

Practica 2

Supuesto 1. Programación Visual con JBuilder9

En la práctica anterior hemos visto como podemos crear un proyecto dónde nos hemos definido nuestras clases y paquetes.

En esta práctica intentaremos dotar a nuestros proyectos de características visuales que nos permitirán desarrollar una interfaz con la que los usuarios de nuestra aplicación van a interactuar.

Creación de un proyecto visual

Lo primero que veremos en esta práctica será como podemos crear un a aplicación que esté basada en una ventana principal desde la que podremos acceder a todas las funcionalidades de nuestra aplicación. Para conseguirlo deberemos crear un nuevo proyecto tal y como vimos en la práctica anterior. Un vez conseguido, seleccionaremos la opción “Nuevo” y en la pestaña “General” seleccionaremos “Aplicación”.



Una vez hecho esto aparecerá el asistente para aplicaciones que consta de 3 pasos:

Paso1: Indicaremos el paquete dónde se creará esta aplicación y el nombre del mismo.

Paso2: seleccionaremos el nombre del marco/ventana principal de la aplicación y el título deseado para la misma. Además en este paso también podremos indicar si nuestra aplicación contendrá una barra de menú, una barra de herramientas, barra de estado, cuadro “Acerca de” y si nuestro marco/ventana principal aparecerá centrado en la pantalla.

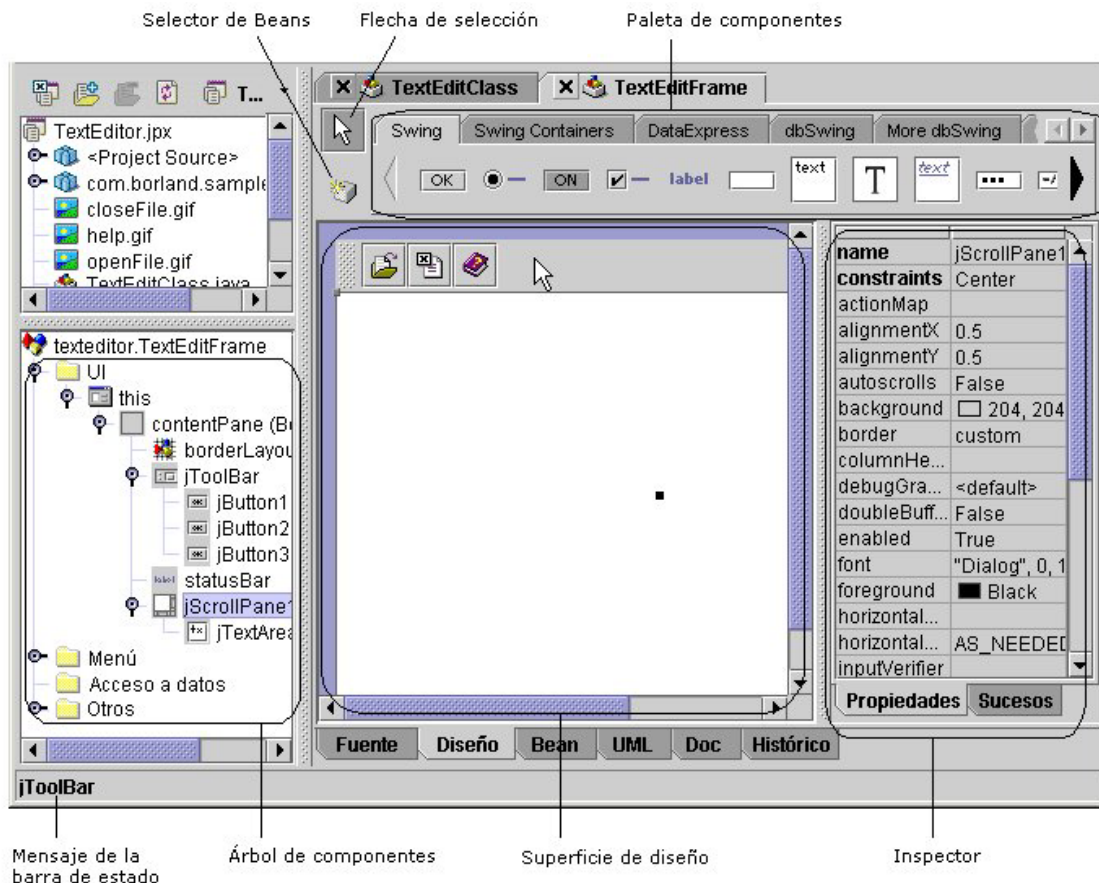
Paso3: Indicaremos la configuración de la aplicación, es decir, dónde se encuentra la clase que contendrá el método main desde la que se inicializará nuestra aplicación.

