

# Programación de Interfaces Gráficas en Java

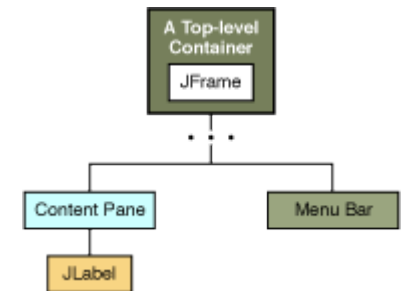
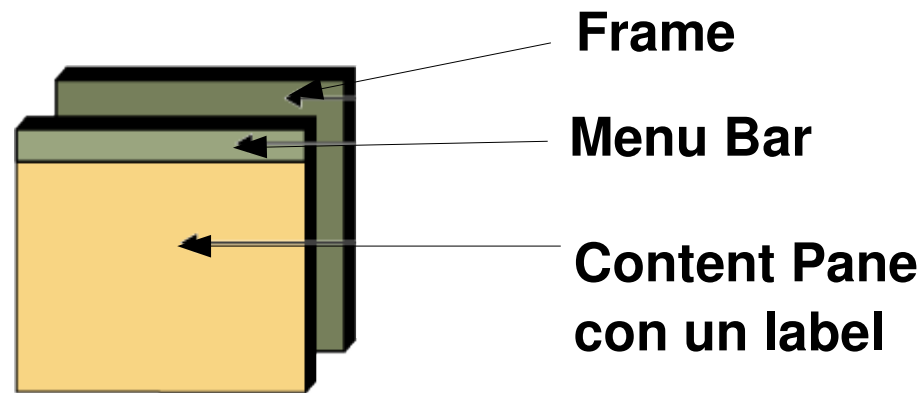
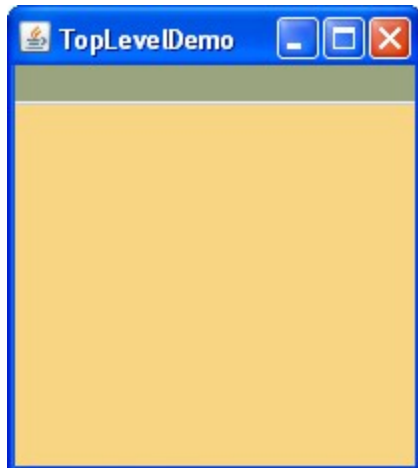
Agustín J. González  
ELO329

# AWT y Swing

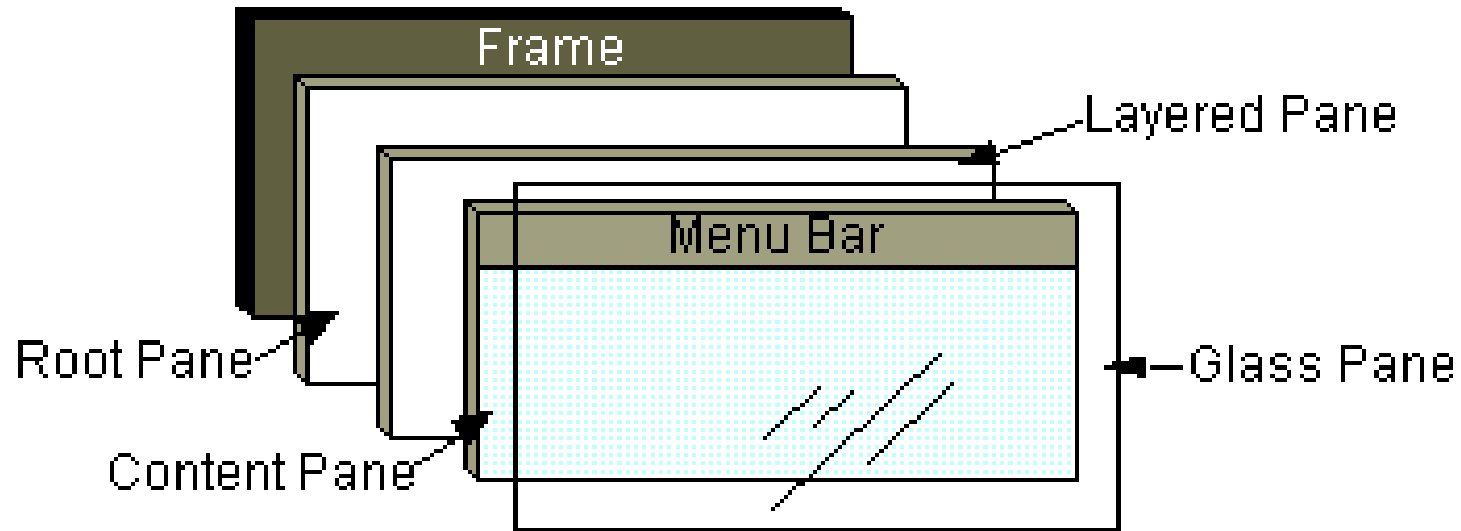
- En sus orígenes Java introdujo la AWT (Abstract Window Toolkit). Ésta “creaba” los objetos delegando su creación y comportamiento a herramientas nativas de la plataforma donde corre la Máquina Virtual Java.
- Este esquema condujo a problemas por diferencias en distintas plataformas y S.O.
- La solución fue desarrollar todos los objetos de la GUI basados sólo en elementos muy básicos y comunes en todas las plataformas. Así surge Swing. (Ver demo de la JFC <http://download.java.net/javadesktop/swingset3/SwingSet3.jnlp>)
- Junto a la JDK vienen demos de Java. Ver la carpeta que contiene el ejecutable javac y podrá encontrar el directorio demo un nivel más arriba.

