

# Prácticas POO

## Curso 10/11

Alejandro Bellogín

Escuela Politécnica Superior  
Universidad Autónoma de Madrid  
Marzo 2011

<http://www.eps.uam.es/~abellogin>

# Esquema

- Explicación de la P3
- Esta práctica y Java
- Esta práctica y NetBeans

# Práctica 3

- Implementación sistema de gestión de tienda virtual
- Con interfaz gráfica
- Sincronizado
- Distribuido (RMI)

# Práctica 3 - Calendario

28 Mar.  
Evaluación P2  
Inicio P3

04 Abr.

11 Abr.

18 Abr.  
Semana Santa

25 Abr.  
Semana Santa

02 May.  
Festivo



09 Abr.

16 May.  
Evaluación P3

# Práctica 3 - Planificación

- Semana 1: **GUIs**
- Semana 2: **sincronización**
- Semana 3: **RMI**
- Semana 4: Semana Santa
- Semana 5: Semana Santa
- Semana 6: fiesta
- Semana 7
- Semana 8: ENTREGA

# Práctica 3

- Implementación sistema de gestión de tienda virtual
- Con interfaz gráfica
- Sincronizado 
- Distribuido (RMI) 

# Explicación P3

- Estructura de datos
  - La de la P2 (XML)
- Interfaz para datos de usuario
  - Funcionalidad de la P2 de manera gráfica
- Gestión de la tienda
  - La de la P2
  - Distinguir roles de cliente y administrador
- Opcional: recomendación

# Observación

- Para facilitar el uso de RMI:
  - Separar bien el cliente del servidor
  - El servidor le devuelve (manda) objetos al cliente



# Interfaces gráficas en Java

- Qué tienen que ver con POO?
- Eventos
- Swing vs AWT
- Componentes más importantes

# Interfaces y POO

- El usuario ve objetos en la pantalla
- El usuario puede manipular los objetos
- Los objetos tienen **comportamiento propio**: distintas formas de responder a una acción del usuario
- Programación basada en eventos

# Programación basada en eventos

- El modo de operación de una interfaz de usuario no se ajusta a un control de flujo estrictamente secuencial
- El usuario tiene un alto grado de libertad en todo momento: normalmente dispone de un amplio conjunto de acciones posibles
- Es el modelo utilizado en las interfaces de usuario actuales basadas en ventanas
  
- La iniciativa no la lleva el programa sino el usuario
  
- Las componentes están a la espera de las acciones del usuario
- Las acciones del usuario generan eventos que se acumulan en una cola
- El sistema de eventos extrae eventos de la cola y los envía a los programas
- Los programas procesan los eventos recibidos respondiendo según el tipo de evento
- Cada tipo de componente se caracteriza por una forma propia de respuesta a los eventos
  
- La ventana recibe eventos sin diferenciar
- Respuesta de ventanas a eventos: repintar, cambiar apariencia y repintar, ejecutar una función (acción)

# Programación basada en eventos

- El modo de operación de una interfaz de usuario no se ajusta a un control de flujo estrictamente secuencial
- El usuario tiene un alto grado de libertad en todo momento: normalmente dispone de un amplio conjunto de acciones posibles
- Es el modelo utilizado en las interfaces de usuario actuales basadas en ventanas

<http://arantxa.ii.uam.es/~castells/docencia/poo/7-guis.pdf>

- La iniciativa no la lleva el programa sino el usuario
- Las componentes están a la espera de las acciones del usuario
- Las acciones del usuario generan eventos que se acumulan en una cola
- El sistema de eventos extrae eventos de la cola y los envía a los programas
- Los programas procesan los eventos recibidos respondiendo según el tipo de evento
- Cada tipo de componente se caracteriza por una forma propia de respuesta a los eventos
- La ventana recibe eventos sin diferenciar
- Respuesta de ventanas a eventos: repintar, cambiar apariencia y repintar, ejecutar una función (acción)

