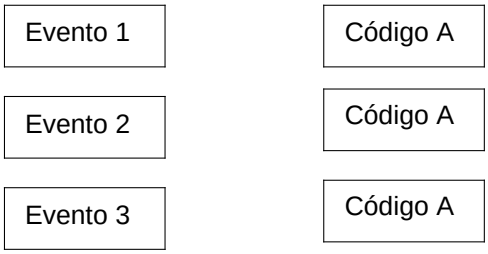
EJERCICIO GUIADO. JAVA: CENTRALIZAR CÓDIGO

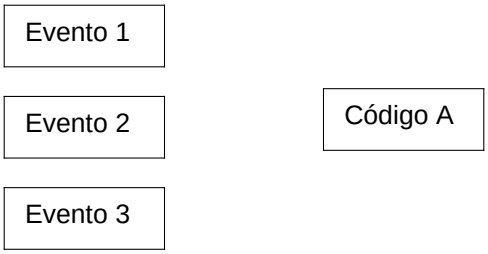
El problema de la repetición de código

Es muy habitual en Java que varios eventos tengan que ejecutar el mismo código. En este caso se plantea la necesidad de "copiar y pegar" ese código en los distintos eventos a programar:



Esta es una mala forma de programación, ya que se necesitara modificar el código, sería necesario realizar la modificación en cada copia del código. Es muy fácil que haya olvidos y aparezcan errores en el programa que luego son muy difíciles de localizar.

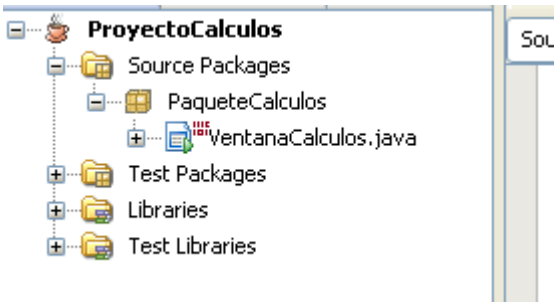
Lo mejor es que el código que tenga que ser ejecutado desde distintos eventos aparezca solo una vez, y sea llamado desde cada evento:



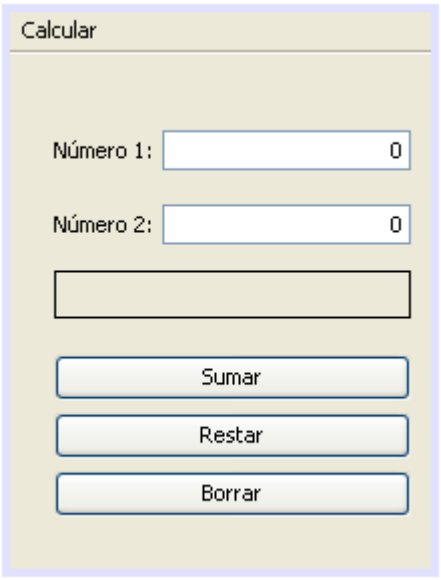
Veamos algunos ejemplos en los que el código se puede repetir y como evitar esta repetición.

Ejercicio guiado 1

- Crea un nuevo proyecto en java que se llame *ProyectoCalculos*. Este proyecto tendrá un paquete llamado *PaqueteCalculos*. Y dentro de él creará un JFrame llamado *VentanaCalculos*. El proyecto tendrá el siguiente aspecto:



