# GUI en Java: conceptes bàsics

## Introducció a la programació amb GUI

La interfase gràfica d'usuari (GUI) és un aspecte molt important de qualsevol aplicació. Un bon disseny permet als usuaris obtenir el màxim rendiment de l'aplicació.

Les principals biblioteques de classes de Java per a la construcció de GUI són **AWT** (*Abstract Window Toolkit*), **SWT** i **Swing**.

Els components de AWT utilitzen utilitats nadiues de les plataformes sobre les quals corren les aplicacions. Per aquest motiu, les aplicacions dissenyades amb AWT tenen una aparença similar a totes les de la plataforma. Per contra, això fa que l'aparença sigui depenent de la plataforma i pot ser que no es visualitzi bé en canviar de plataforma.

Per evitar la dependència de la plataforma, cal utilitzar les **JFC** (***Java Foundation Classes***). En concret, per al disseny gràfic, convé utilitzar les classes de la biblioteca **Swing**.

Les JFC consten de diversos grups de classes, els principals dels quals són:

| | |
|---|---|
| **AWT** | Components del AWT del JDK 1.1.2 i algunes millores posteriors |
| **Java2D** | Classes gràfiques en dues dimensions |
| **Accessibilitat** | Classes per a accessibilitat per a persones amb disminució |
| **Arrosegar i deixar** | *Drag and Drop* Nova generació dels JavaBeans |
| **Swing** | Extensió del AWT amb components JComponent i altres |

L'estructura bàsica de classes consta de **components** i **contenidors**. Els components són controls bàsics (quadres de text, llistes de verificació, botons, etc.). Els contenidors contenen components, la disposició dels quals al seu interior es controla a través de **layouts**. Els contenidors són, a la seva vegada, components i es poden ubicar també dintre d'altres contenidors.

Les accions de l'usuari sobre els components generen **esdeveniments** o **events** (*events*). És tasca del programador escriure els mètodes de gestió dels esdeveniments i associar-los al component o contenidor corresponent. A la biblioteca AWT, tots els components són instàncies de la classe **Component** o d'una derivada d'ella, mentre que els contenidors són instàncies de la classe **Container** o d'una classe derivada d'ella.

El model d'esdeveniments de Java es basa en dues categories d'objectes: els objectes observadors (***listeners***) que gestionen els esdeveniments i disposen de mètodes adients per respondre als mateixos i els objectes font d'esdeveniments, els quals han d'enregistrar els observadors.

Els observadors implementen la interface Listener.

La taula següent resumeix els components del AWT i els esdeveniments específics de cada un d'ells, tot i que cal tenir present que cada component pot produir també els esdeveniments que produeixen les seves superclasses.

| Component | Eventos generados | Significado |
|---|---|---|
| Button | ActionEvent | Clicar en el botón |
| Checkbox | ItemEvent | Seleccionar o deseleccionar un item |
| CheckboxMenuItem | ItemEvent | Seleccionar o deseleccionar un item |
| Choice | ItemEvent | Seleccionar o deseleccionar un item |
| Component | ComponentEvent | Mover, cambiar tamaño, mostrar u ocultar un componente |
| | FocusEvent | Obtener o perder el focus |
| | KeyEvent | Pulsar o soltar una tecla |
| | MouseEvent | Pulsar o soltar un botón del ratón; entrar o salir de un componente; mover o arrastrar el ratón (tener en cuenta que este evento tiene dos Listener) |
| Container | ContainerEvent | Añadir o eliminar un componente de un container |
| List | ActionEvent | Hacer doble click sobre un item de la lista |
| | ItemEvent | Seleccionar o deseleccionar un item de la lista |
| MunuItem | ActionEvent | Seleccionar un item de un menú |
| Scrollbar | AdjustementEvent | Cambiar el valor de la scrollbar |
| TextComponent | TextEvent | Cambiar el texto |
| TextField | ActionEvent | Terminar de editar un texto pulsando Intro |
| Window | WindowEvent | Acciones sobre una ventana: abrir, cerrar, iconizar, restablecer e iniciar el cierre |

Cada objecte que pot rebre una interacció amb l'usuari, pot produir un esdeveniment per al seu control. Per al seguiment de l'esdeveniment, l'objecte enregistra un o més objectes observadors (listeners) que implementen la interface específica de l'esdeveniment.

La interface declara els mètodes de control de l'esdeveniment. Cal definir la funcionalitat de tots aquests mètodes. Sovint no ens interessa definir tots els mètodes, perquè només estem interessats en una de les funcionalitats. Per a això disposem dels adaptadors (Adapter). Es tracta de classes que defineixen els mètodes buits, sense cap funcionalitat. Derivant de les classes adaptadores, només ens cal redefinir els mètodes que ens interessin.