# Desplegando información

- Todos los objetos gráficos en una aplicación Java forman una jerarquía. La mayor jerarquía está en un JFrame, un JDialog, o un JApplet.
- Ahora veremos la estructura de los JFrame.
- Ejemplo:



# Estructura de un JFrame



- El RootPane viene con el JFrame. También lo traen los JInternalFrame y los otros contenedores de ventanas superiores (autónomas): JDialog, JApplet, JFrame.
- El root pane tiene 4 partes: vidrio, panel de capas, panel de contenido, y una barra de menú opcional.

# Panel de vidrio

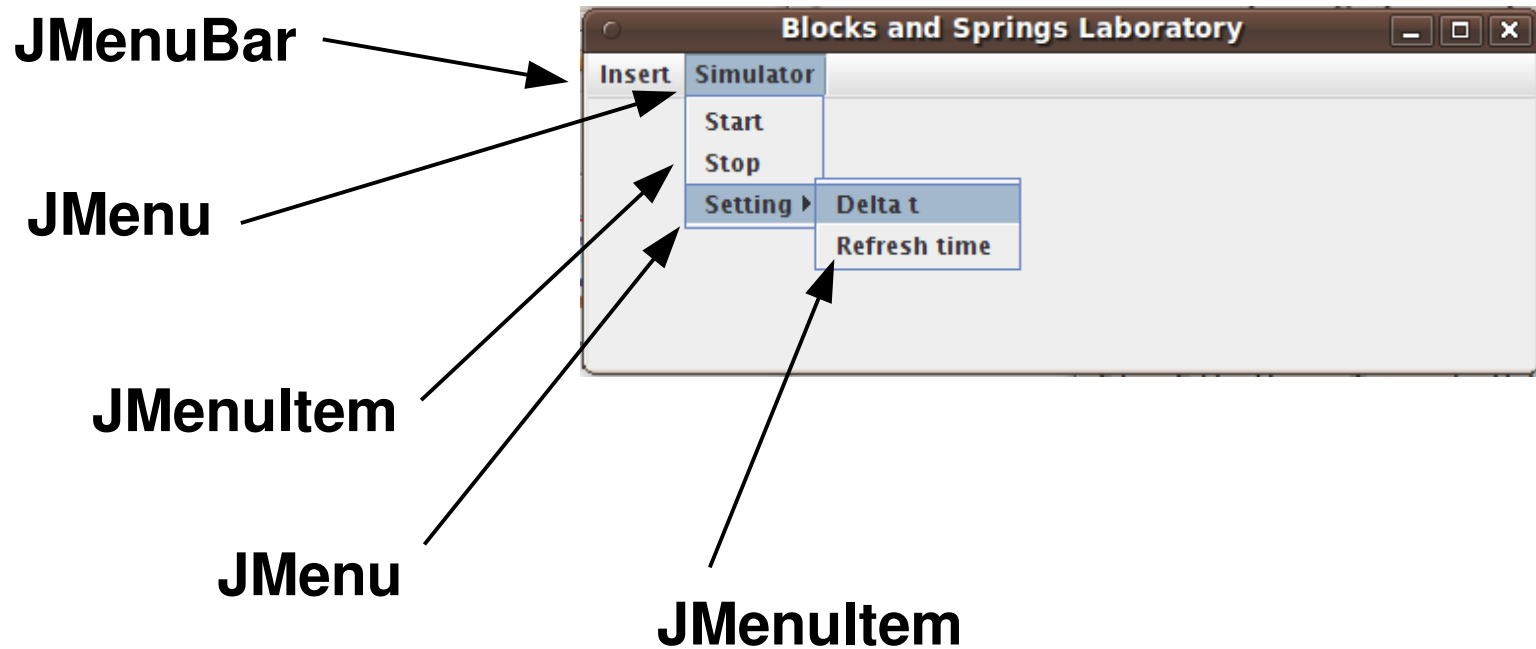
- Oculto por omisión (default).
- Si se hace visible, es como una hoja de vidrio sobre todos las partes del panel raíz.
- Es transparente, a menos que se implemente un método para pintarlo.
- Puede interceptar los eventos de la ventana panel de contenido y menú.
- Ver `GlassPaneDemo.java`

