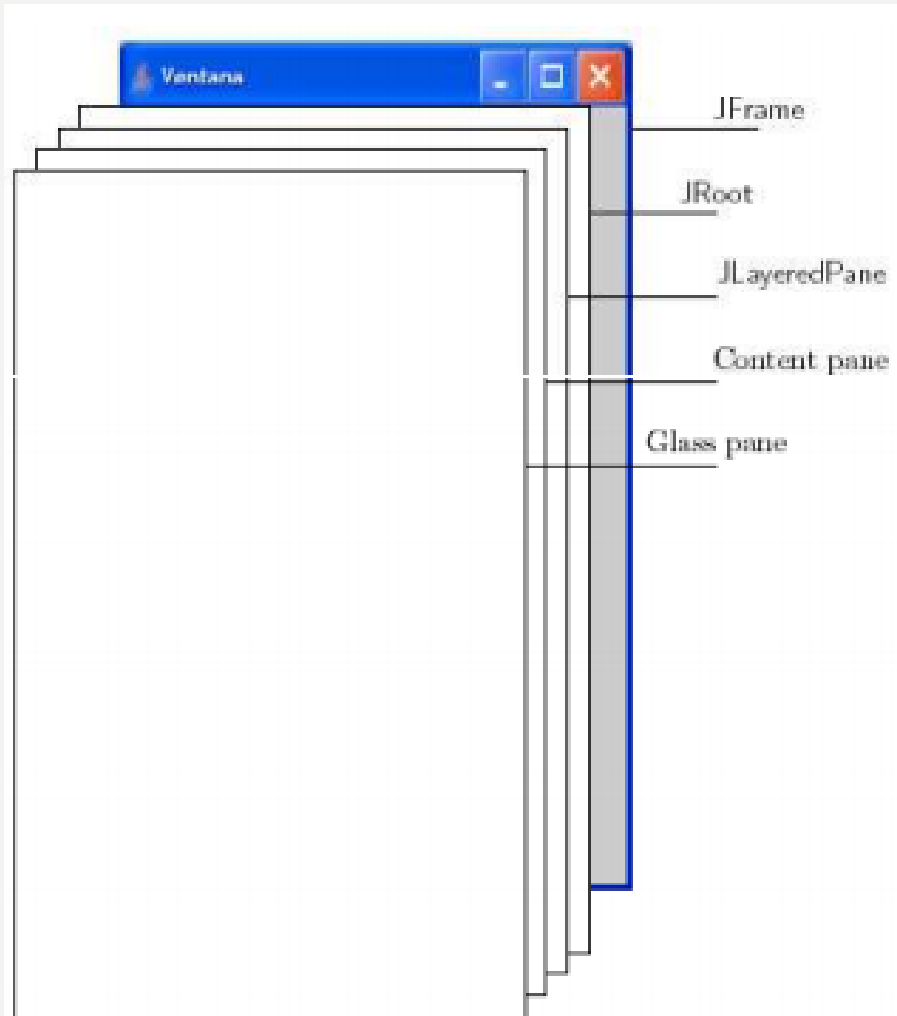




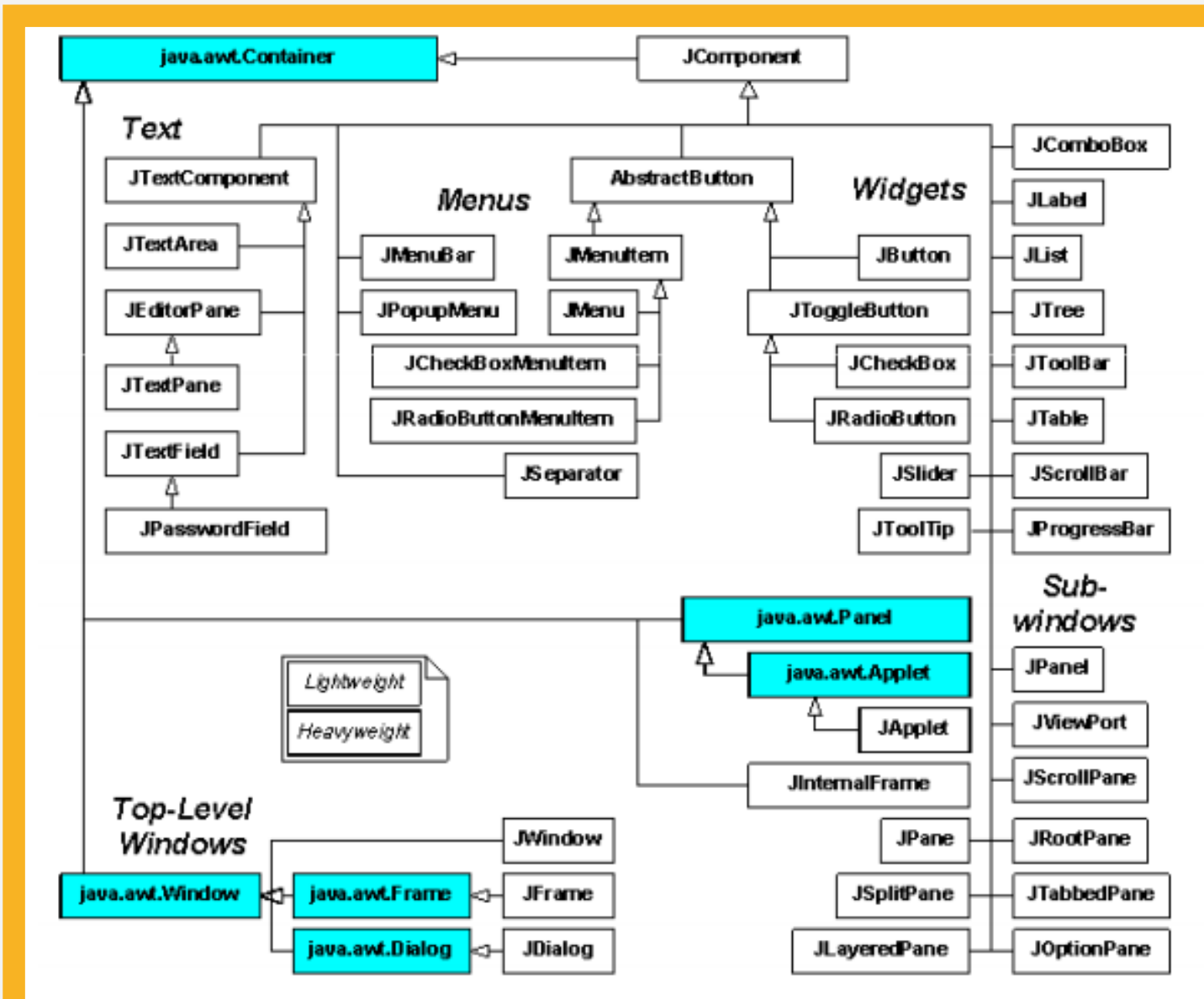
SWING

INTRODUCCIÓ



- Los componentes intermedios son aquellos que agrupan componentes atómicos, los más utilizados: JPanel, JScrollPane, JSplitPane, JTabbedPane y JToolBar.

