SWING

JFC

- JFC es la abreviatura de JavaTM Foundation Classes, que comprende un grupo de características para ayudar a construir interfaces gráficos de usuario (GUIs) y agregar funcionalidad gráfica e interactividad a las aplicaciones de Java.

JFC

- Las características definidas son:
  - Componentes Swing GUI
    - Incluye desde botones, panels y tablas
  - Soporte modificable de 'Look and Feel'
    - En cualquier programa que utilice Swimg se tiene la facultad de seleccionar el look and feel
      - Por ejemplo, el mismo programa puede utilizar ya se el lokk adn feel de java o el de Windows

JFC

- API de accesibilidad
  - Proporciona tecnología de asistencia tales como lectores de pantallas y desplegados en Braile para obtener información de la interfaz del usuario
- API Java 2D ™
  - Proporciona a los desarrolladores la facultad para incorporar gráficas de alta calidad en 2D
  - Esto solo es disponible en JDK 1.2

JFC

- Apoyo de Drag & Drop
  - Permite la posibilidad de arrastrar y soltar entre aplicaciones Java y aplicaciones nativas
  - Esto solo es disponible en JDK 1.2
- Internacionalización
  - Permite a los desarrolladores construir aplicaciones que pueden interacturar con usuarios en el mundo en su lenguaje propio y convenciones culturales

Contenedores de nivel superior

- Swing proporciona tres clases contenedoras de nivel superior
  - JFrame
  - JDialog
  - JApplet

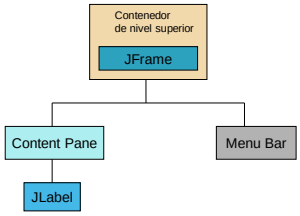
Contenedores de nivel superior

- Cuando se utilizan estas clases se tiene que considerar lo siguiente:
  - Cada componente de la GUI, para que aparezca en pantalla, debe ser parte de una jerarquía de contención
  - Cada componente de la GUI debe estar solamente en un solo lugar
    - Si un componente se encuentra en un contenedor y se intenta agregar a otro, el componente se elimina del primero y se agrega en el segundo

Contenedores de nivel superior

- Cada contenedor de nivel superior contiene un panel de contenido, que generalmente contiene los componentes visibles del contenedor GUI del nivel superior
- Por convención, la barra de menús se coloca en el contenedor de nivel superior, pero fuera del panel de contenido

Jerarquía de los componentes



Jerarquía de los componentes

- Cada aplicación que utiliza componentes Swing debe tener por lo menos un contenedor de nivel superior
- El contenedor de nivel superior es la raíz de la jerarquía de contención
- El JFrame es, por lo general, la raíz de la jerarquía de contención

El JFrame

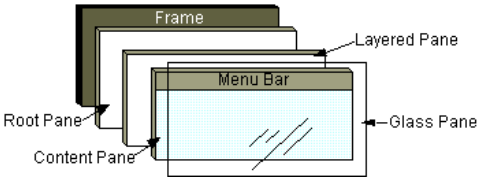


Imagen obtenida de :
http://download.oracle.com/javase/tutorial/uiswing/components/rootpane.html#layeredpane

Adición de objetos a un componente de nivel superior

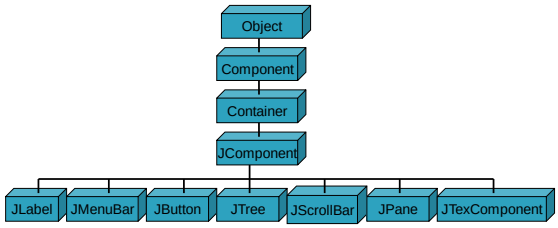
- Para agregar objetos a un componente de nivel superior se utiliza el método getContentPane() el cual devuelve un objeto Container
- Ejemplo

```
miMarco.getContentPane().
            add(miComponente,
            BorderLayout.CENTER);
```

La clase JComponent

- Con excepción de los contenedores de nivel superior, todos los componentes Swing cuyos nombres comienzan con "J" descienden de la clase JComponent
- La clase JComponent hereda de la clase Container, la cual hereda de Component
- La clase Component incluye todo lo necesario para apoyar el dibujado en el lienzo y el manejo de eventos

Jerarquía de clases de Java Swing



Características de JComponent

- Tool tips
  - Se puede proporcionar ayuda a los usuarios de un componente cuando el cursor se posicona sobre el componente
- Pintado y bordes
  - Se puede tener un control específico de la forma en que los gráficos dentro de un componente son dibujados. De la misma forma se puede especificar el borde que muestre un componente