Así si seleccionamos todas las opciones del paso2 nos debe aparecer una aplicación parecida a la que vemos en la figura.

Por defecto JBuilder utiliza como ventana principal un objeto de tipo JFrame aunque dependiendo de la utilidad que deseemos también podrían seleccionarse otras como Frame, Dialog, Aplet, etc.

Las pestañas de Fuente y Diseño

JBuilder nos permite dos formas para trabajar con nuestros programas. Podemos trabajar directamente sobre la fuente de nuestras clases a través de la pestaña fuente como vimos en la anterior práctica o bien trabajar con el “diseñador de componentes” a través de la pestaña de “Diseño”. Cuando vamos cambiando entre estas pestañas podemos ver como los elementos que conforman la interfaz de JBuilder van variando. Cuando seleccionamos la pestaña “Fuente” tendremos acceso al código fuente de la clase y el árbol inferior nos mostrará todos los atributos y métodos de nuestra clase. Mientras que si seleccionamos la pestaña de “Diseño” y tenemos seleccionada una clase visual veremos el aspecto que tendría la misma en lo que llamamos “Superficie de Diseño”. También podemos observar como el árbol inferior a cambiado mostrándonos la jerarquía que conforman todos los componentes de nuestra clase. También aparecerá un nuevo elemento den la parte derecha denominado “Inspector” que nos mostrará las propiedades y sucesos/eventos del componente que tengamos seleccionado. Por último también podremos ver como ha aparecido un nuevo elemento en la parte superior de la superficie de diseño llamada “Paleta de componente” desde como veremos posteriormente podremos añadir los componentes a nuestra aplicación. Podemos ver el aspecto que ofrecería en la pestaña “Diseño” en la figura:







Trabajar con Componentes en JBuilder

Una vez tenemos seleccionada la ventana principal de nuestro proyecto en la pestaña de “Diseño” empezaremos a trabajar con los componentes que conformarán la aplicación.

Agregando Componentes – La paleta de componentes

La agregación de componentes se puede hacer de una manera sencilla y eficaz a través de la paleta de componentes. Esta paleta está dividida en varias pestañas, de entre ellas las que ahora nos van a interesar serán las dos primeras, La de componentes “Swing” y la de “Swing containers”

Que contienen componentes y contenedores como:

 JButton	→ El botón clásico de las aplicaciones
 JLabel	→ Nos permite mostrar una etiqueta
 JTextField	→ Nos permite crear un campo que el usuario rellenará
JPanel	→ Nos permite agrupar componentes
JTabbedPane	→ Nos permite agrupar los componentes en distintas pestañas
 JScrollPane	→ Nos permite agrupar componentes que en caso de ser excesivamente grandes para el área de visualización nos permitirán deslizarlos por la misma.

Los componentes los podemos **agregar** a un determinado contenedor de la siguientes formas:

Seleccione un componente de la paleta y haga clic en la superficie de diseño, en el lugar donde desee que aparezca la esquina superior izquierda del componente. Éste se coloca en la superficie de diseño, con la esquina superior izquierda en el lugar donde se ha hecho clic. El componente aparece también en el árbol de componentes.

Seleccione un componente de la paleta y haga clic en la ubicación jerárquica adecuada, en el árbol de componentes. El componente se coloca en el árbol. Si es visible, también aparece en el lugar adecuado de la superficie de diseño.