JERARQUIA DE CLASSES



ESTRUCTURA BÀSICA D'UNA APLICACIÓ SWING

- Es construeix barrejant components amb les següents regles:
 - Ha d'existir, al menys, un contenidor d'alt nivell (Top-Level Container), que proveeix el suport que les components de Swing necessiten per ser pintades i els events.
 - Altres components penjant del contenidor d'alt nivell (aquest poden ser contenidors senzills o components).

PRINCIPALS ASPECTES D'UNA APLICACIÓ SWING

- Cada aplicació Swing ha de tenir al menys un top-level aquets poden ser:
- **javax.swing.JFrame**: Una finestra independent.
- **javax.swing.JApplet**: Un applet.
- **Diàlegs**: Finestres d'interacció senzilla amb l'usuari com per exemple:
 - `java.swing.JOptionPane`: Finestra de diàleg tipus SI_NO, SI_NO_CANCELAR, ACCEPTAR, etc...
 - `java.swing.JFileChooser`: Finestra per escollir una arxiu.
 - `java.swing.JColorChooser`

JDIALOG

Constructors

Constructor	Description
<code>JDialog()</code>	Creates a modeless dialog without a title and without a specified Frame owner.
<code>JDialog(Dialog owner)</code>	Creates a modeless dialog with the specified Dialog as its owner and an empty title.
<code>JDialog(Dialog owner, boolean modal)</code>	Creates a dialog with an empty title and the specified modality and Dialog as its owner.
<code>JDialog(Dialog owner, String title)</code>	Creates a modeless dialog with the specified title and with the specified owner dialog.
<code>JDialog(Dialog owner, String title, boolean modal)</code>	Creates a dialog with the specified title, modality and the specified owner Dialog.
<code>JDialog(Dialog owner, String title, boolean modal, GraphicsConfiguration gc)</code>	Creates a dialog with the specified title, owner Dialog, modality and GraphicsConfiguration.
<code>JDialog(Frame owner)</code>	Creates a modeless dialog with the specified Frame as its owner and an empty title.
<code>JDialog(Frame owner, boolean modal)</code>	Creates a dialog with an empty title and the specified modality and Frame as its owner.
<code>JDialog(Frame owner, String title)</code>	Creates a modeless dialog with the specified title and with the specified owner frame.
<code>JDialog(Frame owner, String title, boolean modal)</code>	Creates a dialog with the specified title, owner Frame and modality.
<code>JDialog(Frame owner, String title, boolean modal, GraphicsConfiguration gc)</code>	Creates a dialog with the specified title, owner Frame, modality and GraphicsConfiguration.
<code>JDialog(Window owner)</code>	Creates a modeless dialog with the specified Window as its owner and an empty title.
<code>JDialog(Window owner, Dialog.ModalityType modalityType)</code>	Creates a dialog with an empty title and the specified modality and Window as its owner.
<code>JDialog(Window owner, String title)</code>	Creates a modeless dialog with the specified title and owner Window.
<code>JDialog(Window owner, String title, Dialog.ModalityType modalityType)</code>	Creates a dialog with the specified title, owner Window and modality.
<code>JDialog(Window owner, String title, Dialog.ModalityType modalityType, GraphicsConfiguration gc)</code>	Creates a dialog with the specified title, owner Window, modality and GraphicsConfiguration.

GESTORS DE POSICIÓ: BorderLayout

- Un **BorderLayout** té cinc àrees objecte. Aquestes àrees són especificades per les constants del BorderLayout:

Fields		
Modifier and Type	Field	Description
static String	AFTER_LAST_LINE	Synonym for PAGE_END.
static String	AFTER_LINE_ENDS	Synonym for LINE_END.
static String	BEFORE_FIRST_LINE	Synonym for PAGE_START.
static String	BEFORE_LINE_BEGINS	Synonym for LINE_START.
static String	CENTER	The center layout constraint (middle of container).
static String	EAST	The east layout constraint (right side of container).
static String	LINE_END	The component goes at the end of the line direction for the layout.
static String	LINE_START	The component goes at the beginning of the line direction for the layout.
static String	NORTH	The north layout constraint (top of container).
static String	PAGE_END	The component comes after the last line of the layout's content.
static String	PAGE_START	The component comes before the first line of the layout's content.
static String	SOUTH	The south layout constraint (bottom of container).
static String	WEST	The west layout constraint (left side of container).

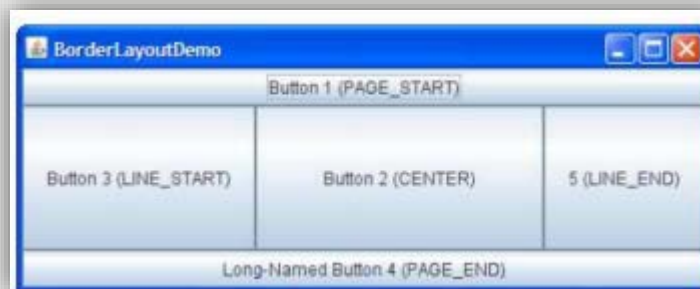
```
... // Panel de contenedores aFrame.getContentPane = () ...
JButton button = new JButton ("Botón 1 (PAGE_START)");
pane.add (botón, BorderLayout.PAGE_START);

// Hacer que el componente de gran centro, ya que es el
// Uso típico de BorderLayout.
button = new JButton ("Botón 2 (Centro)");
button.setPreferredSize (new Dimension (200, 100));
pane.add (botón, BorderLayout.CENTER);

button = new JButton ("Botón 3 (LINE_START)");
pane.add (botón, BorderLayout.LINE_START);

button = new JButton ("Long-nombrada botón 4 (PAGE_END)");
pane.add (botón, BorderLayout.PAGE_END);

button = new JButton ("5 (LINE_END)");
pane.add (botón, BorderLayout.LINE_END);
```



Es el Layout por defecto de un JFrame.

GESTORS DE POSICIÓ: FlowLayout

- El **FlowLayout** proporciona un gestor de disseny molt simple que s'utilitza, per defecte, en els *JPanel*.
- Les dimensions, tant horitzontals com verticals, dels components es respecten.
- Els components se disposen un a continuació de l'anterior, de la mateixa manera que les paraules sobre un text. Quan s'acaba l'espai horitzontal, s'inicia una nova línia.

```
Label label_1=new Label("Aquest era el Label del Nord");  
  
Label label_2=new Label("I aquest el del Sud");  
  
Label label_3=new Label("Era del Est");  
  
Label label_4=new Label("Del Oest");  
  
Label label_5=new Label("Era el Label del Centre");  
  
mainFrame.add(label_1);  
mainFrame.add(label_2);  
mainFrame.add(label_3);  
mainFrame.add(label_4);  
mainFrame.add(label_5);
```



GESTOR DE POSICIÓ: GridLayout

- Disposa els components en una matriu rectangular de files i columnes. El mètode constructor és:
public GridLayout(int files,int columnes)
- No es respecta cap de les dimensions dels components. Les dimensions es forcen per a que totes les cel·les siguin iguals.



