UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Las aplicaciones Swing utilizan los siguientes componentes comúnmente:

1. un *frame*, o ventana principal (JFrame)
2. un *panel*, llamado *pane* (JPanel)
3. un *button* (JButton)
4. un *label* (JLabel)

Un frame es un “contenedor de nivel superior” (top-level container). Existe para brindar un lugar para “pintarse” a sí mismos a los demás componentes. Otros containers top-level usuales son dialogs (JDialog) y applets (Japplet).

El panel es un *container intermedio*. Su único propósito es simplificar el posicionamiento del botón y label dentro de sí. Otros container Swing intermedios como scroll panes (JScrollPane) y tabbed panes (JTabbedPane), juegan un rol más interactivo en un programa GUI.

UNTDF
Universidad Nacional de Tierra del Fuego,
Antártida e Islas del Atlántico Sur

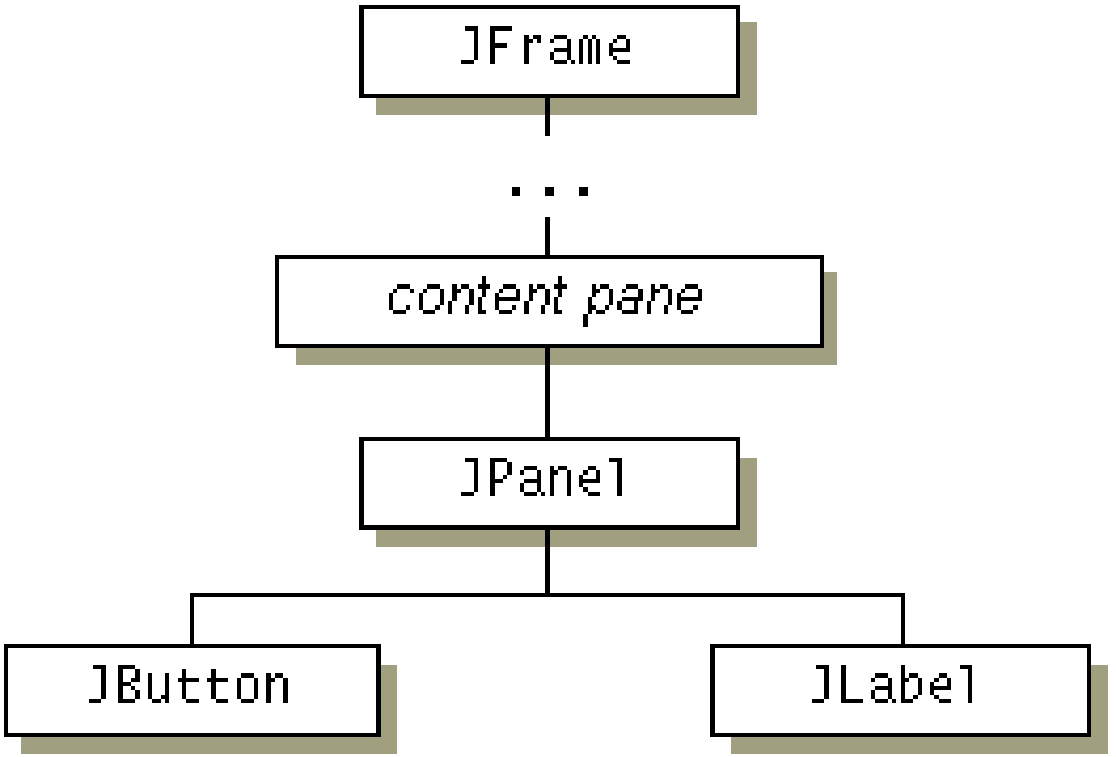
Componentes Swings y Jerarquía

El botón y el label (etiqueta) son componentes atómicos, entidades que presentan bits de información al usuario. Usualmente también reciben input de los usuarios. El API Swing provee muchos componentes atómicos, incluyendo combo boxes (JComboBox), text fields (JTextField), y tables (JTable).

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

El Diagrama muestra una Jerarquía de contención (containment) para la ventana mostrada por una aplicación Swing. Muestra cada container creado o usado por el programa con el componente que éste contiene.



UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Cada container top-level indirectamente contiene un container intermedio conocido como *content pane*. Como regla, el content pane contiene directa o indirectamente, todos los componentes visibles en la ventana GUI a excepción de un menú bar del top level container que está fuera del content pane.

Para agregar un componente a un container se usa un método add, que toma como argumento el componente a ser añadido y a veces como argumento adicional, la información de layout.

```
frame = new JFrame(...);
button = new JButton(...);
label = new JLabel(...);
pane = new JPanel();
pane.add(button);
pane.add(label);
frame.getContentPane().add(pane, BorderLayout.CENTER);
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

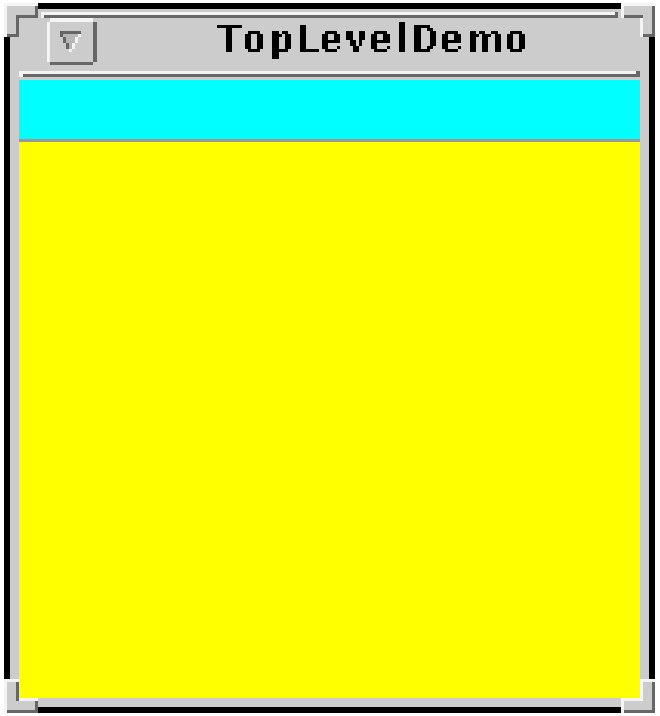
Top-Level Containers

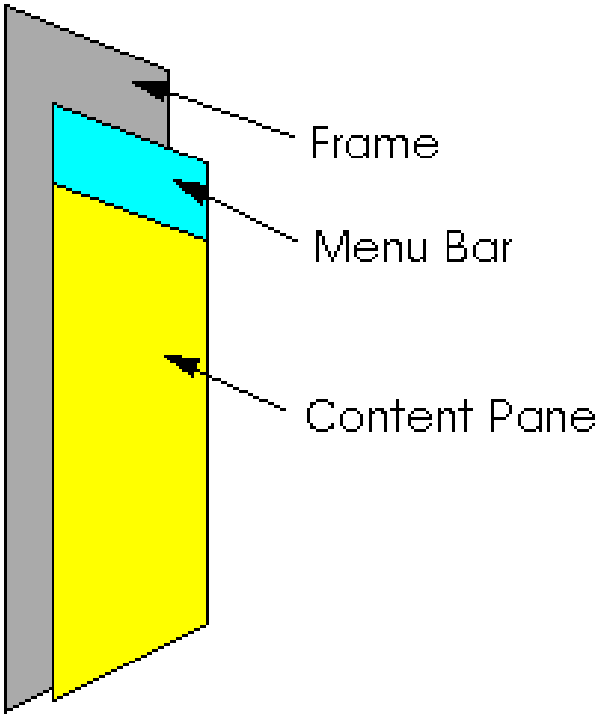
Swing brinda las siguientes clases de top-level container: JFrame, JDialog y JApplet.

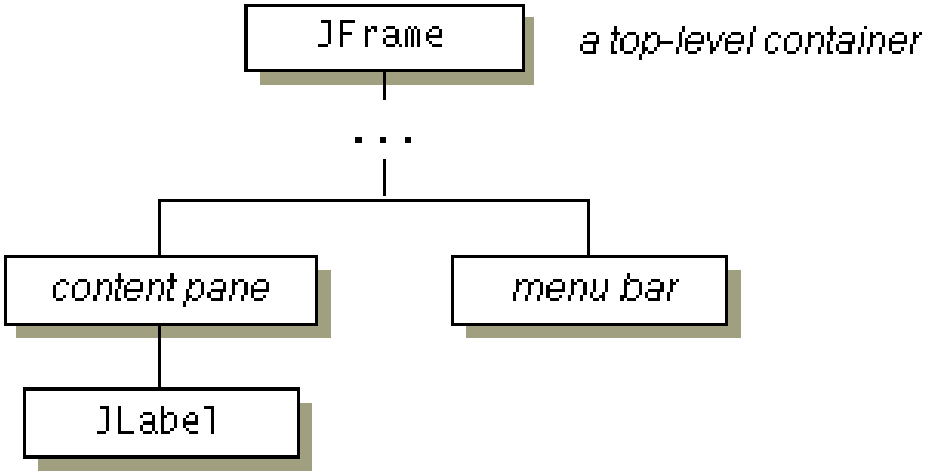
- Para aparecer sobre la pantalla, cada componente GUI debe ser parte de una jerarquía de contención. Cada una tiene un top-level container como raíz.
- Cada top-level container tiene un content pane, el cual contiene los componentes visibles en la interfaz del top-level container.
- Opcionalmente se puede agregar un menu bar a un top-level container. Se posiciona dentro de él, pero fuera del content pane.

UNTDF
Universidad Nacional de Tierra del Fuego,
Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía







UNTDF
Universidad Nacional de Tierra del Fuego,
Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Una aplicación con una GUI basada en Swing tiene como mínimo una jerarquía de contención con JFrame como su raíz.

Por ejemplo, un JFrame que presente un JDialog tiene dos jerarquías de contención.

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Agregar componentes a un Content Pane

frame.getContentPane().add(yellowLabel, BorderLayout.CENTER);

Se puede personalizar el content pane, por ejemplo con el layout manager o agregando un borde.

