# Swing

## *Introducción*

En este laboratorio, utilizaremos una interfaz gráfica que permita realizar una entrada y salida de datos. Para ello usaremos el plugin Visual Editor de Eclipse. También realizaremos un primer contacto con el depurador de Eclipse.

## *Objetivos*

Los objetivos de este laboratorio son:

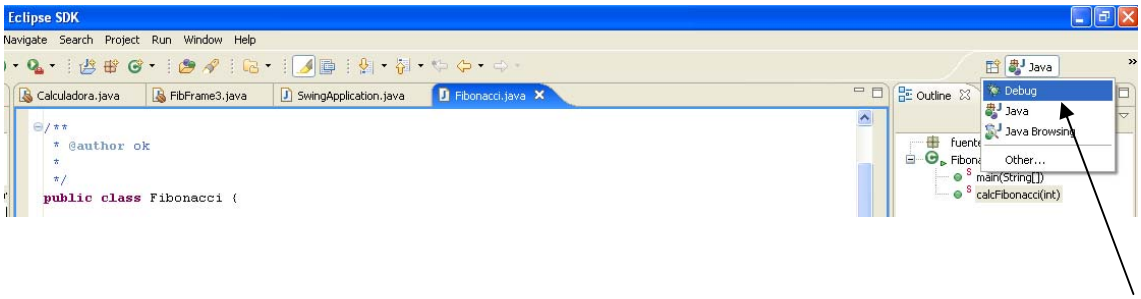
- Utilizar el Debugger
- Desarrollar interfaces gráficos utilizando el plugin Visual Editor de Eclipse.
- Utilizar algunas clases de la librería Swing
- Manejar una excepción.