JLabel i ImageIcon

Constructors

Constructor	Description
<code>JLabel()</code>	Creates a JLabel instance with no image and with an empty string for the title.
<code>JLabel(String text)</code>	Creates a JLabel instance with the specified text.
<code>JLabel(String text, int horizontalAlignment)</code>	Creates a JLabel instance with the specified text and horizontal alignment.
<code>JLabel(String text, Icon icon, int horizontalAlignment)</code>	Creates a JLabel instance with the specified text, image, and horizontal alignment.
<code>JLabel(Icon image)</code>	Creates a JLabel instance with the specified image.
<code>JLabel(Icon image, int horizontalAlignment)</code>	Creates a JLabel instance with the specified image and horizontal alignment.

Constructors

Constructor	Description
<code>ImageIcon()</code>	Creates an uninitialized image icon.
<code>ImageIcon(byte[] imageData)</code>	Creates an ImageIcon from an array of bytes which were read from an image file containing a supported image format, such as GIF, JPEG, or (as of 1.3) PNG.
<code>ImageIcon(byte[] imageData, String description)</code>	Creates an ImageIcon from an array of bytes which were read from an image file containing a supported image format, such as GIF, JPEG, or (as of 1.3) PNG.
<code>ImageIcon(Image image)</code>	Creates an ImageIcon from an image object.
<code>ImageIcon(Image image, String description)</code>	Creates an ImageIcon from the image.
<code>ImageIcon(String filename)</code>	Creates an ImageIcon from the specified file.
<code>ImageIcon(String filename, String description)</code>	Creates an ImageIcon from the specified file.
<code>ImageIcon(URL location)</code>	Creates an ImageIcon from the specified URL.
<code>ImageIcon(URL location, String description)</code>	Creates an ImageIcon from the specified URL.

COMPONENTS: ImageIcon-JLabel

```
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.FlowLayout;
import javax.swing.*.*;

public class Exemple1 extends JFrame{
    private JPanel panel;
    private JLabel etiqueta1, etiqueta2;

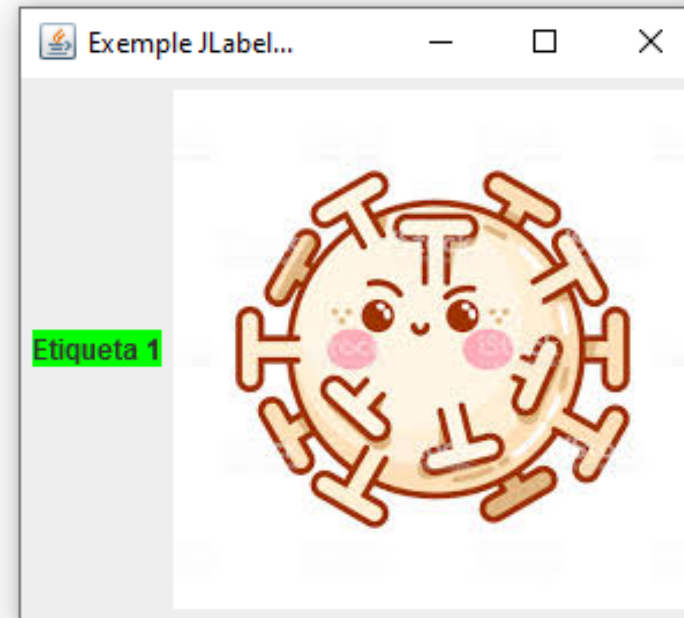
    public Exemple1() {
        panel=new JPanel();
        etiqueta1=new JLabel("Etiqueta 1");
        //PER A CANVIAR EL COLOR DE FONTS:
        etiqueta1.setOpaque(true);
        etiqueta1.setBackground(Color.GREEN);

        ImageIcon imagen=new ImageIcon("imagen.png");

        etiqueta2=new JLabel(imagen);
        panel.add(etiqueta1);
        panel.add(etiqueta2);
        this.add(panel, BorderLayout.CENTER);
        this.setTitle("Exemple JLabel...");
        this.pack();
        this.setVisible(true);
    }

    public static void main(String[] args) {

        new Exemple1();
    }
}
```



JButton

Constructors

Constructor	Description
<code>JButton()</code>	Creates a button with no set text or icon.
<code>JButton(String text)</code>	Creates a button with text.
<code>JButton(String text, Icon icon)</code>	Creates a button with initial text and an icon.
<code>JButton(Action a)</code>	Creates a button where properties are taken from the Action supplied.
<code>JButton(Icon icon)</code>	Creates a button with an icon.

COMPONENTS: JButton

```
import java.awt.BorderLayout;
import javax.swing.*;

public class Exemple2 extends JFrame {

    private JPanel panel;
    private JButton boton1, boton2;

    public Exemple2() {
        panel=new JPanel();

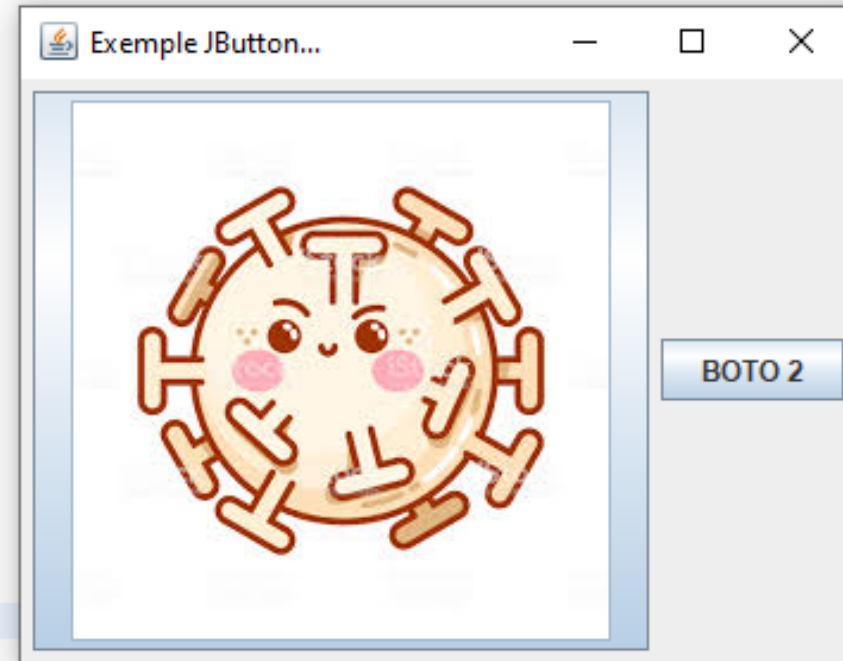
        ImageIcon imagen=new ImageIcon("imagen.png");

        boton1=new JButton(imagen);
        boton2=new JButton("BOTO 2");

        panel.add(boton1);
        panel.add(boton2);
        this.add(panel, BorderLayout.CENTER);
        this.setTitle("Exemple JButton...");
        this.pack();
        this.setVisible(true);
    }

    public static void main(String[] args) {

        new Exemple2();
    }
}
```



JTextArea

Constructors

Constructor	Description
<code>JTextArea()</code>	Constructs a new <code>TextArea</code> .
<code>JTextArea(int rows, int columns)</code>	Constructs a new empty <code>TextArea</code> with the specified number of rows and columns.
<code>JTextArea(String text)</code>	Constructs a new <code>TextArea</code> with the specified text displayed.
<code>JTextArea(String text, int rows, int columns)</code>	Constructs a new <code>TextArea</code> with the specified text and number of rows and columns.
<code>JTextArea(Document doc)</code>	Constructs a new <code>JTextArea</code> with the given document model, and defaults for all of the other arguments (null, 0, 0).
<code>JTextArea(Document doc, String text, int rows, int columns)</code>	Constructs a new <code>JTextArea</code> with the specified number of rows and columns, and the given model.