```
JPanel contentPane = new JPanel();
    contentPane.setLayout(new BorderLayout());
    contentPane.setBorder(someBorder);
    contentPane.add(someComponent,
     BorderLayout.CENTER);
    contentPane.add(anotherComponent,
     BorderLayout.PAGE_END);
    topLevelContainer.setContentPane(contentPane);
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Agregando un Menu Bar

Todos los top-level continers pueden en teoría, tener un menu bar. En la práctica, solo aparecen en frames y applets.
Para agregar un menu bar a un frame o applet, se crea un objeto JMenuBar, se lo llena con menúes y luego se llama setJMenuBar.

```
frame.setJMenuBar(cyanMenuBar);
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Intermediate Swing Containers

No son top-level containers, pero existen para contener otros componentes. Swing provee muchos containers intermedios de propósito general:

Panel	Es el más flexible y más frecuentemente usado. Implementado con la clase JPanel, agregan casi ninguna otra funcionalidad más que la que tienen todos los objetos JComponent. Usados frecuentemente para agrupar componentes, sea porque estén relacionados o porque facilitan su layout. Pueden usar cualquier layout manager y se les puede dar un borde fácilmente. Los content panes de los top-level containers se implementan frecuentemente como instancias de JPanel.
Scroll Pane	Provee barras de desplazamiento alrededor de componentes grandes o expandibles.
Split Pane	Muestra dos componentes en una cantidad fija de espacio, permitiendo al usuario ajustar la cantidad de espacio asignado a cada uno.
Tabbed Pane	Contiene múltiples componentes pero muestra solo uno a la vez. El usuario puede fácilmente cambiar entre componentes.
Tool Bar	Contiene un grupo de componentes (usualmente botones) en una fila o columna, opcionalmente permite al usuario arrastrar la barra de herramientas a una ubicación diferente.

Componentes Swings y Jerarquía

Containers intermedios más especializados:

Internal Frame	Se parece a un frame y tiene mucho de su API, pero debe aparecer dentro de otra ventana.
Layered Pane	Provee una 3° dimensión, profundidad, para posicionar componentes. Se indica la posición y tamaño de cada componente. Un tipo de estos, el desktop pane, está diseñado para contener y manejar frames internos.
Root Pane	Provee un soporte “detrás de la escena” a los top-level containers.

Componentes Atómicos

Existen solo para presentar y a veces aceptar información.
Todos los componentes atómicos descienden de la JComponent Class. Por ello soportan características estándar como tool tips y borders.
Los siguientes componentes existen para aceptar inputs del usuario:

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

JButton, JCheckBox, JRadioButton	Proveen implementaciones de botones fáciles de usar e implementar.
JComboBox	Proveen combo boxes tanto editables como no editables –botones que brindan un menú de posibilidades.
JList	Muestran un grupo de ítems que el usuario puede elegir.