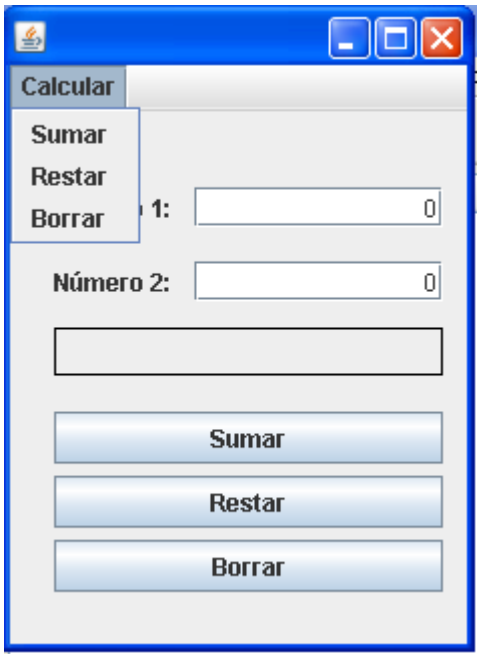
- La *VentanaCalculos* debe estar diseñada de la siguiente forma:



Esta ventana contiene los siguientes elementos:

- Una barra de menús a la que puede llamar *menuBarra*.
- La barra de menús contiene un JMenu con el texto “Calcular” y que se puede llamar *menuCalcular*
- El *menuCalcular* contendrá tres JMenuItem, llamados respectivamente: *menuSumar, menuRestar, menuBorrar* y con los textos *“Sumar”, “Restar” y “Borrar”*.
- Una etiqueta con el texto “Número 1”. (no importa su nombre)
- Una etiqueta con el texto “Número 2”. (no importa su nombre)
- Un cuadro de texto con un 0 y con el nombre *txtNumero1*.
- Un cuadro de texto con un 0 y con el nombre *txtNumero2*.
- Una etiqueta con el nombre *etiResultado*.
- Un botón “Sumar” con el nombre *btnSumar*.
- Un botón “Restar” con el nombre *btnRestar*.
- Un botón “Borrar” con el nombre *btnBorrar*.

- Aquí puedes ver la ventana en ejecución con el menú “Calcular” desplegado:



- El objetivo de programa es el siguiente:
    - El usuario introducirá dos números en los cuadros de texto.
    - Si pulsa el botón Sumar, se calculará la suma.
    - Si pulsa el botón Restar, se calculará la resta.
    - Si pulsa el botón Borrar, se borrarán ambos cuadros de texto.
    - Si elige la opción del menú Calcular-Sumar entonces se calculará la suma.
    - Si elige la opción del menú Calcular-Restar entonces se calculará la resta.
    - Si elige la opción del menú Calcular-Borrar entonces se borrarán ambos cuadros de texto.
    - Si se pulsa enter en alguno de los dos cuadros de texto se debería calcular la suma.

- Este es un ejemplo en el que al activarse uno de varios eventos distintos se tiene que ejecutar el mismo código. Observa el caso de la suma:

Pulsar Botón Sumar

Activar Calcular – Sumar en el menú

Pulsar enter en el primer cuadro de texto

Pulsar enter en el segundo cuadro de texto

Calcular la suma y mostrarla en la etiqueta de resultado

- Para que el código esté “centralizado”, es decir, que aparezca solo una vez, será necesario construir en la clase un *método*. Un *método* en java es el equivalente de una función o procedimiento en C. Veamos como hacerlo:

- Accede al código de tu programa a través del botón *Origen*.

- Un buen sitio para programar tus procedimientos puede ser debajo del constructor. Puedes distinguir fácilmente al constructor porque tiene el mismo nombre que la clase que estás programando, o dicho de otro modo, tiene el mismo nombre que la ventana que estás programando: *VentanaCalculos*.

```
 */
public class VentanaCalculos extends javax.swing.JFrame {

    /**
     * Creates new form VentanaCalculos
     */
    public VentanaCalculos() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this m
```

Este es el constructor

Este es un buen sitio para crear tus propios procedimientos

- Se va a programar un procedimiento que se encargue de recoger los valores de los cuadros de texto. Calculará la suma de dichos valores, y luego mostrará la suma en la etiqueta de resultados.