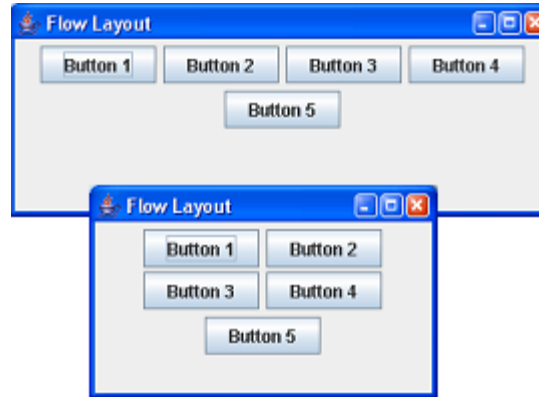
# Swing vs AWT

- Componentes renombradas (en Swing empiezan por J)
- Nuevas componentes
- *Look-and-feel* intercambiable
- No se deben mezclar componentes de Swing y AWT

# Layouts

- Alternativa para no definir posiciones absolutas (impone un orden)
- Layout Managers:
  - BorderLayout
  - FlowLayout
  - GridLayout
  - ...

# Ejemplos de Layout Managers



[Más](#)

# Algunas componentes

- [JButton](#)
- [JCheckBox](#)
- [JComboBox](#)
- [JList](#)
- [JRadioButton](#)
- [JTextField](#) / [JTextArea](#)
- [JLabel](#)
- [JPanel](#)
- [JFrame](#)
- [JMenu](#)
- [JDialog](#)
- [JFileChooser](#)
- [JSeparator](#)
- [JTable](#)





# Algunas componentes

- JButton
- JCheckBox
- JComboBox
- JList
- JRadioButton
- JTextField / JTextArea
- JLabel
- JPanel
- JFrame
- JMenu
- JDialog
- JFileChooser
- JSeparator
- JTable