## Hola món, amb Swing

L'aplicació presenta una finestra amb un missatge de salutació. L'aplicació respon al botó de tancar la finestra.

```
/**
 * HolaMon.java
 * Hola mon amb Swing
 * @author Jose Moreno
 * @version
 */
import javax.swing.*;
public class HolaMon extends JFrame
{
    public static void main( String argv[] ) {
    new HolaMon();
    }
    HolaMon()
    {
        JLabel missatge= new JLabel( "Hola Món!" );
        setDefaultCloseOperation( WindowConstants. DISPOSE_ON_CLOSE );
        getContentPane().add( missatge, "Center" );
        setSize( 200, 100); //definir tamany
        // alternativament, podem definir automàticament el tamany en funció
        // del seu contingut amb
        //pack();
        setVisible( true );  //ver visible la finestra
    }
}
```

## Layouts: BorderLayout

Exemple d'utilització del BorderLayout.

```
/**
 * Layout1.java
 * Exemple d'ús de BorderLayout
 * @author Jose Moreno
 * @version
 */
import java.awt.*;
import java.awt.event.*;
class Layout1 extends Frame
{
    private Label [] etiq;
    public Layout1()
    {
        //crear array de referències a JLabel
        etiq = new Label[5];
        //inicialitzar etiquetes
        etiq[0]=new Label( "Etiqueta A" );
        etiq[0].setAlignment( Label.CENTER );
        etiq[1]=new Label( "Etiqueta B" );
        etiq[1].setAlignment( Label.CENTER );
        etiq[1].setBackground( Color.YELLOW );
        etiq[2]=new Label( "Etiqueta C" );
        etiq[2].setAlignment( Label.CENTER );
        etiq[2].setBackground( Color.BLUE );
        etiq[2].setForeground( Color.WHITE );
        etiq[3]=new Label( "Etiqueta D" );
        etiq[3].setAlignment( Label.CENTER );
        etiq[3].setBackground( Color.CYAN );
        etiq[4]=new Label( "Etiqueta E" );
        etiq[4].setAlignment( Label.CENTER );
        etiq[4].setBackground( Color.BLACK );
        etiq[4].setForeground( Color.PINK );
        //afegir components al panell arrel (JRootPanel) de la finestra
        add( etiq[0], BorderLayout.NORTH );
        add( etiq[1], BorderLayout.SOUTH );
        add( etiq[2], BorderLayout.EAST );
        add( etiq[3], BorderLayout.WEST );
        add( etiq[4], BorderLayout.CENTER );
        //afegir adaptador per a l'esdeveniment de tancament de la finestra
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent e)
            {
                dispose();
                System.exit(0);
            }
        });
    }

    public static void main(String args[])
    {
        Layout1 mainFrame = new Layout1();
        mainFrame.setSize(300, 200);
        mainFrame.setTitle("Prova del BorderLayout");
        mainFrame.setLocationRelativeTo(null);  //centrar en la pantalla
        mainFrame.setVisible(true);
    }
}
```

## Aplicació amb botons de resposta

L'aplicació presenta una finestra amb un missatge o pregunta i dos botons amb textos configurables. Per defecte, els botons presenten els textos "Yes", "No".

El missatge i els textos dels botons s'admeten per la línia d'ordres.

```
/**
 * MostraPregunta.java
 * @author Jose Moreno
 * @version
 */
import java.awt.*;            // classes AWT
import javax.swing.*;         // Components i classes Swing
import javax.swing.border.*;        // Borders per a components Swing
import java.awt.event.*;    // Maneig d'events
public class MostraPregunta {
     public static void main(String[] args) {
          //Crear els components
          JLabel msgLabel = new Jlabel();        // Component per mostrar la pregunta
          JButton yesButton = new Jbutton();   // Botó per a la resposta afirmativa
          JButton noButton = new Jbutton();    // Idem resposta negativa
          //Propietats dels components
          msgLabel.setText(args[0]);             // Missatge a mostrar
          msgLabel.setBorder(new EmptyBorder(10,10,10,10));  // Marge de 10 pixels
          yesButton.setText((args.length >= 2) ? args[1]:"Sí");  // Text per al botó afirmatiu
          noButton.setText ((args.length >= 3) ? args[2]:"No"); // Text per al botó negatiu
          //crear el contenidor per als botons
          JPanel buttonBox = new JPanel();
          //crear finestra principal
          JFrame win = new JFrame("Message");
          //Especificar els Layouts dels contenidors
          win.getContentPane().setLayout(new BorderLayout());
          buttonBox.setLayout(new FlowLayout());
          //Afegir els botons al seu contenidor
          buttonBox.add(yesButton);
          buttonBox.add(noButton);
          //Afegir l'etiqueta (missatge) a la finestra principal, centrada
          win.getContentPane().add(msgLabel, "Center");
          //Afegir el panell de botons (missatge) a la finestra principal, a baix
          win.getContentPane().add(buttonBox, "South");
          //Afegir controladors d'events per als botons
          yesButton.addActionListener(new ActionListener() {  // classe interna
          // mètode que s'invoca en clica al botó yes
          public void actionPerformed(ActionEvent e) { System.exit(0); }
     });
     noButton.addActionListener(new ActionListener() { // classe interna
          // mètode que s'invoca en clica al botó no
          public void actionPerformed(ActionEvent e) { System.exit(1); }
     });
     //Ajustar el tamany de la finestra en funció del seu contingut
     win.pack();
     //Fer visible la finestra
     win.show();
     }
}
```