## *Pasos a seguir*

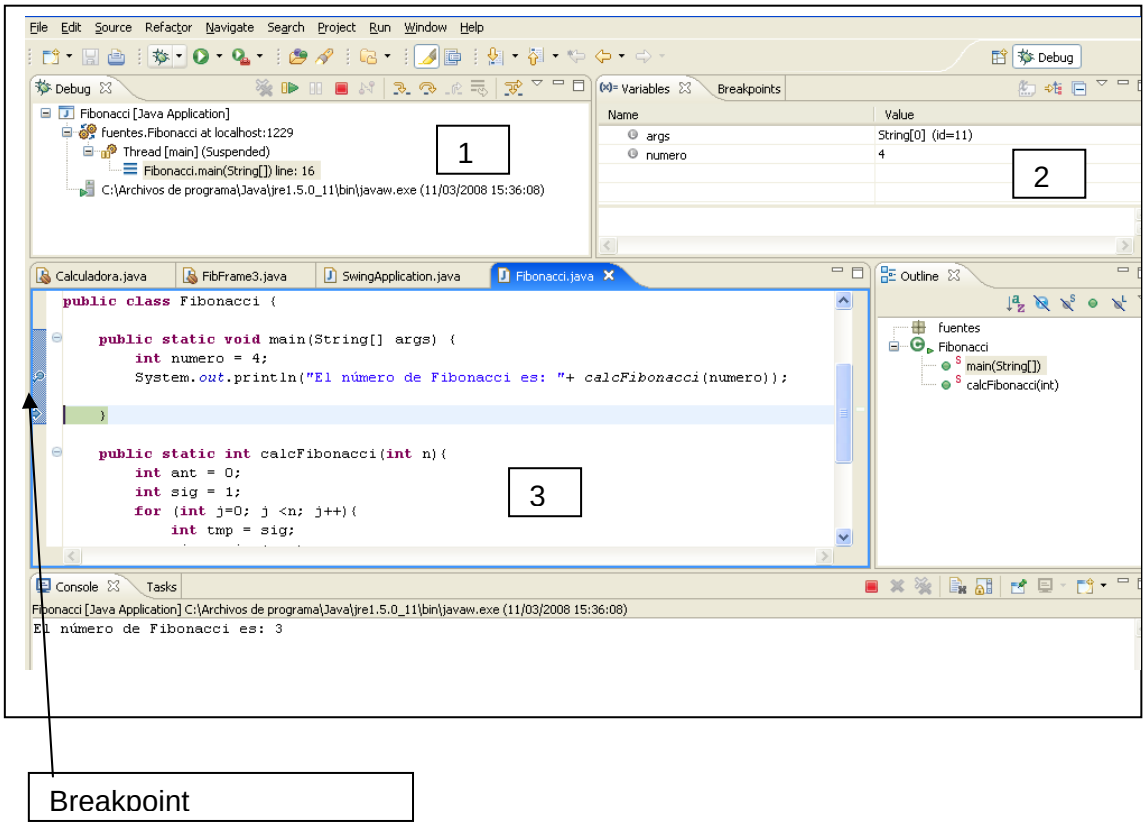
### 1. Debug

En este ejercicio veremos cómo se puede ejecutar un programa en modo “debug”. Esta opción es muy interesante a la hora de encontrar errores. Como ejemplo utilizaremos una clase sencilla como la de Fibonacci.

El entorno Eclipse ofrece un módulo para seguir la ejecución de programas. Lo primero que hay que hacer es abrir dicho módulo. Para ello se escoge la opción: Window->Open Perspective->Debug, tal y como aparece en la siguiente figura:



Automáticamente, se abrirán nuevas ventanas en el entorno de Eclipse. En la siguiente imagen se muestra una imagen después de ejecutarse.




El control de la ejecución del programa se hace en la ventana 1. Por ejemplo, se puede terminar la ejecución del programa, pasar a la siguiente instrucción o entrar en la ejecución de un método. En la ventana 2, se muestran los valores de las variables del programa durante la ejecución. Esta ventana es muy interesante, ya que ofrece la posibilidad de ver que valores tienen las variables en un determinado punto. En la ventana 3, a la izquierda, podemos escoger puntos de control del programa (breakpoints), los cuales son puntos del programa donde se puede parar la ejecución. Al hacer click con el ratón, se añaden o borran los puntos de control.

**Ejercicio:**

1. Crear la siguiente clase Fibonacci:

```
public class Fibonacci {
     public  int calcFibonacci(int n){
        int prev = 0;
        int next = 1;
        for (int j=0; j <n; j++){
            int tmp = next;
            next = next + prev;
            prev = tmp;
        }
        return prev;
    }
}
```

2. Poner un punto de control (Breakpoint) en la instrucción ipini "next=next+prev

3. Crerar una nueva clase con el programa principal, en la cual se calcule el número fibonacci de 4. Ejecutadla viendo los valores que toman las variables en cada paso.