# Layered Pane (panel de capas múltiples)

- Contiene la barra de menú opcional y el panel para poner contenidos.
- Puede también contener otras componentes en orden especificado por eje Z (profundidad).
- Ver más detalles en curso tutorial de Swing
- Ver LayeredDemo.java

# Menús (así es en plural)

- Algunos elementos de un menú



# Menús: Ejemplo

- Crear un frame
  - Crear un menubar
  - Crear a un menu
  - Crear algunos items del menu
  - Capturar eventos
  - Agregar item al menu
  - Agregar el menu al menubar
  - Incorporar el menubar
  - Por ejemplo de menú más completo ver: MenuDemo.java
- ```
JFrame f = new JFrame("MenuT");
JMenuBar mb = new JMenuBar();
JMenu menu = new JMenu("Choose");
JMenuItem item1, item2;
item1 = new JMenuItem("Data 1");
item2 = new JMenuItem("Data 2");
// Action listeners!!
menu.add(item1);
menu.add(item2);
mb.add( menu );
f.setJMenuBar( mb );
```



# Modelos y Vistas de Objetos

- Asociado a cada objeto gráfico debemos distinguir su modelo de su vista.
- El **modelo es el conjunto de atributos de un objeto**, corresponde a la representación en memoria de un objeto. Por ejemplo, para un termómetro, basta el atributo double temperatura.
- La **vista es la apariencia visual** que decidimos dar al objeto. Por ejemplo, un termómetro puede ser digital, columna de mercurio, la intensidad de un color, etc.
- Así una cosa es el cambio del estado de un objeto y otra el cambio en la vista correspondiente a ese nuevo estado.
- Objetos Swing mantienen esa consistencia, pero debemos ocuparnos de ello en objetos gráficos nuestros.

