Índice

1 Funcionalidades Java que se presentan en esta práctica

Los dos conceptos básicos que se manejan en esta práctica son:

- La creación de una aplicación gráfica en Java: aplicación gráfica en el sentido de que no es una aplicación de consola, sino que tiene una interfaz gráfica con la que el usuario interacciona
- El manejo de eventos de Java

2 Una aplicación gráfica en Java

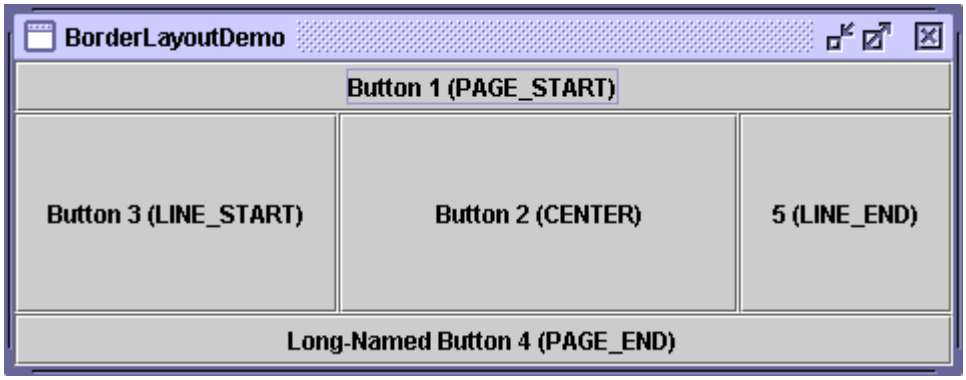
Los elementos de una Interfaz Gráfica de Usuario (en adelante, IGU) que vamos a manejar en esta práctica son:

- Los componentes gráficos que nos sirven para crear las ventanas, botones, etiquetas, etc. típicos de una aplicación gráfica; estos componentes son clases englobadas en el paquete javax.swing

- Los componentes de menú: JMenuBar, JMenu y JMenuItem, clases englobadas también en el paquete javax.swing

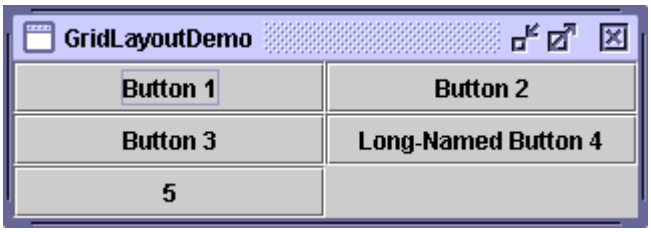
- Los Gestores de Disposición (Layout Managers):
  La manera de construir interfaces gráficas en Java es creando contenedores (generalmente objetos de la clase JPanel) que contienen componentes. La manera en que los componentes se distribuyen en el interior del contenedor se controla mediante estos Gestores de Disposición. Algunos de los que vamos a utilizar en esta práctica son:

  BorderLayout - Se pueden especificar 5 ubicaciones diferentes para los componentes, que se especifican con las constantes cuyo nombre aparece entre paréntesis (BorderLayout.PAGE_START, BorderLayout.CENTER, etc.)