Elija Edición | Añadir componente. Se abre el cuadro de diálogo CVS Añadir: seleccione una biblioteca, elija un componente y pulse Aceptar. El componente aparece en el árbol de componentes. Si es visible, también aparece en el lugar adecuado de la superficie de diseño

Seleccionar un componente:

La selección es tan sencilla como seleccionar la flecha de la paleta de componentes (sino está ya seleccionada por defecto) y hacer clic sobre el componente deseado sobre la superficie de diseño dónde se encuentre. Una vez seleccionado el inspector de componentes nos mostrará todos las propiedades y sucesos/eventos del componente seleccionado.

También podremos seleccionar un componente desde el árbol jerárquico.

Mover/Copiar/Cortar/Pegar componentes:

Una vez seleccionado un componente nos permitirá todas estas funciones a través del menú edición o a través del menú contextual que nos aparece al pulsar sobre el botón derecho. Excepto mover que lo haremos arrastrando con el ratón.

Cambiar las dimensiones de un componentes:

Una vez seleccionado aparecerán unos cuadrados azules y negros con los que podremos redimensionar de forma correcta. Otra posibilidad es utilizar las propiedades del componente a las que accederemos a través del inspector de objetos. Las propiedades que nos interesarán en este caso serán todas aquellas que hagan referencia al tamaño del mismo como podrían ser *minimumSize*, *maximumSize*, *preferredSize*, etc.

Trabajar con los gestores de presentación

Un gestor de presentación es una clase Java que gestiona la posición y tamaño de los componentes de un contenedor. En este apartado veremos como establecer y cambiar los gestores de presentación. También veremos una lista de todos los layouts disponibles en JBuilder así como una pequeña explicación sobre cada uno de ellos. Resaltar que algunos de los gestores de presentación que usa JBuilder son propios y no se implementan en ninguna otro programa para el desarrollo de aplicaciones en Java como son el XYLayout o null.

Establecer y cambiar gestores de presentación

Para establecer el gestor de presentación debemos seleccionar un componente que disponga de la propiedad "Layout" que seleccionaremos desde el "Inspector de Componentes". (Es aconsejable seleccionar el layout deseado antes de empezar a añadir componentes a un contenedor).

Los gestores que podemos usar y sus características las podemos ver en la siguiente tabla:

Nombre	Descripción
Border Layout	Posibilita colocar los componentes en cinco posiciones: centro y a lo largo de los cuatro lados, Norte, Este, Sur y Oeste. Los componentes en el Norte y Sur se expanden horizontalmente para ocupar todo el espacio, mientras que los componentes en el lado Oeste y Este se expanden verticalmente.
Flow Layout	Ordena los componentes como palabras en una página. Líneas de izquierda a derecha y de arriba abajo. Se puede especificar la alineación de las filas y el espacio entre componentes y entre filas.
Grid Layout	Divide el contenedor en un número configurable de filas y columnas. Como el Flow Layout los componentes se añaden de izquierda a derecha y de abajo a arriba. Las dimensiones de la celda son fijas.
Card Layout	Podemos pensar en una presentación al estilo de un mazo de cartas.

