M S c. R e y n a l d o Z e b a l l o s
2007/2008

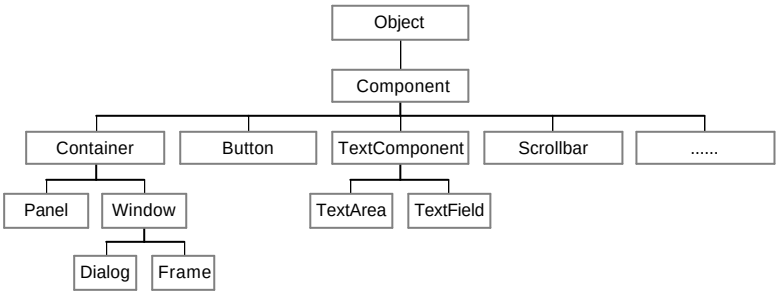
AWT

- Contenedores
- Contenidos

AWT

- El paquete AWT (Abstract Window Toolkit) tiene varias clases para crear y diseñar interfaces de usuario (GUI), es decir ventanas, elementos gráficos, manejo de eventos, etc. la intención es, como para el resto del lenguaje, que el interface sea independiente de la plataforma
- En las versiones de jdk 1.2 y sucesivas este paquete esta mejorado por los paquetes JFC swing
- La jerarquía de clase de awt se basa en la clase component que hereda de object y de la que heredan a su vez los distintos elementos que se pueden colocar en el interfaz diseñado, entre ellos cabe destacar 2 tipos importantes
  - Los contenedores : capaces de incluir dentro de si otros elementos (ej. Una Ventana)
  - Los componentes “elementales” como botones, checkbox, etc..

Jerarquia de clases awt



Container

- Container es la clase base de los contenedores y consta de 2 subclases importantes:
  - Window : para la creación de ventanas genéricas. Esta consta de 2 subclases más específicas
    - Dialog : para crear ventanas en las que se presenta o pide información al usuario.
    - Frame: para crear la ventana principal de la aplicación.
  - Panel contenedor genérico que se utiliza para organizar los demás elementos dentro de él

Componentes “elementales”

- Button: botón similar a los de aceptar y cancelar típicos de aplicaciones de entornos de ventanas.
- *Checkbox:* botones de comprobación. Proporcionan información del tipo *Sí* o *No*
- *Choice :* Los botones de selección en una lista permiten el rápido acceso a una lista de elementos. Por ejemplo, podríamos implementar una selección de paises y mantenerla en un botón Choice.
- *Label :* etiquetas , permiten colocar texto invariable en un panel, para dar información
- *List:* listas, son similares a los Choice pero en este caso los valores aparecen a la vista todo el tiempo y se usa una barra de desplazamiento para ver los que no entran en la zona visible
- Scrollbar: barra de desplazamiento
- *TextArea:* zona, de una o varias líneas, para permitir la entrada de texto o para mostrar información
- TextField: similar a la anterior pero utilizada para cuando el tamaño del texto a introducir es menor(una línea, una palabra)
- ……

Frame

- El elemento básico de una interfaz de usuario es la ventana, los Frames son las más genéricas, poseen un titulo y un borde y los botones de ampliar, cerrar e iconizar.
- Métodos:
  - `Frame ([String título])`
  - `String getTitle()`
  - `void settitle(String titulo)`
  - `void setSize (int anchura, int altura)`: Define el tamaño de la ventana.
  - `void setVisible (boolean b)` Muestra u oculta el frame
  - `boolean isShowing()`
  - `void dispose()` Destruye el frame
- Nota: El origen de coordenadas es la esquina superior izquierda

Construcción de un Interface

- La forma básica de crear una ventana es crear una clase que herede de Frame, y contenga un método `paint()`, que sobreescribirá el correspondiente de la clase Frame. Este método será invocado implícitamente cuando se realiza alguna operación sobre la ventana (iconzarla->desiconizarla) o explícitamente invocando al método `repaint()`.

`public void paint (Graphics g):` este método recibe un parámetro de la clase Graphics que representa el contexto en el que el componente puede realizar su dibujo.

```
import java.awt.*;
public class Ejem1AWT extends Frame {
    public Ejem1AWT() {
        super( "Aplicación 1 con AWT" );
        setSize( 200, 100);    }
    public void paint( Graphics g ) {
        g.drawString("HOLA", 50, 50 );    }
    public static void main( String args[] ) {
        Ejem1AWT a=new Ejem1AWT();
        a.show();     }
}
```

Graphics

- La clase Graphics proporciona métodos para dibujar algunos de sus métodos son
- `void drawString(String str, int x, int y)` dibuja el string indicado en las coordenadas x,y
- `drawRect (int x, int y, int ancho, int alto)` dibuja el rectangulo indicado
- `fillOval( int x, int y, int ancho, int alto)` dibuja un ovalo, relleno con el color actual, del tamaño indicado en la posición x,y.
- `setColor(Color c)` define el color indicado como el actual
- `setFont(Font fuente)` define la fuente indicada (courer, bold, etc) como la actual
- …….