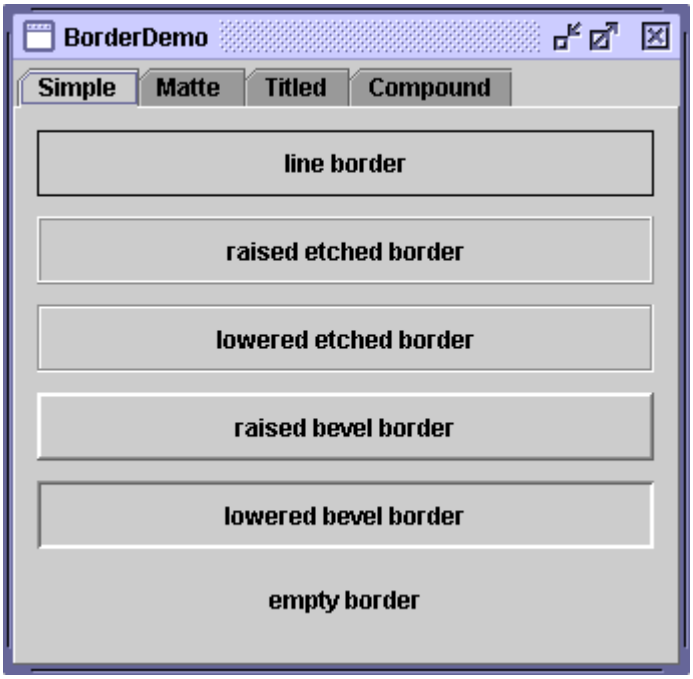


FlowLayout - Los componentes se van incorporando uno junto a otro, en fila, hasta que no hay suficiente espacio en la fila; entonces, se incorporan en una nueva fila

GridLayout - Los componentes manejados por este gestor se disponen en una malla de filas y columnas cuyo número se especifica en el constructor de la clase GridLayout. Todas las celdas disponen de, exactamente, el mismo espacio.



- Los Bordes
  Los bordes son también objetos Java. Con ellos se especifica el espacio que debe quedar inocupado entre componentes (EmptyBorder); también se utilizan para enmarcar un componente. En la siguiente imagen vemos algunos tipos:



3 El manejo de eventos en Java

Con el término *evento* se pretende dar a entender la acción que se desencadena cuando el usuario interacciona con una aplicación gráfica: pinchar con el ratón sobre un botón, seleccionar una opción de menú, etc.
Un componente envía un mensajes de aviso cuando el usuario interacciona con él. Es responsabilidad del programador escribir código que recoja ese aviso y actúe en consecuencia (código *manejador* del evento).

Por ejemplo, al pinchar con el ratón sobre un botón Salir o Cerrar se debe escribir código que recoja el aviso que envía el ratón y cerrar la ventana (o la aplicación).

Cada manejador de eventos consta de tres piezas de código:

- En la declaración de la clase manejadora del evento, una línea de código especifica que la clase o bien implementa una interfaz *escuchadora* o bien extiende una clase que implementa una interfaz escuchadora. Por ejemplo:
  public class MyClass implements ActionListener {
- Otra línea de código registra en uno o más componentes un objeto de la clase que implementa la interfaz *escuchadora*. Por ejemplo:
  someComponent.addActionListener(objectOfMyClass);
- La clase manejadora del evento tiene código que implementa los métodos de la interfaz *escuchadora*. Por ejemplo:
  public void actionPerformed(ActionEvent e) {
  // código que ejecuta las acciones apropiadas en respuesta al evento
  }

4 Tareas

El trabajo en esta práctica consiste en implementar la aplicación gráfica cuyo *ejecutable* se ha proporcionado junto con el guión.
A continuación presentamos una descripción detallada de la funcionalidad de la aplicación y de los componentes de la IGU:

0 La ventana principal de la aplicación

1 La Barra de Menú
1.1 La opción de menú Calculadora
1.1.1 La subopción de menú Limpiar Calculadora
1.2 La opción de menú Área de texto
1.2.1 La subopción de menú Limpiar Área de Texto

2 El contenedor principal (=contenedor asociado a la ventana de la aplicación)
2.1 El contenedor Calculadora
2.1.1 El contenedor Suma
2.1.1.1 El botón Sumar
2.1.1.2 El botón Limpiar
2.1.2 El contenedor Resta
2.2 El contenedor Área de Texto
2.2.1 El Área de Texto
2.3 El contenedor Cerrar
2.3.1 El botón Cerrar

Pasamos a describir cada uno de estos elementos:

0 La ventana de la aplicación

- Se trata de un objeto de la clase JFrame
- No es redimensionable por el usuario
- Su tamaño es de 800 x 600 (ancho x alto)

1 La Barra de Menú

- La barra de menú es un objeto de la clase JMenuBar
- Se incorpora a la ventana de la aplicación utilizando el método JFrame.setJMenuBar()

1.1 La opción de menú Calculadora

- La opción de menú Calculadora es un objeto de la clase JMenu
- El texto que muestra es, precisamente, "Calculadora"
- Se incorpora a la barra de menú utilizando el método JMenuBar.add()
- Despliega su subopción de menú cuando se pincha sobre ella con el ratón o cuando se teclea ALT+C
  Para conseguir esto último se debe utilizar el método JMenu.setMnemonic(); el argumento que se pasa al método es una constante de la clase KeyEvent

