INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Programmation Orientée Objet

Sami Yangui, Ph.D.
A. Prof and CNRS LAAS Researcher

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Lecture 5: Java Swing

November 13, 2019

OUTLINE

- Definition of SWING
- JAVA AWT vs JAVA SWING
- JAVA SWING Classes

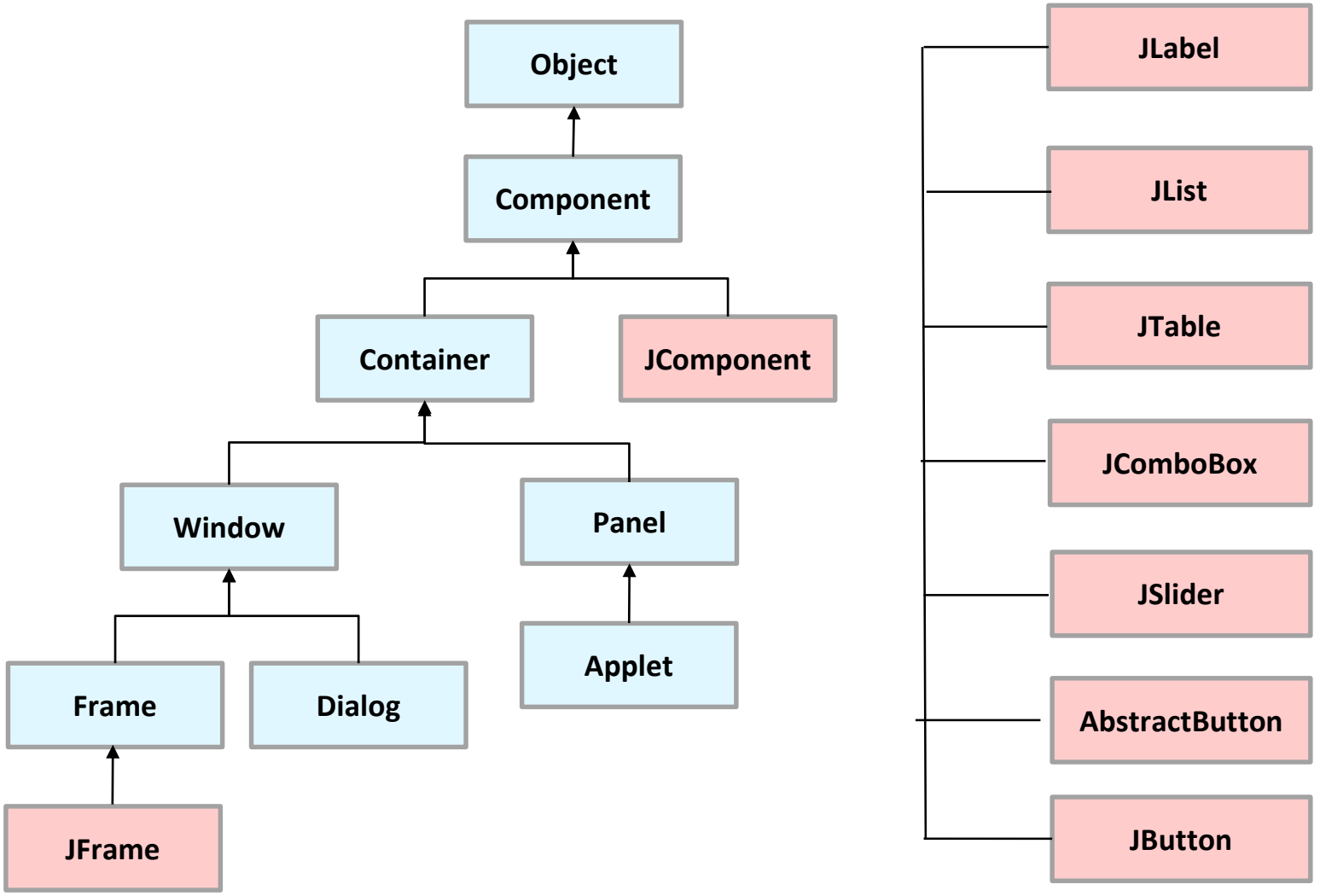
DEFINITION OF SWING

- Part of Java Foundation Classes (JFC) that is used to create window-based applications
  - Built on top of Abstract Windowing Toolkit (AWT) API
- Enables creating applications that use Graphical User Interfaces (GUI) instead of dull console interface
  - The SWING API provides many different classes for creating and handling various types of user interface entities

DIFFERENCES BETWEEN AWT AND SWING

JAVA AWT	JAVA SWING
Components are platform-dependent	Components are platform-independent
Components are heavyweight	Components are lightweight
Does not support pluggable look and feel	Supports pluggable look and feel
Provides less components than SWING	Provides more powerful components such as tables, lists, scrollpanes, etc.
Does not follow MVC	Follows MVC

JAVA SWING CLASSES



COMPONENT CLASS

- Represents an object that:
  - Has a visual representation
  - Can be shown on-screen
  - Can interact with users

– Defines some basic methods that are available to all Swing classes

void	**setSize**(int width, int height) Resizes this component so that it has width width and height height.
void	**setVisible**(boolean b) Shows or hides this component depending on the value of parameter b.
void	**show**() **Deprecated.** *As of JDK version 1.1, replaced by setVisible(boolean).*
void	**show**(boolean b) **Deprecated.** *As of JDK version 1.1, replaced by setVisible(boolean).*
Dimension	**size**() **Deprecated.** *As of JDK version 1.1, replaced by getSize().*

Comprehensive list is available at: https://docs.oracle.com/javase/8/docs/api/java/awt/Component.html

CONTAINER CLASS

- Builds on the basic visual capabilities of the ***Component*** class by adding the ability to hold other containers.

Methods

Modifier and Type	Method and Description
`Component`	`add(Component comp)` Appends the specified component to the end of this container.
`Component`	`add(Component comp, int index)` Adds the specified component to this container at the given position.
