

**UNIVERSIDAD PRIVADA FRANZ TAMAYO**

**DEFENSA HITO 4 - TAREA FINAL**

Nombre Completo: **Landa Garcia Daniela Claudia**

Asignatura: **PROGRAMACIÓN III**

Carrera: **INGENIERÍA DE SISTEMAS**

Paralelo: **PROG (3)**

Docente: **Lic. William R. Barra Paredes**

fecha: **17/06/2020**

Git: **https://github.com/Danylanda/progra3.git**

# Parte Teórica.

## Preguntas.

Responda de manera breve, clara y concisa posible.

* **Defina que es SWING, a que se refiere cuando se habla de AWT.**
  + - Swing es una biblioteca gráfica para Java. Incluye widgets para interfaz gráfica de usuario tales como cajas de texto, botones, listas desplegables y tablas.
    - La Abstract Window Toolkit (AWT, en español Kit de Herramientas de Ventana Abstracta) es un kit de herramientas de gráficos, interfaz de usuario, y sistema de ventanas independiente de la plataforma original de Java.
* **Qué son los componentes SWING, mencionar algunos de ellos.**
  + - Elementos que se añaden a contenedores. Usualmente los componentes tienen aspecto gráfico. Como un botón, label, caja de textos,etc.
* **Importancia de JFRAME.**
  + - Toda aplicación Swing tiene, al menos, un contenedor raíz. La clase JFrame proporciona ventanas al uso, A su vez, JFrame incluye una serie de elementos
* **Qué es un LISTENER en SWING.**
  + - Los Listeners se encargan de controlar los eventos, esperan a que el evento se produzca y realiza una serie de acciones. Según el evento, necesitaremos un Listener que lo controle.
* **Describa cómo funciona paso a paso el componente GRIDLAYOUT.**
  + - El GridBagLayout es útil, ya que divide su ventana en filas y columnas, y usted decide en qué fila y columna colocar los componentes, así como en cuántas filas y columnas es grande el componente.
* **Describa cómo funciona paso a paso el componente FLOWLAYOUT.**
  + - Pone a los componentes que le añadimos alineados de izquierda a derecha, haciendo que cada uno ocupe lo que necesita. Si hay hueco de sobra en horizontal, los componentes aparecerán centrados. Si falta hueco, los componentes se partirán automáticamente en varias filas.
* **Describa cómo funciona paso a paso el componente BORDERLAYOUT**.
  + - El BorderLayout divide la ventana en 5 partes: centro, arriba, abajo, derecha e izquierda.
    - Hará que los componentes que pongamos arriba y abajo ocupen el alto que necesiten, pero los estirará horizontalmente hasta ocupar toda la ventana.
    - Los componentes de derecha e izquierda ocuparán el ancho que necesiten, pero se les estirará en vertical hasta ocupar toda la ventana.
    - El componente central se estirará en ambos sentidos hasta ocupar toda la ventana.

**Parte Práctica.**

* GRIDLAYOUT

Mostrar un ejemplo en donde se use un GRIDLAYOUT.

* Como se usa.
* Cómo funciona
* En el componente principal JFRAME debe mostrarse el uso.

