11.D. Layouts.

Tabla de contenidos

1 Organizadores de contenedores: layout managers.

Los layout managers son fundamentales en la creación de interfaces de usuario, ya que determinan las posiciones de los controles en un contenedor.

En lenguajes de programación para una única plataforma, el problema sería menor porque el aspecto sería fácilmente controlable. Sin embargo, dado que Java está orientado a la portabilidad del código, éste es uno de los aspectos más complejos de la creación de interfaces, ya que las medidas y posiciones dependen de la máquina en concreto.

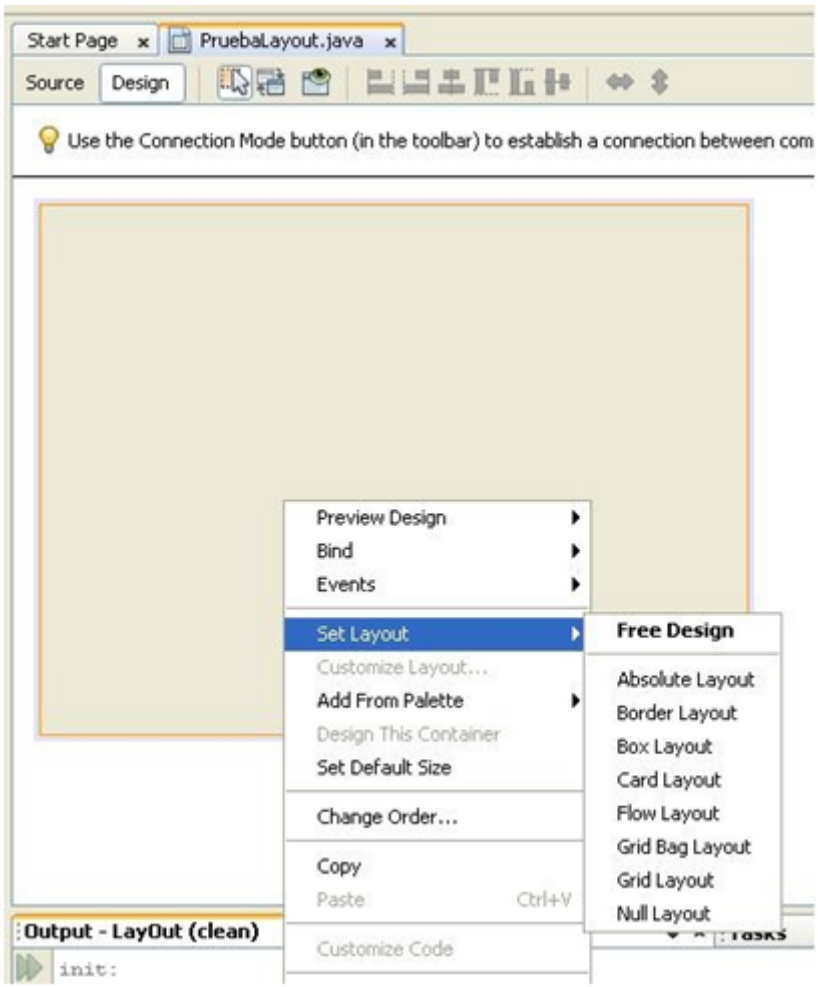
En algunos entornos los componentes se colocan con coordenadas absolutas. En Java se desaconseja esa práctica porque en la mayoría de casos es imposible prever el tamaño de un componente.

Por tanto, en su lugar, se usan organizadores o también llamados **administradores de diseño** o **layout managers** o **gestores de distribución** que permiten colocar y maquetar de forma independiente de las coordenadas.

Debemos hacer un buen diseño de la interfaz gráfica, y así tenemos que elegir el mejor gestor de distribución para cada uno de los contenedores o paneles de nuestra ventana.

Esto podemos conseguirlo con el método `setLayout()`, al que se le pasa como argumento un objeto del tipo de Layout que se quiere establecer.

En NetBeans, una vez insertado un `JFrame`, si nos situamos sobre él y pulsamos botón derecho, se puede ver, como muestra la imagen, que aparece un menú, el cual nos permite elegir el layout que queramos.



El eclipse, una vez incorporado el plugin WindowBuilder, tenemos una utilidad similar.