`void`	`add(Component comp, Object constraints)` Adds the specified component to the end of this container.
`void`	`add(Component comp, Object constraints, int index)` Adds the specified component to this container with the specified constraints at the specified index.
`Component`	`add(String name, Component comp)` Adds the specified component to this container.
`void`	`addContainerListener(ContainerListener l)` Adds the specified container listener to receive container events from this container.

Comprehensive list is available at: https://docs.oracle.com/javase/8/docs/api/java/awt/Container.html

WINDOW CLASS

- Specialized type of ***Container*** object that:
  - Has a border
  - Has a title bar
  - Has buttons that minimize, maximize, and close the window
  - Can be repositioned and possibly even resized by the user.

void	**hide**() **Deprecated.** *As of JDK version 1.5, replaced by* ***setVisible(boolean)***.
boolean	**isActive**() Returns whether this Window is active.
boolean	**isAlwaysOnTop**() Returns whether this window is an always-on-top window.
boolean	**isAlwaysOnTopSupported**() Returns whether the always-on-top mode is supported for this window.

Comprehensive list is available at: https://docs.oracle.com/javase/8/docs/api/java/awt/Window.html

WINDOW CLASS

- (Cont.)

void	**setLocation**(int x, int y) Moves this component to a new location.
void	**setLocation**(**Point** p) Moves this component to a new location.

void	**setOpacity**(float opacity) Sets the opacity of the window.
void	**setShape**(**Shape** shape) Sets the shape of the window.
void	**setSize**(**Dimension** d) Resizes this component so that it has width `d.width` and height `d.height`.
void	**setSize**(int width, int height) Resizes this component so that it has width `width` and height `height`.
void	**setType**(**Window.Type** type) Sets the type of the window.
void	**setVisible**(boolean b) Shows or hides this `Window` depending on the value of parameter `b`.
void	**show**() **Deprecated.** *As of JDK version 1.5, replaced by* ***setVisible(boolean)***.
void	**toBack**() If this Window is visible, sends this Window to the back and may cause it to lose focus or activation if it is the focused or active Window.
void	**toFront**() If this Window is visible, brings this Window to the front and may make it the focused Window.