Ejemplo Graphics

```
import java.awt.*;
/* muestra, cada segundo, el contador
 y cambia el color del rectángulo.*/
public class Ejem2AWT extends Frame {
    int contador=0;
   public Ejem2AWT() {
     super( "Aplicación 2 con AWT" );
     setSize( 400, 200);
   }

   public void paint( Graphics g ) {
     g.drawString("Contador:"+
    contador, 50, 50 );
     contador++;
     if (contador %2 ==0)
       g.setColor(Color.blue);
     else
       g.setColor(Color.red);
     g.fillRect (100,100, 30,100);
   }
```

```
public static void main( String args[] ) {
        Ejem2AWT a=new Ejem2AWT();
        a.show();
        for (int i=0; i<10; i++){
         try{
          Thread.sleep(1000);
         } catch(Exception e){}
          a.repaint();
        }
        a.dispose();
    }
}
```

FlowLayout

- FlowLayout: los componentes se añaden en forma de lista horizontal de izquierda a derecha y de arriba a abajocon espacio entre cada componente.

```
import java.awt.*;
public class FlowLayoutAWT extends Frame {
  public FlowLayoutAWT() {
    super( "Aplicación con FlowLayout" );
    setLayout(new FlowLayout());
    add(new Button("uno"));
    add(new Button("dos"));
    add(new Button("tres"));
    add(new Button("cuatro"));
    setSize( 100, 100);
    show();
  }
  public static void main( String args[] ) {
    FlowLayoutAWT a=new FlowLayoutAWT();
    a.show();
  }
}
```

Cómo colocar los elementos en el GUI

- Aunque es posible definir la posición de un elemento de forma fija como hemos visto en los ejemplos de la clase Graphics, lo recomendable es utilizar layouts (formatos genéricos de posicionamiento que se adaptan al tamaño de la ventana en cada momento)
- Existen varios tipos de layouts :FlowLayout, BorderLayout, GridLayout, GridBagLayout y CardLayout y además se pueden crear layouts personalizados.
- Para seleccionarlos se usa

  `public void setLayout( LayoutManager tipoLayout)`

- Por defecto Panel (y Applet que es heredera suya) tiene como layout por defecto el FlowLayout y Window, Dialog y Frame tienen el BorderLayout

Panel

Un Panel es un contenedor (class Container) básico de objetos.

Métodos

- `Panel ([LayoutManager layout])`: Constructor que crea un nuevo panel con un LayoutManager de tipo por defecto (FlowLayout) a no ser que se especifique otro. Los Layouts permiten que la disposición de los componentes en la ventana sea sencilla de realizar y no dependa de aspectos concretos del entorno como el tamaño de la ventana.
- `public void addNotify ()` : Crea un "duplicado"(peer)del panel. El peer permite modificar la apariencia del panel sin cambiar su funcionalidad. Con esto AWT enviará al panel todos los eventos que ocurran sobre el.

La clase Applet es una subclase de Panel

Ejem. Panel

```
import java.awt.*;
public class LayoutAWT extends Frame {
  public LayoutAWT() {
    super( "Aplicación Layout con AWT" );
    setLayout(new BorderLayout()); //defiono el tipo de layout
    Panel p =new Panel();  //creo un panel
    // 2 formas de añadir botones al panel, mejor la 1ª
    Button b1=new Button("Aceptar");
    p.add(b1);
    p.add(new Button("Cancelar"));
    add("North",p);//coloco el panel en la parte superior de la ventana
    setSize( 300, 200);  // defino el tamaño de la ventana
    show();
  }
  public void paint( Graphics g ) {
    g.drawString("Palabra", 120, 120 );
    //necesario si se añaden botones a un contenedor ya visible
    validate();
  }
  public static void main( String args[] ) {
    LayoutAWT a=new LayoutAWT();
    a.show();
  }
}
```

Componentes AWT

Para añadir un elemento gráfico a un contenedor se utiliza el método

objetocontenedor.add (Component comp)

- Veamos un ejemplo con la clase Button:
  - Button([String etiqueta]);
  - String getLabel()
  - void setLabel(String etiqueta)