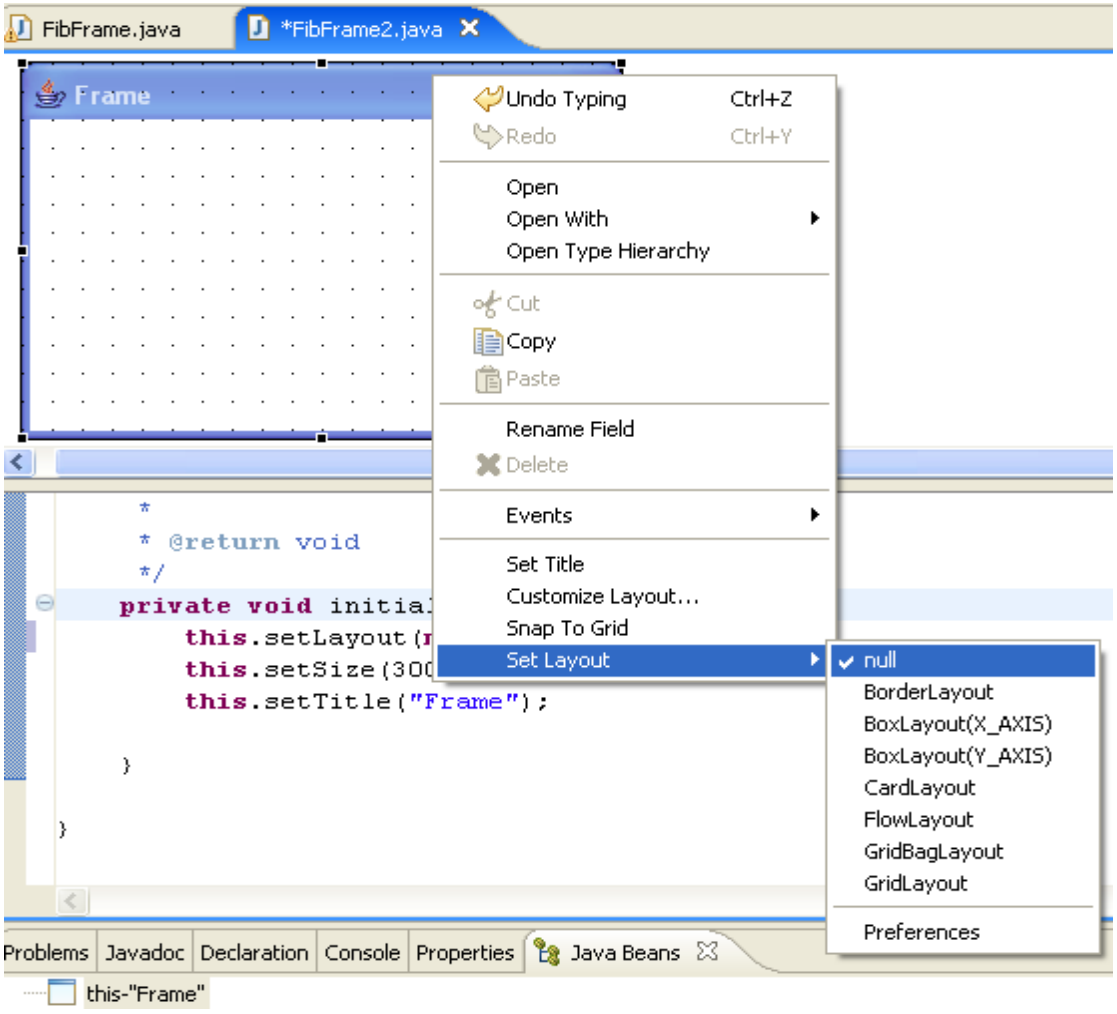
## 2. Swing

### Instalación de las librerías

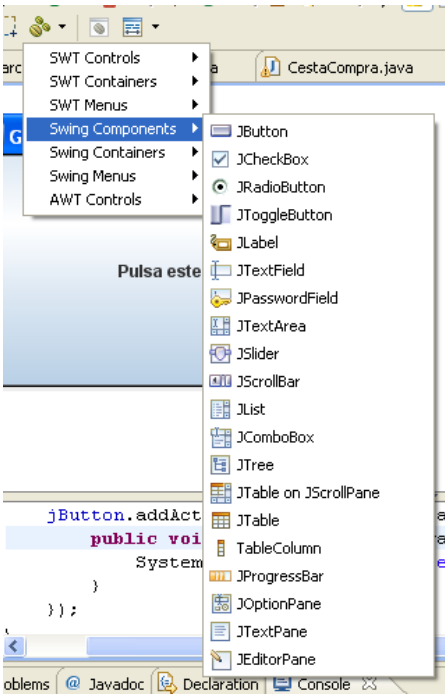
Para crear la interfaz de usuario gráfica vamos a utilizar la librería de Java Swing/AWT. Y para que el desarrollo sea más fácil haremos uso del plugin de Eclipse para trabajar con dicha librería. En el anexo 1 podéis encontrar cómo instalarlo.