JTextArea

```
import java.awt.BorderLayout;
import javax.swing.*;

public class Exemple3 extends JFrame{

    private JPanel panel;
    private JTextArea texto;

    public Exemple3() {
        panel=new JPanel();

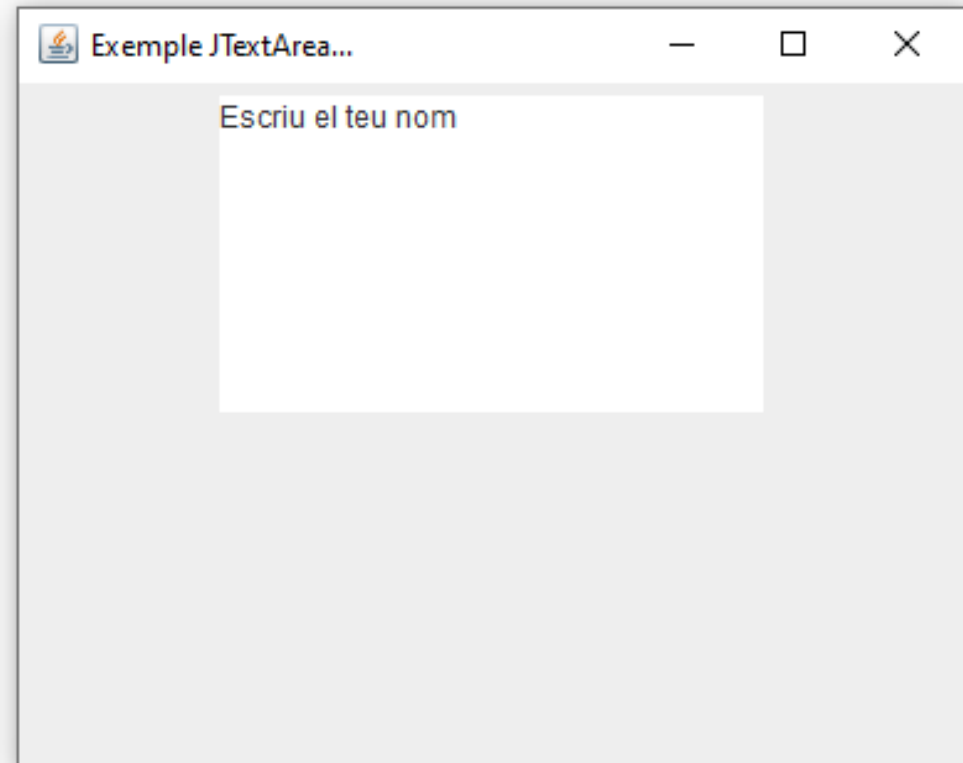
        texto=new JTextArea("Escriu el teu nom",8,20);

        panel.add(texto);

        this.add(panel, BorderLayout.CENTER);
        this.setTitle("Exemple JTextArea...");
        this.pack();
        this.setVisible(true);
    }

    public static void main(String[] args) {

        new Exemple3();
    }
}
```



JTextField

Constructors

Constructor	Description
<code>JTextField()</code>	Constructs a new <code>TextField</code> .
<code>JTextField(int columns)</code>	Constructs a new empty <code>TextField</code> with the specified number of columns.
<code>JTextField(String text)</code>	Constructs a new <code>TextField</code> initialized with the specified text.
<code>JTextField(String text, int columns)</code>	Constructs a new <code>TextField</code> initialized with the specified text and columns.
<code>JTextField(Document doc, String text, int columns)</code>	Constructs a new <code>JTextField</code> that uses the given text storage model and the given number of columns.

JTextField

```
import java.awt.BorderLayout;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;

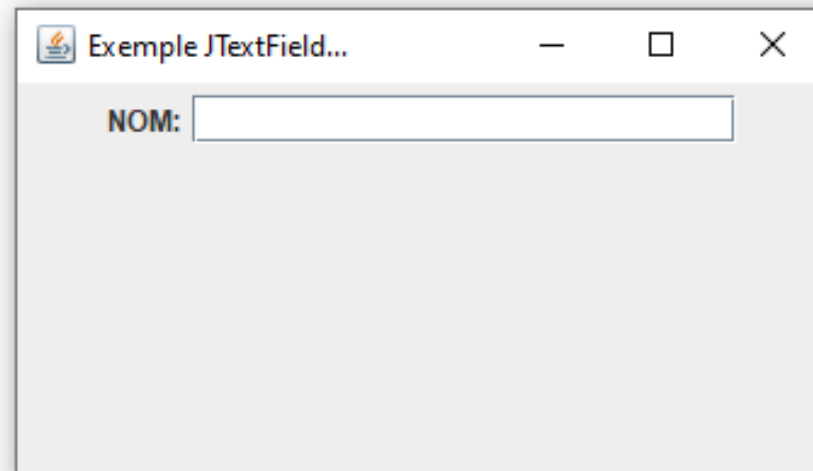
public class Exemple4 extends JFrame {

    private JPanel panel;
    private JTextField texto;
    private JLabel nom;

    public Exemple4() {
        panel=new JPanel();
        nom=new JLabel("NOM:");
        texto=new JTextField(20);
        panel.add(nom);
        panel.add(texto);

        this.add(panel,BorderLayout.CENTER);
        this.setTitle("Exemple JTextField...");
        this.setSize(350, 200);
        this.setVisible(true);
    }
    public static void main(String[] args) {

        new Exemple4();
    }
}
```



JList

Constructors

Constructor	Description
<code>JList()</code>	Constructs a <code>JList</code> with an empty, read-only, model.
<code>JList(E[] listData)</code>	Constructs a <code>JList</code> that displays the elements in the specified array.
<code>JList(Vector<? extends E> listData)</code>	Constructs a <code>JList</code> that displays the elements in the specified <code>Vector</code> .
<code>JList(ListModel<E> dataModel)</code>	Constructs a <code>JList</code> that displays elements from the specified, non-null, model.