JMenu	Incluyen implementaciones de barra de menú, menú e ítems de menú, incluyendo ítems de menú especializados como los ítems de menú con check box.
JSlider	Permite al usuario elegir uno de un rango continuo de valores.
JTextField	Permite que el usuario ingrese una línea simple de texto.

Algunos existen sólo para dar información:

JLabel	Presenta algún texto, un icono o ambos.
JProgressBar	Muestra el progreso hacia una meta.
ToolTip	Muestra una pequeña ventana que describe otro componente.

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Estos proveen información formateada y un modo de editarla:

JColorChooser	Una UI para elegir colores; puede ser usada dentro o fuera de un JDialog.
JFileChooser	Una UI para elegir archivos y directorios.
JTable	Un componente extremadamente flexible que muestra datos en un formato de grilla.
Text Support	Un framework incluyendo todo desde componentes de texto simple, hasta un cjto. completo y extensible para construir editores de texto.
JTree	Un componente que muestra datos jerárquicos.

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Manejador de Eventos

Cada vez que un usuario tipea un carácter o aprieta el botón del mouse, ocurre un evento. Para notificarse del evento cada objeto tiene que implementar la interface apropiada que se registra como un *event listener* en el apropiado *event source*.

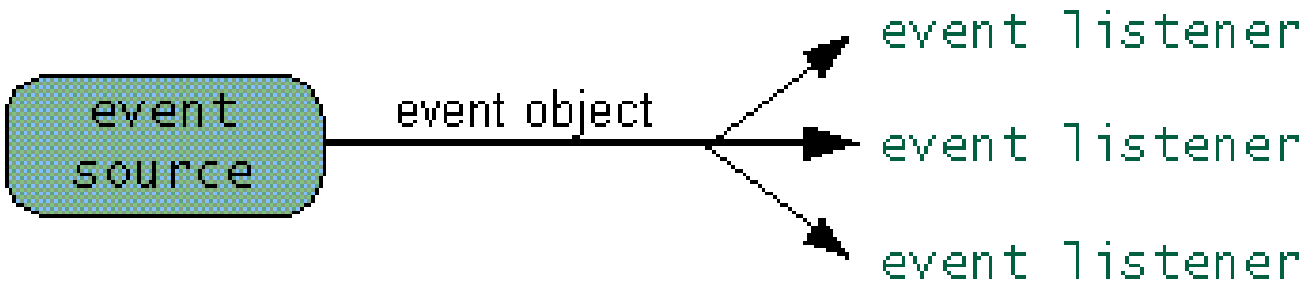
Eventos que pueden generar muchos tipos de eventos:

Actos que resultan en un evento	Tipo de Listener
Usuario hace click en botón, presiona Return mientras tipea en un text field, o elige un ítem de menú.	ActionListener
Usuario cierra un frame (main window)	WindowListener
Usuario presiona el botón del mouse mientras el cursor esta sobre un componente.	MouseListener
Usuario mueve el mouse sobre un componente	MouseMotionListener
Componente se vuelve visible	ComponentListener
Componente toma el foco del teclado	FocusListener
Tabla o lista de selección cambian	ListSelectionListener

UNTDF
Universidad Nacional de Tierra del Fuego,
Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Cada evento se representa por un objeto que da información acerca del evento e identifica la fuente del mismo.
Cada fuente de eventos puede tener muchos listeners registrados.
A su vez un listener simple, también puede registrarse con muchas fuentes de eventos



UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Cómo implementar un manejador de eventos
Cada manejador de eventos requiere el siguiente código:
Ejemplo de la declaración de la clase event handler
1. En la declaración para la clase event handler, el código que especifica que la clase implementa un listener interface.
public class MyClass implements ActionListener {
2. Código que registra una instancia del event handler como un listener sobre uno o más componentes.
someComponent.addActionListener(instanceOfMyClass);
3.Código que implementa los métodos en la interface listener.