### Diseñar la interfaz gráfica

Primero crearemos una ventana, seleccionando File->New>"Visual Class" y escogiendo una clase del tipo Swing/JFrame. A continuación quitaremos el administrador de diseño de la siguiente manera:

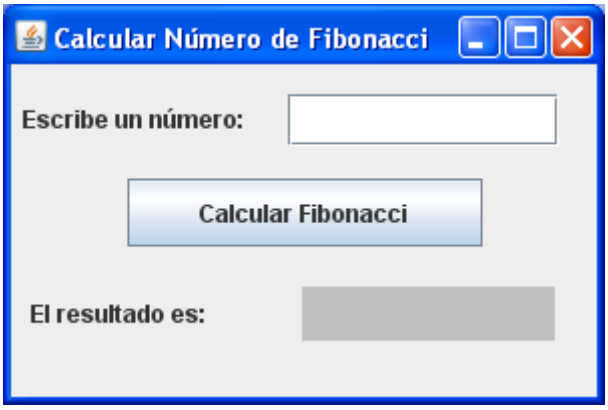



Para poner componentes en el frame, los seleccionaremos de la paleta de componentes (ver siguiente figura) y lo colocaremos en la posición del frame deseada. Para poner el texto asociado al componente, se puede hacer posicionándonos encima del mismo y haciendo click derecho y seleccionando “Set Text”.




Cread el siguiente frame con las etiquetas, botón, caja de texto y área de texto (coloreada con gris, lo cual se consigue añadiendo la siguiente instrucción:

jTextArea.setBackground(Color.lightGray);




Hay que añadir el código que debe ejecutarse cuando se pulse el botón. Para ello hay que posicionarse primero en el botón, seleccionar la opción “Events” tras hacer click derecho y escoger el evento. Aunque el evento apropiado tendría que ser “actionPerformed” (para programar la acción “pulsar el botón”), en este caso escoged el evento mouseClicked (se hace en “add events”). Poned también el siguiente código como respuesta a dicho evento (tened cuidado con los nombres de las variables de los componentes y de las clases).

```
int numero = Integer.parseInt(textField1.getText());

textField2.setText(Integer.toString(Fibonacci.calcFibonacci(numero)));
```

**Guardad los ficheros**: Es conveniente que guardéis los ficheros para no perder el trabajo.

**Definir la clase principal**

En esta clase crearemos y ejecutaremos un objeto de la interfaz. Definid una clase Principal con el siguiente método main:

```
public static void main(String[] args) {
      // TODO Auto-generated method stub
      FibFrame fibFrame = new FibFrame();
      Fibonacci fib=new Fibonacci();
      fibFrame.setFibonacci(fib);
      fibFrame.setVisible(true);}
```

**Ejecutar la aplicación.**

Puede suceder que no siempre se calcule el número de Fibonacci tras hacer click en el botón (PROBAD A ARRASTRAR EL RATÓN SOBRE EL BOTÓN, ESTO ES, HACER "DRAG" EN VEZ DE "CLICK"). El problema es que EL EVENTO APROPIADO NO ES "mouseClicked" SINO "actionPerformed", que responde a cualquier evento sobre el botón: hacer "clic" en el mismo, arrastrar el ratón dentro del botón (hacer "drag"), etc. Definir el comportamiento anterior, pero esta vez para el evento "actionPerformed".

¿Qué sucede si se mete un carácter no numérico en el "textField1". Comprobad que se lanza una excepción. Para arreglar el problema hay que cambiar el código, y poner las instrucciones que pueden levantar excepciones dentro de un "try{...}", y las instrucciones de atención a la excepción dentro de un bloque "catch". En este caso, habría que añadir código con este esquema:

```
try{...}

catch(NumberFormatException ex){..}
```

**Referencias:**

[1] http://www.eclipse.org/vep/WebContent/main.php (Visual Editor for Eclipse 3.3)

[2] http://java.sun.com/docs/books/tutorial/information/download.html

[3]http://help.eclipse.org/help32/index.jsp?topic=/org.eclipse.ve.doc/topics/cve_overview.html