1.1.1 La subopción de menú Limpiar Calculadora

- La subopción de menú Limpiar Calculadora es un objeto de la clase JMenuItem
- El texto que muestra es, precisamente, "Limpiar CALCULADORA"
- Se incorpora a la opción de menú utilizando el método JMenu.add()
- Cuando se selecciona esta subopción de menú se deben inicializar todas las cajas de texto del contenedor Calculadora al valor "0.0" (las tres de la suma y las tres de la resta).
  Además, debe aparecer una línea en el área de texto con el literal: "Has seleccionado la opción de menú: "Limpiar CALCULADORA"
- Esta subopción de menú se selecciona cuando se pincha sobre ella con el ratón o cuando, estando desplegada la opción de menú, se teclea el carácter "L".
  Para conseguir esto último se debe utilizar el método JMenuItem.setMnemonic(); el argumento que se pasa al método es una constante de la clase KeyEvent.
- Para que la aplicación responda como se ha descrito en las líneas anteriores cuando se selecciona esta opción de menú, esta ha de tener registrado un objeto auditor o "escuchador" de eventos. Se recomienda que el objeto auditor de eventos que se registre asociado a esta subopción de menú implemente la interfaz ActionListener.

1.2 La opción de menú Área de texto

- La opción de menú Área de texto es un objeto de la clase JMenu
- El texto que muestra es, precisamente, "Área de texto"
- Se incorpora a la barra de menú utilizando el método JMenuBar.add()
- Despliega su subopción de menú cuando se pincha sobre ella con el ratón (automático) o cuando se teclea ALT+R.
  Para conseguir esto último se debe utilizar el método JMenu.setMnemonic(); el argumento que se pasa al método es una constante de la clase KeyEvent

1.2.1 La subopción de menú Limpiar Área de Texto

- La subopción de menú Limpiar Calculadora es un objeto de la clase JMenuItem
- El texto que muestra es, precisamente, "Limpiar ÁREA DE TEXTO"
- Se incorpora a la opción de menú utilizando el método JMenu.add()
- Cuando se selecciona esta subopción de menú se debe dejar en blanco el área de texto del contenedor Área de Texto, eliminando todo el texto que estuviese mostrando hasta ese momento
- Esta subopción de menú se selecciona cuando se pincha sobre ella con el ratón o cuando, estando desplegada la opción de menú, se teclea el carácter "I".

Para conseguir esto último se debe utilizar el método JMenuItem.setMnemonic(); el argumento que se pasa al método es una constante de la clase KeyEvent.

- Para que la aplicación responda como se ha descrito en las líneas anteriores cuando se selecciona esta opción de menú, esta ha de tener registrado un objeto auditor o "escuchador" de eventos. Se recomienda que el objeto auditor de eventos que se registre asociado a esta subopción de menú implemente la interfaz ActionListener.
  Este objeto puede ser de la misma clase o no que el objeto auditor registrado para la subopción de menú Limpiar Calculadora. En el primer caso el código de la clase deberá distinguir qué subopción de menú es la que ha generado el evento (para ello se puede utilizar el método ActionEvent.getSource() que devuelve una referencia de tipo Object apuntando al objeto que ha generado el evento); en el segundo caso el código no necesita discriminar el objeto generador porque está implícito en la declaración de la clase pero, por contra, hemos de crear una clase más.

2 El contenedor principal (=contenedor asociado a la ventana de la aplicación)

- Este contenedor es un objeto de la clase JPanel
- Se incorpora a la ventana de la aplicación utilizando el método: JFrame.setContentPane()
- El tamaño de este contenedor se puede sugerir utilizando el método: JPanel.setPreferredSize()
- El gestor de disposición asociado a este contenedor es de la clase BorderLayout y se establece utilizando el método JPanel.setLayout()
- Para mejorar el aspecto de la IGU, se puede asociar a este contenedor un borde vacío. Un objeto borde vacío se crea invocando el método estático BorderFactory.createEmptyBorder() y se asocia al contenedor mediante el método JPanel.setBorder()