JList

```
import java.awt.BorderLayout;

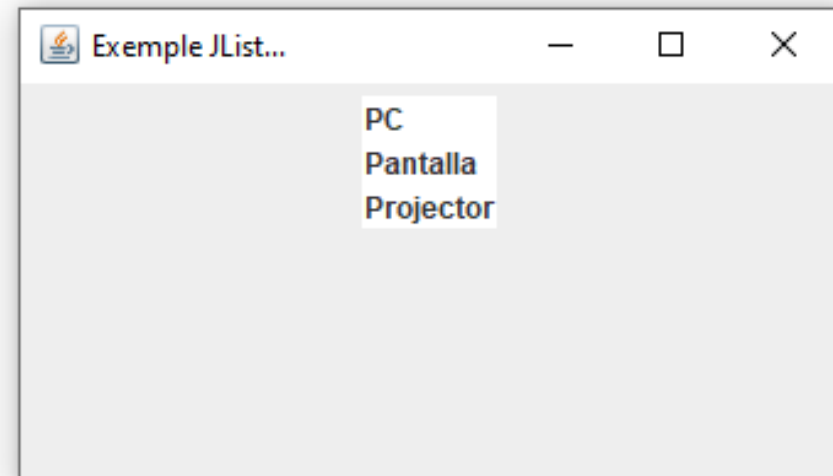
import javax.swing.*;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class Exemple5 extends JFrame {
    private JPanel panel;
    private JList lista;

    public Exemple5() {
        panel=new JPanel();
        String []datos= {"PC","Pantalla","Projector"};
        lista=new JList(datos);
        panel.add(lista);

        this.add(panel,BorderLayout.CENTER);
        this.setTitle("Exemple JList...");
        this.setSize(350, 200);
        this.setVisible(true);
    }
    public static void main(String[] args) {

        new Exemple5();
    }
}
```



JComboBox

Constructors

Constructor	Description
<code>JComboBox()</code>	Creates a JComboBox with a default data model.
<code>JComboBox(E[] items)</code>	Creates a JComboBox that contains the elements in the specified array.
<code>JComboBox(Vector<E> items)</code>	Creates a JComboBox that contains the elements in the specified Vector.
<code>JComboBox(ComboBoxModel<E> aModel)</code>	Creates a JComboBox that takes its items from an existing ComboBoxModel.

JComboBox

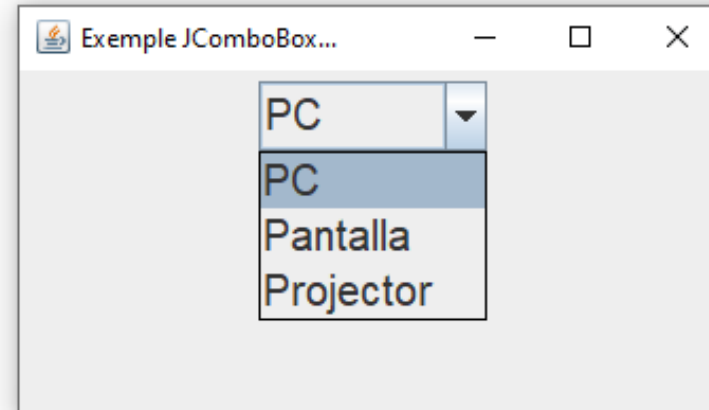
```
import java.awt.BorderLayout;
import java.awt.Font;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JPanel;

public class Exemple6 extends JFrame{
    private JPanel panel;
    private JComboBox cb;

    public Exemple6() {
        panel=new JPanel();
        String []datos= {"PC","Pantalla","Projector"};
        cb=new JComboBox();
        //CANVIEM EL TIPUS DE FONT
        cb.setFont(new Font("Arial",Font.PLAIN,20));
        cb.addItem("PC");
        cb.addItem("Pantalla");
        cb.addItem("Projector");
        panel.add(cb);

        this.add(panel,BorderLayout.CENTER);
        this.setTitle("Exemple JComboBox...");
        this.setSize(350, 200);
        this.setVisible(true);
    }
    public static void main(String[] args) {

        new Exemple6();
    }
}
```



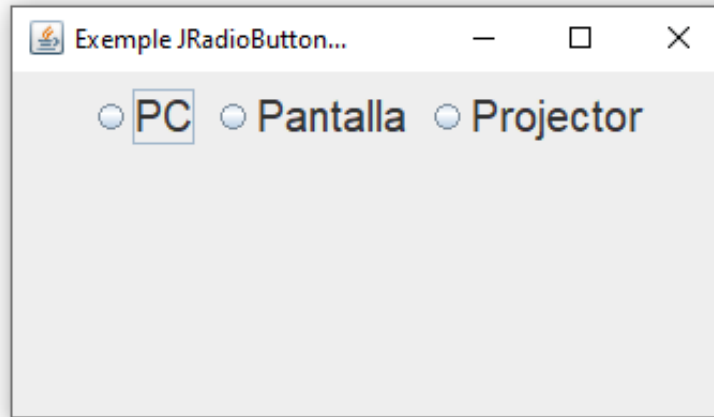
JRadioButton

Constructors

Constructor	Description
<code>JRadioButton()</code>	Creates an initially unselected radio button with no set text.
<code>JRadioButton(String text)</code>	Creates an unselected radio button with the specified text.
<code>JRadioButton(String text, boolean selected)</code>	Creates a radio button with the specified text and selection state.
<code>JRadioButton(String text, Icon icon)</code>	Creates a radio button that has the specified text and image, and that is initially unselected.
<code>JRadioButton(String text, Icon icon, boolean selected)</code>	Creates a radio button that has the specified text, image, and selection state.
<code>JRadioButton(Action a)</code>	Creates a radiobutton where properties are taken from the Action supplied.
<code>JRadioButton(Icon icon)</code>	Creates an initially unselected radio button with the specified image but no text.
<code>JRadioButton(Icon icon, boolean selected)</code>	Creates a radio button with the specified image and selection state, but no text.

JRadioButton

```
import java.awt.BorderLayout;
import java.awt.Font;
import javax.swing.*;
public class Exemple7 extends JFrame{
    private JPanel panel;
    private JRadioButton pc,pantalla,projector;
    private ButtonGroup grupo;
    public Exemple7() {
        panel=new JPanel();
        grupo=new ButtonGroup();
        pc=new JRadioButton("PC",false);
        pantalla=new JRadioButton("Pantalla",false);
        projector=new JRadioButton("Projector",false);
        //CANVIEM EL TIPUS DE FONT
        pc.setFont(new Font("Arial",Font.PLAIN,20));
        pantalla.setFont(new Font("Arial",Font.PLAIN,20));
        projector.setFont(new Font("Arial",Font.PLAIN,20));
        //AFEGIM AL GRUP
        grupo.add(pc);
        grupo.add(pantalla);
        grupo.add(projector);
        //AFEGIM AL PANEL
        panel.add(pc);
        panel.add(pantalla);
        panel.add(projector);
        this.add(panel,BorderLayout.CENTER);
        this.setTitle("Exemple JRadioButton...");
        this.setSize(350, 200);
        this.setVisible(true);
    }
    public static void main(String[] args) {
        new Exemple7();
    }
}
```



JCheckBox

Constructors

Constructor	Description
<code>JCheckBox()</code>	Creates an initially unselected check box button with no text, no icon.
<code>JCheckBox(String text)</code>	Creates an initially unselected check box with text.
<code>JCheckBox(String text, boolean selected)</code>	Creates a check box with text and specifies whether or not it is initially selected.
<code>JCheckBox(String text, Icon icon)</code>	Creates an initially unselected check box with the specified text and icon.
<code>JCheckBox(String text, Icon icon, boolean selected)</code>	Creates a check box with text and icon, and specifies whether or not it is initially selected.
<code>JCheckBox(Action a)</code>	Creates a check box where properties are taken from the Action supplied.
<code>JCheckBox(Icon icon)</code>	Creates an initially unselected check box with an icon.
<code>JCheckBox(Icon icon, boolean selected)</code>	Creates a check box with an icon and specifies whether or not it is initially selected.