## Aplicació amb botons i camp de text

El següent exemple és ja una aplicació completa que respon al tancament de la finestra i que conté dos botons i un camp de text. El camp de text és editable. A més, en clicar cada botó, es modifica el contingut del camp de text.

```
/**
 * TextBotonsAp.java
 * Exemple de finestra amb dos botons i text
 * @author Jose Moreno
 * @version
 */
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
class PanellBotoText extends JPanel
{
    JButton boto1 = new JButton( "Botó 1" );
    JButton boto2 = new JButton( "Botó 2" );
    JTextField text = new JTextField( 20 );
    public PanellBotoText()
    {
        ActionListener al = new ActionListener() {
            public void actionPerformed( ActionEvent evt ) {
                String nomBoto = ( (JButton)evt.getSource()).getText();
                text.setText( "Premut "+nomBoto );
            }
        };
        //definir text alternatiu del camp de text
        text.setToolTipText( "Camp de text" );
        //afegir camp de text al panell
        add( text );
        //afegir observador a boto1
        boto1.addActionListener( al );
        //definir text alternatiu boto1
        boto1.setToolTipText( "Botó 1" );
        //afegir boto1 al panell
        add( boto1 );
        //afegir observador a boto2
        boto2.addActionListener( al );
        //definir text alternatiu boto2
        boto2.setToolTipText( "Botó 2" );
        //afegir boto2 al panell
        add( boto2 );
    }
}
// classe principal
public class TextBotonsAp
{
    public static void main( String args[] ) {
        JFrame finestra = new JFrame( "Finestra amb dos botons i text" );
        // afegir observador tancament finestra
        finestra.addWindowListener( new WindowAdapter() {
            public void windowClosing( WindowEvent evt ){
                System.exit( 0 );
            }
        } );
        //afegir panell contenidor dels botons i camp de text
        finestra.getContentPane().add( new PanellBotoText(), BorderLayout.CENTER );
        //definir dimensions finestra
        finestra.setSize( 300,100 );
```

```
        //fer visible la finestra
        finestra.setVisible( true );
    }
}
```

## Applet senzill amb JApplet

Aquest senzill applet només mostra un text. Es pot incrustar l'etiqueta de applet (la que aniria al codi html) per facilitar-ne la prova. D'aquesta manera, es pot executar amb AppletViewer directament l'arxiu JAppletExemple1.java.

```
/**
 * JAppletExemple1.java
 * @author Jose Moreno
 * @version
 */
// Incrustem l'etiqueta applet per a Appletviewer.
// <applet code=JAppletExemple1 width=100 height=50>
// </applet>
import javax.swing.*;
import java.awt.*;
public class JAppletExemple1 extends JApplet {
    public void init() {
        getContentPane().add(new JLabel("Applet!"));
    }
    public void paint(Graphics g) {

    }
}
```

## Aplicació senzilla amb menú (Swing)

Aquest exemple mostra com fer una aplicació amb interfície gràfica d'usuari utilitzant la biblioteca Swing. L'aplicació disposa d'un menú elemental.

La classe guiFrame defineix la interfície gràfica.

```
/**
 * GuiFrame.java
 * Aplicació GUI amb JFC
 * @author Jose Moreno
 * @version
 */
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class GuiFrame extends JFrame {
    public guiFrame() {
        JMenuBar menuBar = new JMenuBar();
        JMenu menuFile = new JMenu();
        JMenuItem menuFileExit = new JMenuItem();
        menuFile.setText("Arxiu");
        menuFileExit.setText("Sortir");
        // Afegir observador per a l'opció de menu sortir
        menuFileExit.addActionListener
        (
            new ActionListener() {
                public void actionPerformed(ActionEvent e) {
```

```
                                GuiFrame.this.windowClosed();
                        }
                }
        );
        menuFile.add(menuFileExit);
        menuBar.add(menuFile);
        setTitle("GUI amb menú");
        setJMenuBar(menuBar);
        setSize(new Dimension(400, 400));
        // Afegir observador de la finestra per tancar-la
        this.addWindowListener
        (
                new WindowAdapter() {
                        public void windowClosing(WindowEvent e) {
                                GuiFrame.this.windowClosed();
                        }
                }
        );
    }
    /**
     * windowClosed()
     * procediment per tancar l'aplicació
     */
    protected void windowClosed() {
        // TODO: Comprovar que es pot tancar l'aplicació
        // Acabar l'execució de l'aplicació
        System.exit(0);
    }
}
```

La interfície gràfica es carrega des de la classe principal continguda a Gui.java.

```
/**
 * Gui.java
 * Aplicació GUI amb JFC contenint menús. Principal.
 * @author Jose Moreno
 * @version
 */
public class Gui {
    public static void main(String[] args) {
        // instanciar la finestra
        GuiFrame finestra = new GuiFrame();
        // mostrar
        finestra.setVisible(true);
    }
}
```