[4]:http://www.apl.jhu.edu/~hall/java/Swing-Tutorial/ Tutorial Swing

[5]: http://www.programacion.com/java/tutorial/swing/5/

# Anexo 1: Instalación de Visual Editor Project

**Paso1: Descargar el plugin:**

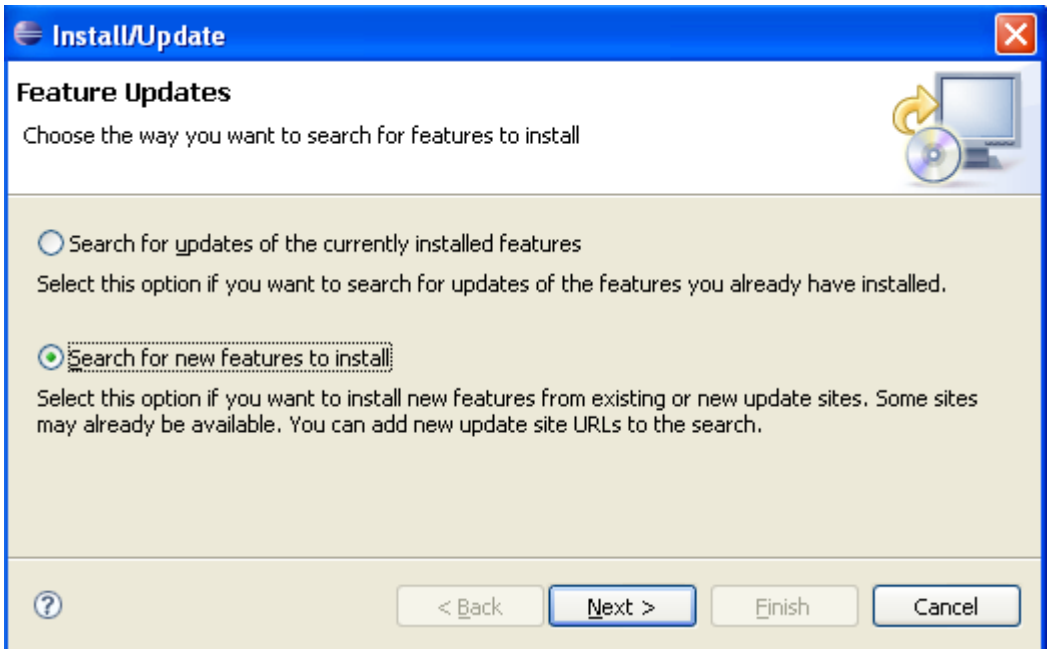
En la página http://www.eclipse.org/vep/WebContent/main.php encontramos la información relativa al proyecto **Visual Editor Project**. Iremos al enlace 'download page' y pincharemos en el enlace de la última versión. En la página que aparece descargaremos el SDK, paquete para desarrolladores que contiene los fuentes y documentación. Al inicio de la página nos advierten mediante una nota que es necesario tener los siguientes programas y versiones:

- Eclipse build eclipse-SDK-3.2: (download win32 zip)
- EMF build 2.2.0: (download zip)
- GEF Build 3.2: (download zip)

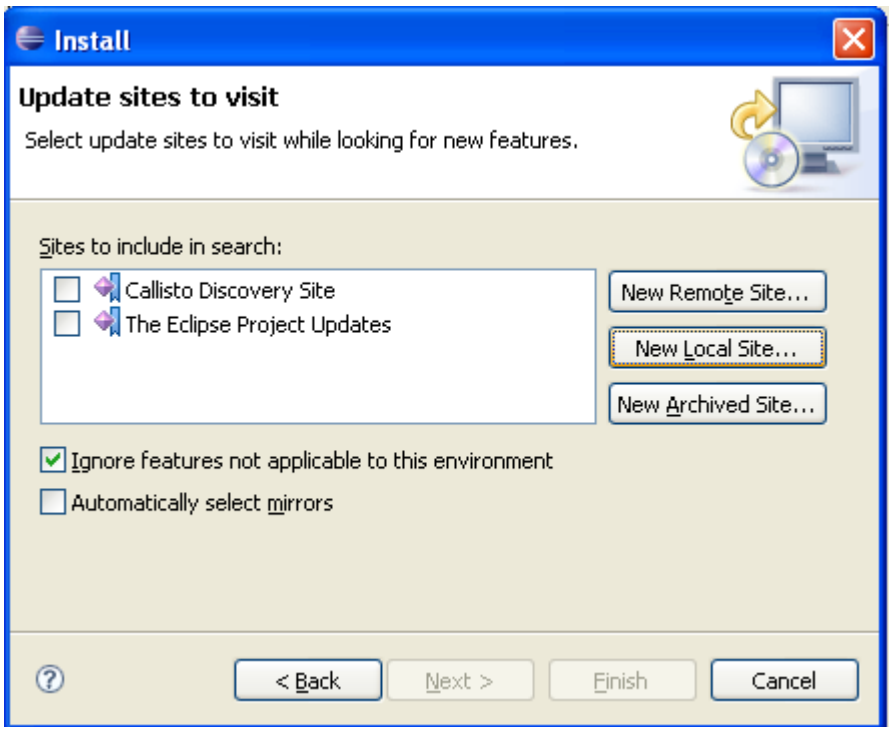
Por lo que si no las tenemos deberemos descargarlas también.