JCheckBox

```
import java.awt.BorderLayout;
import java.awt.Font;
import javax.swing.*;

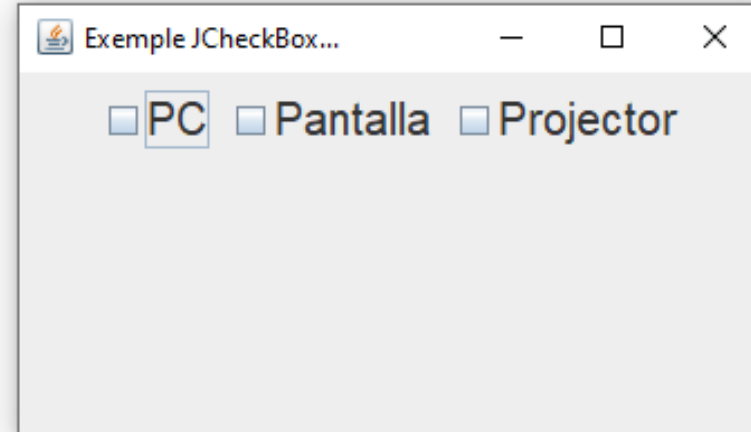
public class Exemple8 extends JFrame{
    private JPanel panel;
    private JCheckBox pc,pantalla,projector;

    public Exemple8() {
        panel=new JPanel();

        pc=new JCheckBox("PC",false);
        pantalla=new JCheckBox("Pantalla",false);
        projector=new JCheckBox("Projector",false);
        //CANVIEM EL TIPUS DE FONT
        pc.setFont(new Font("Arial",Font.PLAIN,20));
        pantalla.setFont(new Font("Arial",Font.PLAIN,20));
        projector.setFont(new Font("Arial",Font.PLAIN,20));

        //AFEGIM AL PANEL
        panel.add(pc);
        panel.add(pantalla);
        panel.add(projector);
        this.add(panel,BorderLayout.CENTER);
        this.setTitle("Exemple JCheckBox...");
        this.setSize(350, 200);
        this.setVisible(true);
    }

    public static void main(String[] args) {
        new Exemple8();
    }
}
```



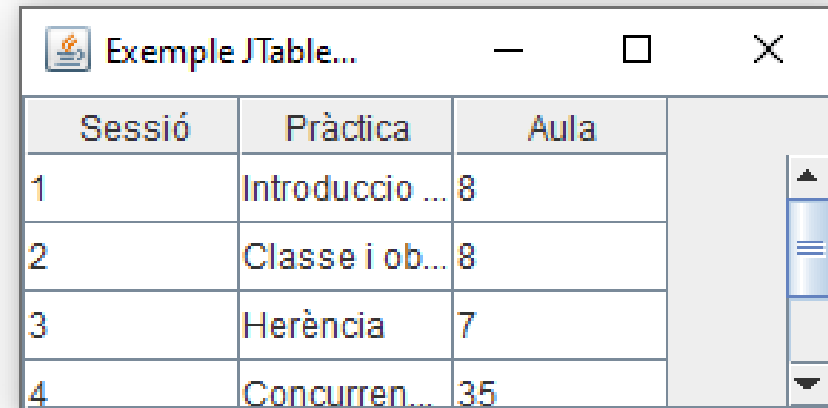
JTable

Constructors

Constructor	Description
<code>JTable()</code>	Constructs a default JTable that is initialized with a default data model, a default column model, and a default selection model.
<code>JTable(int numRows, int numColumns)</code>	Constructs a JTable with <code>numRows</code> and <code>numColumns</code> of empty cells using <code>DefaultTableModel</code> .
<code>JTable(Object[][] rowData, Object[] columnNames)</code>	Constructs a JTable to display the values in the two dimensional array, <code>rowData</code> , with column names, <code>columnNames</code> .
<code>JTable(Vector<? extends Vector> rowData, Vector<?> columnNames)</code>	Constructs a JTable to display the values in the Vector of Vectors, <code>rowData</code> , with column names, <code>columnNames</code> .
<code>JTable(TableModel dm)</code>	Constructs a JTable that is initialized with <code>dm</code> as the data model, a default column model, and a default selection model.
<code>JTable(TableModel dm, TableColumnModel cm)</code>	Constructs a JTable that is initialized with <code>dm</code> as the data model, <code>cm</code> as the column model, and a default selection model.
<code>JTable(TableModel dm, TableColumnModel cm, ListSelectionModel sm)</code>	Constructs a JTable that is initialized with <code>dm</code> as the data model, <code>cm</code> as the column model, and <code>sm</code> as the selection model.

JTable

```
public class Exemple9 extends JFrame{
    private JTable taula;
    private JScrollPane jsp;
    public Exemple9() {
        //NOM COLUMNES
        String []nomCol= {"Sessió","Pràctica","Aula"};
        //DADES TAULA
        String [][]datos= {{ "1","Introduccio a java","8"}, {"2","Classe i objectes","8"}, {"3","Herència","7"},
                             {"4","Concurrencia","35"}, {"5","Entrada/Sortida","10"}, {"6","Xarxes","15"} };
        taula=new JTable(datos,nomCol);
        taula.setRowHeight(24); //Alçada de cada fila
        taula.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
        jsp=new JScrollPane(taula);
        //AFEGIM AL JFrame
        this.add(jsp,BorderLayout.CENTER);
        this.setTitle("Exemple JTable...");
        this.setSize(300, 150);
        this.setVisible(true);
    }
    public static void main(String[] args) {
        new Exemple9();
    }
}
```



The screenshot shows a Java Swing window titled "Exemple JTable...". Inside the window is a JTable with 3 columns and 4 rows. The columns are labeled "Sessió", "Pràctica", and "Aula". The rows contain data as follows:

Sessió	Pràctica	Aula
1	Introduccio ...	8
2	Classe i ob...	8
3	Herència	7
4	Concurren...	35