Características de JComponent

- Modificación de look and feel
  - La arquitectura de Swing está diseñada de tal manera que se puede cambiar el "look and feel" de la aplicación GUI
  - El "look" se refiere a la apariencia de los artefactos GUI y "feel" se refiere a la manera en que los artefactos se comportan

Características de JComponent

- Propiedades definidas por el usuario
  - Las propiedades son valores de configuración manejados por el usuario como pares *clave/valor*
  - En cada par, la clave y el valor son valores tipo String
  - La clave identifica y es usada para recuperar el valor
    - Por ejemplo cuando se descargan archivos se puede utilizar una propiedad llamada "descarga.ultimoDirectorio" para saber cual fué el último directorio que se descargó

Características de JComponent

- Apoyo para el layout
  - Swing utiliza administradores de layout para disponer en pantalla los componentes

Características de JComponent

- Apoyo para la accesibilidad
  - Esta característiva se refiere al uso de tecnologías adaptadas a usuarios con necesidades especiales
  - Ejemplos de estas tecnologías son las interfaces de voz, lectores de pantalla o dispositivos de entrada alternos
  - Estas tecnologías son útiles no únicamente a personas con discapacidades, sino también para personas que utilizan computadoras en ambientes fuera de oficinas

Características de JComponent

- Apoyo para drag and drop
  - Se refiere a la habilidad que tenga una aplicación para transferir información entre componentes

Características de JComponent

- Asociación de teclas
  - Esta característica hace que un componente reaccione al presionado de teclas en el teclado
    - Un ejemplo de esto es cuando un botón tiene el foco, teclear un espacio es como si se efectuara un click sobre el botón del mouse

Aplicación de JComponent

»

Ventanas

- Clase Jframe: Ventana con título y marco

Método	Descripción
void setLocation(int x, int y)	Posiciona la ventana en la pantalla. (x, y) son las coordenadas de su vértice superior izquierdo
void setSize(int ancho, int alto)	Asigna el tamaño de la ventana
void setVisible(boolean b) void	Ventana visible (x verdadero) u oculta (x falso)
pack()	Reduce la ventana al tamaño de sus componentes
void setDefaultCloseOperation (int acción);	Acción al cerrar la ventana (JFrame.EXIT_ON_CLOSE)

Agregar Componentes a una Ventana

- Para agregar componentes a una ventana (JFrame) se tiene que agregar a su Content Pane
  - Por uso común los métodos add, remone y setLayout de un Jframe están redefinidos sobre su Content Pane
    - Por ejemplo
      add(a);
    - Es equivalente a
      getContentPane().add(a);

Algunos métodos de los páneles contenedores

Método(s)	Acción
void setBackground(Color c) Color getBackground()	Permite cambiar y obtener el color de fondo
int getWidth() int getHeight()	Regresa la altura y el ancho del componente
void setLayout(LayoutManager mgr)	Permite configurar la forma en que se distribuyen los componentes en el contenedor
void setPreferredSize(Dimension preferredSize)	Determina el tamaño del componente, ejemplo: panel.setPreferredSize(new Dimension(200,100));
setLocatioin(int x, int y)	Mueve el componente a una nueva coordenada

Paneles

- Swing declara varios tipos de paneles contenedores. Algunos de los más usados son:
  - JPanel
  - JScrollPane

Agregar y eliminar componentes a un Panel

- Para agregar componentes a un panel se utilizan

Component add(Component comp)
Component add(Component comp, int index)

- Para la eliminación de componentes se utiliza

void remove(Component comp)

Administradores de distribución

- Determina la manera en que se distribuyen los componentes dentro de un contenedor
- Dentro de los administradores comunes están
  - FlowLayout
  - GridLayout
  - BorderLayout

FlowLayout

- Los componentes se van colocando de izquierda a derecha
- Cuando una línea se completa se pasa a la siguiente línea
- Conserva el tamaño de lo componentes
- Constructor
  - FlowLayout()
- Administrador por defecto en JPanel

GridLayout

- Rejilla de celdas iguales
- Se llena:
  - De izquierda a derecha
  - De arriba hacia abajo (línea a línea)
- Cambia el tamaño de los componentes
- Constructores

GridLayout() – Una columna, un renglón
GridLayout(int filas, int columnas)

BorderLayout

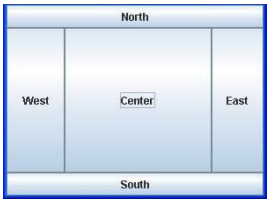
- Coloca los 4 componentes en los 4 puntos cardinales y en el centro
- El central se expande, ocupando la mayor área
- Constructor
  BorderLayout()