**Paso 2: Instalar el plugin**

- Copiamos los archivos descargados en C:\Archivos de programa\eclipse3.2\plugins. Cada vez que se descomprima un archivo le podemos dar a la carpeta el mismo nombre que el fichero .zip.
- Abrimos Eclipse.
- vamos a Help / Software Updates / Find and Install...
- Seleccionamos la opción 'Search for new features to install' como se mestra en la siguiente imagen y pulsamos en Next.

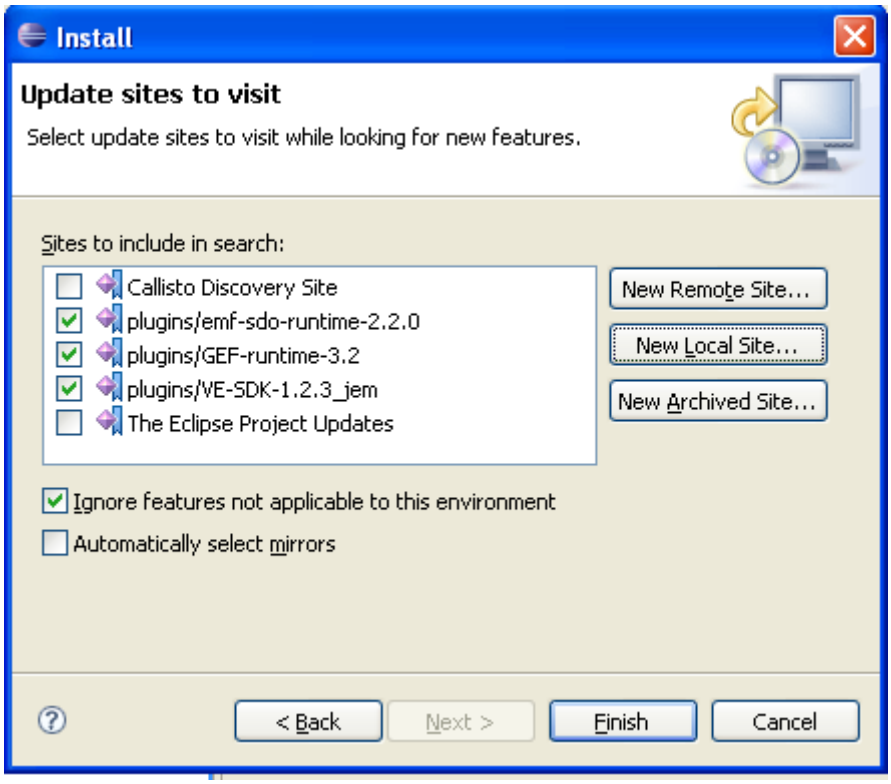


En la siguiente pantalla seleccionaremos la opción 'New Local Site…'




Y seleccionaremos las carpetas que hemos descomprimido.

Después de seleccionar las tres carpetas descargadas y descomprimidas la pantalla será:



Seleccionamos ‘Finish’.

Busca los componentes a instalar y si todo ha ido bien y no hay incompatibilidad de versiones nos los muestra para seleccionar. Una vez seleccionados pulsamos ‘Next’ y nos aparece una nueva pantalla en la que tendremos que tendremos que aceptar las condiciones y términos de uso.

Por último, pulsamos ‘Finish’. Una vez instalados los plugins nos pedirá reiniciar el Eclipse. Le decimos que sí.