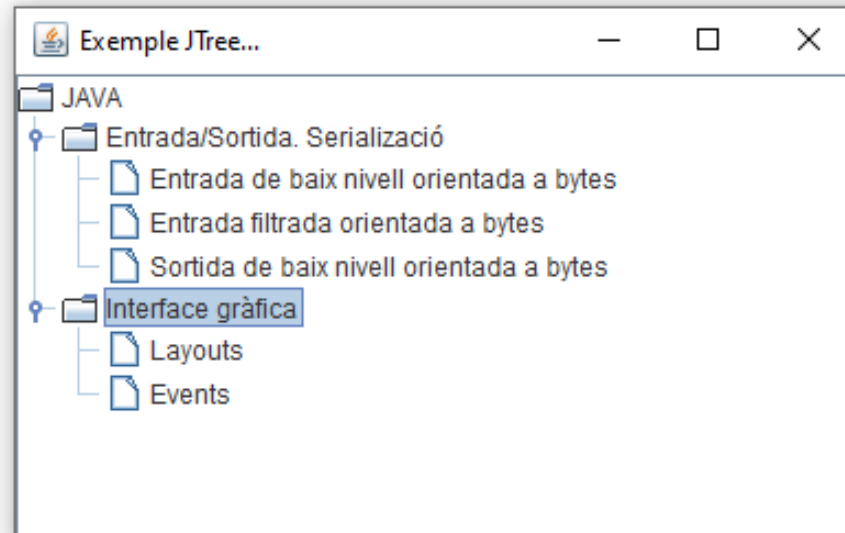
JTree

Constructors

Constructor	Description
<code>JTree()</code>	Returns a <code>JTree</code> with a sample model.
<code>JTree(Object[] value)</code>	Returns a <code>JTree</code> with each element of the specified array as the child of a new root node which is not displayed.
<code>JTree(Hashtable<?,?> value)</code>	Returns a <code>JTree</code> created from a <code>Hashtable</code> which does not display with root.
<code>JTree(Vector<?> value)</code>	Returns a <code>JTree</code> with each element of the specified <code>Vector</code> as the child of a new root node which is not displayed.
<code>JTree(TreeModel newModel)</code>	Returns an instance of <code>JTree</code> which displays the root node -- the tree is created using the specified data model.
<code>JTree(TreeNode root)</code>	Returns a <code>JTree</code> with the specified <code>TreeNode</code> as its root, which displays the root node.
<code>JTree(TreeNode root, boolean asksAllowsChildren)</code>	Returns a <code>JTree</code> with the specified <code>TreeNode</code> as its root, which displays the root node and which decides whether a node is a leaf node in the specified manner.

JTree

```
public class Exemple10 extends JFrame{
    private JScrollPane jsp;
    private JTree arbol;
    public Exemple10() {
        //NOM COLUMNES
        DefaultMutableTreeNode asig=new DefaultMutableTreeNode("JAVA");
        DefaultTreeModel java=new DefaultTreeModel(asig);
        DefaultMutableTreeNode tema =null;
        DefaultMutableTreeNode seccion=null;
        tema=new DefaultMutableTreeNode("Entrada/Sortida. Serializació");
        asig.add(tema);
        seccion= new DefaultMutableTreeNode("Entrada de baix nivell orientada a bytes");
        tema.add(seccion);
        seccion= new DefaultMutableTreeNode("Entrada filtrada orientada a bytes");
        tema.add(seccion);
        seccion= new DefaultMutableTreeNode("Sortida de baix nivell orientada a bytes");
        tema.add(seccion);
        tema=new DefaultMutableTreeNode("Interface gràfica");
        asig.add(tema);
        seccion= new DefaultMutableTreeNode("Layouts");
        tema.add(seccion);
        seccion= new DefaultMutableTreeNode("Events");
        tema.add(seccion);
        arbol=new JTree(java);
        jsp=new JScrollPane(arbol);
        this.add(jsp,BorderLayout.CENTER);
        this.setTitle("Exemple JTree...");
        this.setSize(400, 250);
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setVisible(true);
    }
    public static void main(String[] args) {
        new Exemple10();
    }
}
```



JMenu

Constructors

Constructor	Description
<code>JMenu()</code>	Constructs a new JMenu with no text.
<code>JMenu(String s)</code>	Constructs a new JMenu with the supplied string as its text.
<code>JMenu(String s, boolean b)</code>	Constructs a new JMenu with the supplied string as its text and specified as a tear-off menu or not.
<code>JMenu(Action a)</code>	Constructs a menu whose properties are taken from the Action supplied.

JMenu

```
import java.awt.BorderLayout;
import javax.swing.*;

public class Exemple11 extends JFrame {
    private JMenuBar menu;

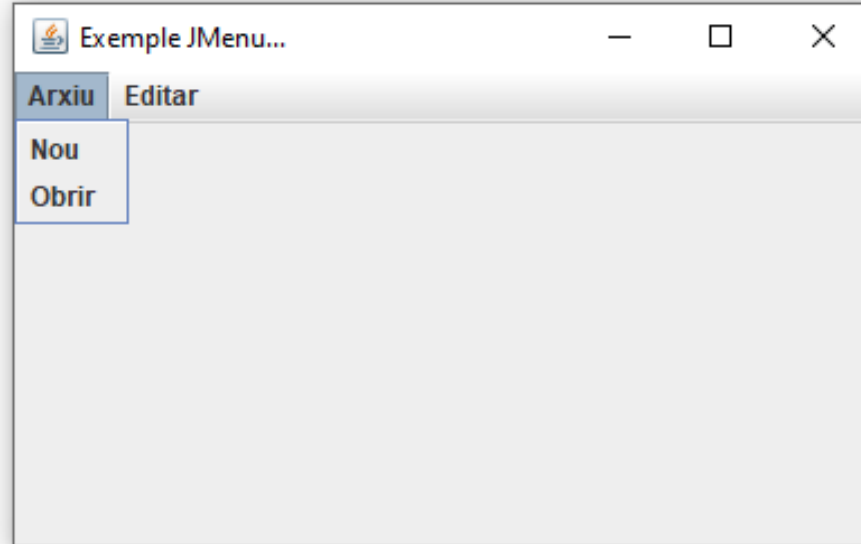
    public Exemple11() {

        menu=new JMenuBar();
        JMenu arxiu=new JMenu("Arxiu");
        JMenuItem nou=new JMenuItem("Nou");
        arxiu.add(nou);
        JMenuItem obrir=new JMenuItem("Obrir");
        arxiu.add(obrir);

        JMenu editor=new JMenu("Editor");
        JMenuItem copiar=new JMenuItem("Copiar");
        editor.add(copiar);
        JMenuItem pegar=new JMenuItem("Pegar");
        editor.add(pegar);
        menu.add(arxiu);
        menu.add(editor);

        this.add(menu,BorderLayout.NORTH);
        this.setTitle("Exemple JMenu...");
        this.setSize(400, 250);
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setVisible(true);
    }

    public static void main(String[] args) {
        new Exemple11();
    }
}
```



JTabbedPane

Constructors

Constructor	Description
<code>JTabbedPane()</code>	Creates an empty <code>TabbedPane</code> with a default tab placement of <code>JTabbedPane.TOP</code> .
<code>JTabbedPane(int tabPlacement)</code>	Creates an empty <code>TabbedPane</code> with the specified tab placement of either: <code>JTabbedPane.TOP</code> , <code>JTabbedPane.BOTTOM</code> , <code>JTabbedPane.LEFT</code> , or <code>JTabbedPane.RIGHT</code> .
<code>JTabbedPane(int tabPlacement, int tabLayoutPolicy)</code>	Creates an empty <code>TabbedPane</code> with the specified tab placement and tab layout policy.