Comprehensive list is available at: https://docs.oracle.com/javase/8/docs/api/java/awt/Window.html

WINDOW CLASS

```
Window w = new Window(Window owner, GraphicsConfiguration gc);

Rectangle bounds = gc.getBounds();

w.setLocation(10 + bounds.x, 10 + bounds.y);
```

FRAME CLASS

- Type of ***Window*** that serves as the basis for Java GUI applications.
- An AWT class that has been improved upon by the ***JFrame*** class.

`String`	**`getTitle`**`()` Gets the title of the frame.
`boolean`	**`isResizable`**`()` Indicates whether this frame is resizable by the user.
`boolean`	**`isUndecorated`**`()` Indicates whether this frame is undecorated.
`protected String`	**`paramString`**`()` Returns a string representing the state of this `Frame`.
`void`	**`remove`**`(`**`MenuComponent`**` m)` Removes the specified menu bar from this frame.
`void`	**`removeNotify`**`()` Makes this Frame undisplayable by removing its connection to its native screen resource.
`void`	**`setBackground`**`(`**`Color`**` bgColor)` Sets the background color of this window.
`void`	**`setCursor`**`(int cursorType)` **Deprecated.** *As of JDK version 1.1, replaced by `Component.setCursor(Cursor)`.*

Comprehensive list is available at: https://docs.oracle.com/javase/8/docs/api/java/awt/Frame.html

FRAME CLASS

- (Cont.)

void	**setResizable**(boolean resizable) Sets whether this frame is resizable by the user.
void	**setShape**(**Shape** shape) Sets the shape of the window.
void	**setState**(int state) Sets the state of this frame (obsolete).
void	**setTitle**(**String** title) Sets the title for this frame to the specified string.
void	**setUndecorated**(boolean undecorated) Disables or enables decorations for this frame.

Comprehensive list is available at: https://docs.oracle.com/javase/8/docs/api/java/awt/Frame.html

FRAME CLASS

```
Frame f = new Frame(GraphicsConfiguration gc);

Rectangle bounds = gc.getBounds();

f.setLocation(10 + bounds.x, 10 + bounds.y);
```

JFRAME CLASS

- The SWING version of the older ***Frame*** class. Most of the SWING applications include at least one ***JFrame*** object.

Constructor	**Description**
`JFrame ( )`	Creates a new frame with no title.
`JFrame (String title)`	Creates a new frame with the specified title.
Method	**Description**
`void add (Component c)`	Adds the specified component to the frame.

JFRAME CLASS

Method	Description
`JMenuBar getJMenuBar ( )`	Gets the menu for this frame.
`void pack ( )`	Adjusts the size of the frame to fit the components added to it.
`void remove (Component c)`	Removes the specified component from the frame.

JFRAME CLASS

Method	Description
`void remove (Component c)`	Removes the specified component from the frame.
`void setDefaultCloseOperation`	Sets the action taken when the user closes the frame. Always specify JFrame.EXIT ON CLOSE.

JFRAME CLASS

Method	Description
`void setIconImage (Icon image)`	Sets the icon displayed when the frame is minimized.
`void setLayout (LayoutManager layout)`	Sets the layout manager used to control how components are arranged when the frame is displayed. The default is the BorderLayout manager.

JFRAME CLASS

Method	Description
`void setLocation (int x, int y)`	Sets the x and y position of the frame on-screen. The top-left corner of the screen is 0, 0.
`void setLocationRelativeTo (Component c)`	Centers the frame on-screen if the parameter is null.

