Layouts Java

José León

November 2016

Los layouts son los que, en una interfaz gráfica, nos permite establecer donde y como se posicionan los diferentes componentes. Además de ello cambia el tamaño de los componentes para ajustarlo al tamaño de la ventana. Podemos encontrar varios tipos de layouts los cuales se describirán a continuación.

1. Layout Null

Es cuando no se usa ningún layout y nosotros establecemos donde va cada componente, el problema con este es que; al agrandar la ventana, cambiar de sistema operativo, de fuente, etc.; los componentes pueden tener letras faltantes, ser muy pequeña y otros problemas de visualización.

2. FlowLayout

Es el layout predeterminado para todos los JPanel, coloca a los componentes en linea, uno tras de otro, creando nuevas lineas si no alcanzan en la actual.

Figura 1: Ejemplo de Flow Layout

3. BoxLayout

Comparte las características del *FlowLayout* con la diferencia de que permite colocar tanto filas como columnas de componentes.

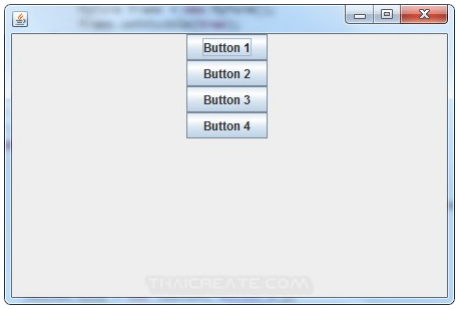

Figura 2: Ejemplo de Box Layout

4. GridLayout

Este *layout* permite crear una cuadricula con los componentes de modo que todos tengan el mismo ancho y largo, además de darnos la posibilidad de establecer en cuantas filas y columnas queremos mostrarlos.

Figura 3: Ejemplo de Grid Layout

5. BorderLayout

Nos permite colocar dividir la ventana y colocar los componentes en las cuatro direcciones cardinales además del centro, se coloca un solo componente por cada parte y el *layout* hará que este se estire hasta ocupar todo el espacio, haciendo que la ventana no tenga espacios vacíos.

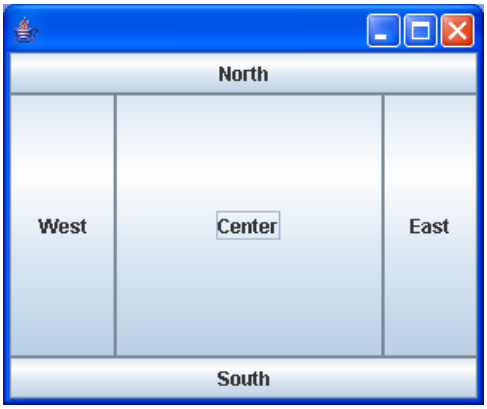

Figura 4: Ejemplo de Border Layout

6. GridBagLayout

Es uno de los mas versátiles y complejos, nos permite colocar los componentes en un sistema de celdas, sin embargo, un componente puede ocupar mas de una celda, además de la posibilidad para que el alto de cada celda sea diferente, también se puede establecer las acciones al agrandar la pantalla, y en caso de no querer que el componente se agrande, establecer una posición dentro de la misma celda con el fin de acomodarlo.

Figura 5: Ejemplo de GridBag Layout

7. CardLayout

El *CardLayout* nos permite hacer que un espacio definido sea ocupado por varios componentes de diferentes tipos, superponiendolos unos con otros y mostrando solo los que se indiquen en *ComboBox* o por ejemplo en una *Tabbed Pane*.

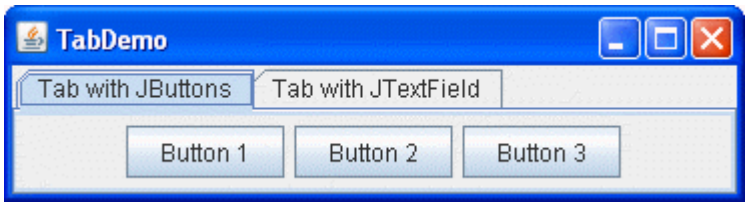

Figura 6: Ejemplo de Card Layout

8. SpringLayout

Nos permite establecer una distancia a la que se encuentra cada uno de los componentes respecto a sus bordes, volviendolo muy flexible.

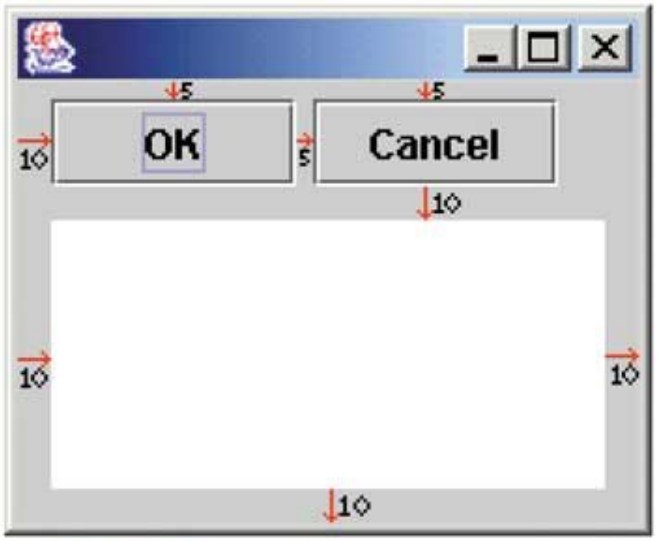

Figura 7: Ejemplo de Spring Layout

9. GroupLayout

Posee la caracteristica del *BoxLayout* con la diferencia de que las dimensiones en vertical y horizontal se tratan por separado de modo que cada componente debe ser establecido en cada una de ellas.

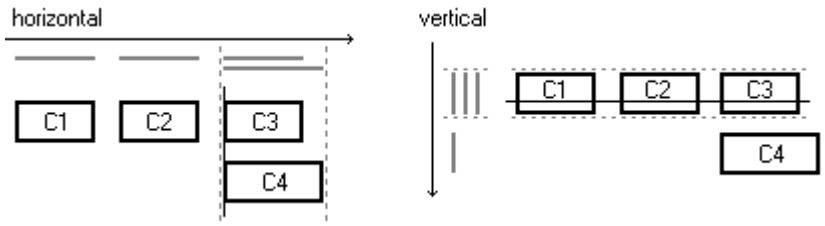

Figura 8: Ejemplo de Group Layout

El material fue extraído de las paginas citadas a continuación. [1] [2] [3]

Referencias

[1] Corporativo. *Uso de Layouts*. 2014. URL: `http://chuwiki.chuidiang.org/index.php?title=Uso_de_Layouts`.

[2] S. Jaiswal. *Layout Manager*. 2011. URL: `http://www.javatpoint.com/java-layout-manager`.

[3] Oracle. *A Visual Guide to Layout Managers*. URL: `https://docs.oracle.com/javase/tutorial/uiswing/layout/visual.html`.