```
public void actionPerformed(ActionEvent e) {
     //código que reacciona a la acción...
 }
```

UNTDF
Universidad Nacional de Tierra del Fuego,
Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Catálogo de componentes Swing

Swing incluye diferentes tipos de botones. Botones, checkboxes, radio buttons y menu items heredan de AbstractButton.

```
import java.awt.Container;
import java.awt.FlowLayout;

import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JToggleButton;
import javax.swing.border.TitledBorder;
import javax.swing.plaf.basic.BasicArrowButton;

public class Buttons extends JFrame
{
 JButton jb = new JButton("JButton");
 BasicArrowButton
     up = new BasicArrowButton(BasicArrowButton.NORTH),
     down = new BasicArrowButton(BasicArrowButton.SOUTH),
     right = new BasicArrowButton(BasicArrowButton.EAST),
     left = new BasicArrowButton(BasicArrowButton.WEST);
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

```
public Buttons()
{
    Container cp = getContentPane();
    cp.setLayout(new FlowLayout());
    cp.add(jb);
    cp.add(new JToggleButton("JToggleButton"));
    cp.add(new JCheckBox("JCheckBox"));
    cp.add(new JRadioButton("JRadioButton"));
    JPanel jp = new JPanel();
    jp.setBorder(new TitledBorder("Directions"));
    jp.add(up);
    jp.add(down);
    jp.add(left);
    jp.add(right);
    cp.add(jp);
 }
}
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Button groups

Si se quiere que un grupo de botones se comporten de una cierta forma, se los debe agregar a un buttom group; cualquier Abstract Buttom puede ser agregado a un Buttom Group.

Para evitar repetir código, este ej. usa “reflection” para generar distintos tipos de botones. Esto se ve en makeBPanel() que crea un grupo de botones y un JPanel. El segundo argumento es un array de String.

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

```
import java.awt.Container;
import java.awt.FlowLayout;
import java.lang.reflect.Constructor;

import javax.swing.AbstractButton;
import javax.swing.ButtonGroup;
import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JToggleButton;
import javax.swing.border.TitledBorder;

public class ButtonGroups extends JFrame
{
 static String[] ids = {"Ariel", "Martín", "Diego",
"Jorge", "Gustavo", "Sergio"};
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

```
static JPanel makeBPanel(Class bClass, String[] ids) {
ButtonGroup bg = new ButtonGroup();
JPanel jp = new JPanel();
String title = bClass.getName();
title = title.substring(title.lastIndexOf('.') + 1);
jp.setBorder(new TitledBorder(title));
for(int i=0; i<ids.length; i++) {
    AbstractButton ab = new JButton("failed");
    try {
        Constructor ctor = bClass.getConstructor(
            new Class[] {String.class});
        ab =(AbstractButton)ctor.newInstance(
            new Object[]{ids[i]});
    } catch(Exception ex) {
        System.err.println("no puedo crear"+ bClass);
    }
    bg.add(ab);
    jp.add(ab);
}
return jp;
}
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

```
public ButtonGroups()
{
    Container cp = getContentPane();
    cp.setLayout( new FlowLayout());
    cp.add(makeBPanel(JButton.class,ids));
    cp.add(makeBPanel(JToggleButton.class,ids));
    cp.add(makeBPanel(JCheckBox.class,ids));
    cp.add(makeBPanel(JRadioButton.class,ids));
}
public static void main(String[] args)
{
    ButtonGroups bg= new ButtonGroups();
    bg.setSize(600,400);
    bg.setVisible(true);
}
}
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