2.1 El contenedor Calculadora

- El contenedor Calculadora es un objeto de la clase JPanel
- Se incorpora al contenedor principal utilizando el método JPanel.add(), especificando la posición con el valor de la constante BorderLayout.PAGE_START, definida en la clase BorderLayout
- El tamaño de este contenedor se puede sugerir utilizando el método: JPanel.setPreferredSize()
- El gestor de disposición asociado a este contenedor es de la clase GridLayout con 2 filas y 1 columna, para albergar los contenedores Suma y Resta; el gestor de disposición se establece utilizando el método JPanel.setLayout()
- Este panel tiene un borde titulado "Calculadora". Para crearlo se utiliza el método BorderFactory.createTitledBorder(). Para mejorar el aspecto de la IGU se puede añadir a este borde un borde vacío. Ambos bordes se pueden combinar en uno utilizando el método:
  BorderFactory.createCompoundBorder(un_borde, el_otro_borde)

2.1.1 El contenedor Suma

- El contenedor Suma es un objeto de la clase JPanel
- Se incorpora al contenedor Calculadora utilizando el método JPanel.add()
- El tamaño de este contenedor se puede sugerir utilizando el método JPanel.setPreferredSize()
- El gestor de disposición asociado a este contenedor es de la clase FlowLayout, para que los componentes que se vayan añadiendo aparezcan en fila; los componentes se incorporan utilizando el método JPanel.add()

- Los componentes que constituyen este contenedor son:
  1. Un cuadro de texto para que el usuario introduzca el primer sumando (objeto de la clase JTextField)
  2. Un etiqueta con el símbolo de la suma (objeto de la clase JLabel)
  3. Un cuadro de texto para que el usuario introduzca el segundo sumando
  4. Un botón con el texto "Sumar"
  5. Un cuadro de texto no editable (i.e., que el usuario no puede escribir en él) que es un objeto de la clase JTextField, para que la aplicación muestre el resultado de la suma
  6. Un botón con el texto "Limpiar"

A continuación pasamos a describir los componentes para los que se precisa un mayor detalle:

2.1.1.1 El botón Sumar

- El botón Sumar es un objeto de la clase JButton
- El texto del botón es, precisamente, "Sumar"
- El botón tiene asociado un tooltip con el texto "Sumar". Buscar en la clase el método adecuado para establecerlo
- Cuando se pincha con el ratón sobre este botón se debe mostrar en la caja de texto Resultado el valor de la suma de los valores contenidos en las cajas de texto Primer Sumando y Segundo Sumando. Para ello, se tiene que:
  1. Leer el valor contenido en las cajas de texto de los sumandos
  2. Convertir estos textos a números
  3. Sumar los valores numéricos
  4. Escribir el resultado en la caja de texto Resultado

  Además, al pulsar el botón también debe aparecer una línea en el área de texto con el literal: "SUMA: 8.3 + 1.2 = 9.5" (suponiendo que "8.3" y "1.2" son los valores de los sumandos).

  Por último, la aplicación también debe hacer un sencillo control de errores: si alguna de las cajas de texto de los sumando o bien está vacía o bien no contiene un número válido (por ejemplo, porque se introduzca una cadena en lugar de un número), debe establecer el valor "0.0" en las tres cajas de texto implicadas. El literal que se escribe en el área de texto será entonces: "SUMA: 0.0 + 0.0 = 0.0". Este sencillo control de errores se puede hacer con bloques try/catch utilizando la excepción NumberFormatException
- Para que la aplicación responda como se ha descrito en las líneas anteriores cuando se pincha con el ratón sobre este botón, este ha de tener registrado un objeto auditor o "escuchador" de eventos. Se recomienda que la clase del objeto auditor de eventos que se registre asociado a este botón extienda la clase MouseAdapter; si se hace así, es suficiente con sobreescribir el método MouseAdapter.mousePressed()
- NOTA: cuando este botón tiene el foco (el literal "Sumar" aparece rodeado con un pequeño rectángulo azul o morado) no se exige que responda al pulsado de la tecla ENTER, que es el comportamiento al que estamos habituados. La respuesta ante esta acción sólo se produce si escribimos código para ello. Si alguien se anima, puede hacerlo.

