UT2. Creación de componentes visuales

24 horas

Contenidos

1. Desarrollo de software basado en componentes. Reutilización del software. Beneficios.
2. Componentes visuales: concepto de componente; propiedades y atributos.
3. Eventos:
   a. Componentes y eventos
   b. Listeners
   c. Métodos y eventos
4. Persistencia del componente.
5. Herramientas para desarrollo de componentes visuales.
6. Empaquetado de componentes.

Resultados de Aprendizaje

RA2: Genera interfaces naturales de usuario utilizando herramientas visuales	5%
a) Se han identificado las herramientas disponibles para el aprendizaje automático relacionadas con las interfaces de usuario.	40% .
b) Se ha creado una interfaz natural de usuario utilizando las herramientas disponibles.	25% .
c) Se ha utilizado el reconocimiento de voz para implementar acciones en las interfaces naturales de usuario.	25% .
d) Se ha incorporado la detección del movimiento del cuerpo para implementar acciones en las interfaces naturales de usuario.	5% .
e) Se han integrado elementos de detección de partes del cuerpo para implementar acciones en las interfaces naturales de usuario.	5% .
f) Se ha integrado la realidad aumentada en los interfaces de usuario.	5%

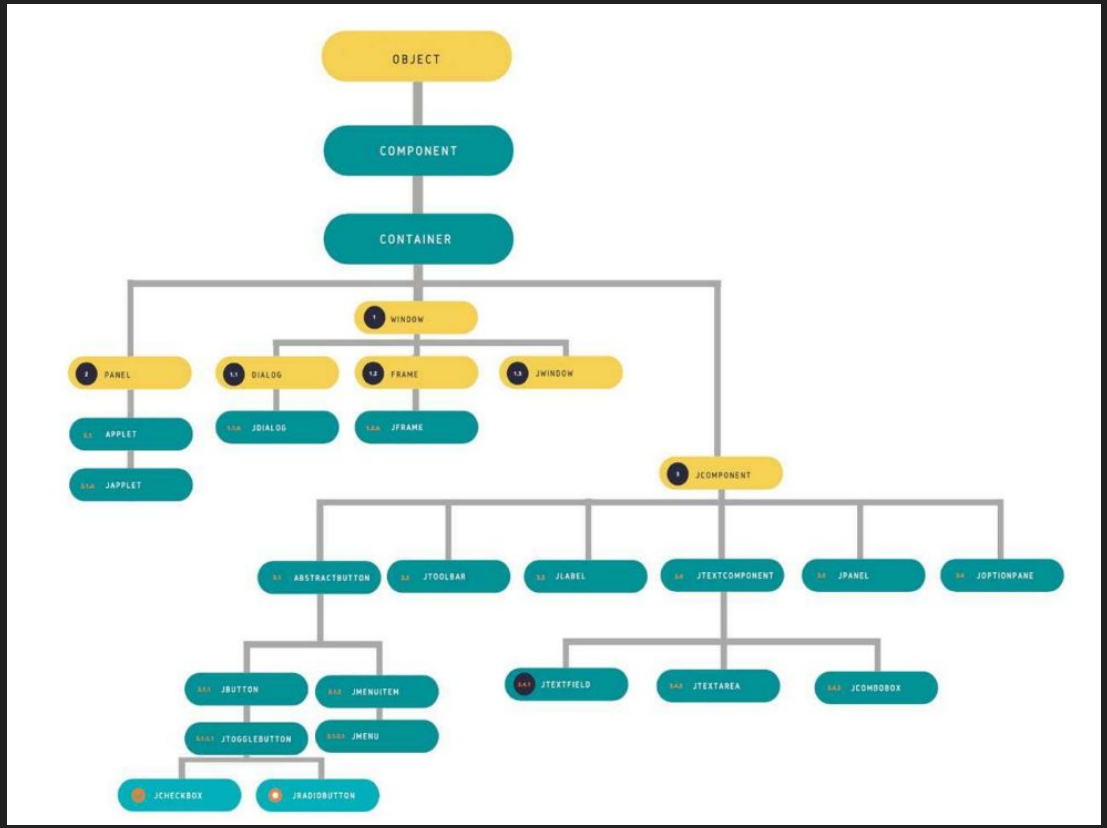
Resultados de Aprendizaje

RA3: Crea componentes visuales valorando y empleando herramientas específicas	15%
a) Se han identificado las herramientas para diseño y prueba de componentes.	10%
b) Se han creado componentes visuales.	10%
c) Se han definido sus métodos y propiedades con asignación de valores por defecto.	10% .
d) Se han determinado los eventos a los que debe responder el componente y se les han asociado las acciones correspondientes.	20% .
e) Se han realizado pruebas unitarias sobre los componentes desarrollados.	10%
f) Se han documentado los componentes creados.	10%
g) Se han empaquetado componentes.	10%
h) Se han programado aplicaciones cuyo interfaz gráfico utiliza los componentes creados.	20% .

Componentes visuales

Las aplicaciones informáticas requieren una interacción constante entre usuario e interfaz. Los elementos visuales que permiten la comunicación son los componentes visuales.