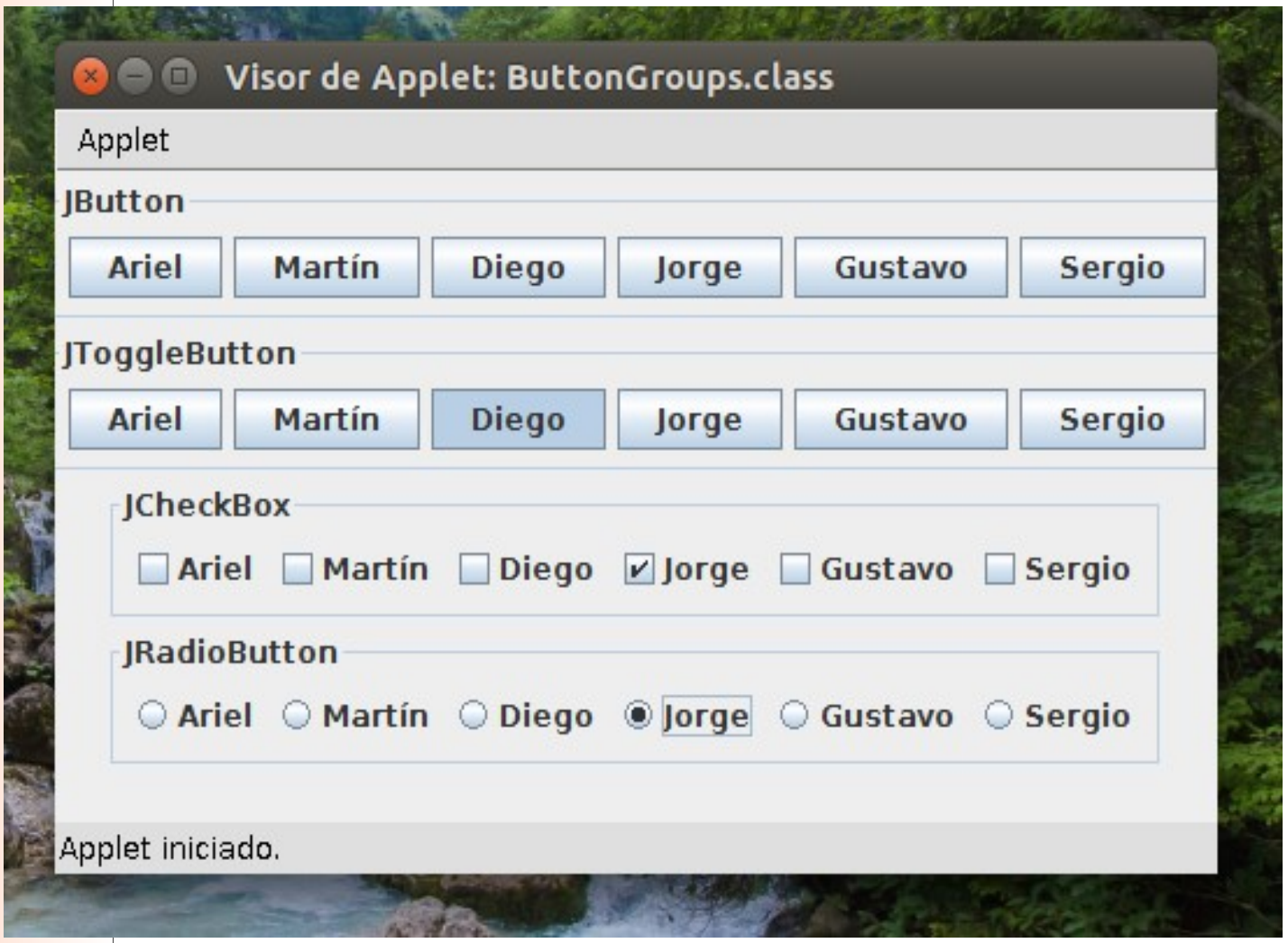
El título para el borde de cada grupo se toma del nombre de la clase.

El método getConstructor() produce un objeto Constructor que toma el array de argumentos de los tipos en el array Class pasados al getConstructor(). Todo lo que se hace es llamar newInstance(), pasándola a un array de objetos conteniendo los argumentos actuales; en este caso el String del array ids.

Para obtener comportamiento exclusivo con botones, se crea un button group y se agrega cada botón que se quiere se comporte como el grupo.

UNTDF
Universidad Nacional de Tierra del Fuego,
Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía



UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Icons

Se puede usar un Icon dentro de un JLabel o cualquier cosa que herede de AbstractButton.