BorderLayout (cont)

- El método add tiene un parámetro adicional (índice) para indicar la zona

Component add(Component comp, int index)

BorderLayout.NORTH
BorderLayout.SOUTH
BorderLayout.WEST
BorderLayout.EAST
BorderLayout.CENTER



Componentes

»

Componentes

- Etiqueta (JLabel)
- Botones
- Campo de Texto
- Area de texto
- Menú desplegable
- Lista de opciones

JLabel

- Texto de solo lectura
- Constructor default
  JLabel()
- Obtener texto de la etiqueta

String getText()

- Cambiar texto de la etiqueta

void setText(String text)

Botones

- Jbutton
- JCheckBox
- JRadioButton

Todos heredan de AbstractButton:
String getText()
void setText(String text)
boolean isSelected()
void setSelected(boolean b)

JButton

- Botón que puede ser presionado
- Constructor default

JButton(String text)

JCheckBox

- Casilla que puede ser marcada
- Constructores comúnes

JCheckBox()
- Crea un botón 'check box' sin texto y sin seleccionar inicialmente

JCheckBox(String text)

JCheckBox(String text, boolean state)
- Crea un botón 'check box' con texto y especifica si o no es inicialmente seleccionado

JRadioButton

- Casilla que puede ser marcada
- Constructores comunes

JRadioButton(String text)
JRadioButton(String text, boolean selected)

Grupo de botones

- ButtonGroup
  - Permite agrupar varias casillas
- Constructor defautl
  ButtonGroup()
- Agregar un botón al grupo:
  void add(AbstractButton b)

Componentes de texto

- Los componentes de texto en swing muestran texto y, de manera opcional, permiten su edición
- Swing provee seis componentes de texto, junto con sus clases e interfaces de apoyo
- Todos los componente de swing heredan de la misma superclase: JTextComponent

Jerarquía de JTextComponent

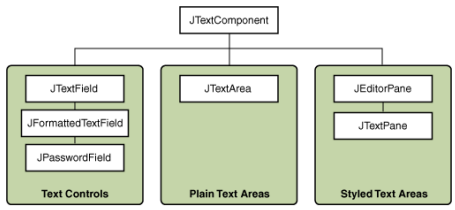


Imagen obtenida de :
http://download.oracle.com/javase/tutorial/uiswing/components/text.html

Aquí comentaremos JTextField y JTextArea

JTextField

- Línea de texto que puede ser editada por el usuario
- Constructores:
  JTextField()
  JTextField(int columns)
  JTextField(String text)
  JTextField(String text, int columns)
- Regresa texto
  String getText()
- Cambia texto
  void setText(String t)

JTextArea

- Area de texto de varias líneas que puede ser configurada para ser editada por el usuario o de sólo lectura
- Constructores:
  JTextArea()
  JTextArea(int rows, int columns)
  JTextArea(String text)
  JTextArea(String text, int rows, int columns)
- Configuración si es editable o no
  void setEditable(boolean b)
- Regresa texto
  String getText()
- Escribe texto
  void setText(String t)

Menú desplegable (JComboBox)

- Menú desplegable con varias opciones
- La opción elegida en cada momento aparece en el nombre del menú
- Constructores
  JComboBox()
  JComboBox(Object[] items)
- Regresa la opción seleccionada
  Object getSelectedItem()
  int getSelectedIndex()
- Agregar una opción al final de la lista
  void addItem(Object anObject)

Lista de Opciones (JList)

- Puede configurarse para que permita elegir una o varias opciones
- Constructor default
  JList(Object[] listData)

Lista de Opciones (JList)

Configurar el tipo de selección
void setSelectionMode(int selectionMode)

ListSelectionModel.SINGLE_SELECTION
ListSelectionModel.SINGLE_INTERVAL_SELECTION
ListSelectionModel.MULTIPLE_INTERVAL_SELECTION (valor por default)

Lista de Opciones (JList)

- Regresar la(s) opción(es) seleccionada(s)
  int getSelectedIndex()
  int[] getSelectedIndices()
  E getSelectedValue()
  List<E> getSelectedValuesList()

Cambia la(s) opción(es) seleccionada(s)
  void setSelectedIndex(int index)
  void setSelectedIndices(int[] indices)

JScrollPane

»

JScrollPane

- Panel que incluye barras de desplazamiento
- Sirve para desplazarse por componentes demasiado grandes para ser mostrados en su totalidad
- Se utiliza principalmente con áreas de texto y listas