Debes conocer

En la siguiente web puedes ver gráficamente los distintos layouts

LayOuts (en inglés)

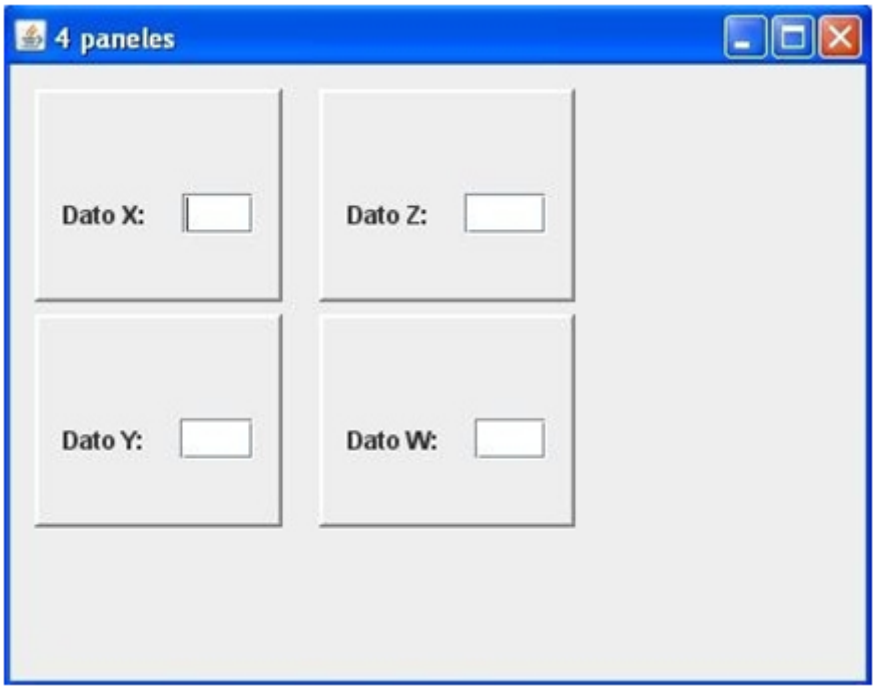
Autoevaluación

Cuando programamos en Java es aconsejable establecer coordenadas absolutas, siempre que sea posible, en nuestros componentes.

- Verdadero.
- Falso.

2 Contenedor ligero: JPanel.

Es la clase utilizada como contenedor genérico para agrupar componentes.



Normalmente cuando una ventana de una aplicación con interfaz gráfica cualquiera presenta varios componentes, para hacerla más atractiva y ordenada al usuario se suele hacer lo siguiente:

- Crear un marco, un `JFrame`.

- Para organizar mejor el espacio en la ventana, añadiremos varios paneles, de tipo `JPanel`. (Uno para introducir los datos de entrada, otro para mostrar los resultados y un tercero como zona de notificación de errores.)

- Cada uno de esos paneles estará delimitado por un borde que incluirá un título. Para ello se usan las clases disponibles en `BorderFactory` (`BevelBorder` , `CompoundBorder` , `EmptyBorder` , `EtchedBorder` , `LineBorder` , `LoweredBevelBorder` , `MatteBorder` y `TitledBorder`) que nos da un surtido más o menos amplio de tipos de bordes a elegir.

- En cada uno de esos paneles se incluyen las etiquetas y cuadros de texto que se necesitan para mostrar la información adecuadamente.

Con Eclipse o con NetBeans es tan fácil como arrastrar tantos controles `JPanel` de la paleta hasta el `JFrame`. Por código, también es muy sencillo, por ejemplo podríamos crear un panel rojo, darle sus características y añadirlo al JFrame del siguiente modo:

```
…
```

```
JPanel panelRojo = new JPanel();
```

```
panelRojo.setBackground(Color.RED);
```

```
panelRojo.setSize(300,300);
```

```
// Se crea una ventana
```

```
JFrame ventana=new JFrame("Prueba en rojo");
```

```
ventana.setLocation(100,100);
```

```
ventana.setVisible(true);
```

```
// Se coloca el JPanel en el content pane
```

```
Container contentPane=ventana.getContentPane();
```

```
contentPane.add(panelRojo);
```

```
…
```

Cuando en Sun desarrollaron Java, los diseñadores de Swing, por alguna circunstancia, determinaron que para algunas funciones, como añadir un `JComponent`, los programadores no pudiéramos usar `JFrame.add`, sino que en lugar de ello, primeramente, tuviéramos que obtener el objeto `Container` asociado con `JFrame.getContentPane()`, y añadirlo.

Sun se dio cuenta del error y ahora permite utilizar `JFrame.add`, desde Java 1.5 en adelante. Sin embargo, podemos tener el problema de que un código desarrollado y ejecutado correctamente en 1.5 falle en máquinas que tengan instalado 1.4 o anteriores.