# Pintado de Componentes

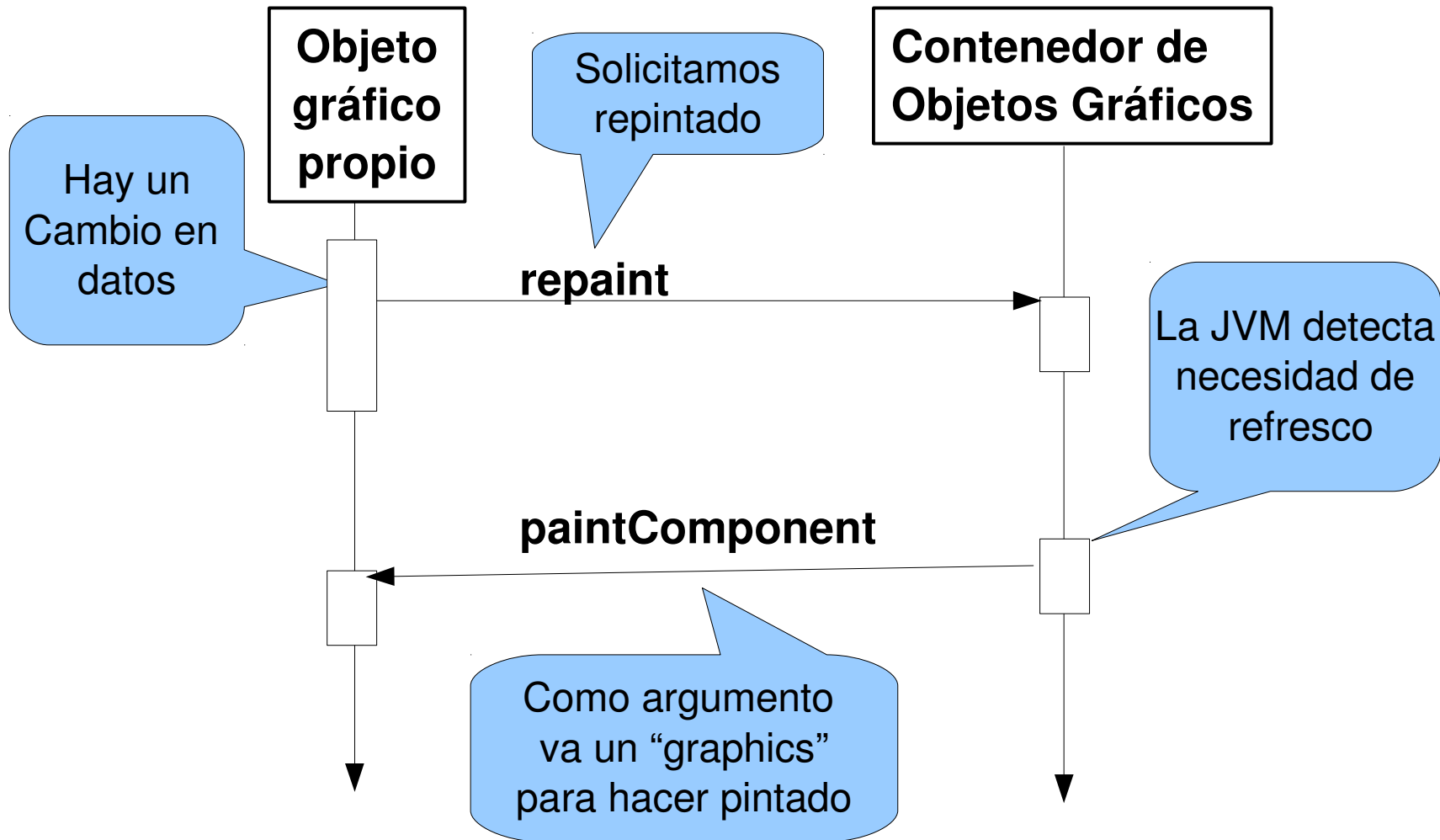
- En general hay que tratar de usar componentes estándares de Swing. Ellas se encargan de hacer su (re)pintado en pantalla cuando corresponda.
- Este es el caso de Labels, buttons, componentes de texto, icons, borders.
- Si luego de hacer visible un contenedor de componentes éste se modifica, llamar a `validate()`. Con esto logramos actualizar su despliegue. Ver `CreaBotones.java`
- Para crear objetos gráficos nuevos, podemos heredar de `JPanel`.
- Cuando la interfaz posee objetos “dibujados” por la aplicación, debemos redefinir el método:  
`protected void paintComponent(Graphics)`. Éste es invocado cada vez que una componente gráfica requiere ser re-pintada.

# Método: repaint

- Cuando una componente cambia alguno de sus atributos, por ejemplo un label cambia su texto, este método es invocado por la componente sobre un panel (en rigor sobre una instancia JComponent) que lo contiene. Se consigue así itinerar el repintado de las componente gráficas.
- Si nosotros hemos construido líneas, círculos, etc. debemos llamar a repaint() para solicitar al panel el llamado a paintComponent en forma explícita.
- Ver ejemplo: Sketch.java y MouseTest.java

# Método: repaint

- Diagrama de secuencia para repintado.



# Java 2D

- Java 2D provee gráficos, texto e imágenes de dos dimensiones a través de extensiones de Abstract Windowing Toolkit (AWT)
- Incluye clases para Rectángulos, Líneas, Elipses.
- La clase Graphics2D, a través de su método draw, permite dibujar estos objetos debido a que todos ellos implementan la interfaz shape.
- Ver demo: ShapesDemo2D.java