  Los procedimientos en java tienen prácticamente la misma estructura que en C. Programe lo siguiente:

```
public class VentanaCalculos extends javax.swing.JFrame {

    /**
     * Creates new form VentanaCalculos
     */
    public VentanaCalculos() {
        initComponents();
    }

    void Sumar() {
        String cad1, cad2;
        int a,b,s;

        cad1 = txtNumero1.getText();
        cad2 = txtNumero2.getText();
        a = Integer.parseInt(cad1);
        b = Integer.parseInt(cad2);
        s=a+b;
        etiResultado.setText(""+s);
    }

    /** This method is called from within the constructor to
```

Este es el procedimiento que tienes que introducir en el programa.

- Si observas el código, es el típico procedimiento de C, cuya cabecera comienza con *void* y el nombre que le hayas asignado (en nuestro caso *Sumar*)

```
void Sumar() {
      ....
}
```

  Si estudias las líneas del código, verás que lo que hace es recoger el contenido de los dos cuadros de texto en dos variables de cadena llamadas *cad1 y cad2*.

  Luego convierte dichas cadenas en números que almacena en dos variables enteras llamadas a y b.

  Finalmente calcula la suma en una variable s y presenta el resultado en la etiqueta etiResultado.

- Hay que destacar que este código no pertenece ahora mismo a ningún evento en concreto, por lo que no tiene efecto ninguno sobre el programa. Será necesario pues asociar los eventos correspondientes con este procedimiento.

- Interesa que al pulsar el botón “Sumar” se ejecute la suma, así pues entre en el evento *actionPerformed* del botón “Sumar” y añada la siguiente línea:

```
Sumar();
```

- Como también interesa que al pulsar la opción del menú “Calcular-Sumar” se ejecute la suma, entre en el evento *actionPerformed* de la opción del menú “Sumar” y añade de nuevo la siguiente línea:

```
Sumar();
```

- También se quiere que al pulsar la tecla enter en el cuadro de texto del número 1 se ejecute la suma. Por lo tanto, en el evento *actionPerformed* del cuadro de texto txtNumero1 hay que añadir la siguiente línea:

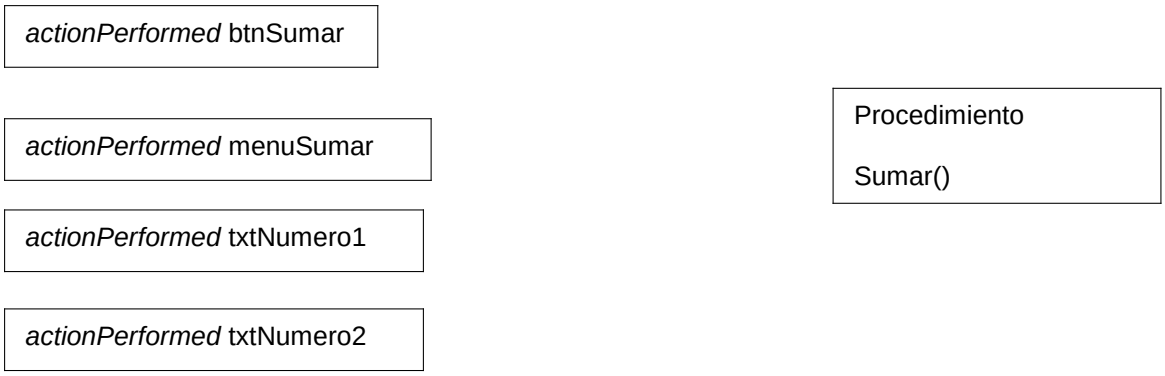
```
Sumar();
```

- Y como también se quiere que al pulsar la tecla enter en el cuadro de texto del número 2 se ejecute la suma, también habrá que introducir en su *actionPerformed* la siguiente línea:

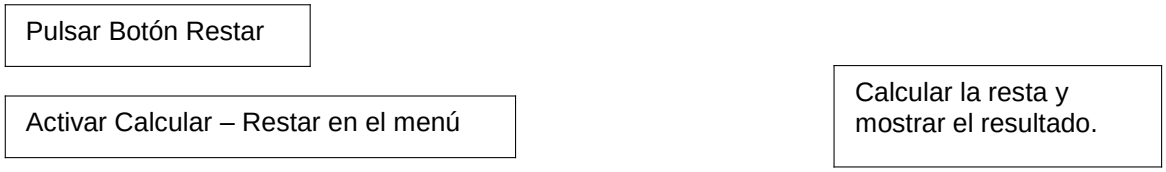
```
Sumar();
```

- Antes de continuar, ejecute el programa, introduzca dos números, y compruebe como se calcula la suma al pulsar el botón Sumar, o al activar la opción del menú Calcular–Sumar, o al pulsar Enter en el primer cuadro de texto, o al pulsar Enter en el segundo cuadro de texto.

  En cada uno de los eventos hay una llamada al procedimiento *Sumar*, que es el que se encarga de realizar la suma.



- En el caso de la resta sucede igual. Tenemos que varios eventos distintos deben provocar que se realice una misma operación. En este caso tenemos lo siguiente:



- Para centralizar el código, crearemos un método *Restar* que se encargará de hacer la resta de los números introducidos en los cuadros de texto. Este método se puede colocar debajo del anterior método *Sumar*:

```
    void Sumar() {
        String cad1, cad2;
        int a,b,s;

        cad1 = txtNumero1.getText();
        cad2 = txtNumero2.getText();
        a = Integer.parseInt(cad1);
        b = Integer.parseInt(cad2);
        s=a+b;
        etiResultado.setText(""+s);
    }

    void Restar() {
        String cad1, cad2;
        int a,b,r;

        cad1 = txtNumero1.getText();
        cad2 = txtNumero2.getText();
        a = Integer.parseInt(cad1);
        b = Integer.parseInt(cad2);
        r=a-b;
        etiResultado.setText(""+r);
    }
```

Programa este procedimiento.

- El código de este procedimiento es prácticamente idéntico al del procedimiento Sumar, así que no se comentará.

- Ahora, es necesario que cuando se activen los eventos indicados antes, estos hagan una llamada al procedimiento Restar para que se efectúe la resta. Así pues, entre en el evento *actionPerformed* del botón “Restar” y añada esta línea de código:

```
Restar();
```

- Igualmente, entre en el evento *actionPerformed* de la opción del menú “Calcular – Restar” y añada la misma llamada:

```
Restar();
```

- Ejecute el programa y compruebe como funciona el cálculo de la resta, da igual que lo haga pulsando el botón “Restar” o la opción del menú “Restar”. Ambos eventos llaman al mismo método:

actionPerformed btnRestar

actionPerformed menuRestar

Procedimiento

Restar()

- Finalmente se programará el borrado de los cuadros de texto a través del botón “Borrar” y de la opción del menú “Borrar”. En primer lugar, programa el siguiente método (puedes hacerlo debajo del método “Restar”):

```
}

void Restar() {
    String cad1, cad2;
    int a,b,r;

    cad1 = txtNumero1.getText();
    cad2 = txtNumero2.getText();
    a = Integer.parseInt(cad1);
    b = Integer.parseInt(cad2);
    r=a-b;
    etiResultado.setText(""+r);
}

void Borrar() {
    txtNumero1.setText("");
    txtNumero2.setText("");
}
```

Programa el procedimiento Borrar...

- Ahora programa las llamadas al procedimiento borrar desde los distintos eventos. En el evento *actionPerformed* del botón “Borrar” y en el evento *actionPerformed* de la opción del menú “Borrar” programa la siguiente llamada:

```
Borrar();
```

- Ejecuta el programa y prueba su funcionamiento.

CONCLUSIÓN

En java se pueden programar procedimientos al igual que en C. Normalmente, estos procedimientos se programarán debajo del constructor, y tienen la misma estructura que en C.

Se puede llamar a un mismo procedimiento desde distintos eventos, evitando así la repetición de código.