Tool Tips

Todas las clases que se usan cuando se crean interfaces se derivan de JComponent, que contiene un método llamado setToolTipText(String). De modo que para cualquier componente que se coloque en el form, se escribe lo siguiente:

jc.setToolTipText(“Un Tip”);

y cuando el mouse permanezca sobre el componente por un período de tiempo, aparecerá una caja delgada con el texto donde está el mouse.

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

TextFields

```
import java.awt.Container;
import java.awt.FlowLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JTextField;
import javax.swing.event.DocumentEvent;
import javax.swing.event.DocumentListener;
import javax.swing.text.AttributeSet;
import javax.swing.text.BadLocationException;
import javax.swing.text.PlainDocument;
public class TextField extends JFrame {
  JButton
        B1 = new JButton("Get Text"),
        B2 = new JButton("Set Text");
  JTextField
        t1 = new JTextField(30),
        t2 = new JTextField(30),
        t3 = new JTextField(30);
    String s = new String();
    UpperCaseDocument Ucd = new UpperCaseDocument();
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

```
public TextField() {
        t1.setDocument(Ucd);
        Ucd.addDocumentListener(new T1());
        B1.addActionListener(new B1());
        B2.addActionListener(new B2());
        DocumentListener dl = new T1();
        t1.addActionListener(new T1A());

        Container cp = getContentPane();
        cp.setLayout(new FlowLayout());
        cp.add(B1);
        cp.add(B2);
        cp.add(t1);
        cp.add(t2);
        cp.add(t3);
}
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

```
class T1 implements DocumentListener {
        public void changedUpdate(DocumentEvent e) { }
        public void insertUpdate(DocumentEvent e) {
            t2.setText(t1.getText());
            t3.setText("Text: "+ t1.getText());
    }
        public void removeUpdate(DocumentEvent e) {
            t2.setText(t1.getText());
    }
}

 class T1A implements ActionListener {
        private int count =0;
        public void actionPerformed(ActionEvent e) {
            t3.setText("t1 Action Event "+ count++);
    }
}
```

Componentes Swings y Jerarquía

```
class B1 implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            if(t1.getSelectedText() == null)
                s = t1.getText();
            else
                s = t1.getSelectedText();
            t1.setEditable(true);
    }
}
    class B2 implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            Ucd.setUpperCase(false);
            t1.setText("Insertado por Button 2: "+ s);
            Ucd.setUpperCase(true);
            t1.setEditable(false);
    }
}
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

```
class UpperCaseDocument extends PlainDocument {
  private boolean upperCase = true;
  public void setUpperCase(boolean flag) {
            upperCase = flag;
    }
  public void insertString(int offset, String str,
AttributeSet attSet)
            throws BadLocationException {
      if(upperCase)str = str.toUpperCase();
            super.insertString(offset, str, attSet);
    }
  }
public static void main(String[] args)
    {
       TextField tf=new TextField();
       tf.setSize(600,400);
       tf.setVisible(true);
    }
}
```

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Otros componentes a analizar:

Borders JscrollPanes JtextPane (mini editor)

CheckBoxes RadioButtons ComboBoxes

ListBoxes Tabbed panes MessageBoxes

Menus Pop-up menus Drawing

DialogBoxes FileDialogs

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

Seleccionando “Look and Feel”

Uno de los aspectos interesantes de swing es el “Pluggable Look & Feel” (apariencia conectable). Esto permite al programa emular el “look” de varios entornos operativos. Mediante esta herramienta, el programa Java se ve como si fuera creado específicamente para ese sistema.
El código para seleccionar uno de estos comportamientos es simple, pero se debe ejecutar antes de crear cualquier componente visual.

Actualmente la plataforma default es la cross-platform (“metal”), característica de los programas swing. Pero si se quiere cambiar el look & feel, se inserta el siguiente código al principio de main() antes de agregar los componentes.

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Componentes Swings y Jerarquía

```
try {
    UIManager.setLookAndFeel(UIManager.
getSystemLookAndFeelClassName());
    } catch(Exception e) { }
```

No se necesita incluir nada en el catch porque el UIManager establecerá por default la alternativa look & feel, si alguna otra falla.

The Swing Tutorial:

http://docs.oracle.com/javase/tutorial/uiswing/index.html