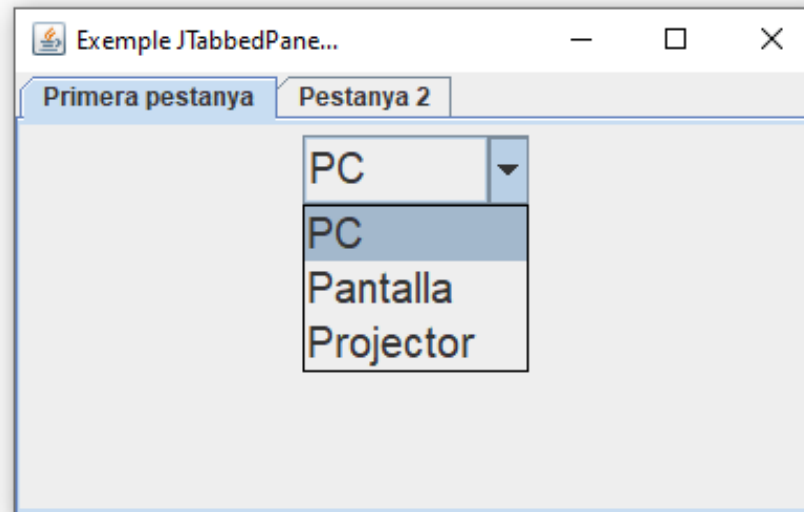
JTabbedPane

```
public class Exemple12 extends JFrame{
    private JTabbedPane panel1,panel2;
    private JComboBox cb;
    private JTextArea texto;
    public Exemple12() {

        panel1=new JTabbedPane();
        //AFEGIM COMBOBOX
        cb=new JComboBox();
        //CANVIEM EL TIPUS DE FONT
        cb.setFont(new Font("Arial",Font.PLAIN,20));
        cb.addItem("PC");
        cb.addItem("Pantalla");
        cb.addItem("Projector");
        JPanel panel2=new JPanel();
        panel2.add(cb);
        //AFEGIM PANEL CON JCOMBOBOX A LA PRIMERA PESTANYA
        panel1.addTab("Primera pestanya", panel2);
        //AFEGIM JTextArea

        texto=new JTextArea("Escriu el teu nom",8,20);
        //AFEGIM JTEXTAREA DIRECTAMENT A LA SEGONA PESTANYA
        panel1.addTab("Pestanya 2",texto);
        //AFEGIM AL JTABBEDPANE AL JFRAME

        this.add(panel1);
        this.setTitle("Exemple JTabbedPane...");
        this.setSize(400, 250);
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setVisible(true);
    }
    public static void main(String[] args) {
        new Exemple12();
    }
}
```



JSplitPane

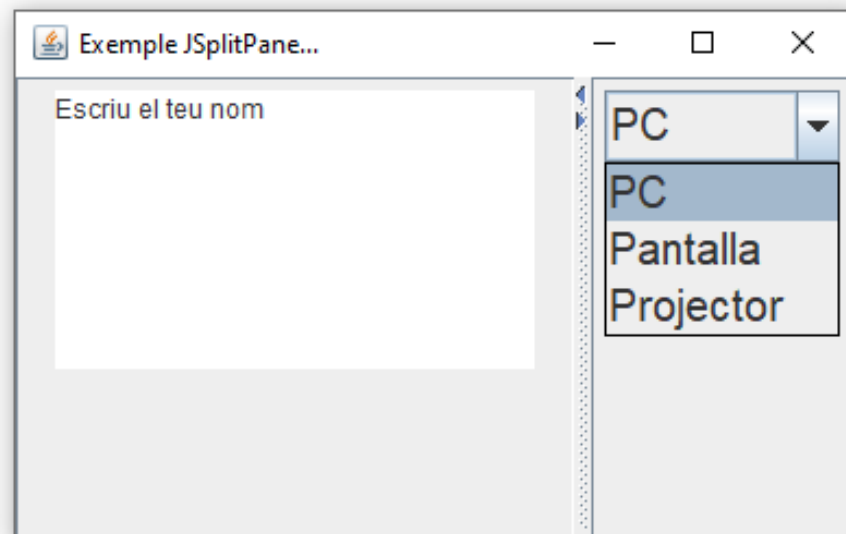
Constructors

Constructor	Description
<code>JSplitPane()</code>	Creates a new JSplitPane configured to arrange the child components side-by-side horizontally, using two buttons for the components.
<code>JSplitPane(int newOrientation)</code>	Creates a new JSplitPane configured with the specified orientation.
<code>JSplitPane(int newOrientation, boolean newContinuousLayout)</code>	Creates a new JSplitPane with the specified orientation and redrawing style.
<code>JSplitPane(int newOrientation, boolean newContinuousLayout, Component newLeftComponent, Component newRightComponent)</code>	Creates a new JSplitPane with the specified orientation and redrawing style, and with the specified components.
<code>JSplitPane(int newOrientation, Component newLeftComponent, Component newRightComponent)</code>	Creates a new JSplitPane with the specified orientation and the specified components.

JSplitPane

```
import java.awt.Font;

public class Exemple13 extends JFrame {
    private JSplitPane panel;
    JPanel panel1, panel2;
    private JComboBox cb;
    private JTextArea texto;
    public Exemple13() {
        cb=new JComboBox();
        //CANVIEM EL TIPUS DE FONT
        cb.setFont(new Font("Arial",Font.PLAIN,20));
        cb.addItem("PC");
        cb.addItem("Pantalla");
        cb.addItem("Projector");
        panel2=new JPanel();
        panel2.add(cb);
        //AFEGIM JTextArea
        texto=new JTextArea("Escriu el teu nom",8,20);
        panel1=new JPanel();
        panel1.add(texto);
        panel=new JSplitPane(JSplitPane.HORIZONTAL_SPLIT,panel1, panel2);
        panel.setOneTouchExpandable(true);
        this.add(panel);
        this.setTitle("Exemple JSplitPane...");
        this.setSize(400, 250);
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setVisible(true);
    }
    public static void main(String[] args) {
        new Exemple13();
    }
}
```



JToolBar

Constructors

Constructor	Description
<code>JToolBar()</code>	Creates a new tool bar; orientation defaults to HORIZONTAL.
<code>JToolBar(int orientation)</code>	Creates a new tool bar with the specified orientation.
<code>JToolBar(String name)</code>	Creates a new tool bar with the specified name.
<code>JToolBar(String name, int orientation)</code>	Creates a new tool bar with a specified name and orientation.

JToolBar

```
import java.awt.BorderLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JTextArea;
import javax.swing.JToolBar;
public class Exemple14 {

    public Exemple14() {
        JFrame ventanaPrincipal = new JFrame("JToolBar.....");
        JTextArea componentePrincipal = new JTextArea(25, 80);
        JToolBar toolBar = getToolBar();
        ventanaPrincipal.add(componentePrincipal);
        ventanaPrincipal.add(toolBar, BorderLayout.NORTH);
        ventanaPrincipal.pack();
        ventanaPrincipal.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ventanaPrincipal.setLocationRelativeTo(null);
        ventanaPrincipal.setVisible(true);
    }

    private JToolBar getToolBar() {
        JToolBar barraBotones = new JToolBar();
        barraBotones.add(new JButton("Botó 1"));
        barraBotones.add(new JButton("Botó 2"));
        barraBotones.add(new JButton("Botó 3"));
        barraBotones.add(new JButton("Botó 4"));
        barraBotones.add(new JButton("Botó 5"));
        return barraBotones;
    }

    public static void main(String[] args) {
        new Exemple14();
    }
}
```