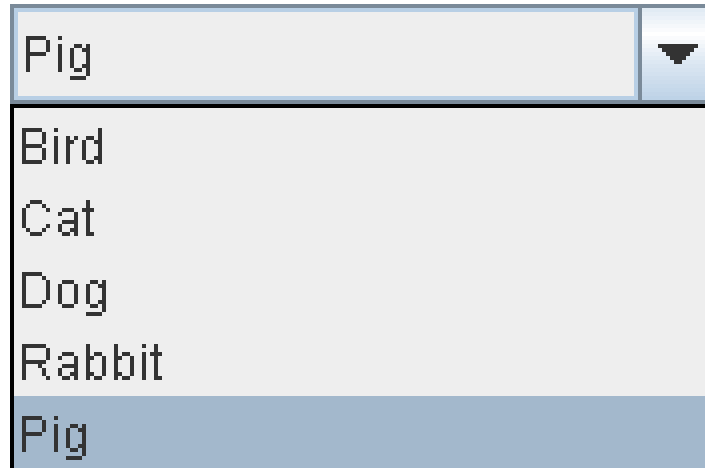
# Componentes Swing: JButton



# Componentes Swing: JCheckBox



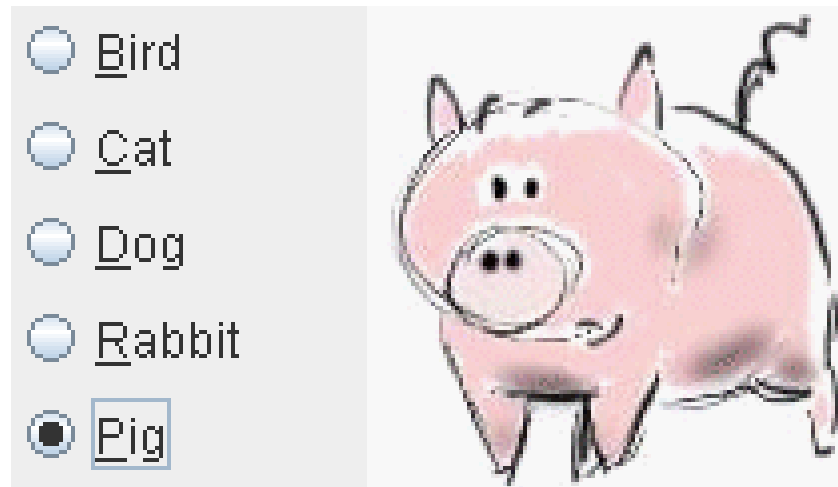
# Componentes Swing: JComboBox



# Componentes Swing: JList



# Componentes Swing: JRadioButton

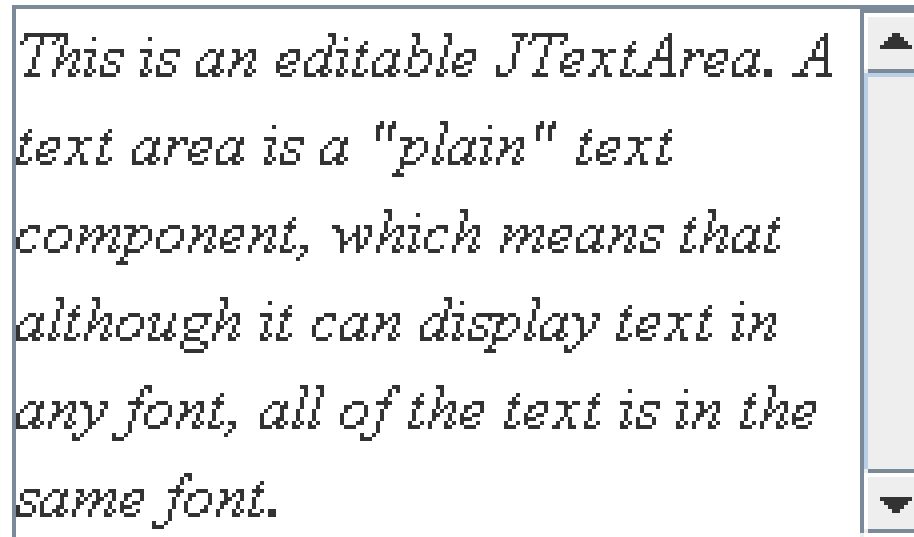


# Componentes Swing: JTextField

City:



# Componentes Swing: JTextArea

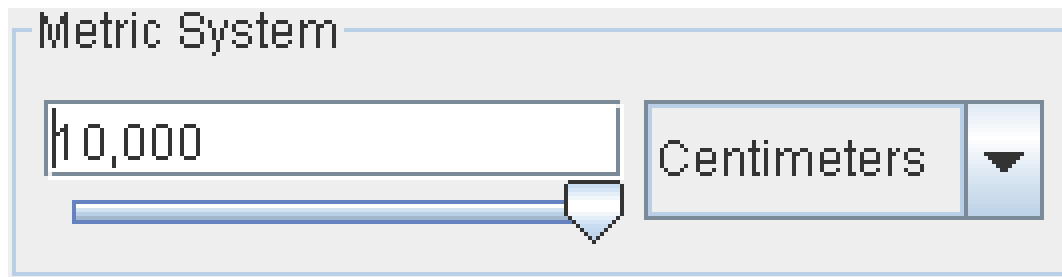




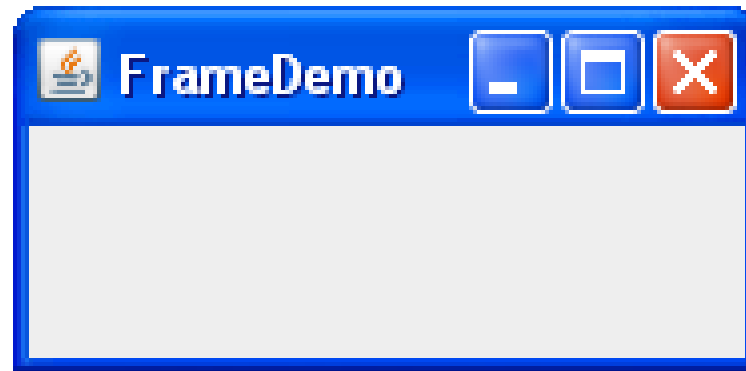
# Componentes Swing: JLabel



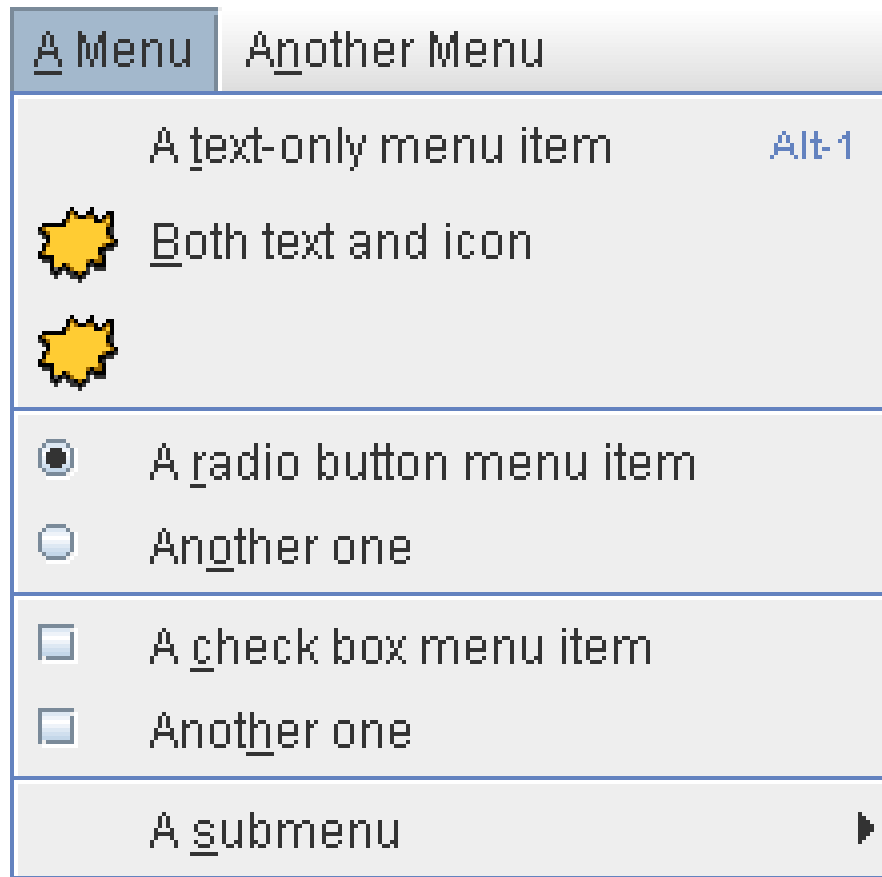
# Componentes Swing: JPanel



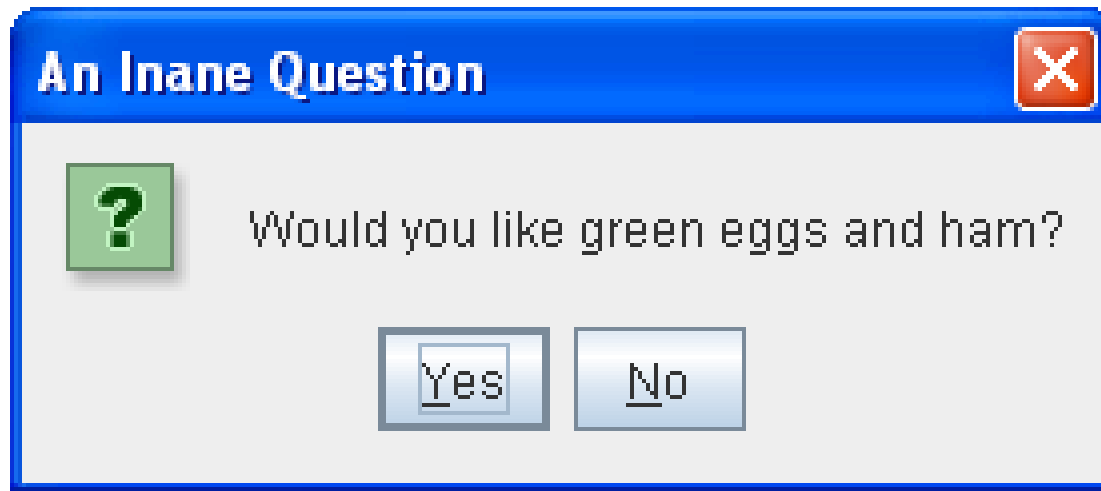
# Componentes Swing: JFrame



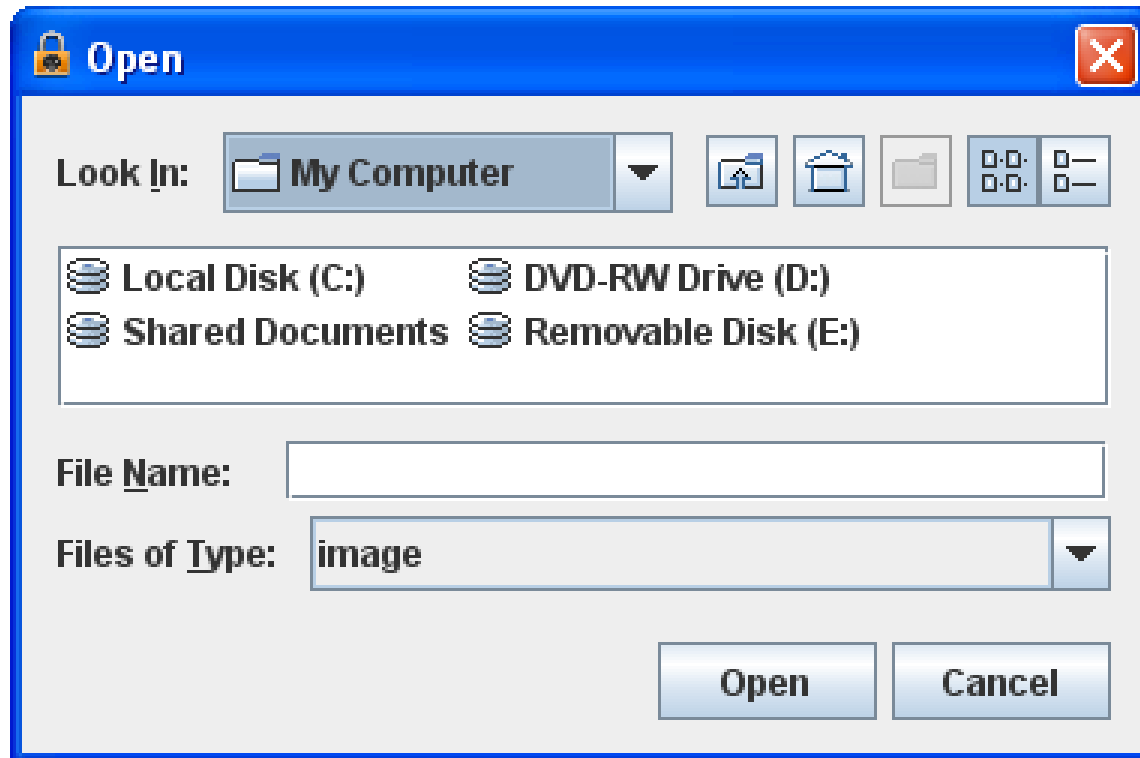
# Componentes Swing: JMenu



# Componentes Swing: JDialog



# Componentes Swing: JFileChooser



# Componentes Swing: JSeparator



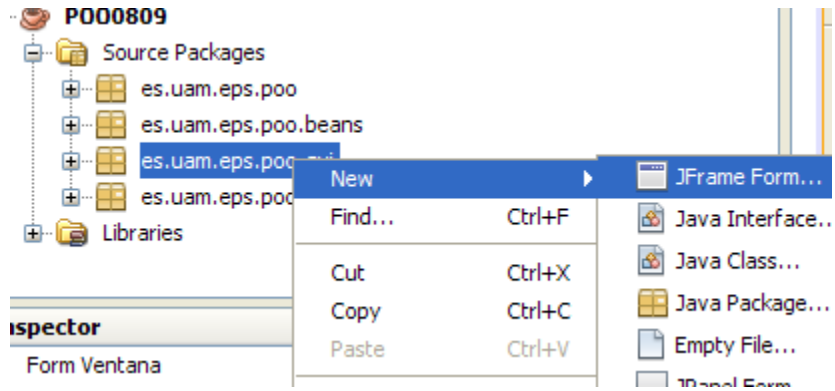
# Componentes Swing: JTable

Host	User	Password	Last Modified
Biocca Games	Freddy	!#asf6Awwzb	Mar 16, 2006
zabble	ichabod	Tazb!34\$fZ	Mar 6, 2006
Sun Developer	fraz@hotmail.co...	AasW541!fbZ	Feb 22, 2006
Heirloom Seeds	shams@gmail....	bkz[ADF78!	Jul 29, 2005
Pacific Zoo Shop	seal@hotmail.c...	vbAf1 24%z	Feb 22, 2006