JScrollPane

- Constructores
  JScrollPane(Component view)
  JScrollPane(Component view,
  int vsbPolicy,
  int hsbPolicy)

ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED
ScrollPaneConstants.VERTICAL_SCROLLBAR_NEVER
ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS

ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED
ScrollPaneConstants.HORIZONTAL_SCROLLBAR_NEVER
ScrollPaneConstants.HORIZONTAL_SCROLLBAR_ALWAYS

Manejo de eventos

Interfaces gráficas de usuario (GUI)

- Muchos de los programas basados en GUIs (*Graphical User Interface*) son un ejemplo de programa dirigido por eventos
- Las GUIs facilitan la interacción del usuario con el programa

Interfaces gráficas de usuario (GUI)

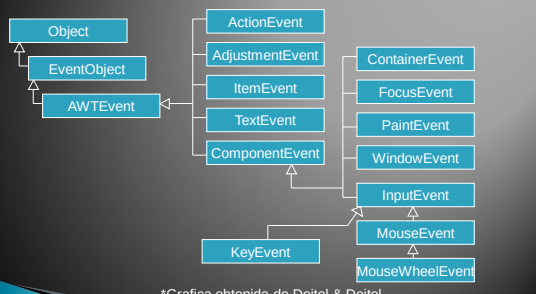
Eventos

- Cuando un usuario interactúa con la interfaz gráfica, genera eventos
- Algunos eventos comunes son:
  - mover el ratón,
  - hacer clic en botón en la pantalla,
  - escribir en un campo de texto,
  - seleccionar un elemento de un menú
  - cerrar una ventana, etc.

Eventos

- Cuando ocurre una interacción con el usuario, se envía un mensaje al programa
- La información de los eventos de la interfaz gráfica se almacena en un objeto de una clase que extiende AWTEvent
- En la siguiente figura se muestran algunas clases de eventos del paquete java.awt.event
- Los tipos de eventos del paquete java.awt.event se utilizan con componentes de AWT y de Swing

Paquete java.awt.event



*Grafica obtenida de Deitel & Deitel

Manejo de eventos

- El mecanismo del manejo de eventos consta de tres partes
  - El origen del evento
    - Es el componente específico, de la interfaz gráfica, con el cual interactúa el usuario
  - El objeto del evento
    - El objeto del evento encapsula la información acerca del evento que ocurrió
  - El componente de escucha del evento
    - Es un objeto que recibe la notificación del origen del evento cuando éste ocurre
    - El origen del evento mantiene una lista de sus componentes de escucha registrados, y notifica a cada uno de ellos cuando ocurre un evento

Manejo de eventos

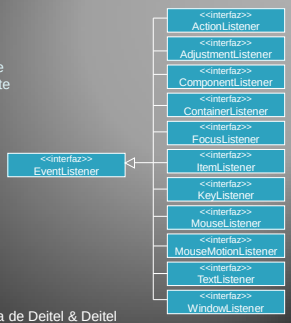
- Para procesar un evento de interfaz gráfica de usuario se tienen que realizar las siguientes tareas:
  - Registrar un componente de escucha del evento para el componente de la interfaz gráfica que se espera genere ese evento
  - Implementar un (conjunto de) método(s) manejador(es) del evento

Manejo de eventos

- Un componente de escucha para un evento de la interfaz gráfica de usuario es un objeto de una clase que implementa a una o más de las interfaces de componentes de escucha de eventos de los paquetes
  java.awt.event
  javax.swing.event

Interfaces

Interfaces de componentes de escucha de eventos del paquete java.awt.event



*Grafica obtenida de Deitel & Deitel

Interfaz de componente de escucha de eventos

- Cada interfaz de componente de escucha de eventos especifica uno o más métodos manejadores de eventos que deben declararse en la clase que implemente la interfaz de componente de escucha de eventos

Modelo de delegación de eventos

- El uso de componentes de escucha de eventos se conoce como el modelo de delegación de eventos
  - El procesamiento de un evento se delega a un objeto específico en el programa

Bibliografía

- Kathy Walrath…[et al.]. The JFC Swing Tutorial.: a guide to constructing GUIs. Second Ed. The Java Series. Sun Microsystems
- Deitel & Deitel. Java™- Cómo programar. Pearson Prentice-Hall. Quinta Ed.
- F.J. Ceballos. Java 2 Curso de programación. Alfaomega Ra-Ma. 2da. Ed
- Material en linea

http://download-llnw.oracle.com/javase/tutorial/essential/environment/

Bibliografía (cont)

- C. Horstman. Big Java. Wiley. 3era Ed.