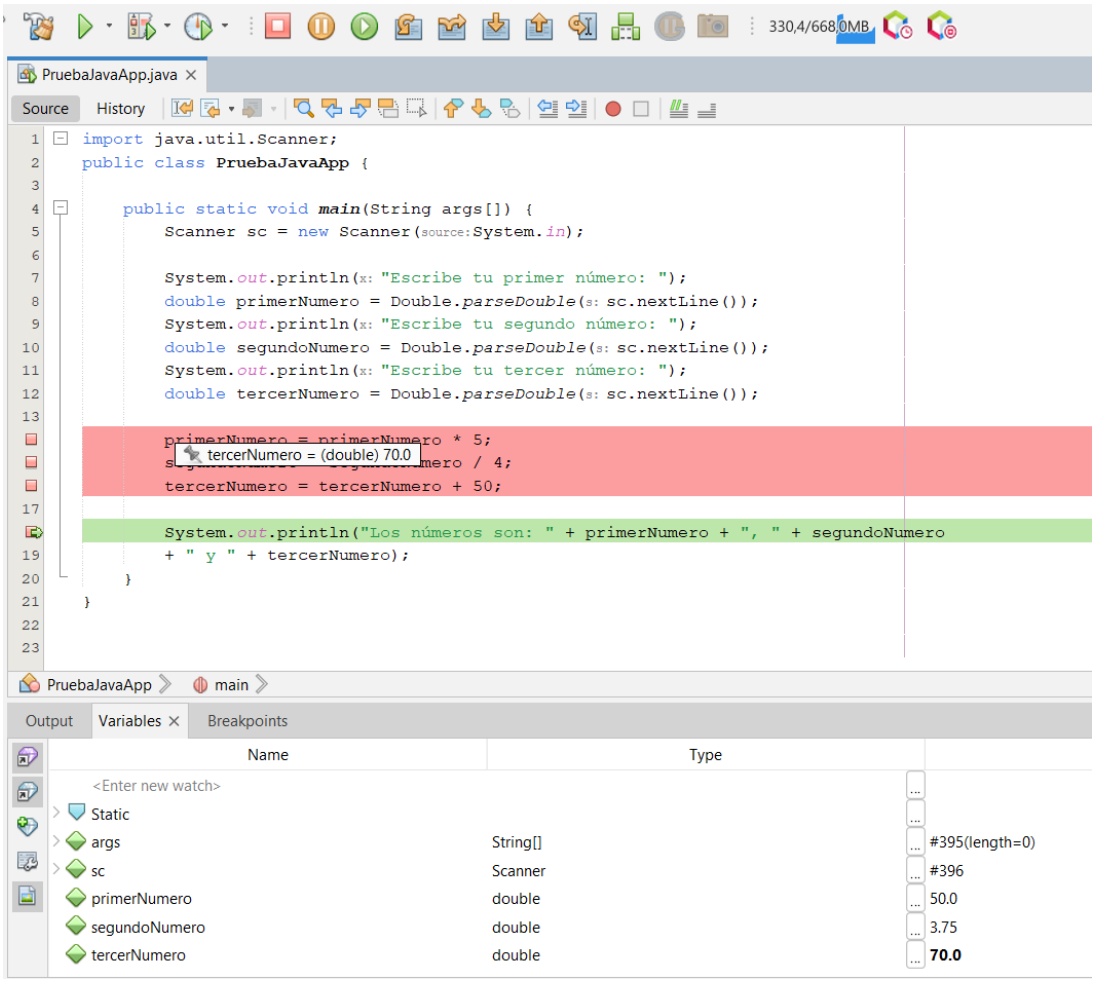
```
# styles.css   <> index.html   J PruebaApp.j
C: > Users > pedro > Downloads > PruebaApp.java > PruebaApp > main(String[])
1   import java.util.Scanner;
2   public class PruebaApp{
        Run | Debug
3       public static void main(String[] Args) { Args = String[0]@9
4           Scanner sc = new Scanner(System.in); sc = Scanner@10
5
6           System.out.println(x:"Escribe un número al azar: ");
7           double primerNumero = Double.parseDouble(sc.nextLine()); primerNumero = 20,000000, sc = Scanner@10
8           System.out.println(x:"Escribe otro número al azar: ");
9           double segundoNumero = Double.parseDouble(sc.nextLine()); segundoNumero = 65,000000, sc = Scanner@10
10          System.out.println(x:"Escribe un último número al azar: ");
11          double tercerNumero = Double.parseDouble(sc.nextLine()); tercerNumero = 10,000000, sc = Scanner@10
12
13          primerNumero = primerNumero * 2; primerNumero = 20,000000
14          segundoNumero = segundoNumero + 50; segundoNumero = 65,000000
15          tercerNumero = tercerNumero / 2; tercerNumero = 10,000000
16
17          System.out.println(x:"Los números han quedado así");
18          System.out.print(primerNumero + ", " + segundoNumero + ", " + tercerNumero);
19      }
20  }
21
```

Si le hacemos click a Run and Debug, podemos ver que nuestro programa empezará a ejecutarse y se parará en los breakpoints que hemos establecido. Con los botones de arriba, decidiremos avanzar de breakpoint o cualquier otra opción que deseemos. De esta manera, podremos hacer debug en Visual Studio Code.

En **NetBeans** deberemos crear nuestro programa en Java. De la misma manera que hemos hecho en Eclipse y Visual Studio, deberemos crear nuestros breakpoints para ir parando nuestro programa según deseemos.



Este programa tiene el mismo funcionamiento que el programa de Visual Studio Code. El programa pide 3 números por teclado y realiza 3 operaciones básicas con esos números. Haremos Ctrl + Shift + F5 para entrar en el menú de Debug.



Una vez en este menú, podemos ir avanzando por cada breakpoint con los botones de arriba. Las variables nos aparecerán abajo del todo, donde podremos ir viendo cómo van cambiando dependiendo de las operaciones que les realicemos. De esta manera, podremos hacer Debug en NetBeans.