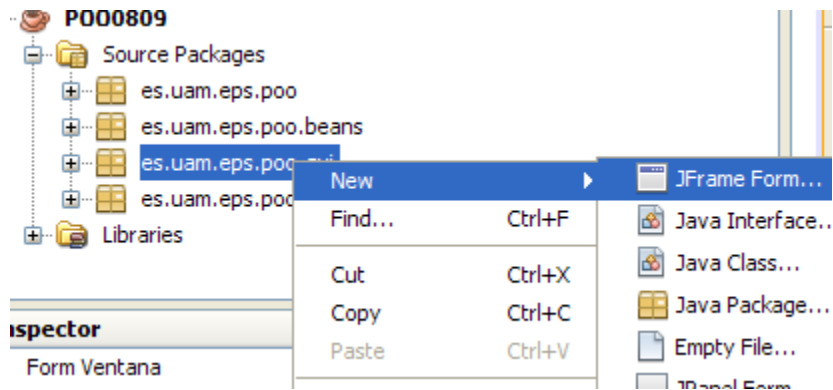




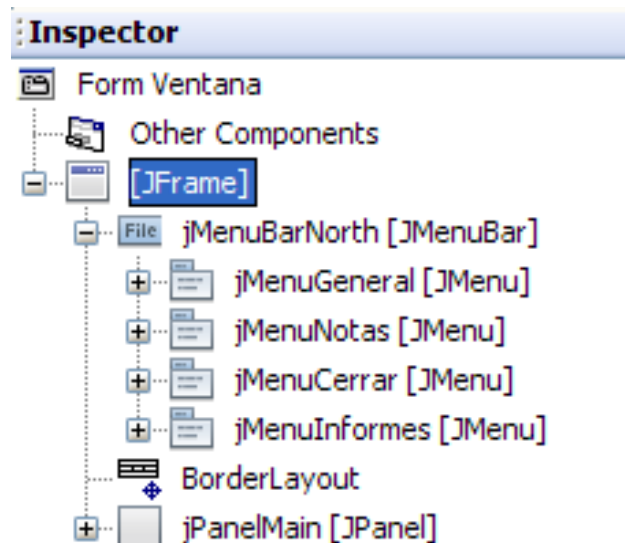
# GUIs + NetBeans (I)



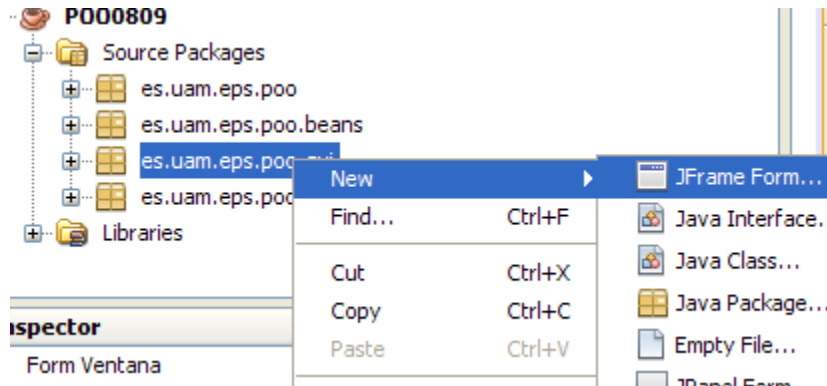
# GUIs + NetBeans (I)