|  |
| --- |
| * [GridLayout():](https://docs.oracle.com/javase/7/docs/api/java/awt/GridLayout.html#GridLayout()) empty constructor with one column and as many rows as the components. * [GridLayout(int rows, int cols):](https://docs.oracle.com/javase/7/docs/api/java/awt/GridLayout.html#GridLayout(int,%20int)) a constructor with specified rows and columns. * [GridLayout(int rows, int cols, int hgap, int vgap):](https://docs.oracle.com/javase/7/docs/api/java/awt/GridLayout.html#GridLayout(int,%20int,%20int,%20int)) a constructor with specified rows, columns, horizontal and vertical gaps.   The arguments that are used are:   * rows: is the number of rows in the grid. * cols: represents the number of columns in the grid. * hgap: is the horizontal space between the cells. * vgap: is the vertical space between the cells * package com.javacodegeeks.desktop.gridlayout; * import java.awt.GridLayout; * import javax.swing.JButton; * import javax.swing.JFrame; * import javax.swing.JTextArea; * public class GridLayoutTest { * private static JButton[] arrayBtn; * public static void main(String[] args) { * // the frame that contains the components * JFrame frame = new JFrame("GridLayoutTest from JCG"); * frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE); * // set the size of the frame * frame.setSize(350, 350); * // set the rows and cols of the grid, as well the distances between them * GridLayout grid = new GridLayout(5, 3, 10, 10); * // what layout we want to use for our frame * frame.setLayout(grid); * // add a text field with a specified text to the frame * JTextArea text = new JTextArea(); * text.setText("Result"); * text.setEditable(false); * frame.add(text); * // add buttons to the frame * frame.add(new JButton("+")); * frame.add(new JButton("=")); * arrayBtn = new JButton[10]; * // add JButtons dynamically * for(int i=0; i < arrayBtn.length; i++) { * arrayBtn[i] = new JButton(Integer.toString(i)); * frame.add(arrayBtn[i]); * } * frame.setVisible(true); * } * } |

* FLOWLAYOUT

Mostrar un ejemplo en donde se use un FLOWLAYOUT.

* + Como se usa.
  + Cómo funciona
  + En el componente principal JFRAME debe mostrarse el uso.

|  |
| --- |
| 1. **FlowLayout ():** crea un diseño de flujo con alineación centrada y un espacio horizontal y vertical predeterminado de 5 unidades. 2. **FlowLayout (int align):** crea un diseño de flujo con la alineación dada y un espacio horizontal y vertical predeterminado de 5 unidades. 3. **FlowLayout (int align, int hgap, int vgap):** crea un diseño de flujo con la alineación dada y el espacio horizontal y vertical dado. 4. **import** java.awt.\*; 5. **import** javax.swing.\*; 7. **public** **class** MyFlowLayout{ 8. JFrame f; 9. MyFlowLayout(){ 10. f=**new** JFrame(); 12. JButton b1=**new** JButton("1"); 13. JButton b2=**new** JButton("2"); 14. JButton b3=**new** JButton("3"); 15. JButton b4=**new** JButton("4"); 16. JButton b5=**new** JButton("5"); 18. f.add(b1);f.add(b2);f.add(b3);f.add(b4);f.add(b5); 20. f.setLayout(**new** FlowLayout(FlowLayout.RIGHT)); 21. //setting flow layout of right alignment 23. f.setSize(300,300); 24. f.setVisible(**true**); 25. } 26. **public** **static** **void** main(String[] args) { 27. **new** MyFlowLayout(); 28. } 29. } |

* BORDERLAYOUT

Mostrar un ejemplo en donde se use un BORDERLAYOUT.

* + Como se usa.
  + Cómo funciona
  + En el componente principal JFRAME debe mostrarse el uso.

|  |
| --- |
| Como muestra la imagen anterior, un BorderLayoutobjeto tiene cinco áreas   * PAGE\_START * PAGE\_END * LINE\_START * LINE\_END * CENTER   ...*//Container pane = aFrame.getContentPane()*...  JButton button = new JButton("Button 1 (PAGE\_START)");  pane.add(button, BorderLayout.PAGE\_START);  //Make the center component big, since that's the  //typical usage of BorderLayout.  button = new JButton("Button 2 (CENTER)");  button.setPreferredSize(new Dimension(200, 100));  pane.add(button, BorderLayout.CENTER);  button = new JButton("Button 3 (LINE\_START)");  pane.add(button, BorderLayout.LINE\_START);  button = new JButton("Long-Named Button 4 (PAGE\_END)");  pane.add(button, BorderLayout.PAGE\_END);  button = new JButton("5 (LINE\_END)");  pane.add(button, BorderLayout.LINE\_END); |