	<p>Durante tiempo de ejecución únicamente una de estas cartas es visible. Durante diseño se muestra como un JTabbedPane para facilitar su diseño. En el código se pueden utilizar los métodos show, next, previous, first y last para seleccionar las cartas.</p>
GridBag Layout	<p>Permite realizar casi cualquier presentación, usando un conjunto Complejo de restricciones. Se puede crear un GridBag de una de las Siguietes tres maneras:</p> <p>Usando la orden Customize layout... en el formulario para mostrar el cuadro de diálogo de personalización, que permite ajustar visualmente la colocación y restricciones de los componentes.</p> <p>Marcando las restricciones en la ficha Layout de la hora de propiedades de cada componente.</p> <p>Creando un Absolute Layout y cambiando a GridBag para obtener las restricciones de forma automática. Si se usa esta opción puede necesitar volver a las dos opciones anteriores para cambiar algunos detalles.</p> <p>Se usa particularmente para aplicaciones multiplataforma, pues Posibilita la creación de presentaciones independientes del formulario que mantienen la consistencia de presentación incluso cuando cambia el look&feel de la plataforma</p>
Box Layout	<p>Es parte del API de Swing. Permite colocar múltiples componentes verticalmente u horizontalmente. Es parecido al GridLayout excepto en que utiliza una única dimensión.</p>
XYLayout	<p>XYLayout es un gestor de diseño personalizado de JBuilder. XYLayout sitúa los componentes en un contenedor en coordenadas x,y específicas, relativas a la esquina superior izquierda del contenedor. Independientemente del tipo de pantalla, el contenedor siempre mantendrá las posiciones x,y relativas de los componentes. Sin embargo, cuando redimensiona un contenedor con un XYLayout, los componentes no vuelven a posicionarse ni redimensionarse.</p>
Null	<p>Diseño Null significa que no se ha asignado ningún gestor de diseño al contenedor. El diseño Null (de Swing) es muy similar al XYLayout, en el sentido de que permite situar los componentes en los contenedores en unas determinadas coordenadas x,y relativas a la esquina superior izquierda del contenedor. Las coordenadas x,y de cada componente se especifican en su propiedad constraints. Más adelante, puede cambiar a un diseño portable más apropiado a sus necesidades.</p>
PanelLayout	<p>PanelLayout permite especificar el tamaño de un componente en relación con componentes del mismo nivel. PaneLayout, aplicado a un panel o a un marco, permite controlar el porcentaje del contenedor que tendrán los componentes en relación a los otros, pero no crea barras de división desplazables entre los paneles.</p>
VerticalFlowLayout	<p>Este gestor distribuye los componentes en columnas, de arriba abajo y de izquierda a derecha, en función del tamaño recomendado natural de cada contenedor. VerticalFlowLayout alinea tantos componentes en una columna como le es posible y se desplaza a la línea siguiente. Normalmente, VerticalFlowLayout se utiliza para ordenar botones en</p>

	un panel.
OverlayLayout	OverlayLayout2 es el diseño OverlayLayout de Swing integrado como bean para que pueda seleccionarse en el Inspector. Es muy similar a CardLayout en el sentido de que superpone los componentes. A diferencia de CardLayout que sólo permite ver los componentes de uno en uno, permite ver simultáneamente varios componentes; basta declararlos transparentes en el contenedor. Por ejemplo, pueden superponerse varias imágenes transparentes en el contenedor para obtener un gráfico compuesto.

Eventos(Sucesos)

La gestión de eventos es una de las funcionalidades más importantes a la hora de trabajar con componentes. Los eventos nos permiten capturar ciertos acontecimientos que tienen relación con un determinado componente. Algunos ejemplos de eventos serían por ejemplo cuando hacemos clic sobre un componente JButton o cuando cambiamos el texto de un JTextField o cuando pasamos el ratón por encima del componente, etc.

Definir manejadores de eventos/sucesos

Un manejador de evento es un objeto que nos permite insertar código que se ejecutará cuando determinado suceso se produzca. Para definirlos, la forma estándar, es seleccionar la pestaña de sucesos en el “Inspector de Componentes” una vez hemos seleccionado el componente con el que deseamos trabajar. Posteriormente elegiremos el evento que deseamos gestionar y haremos un doble clic en el campo derecho al nombre del evento. Con esta acción conseguiremos rellenar esta campo con el nombre por defecto y nos transportará la método que deberemos implementar para tratar el evento. Aquí será donde escribamos las sentencias Java que nos permiten actuar con el entorno del componente con el que trabajamos.

Algunos eventos importantes que actúan sobre componentes como JButton, JRadioButton, JCheckBox y muchos otros, es el evento/suceso “actionPerformed”. Este evento nos permite capturar cuando el usuario presiona sobre el botón, círculo o cuadrado.

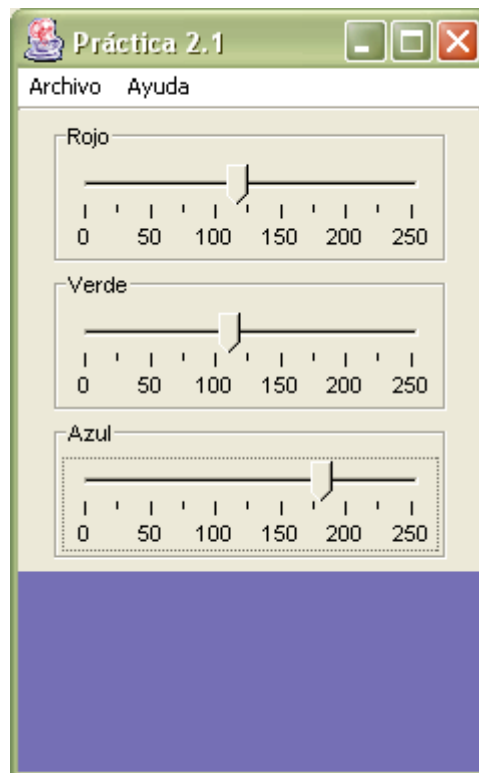
Otro tipo de eventos/sucesos son los que se producen al accionar el ratón, como:

- mouseClicked → Cuando realizamos un clic sobre el componente
- mouseDragged → Cuando intentamos arrastrar el componente
- mouseEntered → Cuando el ratón entra dentro de las dimensiones del componente
- mouseExited → Cuando el ratón sale del componente
- mousePressed → Cuando sobre el componente se presiona el botón del ratón
- MouseReleased → Cuando soltamos el botón del ratón

Alguno de los métodos importantes al gestionar los eventos del ratón son `getX()` y `getY()` al cual accederemos a través del parámetro que se le pasa al método que vamos a implementar. Estos métodos nos indican la posición del ratón al producirse el evento/suceso. Para más información podemos consultar los tutoriales de que dispone JBuilder o acudir a la documentación de la API en la página <http://java.sun.com>
 Para borrar los manejadores de evento que hemos creado con anterioridad nos debemos situar sobre el campo derecho del evento/suceso que hemos seleccionado anteriormente y borrar el texto que está escrito pulsando posteriormente sobre enter.

Actividad propuesta:

Un ejemplo muy típico de cuadro de diálogo es el que ofrece al usuario la posibilidad de seleccionar un color. En esta actividad vamos a intentar llevar a cabo esta tarea, creando un cuadro de diálogo con tres controles de selección del grado de cada una de las componentes RGB que queremos en el color a seleccionar. El aspecto final del formulario debe ser el siguiente:



Los paso que debe seguir hasta conseguir construir el formulario serian los siguientes:

1. Crear un nuevo proyecto y una aplicación.
 - Crear un nuevo proyecto denominado “Práctica21” y crear una aplicación según hemos visto al comienzo. Con todas las opciones del Paso2 excepto la de la barra de herramientas ya que no la vamos a utilizar.
 - El nombre del marco/ventana principal será el de `JFrameSelectorColor`.

2. Añadir componentes.

- Cambiamos el layout del componente de la ventana a “**BorderLayout**”
- Añadimos un *JPanel* en el sur del formulario.
- El panel aparece con el nombre por defecto (*JPanel1*). Renómbrelo a *JPanelColor*. Cambiar el gestor de presentación del panel también a “**BorderLayout**”. Y cambiar el tamaño del mismo para que tenga mayores dimensiones.
- Añada otro nuevo *JPanel* a la zona centro del formulario, configurando este nuevo panel con el modo de presentación “**GridBagLayout**”.
- Añadir tres componentes *JSlider* al panel que acabamos de crear cambiando su nombre por *JSliderRojo*, *JSliderVerde* y *JSliderAzul*. Seleccione los tres controles y de a la propiedad *Maximum* el valor 255. Y cambiar todas las propiedades hasta que la apariencia sea parecida al ejemplo que tenemos en la página anterior.

3. Añadir bordes.

- Cambiar la propiedad *Border* de los *JSlider* a **Titled**.
- Pulsaremos posteriormente sobre los puntos suspensivos que aparecen a la derecha de **Titled**. Debemos indicar en **Title** Rojo, Verde y Azul respectivamente.

Añadir el código.

- Seleccionamos cada uno de los *JSlider* y los vamos modificando.
- Para añadir el código seleccionamos el suceso *StateChanged* haciendo doble clic en el campo derecho.
- En el método al que nos lleva debemos añadir la siguiente sentencia:
`this.jPanelColor.setColor(new Color(this.jSliderRojo.getValue(), this.jSliderVerde.getvalue(), this.jSliderAzul.getValue()));`
- Esto lo realizaremos para los tres componentes *JSlider* añadidos.