JFRAME CLASS

Method	Description
`void setResizeable (boolean value)`	Sets whether or not the size of the frame can be changed by the user. The default setting is true (the frame can be resized).

JFRAME CLASS

Method	Description
`void setSize (int width, int height)`	Sets the size of the frame to the specified width and height.
`void setJMenuBar(JMenuBarMenu)`	Sets the menu for this frame.

JCOMPONENT CLASS

- The basis for all other SWING components except for frames.

JPANEL CLASS

- Allows organizing and controlling the layout of other components such as labels, buttons, text fields, etc.
- Associated to a frame.
  - when the frame is displayed, the components that were added to its panels are made visible.

JPANEL CLASS

Constructor	Description
`JPanel ()`	Creates a new panel.
`JPanel (boolean isDoubleBuffered)`	Creates a new panel. If the parameter is true, the panel uses a technique called double-buffering.

JPANEL CLASS

Constructor	Description
`JPanel (LayoutManager layout)`	Creates a new panel with the specified layout manager. The default layout manager is FlowLayout.

JPANEL CLASS

Method	Description
`void add (Component c)`	Adds the specified component to the panel.
`void remove (Component c)`	Removes the specified component from the panel.

JPANEL CLASS

Method	Description
`void setLayout (LayoutManager layout)`	Sets the layout manager used to control how components are arranged when the panel is displayed. The default is the FlowLayout manager.

JPANEL CLASS

Method	Description
`void setLocation (int x, int y)`	Sets the x and y position of the frame-screen. The top-left corner of the screen is 0, 0.

JPANEL CLASS

Method	Description
`void setSize (int width, int height)`	Sets the size of the frame to the specified width and height.
`void setToolTipText (String text)`	Sets the tooltip text that's displayed if the user rests the mouse over an empty part of the panel.

JLABEL CLASS

- Creates a label that displays a simple text value.

Constructor	Description
`JLabel ( )`	Creates a new label with no initial text.
Method	**Description**
`String getText ( )`	Returns the text displayed by the label.
`void setText (String text)`	Sets the text displayed by the label.

JLABEL CLASS

Method	Description
`void setToolTipText (String text)`	Sets the tooltip text that's displayed if the user rests the mouse over the label for a few moments.
`void setVisible (boolean value)`	Shows or hides the label.

JBUTTON CLASS

- Creates a button the user can click
- The constructors of the ***JButton*** class are similar to the constructors for the ***JLabel*** class.
  - We either create an empty button or a button with text.

Constructor	Description
`JButton ( )`	Creates a new button with no initial text.
`JButton (String text)`	Creates a new button with the specified text.

JBUTTON CLASS

Method	Description
`doClick ( )`	Triggers an action event for the button as if the user clicked it.
`String getText ()`	Returns the text displayed by the button.

JBUTTON CLASS

Method	Description
`void setBorderPainted (boolean value)`	Shows or hides the button's border. The default setting is true (the border is shown).
`void setContentAreaFilled (boolean value)`	Specifies whether or not the button's background should be filled or left empty. The default setting is true (the background is filled in).

JBUTTON CLASS

Method	Description
`void setContentAreaFilled (boolean value)`	Specifies whether or not the button's background should be filled or left empty. The default setting is true (the background is filled in).
`void setEnabled (boolean value)`	Enables or disables the button. The default setting is true (enabled).

JBUTTON CLASS

Method	Description
`void setRolloverEnabled (boolean value)`	Enables or disables the rollover effect, which causes the border to get thicker when the mouse moves over the button. The default setting is true (rollover effect enabled).

JBUTTON CLASS

Method	Description
`void setText (String text)`	Sets the text displayed by the button.
`void setToolTipText (String text)`	Sets the tooltip text that's displayed if the user lets the mouse rest over the button.
`void setVisible (boolean value)`	Shows or hides the button. The default setting is true (the button is visible).