2.1.1.2 El botón Limpiar

- El botón Limpiar es un objeto de la clase JButton
- El texto del botón es, precisamente, "Limpiar"
- El botón tiene asociado un tooltip con el texto "Limpiar operadores y resultado". Buscar en la clase el método adecuado para establecerlo

- Cuando se pincha con el ratón sobre este botón se debe mostrar establecer el valor "0.0" en las tres cajas de texto implicadas
  Además, al pulsar el botón también debe aparecer una línea en el área de texto con el literal: "Se ha limpiado el contenedor Sumar".
- Para que la aplicación responda como se ha descrito en las líneas anteriores cuando se pincha con el ratón sobre este botón, este ha de tener registrado un objeto auditor o "escuchador" de eventos. Se recomienda que la clase del objeto auditor de eventos que se registre asociado a este botón extienda la clase MouseAdapter; si se hace así, es suficiente con sobreescribir el método MouseAdapter.mousePressed().

2.1.2 El contenedor Resta

- Todo lo dicho para el contenedor Suma es válido para el contenedor Resta, cambiando la lógica de sumar dos números por la de restarlos.

2.2 El contenedor Área de Texto

- El contenedor Área de Texto es un objeto de la clase JPanel
- Se incorpora al contenedor principal utilizando el método JPanel.add(), especificando la posición con el valor de la constante BorderLayout.CENTER, definida en la clase BorderLayout
- El tamaño de este contenedor se puede sugerir utilizando el método: JPanel.setPreferredSize()
- El gestor de disposición asociado a este contenedor es de la clase FlowLayout
- Este panel tiene un borde titulado "Área de texto". Para crearlo se utiliza el método BorderFactory.createTitledBorder(). Para mejorar el aspecto de la IGU se puede añadir a este borde un borde vacío. Ambos bordes se pueden combinar en uno utilizando el método BorderFactory.createCompoundBorder(un_borde, el_otro_borde)

2.2.1 El Área de Texto

- El Área de Texto es un objeto de la clase JScrollPane cuyo cliente (=el contenido que se maneja en el interior) es un objeto de la clase JTextArea
- El cliente de este panel se establece en la llamada al constructor; por tanto, se crea antes el objeto de la clase JTextArea y se pasa como argumento (o como uno de los argumentos) al constructor del panel
- El tamaño del panel se sugiere utilizando el método JScrollPane.setPreferredSize()
- El panel se incorpora al contenedor Área de Texto utilizando el método JPanel.add()
- El objeto de la clase JTextArea no es editable; i.e., el usuario no puede escribir en él
- El objeto de la clase JTextArea tiene la propiedad de cambiar a la siguiente línea cuando el texto alcanza el extremo de la derecha del panel

2.3 El contenedor Cerrar

- El contenedor Cerrar es un objeto de la clase JPanel
- Se incorpora al contenedor principal utilizando el método JPanel.add(), especificando la posición con el valor de la constante BorderLayout.PAGE_END, definida en la clase BorderLayout
- El tamaño de este contenedor se puede sugerir utilizando el método: JPanel.setPreferredSize()
- El gestor de disposición asociado a este contenedor es de la clase BorderLayout
- Para mejorar el aspecto de la IGU, se puede asociar a este contenedor un borde vacío. Un objeto borde vacío se crea invocando el método estático

BorderFactory.createEmptyBorder() y se asocia al contenedor mediante el método JPanel.setBorder()

2.3.1 El botón Cerrar

- El botón Cerrar es un objeto de la clase JButton
- Se incorpora al contenedor Cerrar utilizando el método JPanel.add(), especificando la posición con el valor de la constante BorderLayout.LINE_END, definida en la clase BorderLayout
- El texto del botón es, precisamente, "Cerrar"
- El botón tiene asociado un tooltip con el texto "Cerrar"
- Cuando se pincha con el ratón sobre este botón se debe cerrar la ventana y finalizar la aplicación. La sentencia que realiza esta acción es: System.exit(0)
- Para que la aplicación responda como se acaba de describir cuando se pincha con el ratón sobre este botón, este ha de tener registrado un objeto auditor o "escuchador" de eventos. Se recomienda que la clase del objeto auditor de eventos que se registre asociado a este botón extienda la clase MouseAdapter; si se hace así, es suficiente con sobreescribir el método: MouseAdapter.mousePressed()