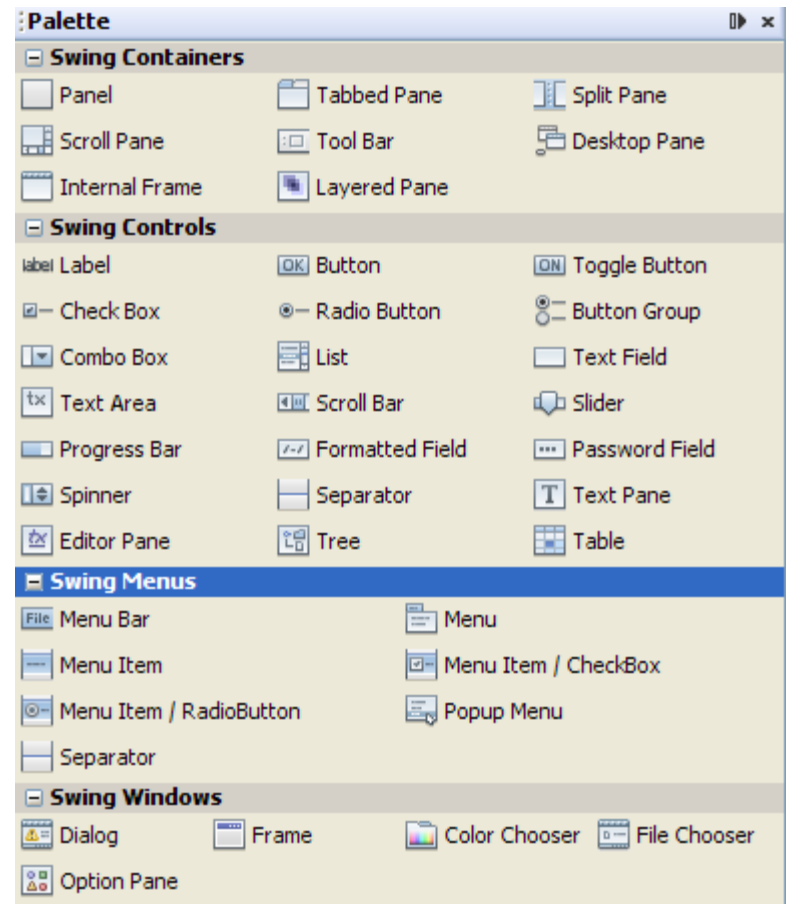
## Inspector



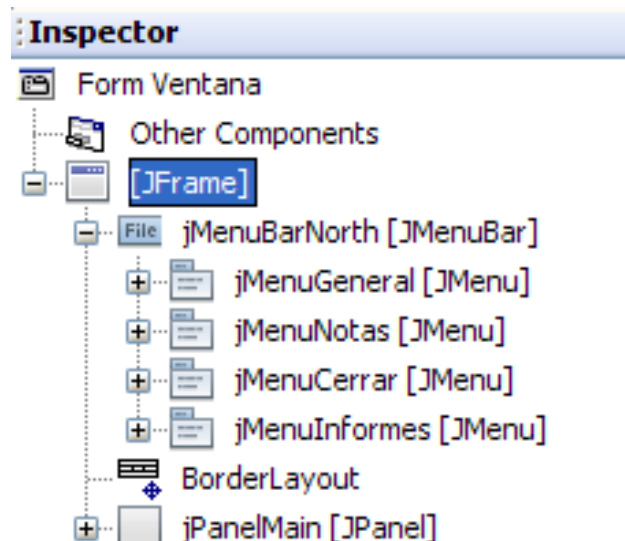
# GUIs + NetBeans (I)