Un componente es un código ya implementado y reutilizable que puede interactuar con otros componentes.



Propiedades y atributos

Las propiedades de un componente definen los datos públicos que forman la apariencia y comportamiento de un objeto. Pueden modificar su valor a través de los métodos.

Por ejemplo el componente *JButton*: una de sus propiedades es *font*, que podrá ser consultada o modificada.

Recuerda los tipos de ámbitos: público, privado y estático.

Eventos

Sin ellos, solo tenemos ventanas que no hacen nada para interactuar con el usuario.

Para programarlos hay que detectar el evento y asociar una acción a él.

Eventos

Clase	Descripción
EventObject	Clase principal de la que derivan TODOS los eventos.
MouseEvent	Eventos relativos a la acción del ratón sobre el componente.
ComponentEvent	Eventos relacionados con el cambio de un componente, de tamaño, posición…
ContainerEvent	Evento producido al añadir o eliminar componente sobre un objeto de tipo Container.
WindowsEvent	Este tipo de eventos se produce cuando una ventana ha sufrido algún tipo de variación, desde su apertura o cierre hasta el cambio de tamaño.
ActionEvent	Evento que se produce al detectarse la acción sobre un componente. Es uno de los más comunes, puesto que modela acciones tales como la pulsación sobre un botón o el check en un menú de selección.

Componentes y eventos asociados

- **JTextField** → **ActionEvent:** Detecta la pulsación de Enter en un campo de texto.
- **JButton** → **ActionEvent:** Detecta la pulsación de un botón.
- **JComboBox** → **ActionEvent, ItemEvent:** Detecta la selección de uno de los valores.
- **JCheckBox** → **ActionEvent, ItemEvent:** Detecta el marcado de una de las celdas.
- **JTextComponent** → **TextEvent:** Se produce un cambio en el texto.
- **JScrollBar** →**AdjustmentEvent:** Detecta el movimiento de la barra scroll.

Listeners

Se quedan escuchando a la espera de un evento.

Todo evento necesita un listener que controle su activación.

```
boton.addActionListener(this);
//
public void actionPerformed(ActionEvent e) {
... ;
}
```

```
boton.addActionListener(new
ClaseClicEnBoton());
//
class ClaseClicEnBoton implements
ActionListener {
  public void actionPerformed(ActionEvent e) {
  ... ;
  }
}
```

```
boton.addActionListener(new ActionListener()
{
public void actionPerformed(ActionEvent e) {
clicEnBoton(e); }
});

private void clicEnBoton(ActionEvent e) {
... ;
}
```

KeyListener

Modo de pulsación	Definición
keyPressed	Se produce al pulsar la tecla.
keyTyped	Se produce al pulsar y soltar la tecla.
keyReleased	Se produce al soltar una tecla.

Ejemplo

Descarga del Aula Virtual el código *KeyListenerExample.java*

ActionListener

Componente	Descripción
JButton	Al hacer click sobre el botón o pulsar la tecla Enter con el foco situado sobre el componente.
JTextField	Al pulsar la tecla Enter con el foco situado sobre la caja de texto.
JMenuItem	Al seleccionar alguna opción del componente menú.
JList	Al hacer doble click sobre uno de los elementos del componente lista.

MouseListener

Modo de pulsación	Definición
mouseClicked	Se produce al pulsar y soltar con el puntero del ratón sobre el componente.
mouseExited	Se produce al salir de un componente utilizando el puntero del ratón.
mousePressed	Se produce al presionar sobre el componente con el puntero.
mouseReleased	Se produce al soltar el puntero del ratón.
mouseEntered	Se produce al acceder a un componente utilizando el puntero del ratón.

Ejemplo

Descarga el código *MouseListenerExample.java* del aula y estúdialo.

MouseMotionListener

Modo de acción	Definición
mouseMoved	Se produce al mover sobre un componente el puntero del ratón.
mouseDragged	Se produce al arrastrar un elemento haciendo click previamente sobre él.

Actividad 1

Investiga el uso de *FocusListener* y *MouseMotionListener.*

Modifica los códigos vistos en clase para añadir éstos.

Métodos de los eventos

Nombre Listener	Métodos	
ActionListener	public void actionPerformed(ActionEvent e)	
KeyListener	keyPressed	public void keyPressed(KeyEvent e)
	keyTyped	public void keyTyped(KeyEvent e)
	keyRelease	public void keyReleased(KeyEvent e)
FocusListener	Obtención del foco	public void focusGained(FocusEvent e)
	Pérdida del foco	public void lostGained(FocusEvent e)
MouseListener	mouseClicked	public void mouseClicked(MouseEvent e)
	mouseExited	public void mouseExited(MouseEvent e)
	mousePressed	public void mousePressed(MouseEvent e)
	mouseReleased	public void mouseReleased(MouseEvent e)
	mouseEntered	public void mouseEntered(MouseEvent e)
MouseMotionListener	mouseMoved	public void mouseMoved(MouseEvent e)
	mouseDragged	public void mouseDragged(MouseEvent e)

Práctica 2

Realiza el ejercicio que encontrarás en el Aula Virtual.