Paleta



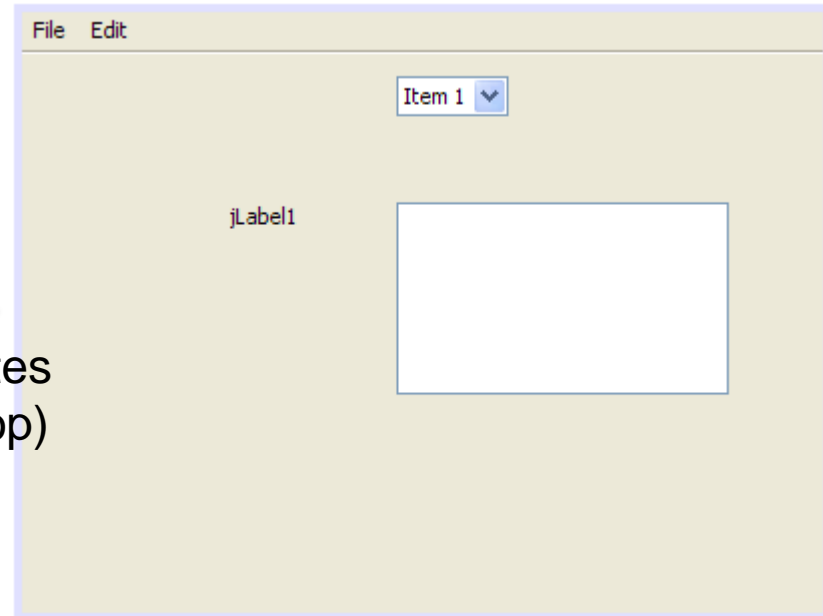
Inspector



# GUIs + NetBeans (II)



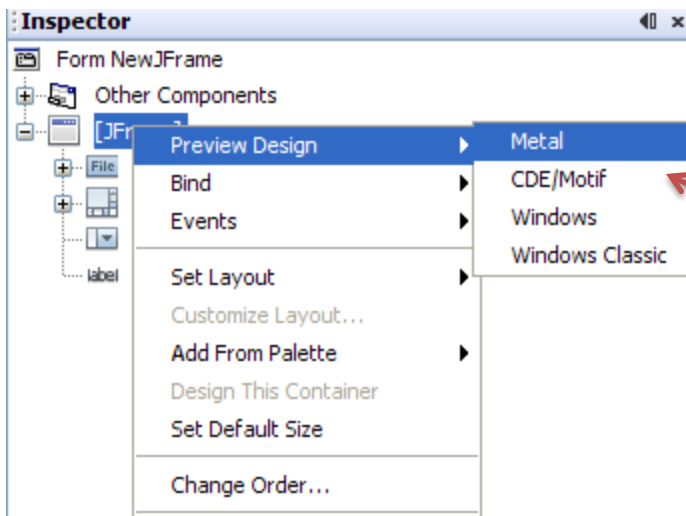
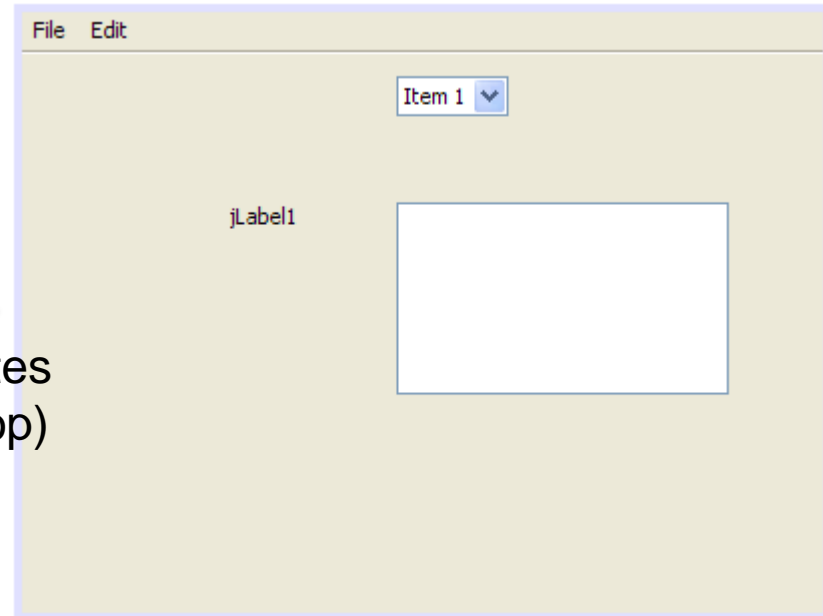
+ 4  
→  
componentes  
(drag & drop)



# GUIs + NetBeans (II)



+ 4  
→  
componentes  
(drag & drop)



distintos  
look & feel

