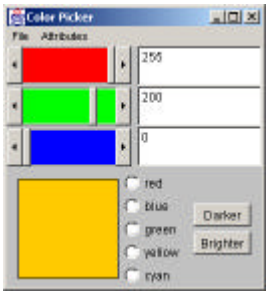
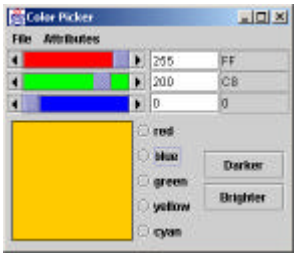
Graphical User Interfaces (GUI)

- **GUI contains elements**
  - which can be presented in a window
  - which can react on events (mouse, keys)



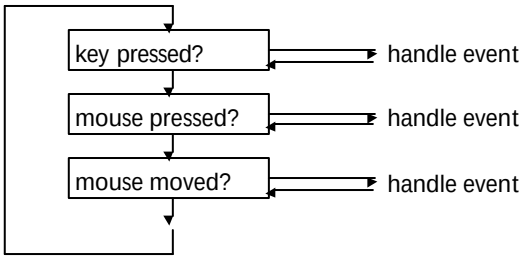


Java GUIs

- **AWT (Abstract Windowing Toolkit)**
  - since JDK 1.0
  - uses native controls
  - appearance / behavior depends on platform
  - least common denominator of functionality
- **Swing**
  - since JDK 1.2
  - implemented completely in Java (light weight)
  - several "Look & Feel" modes on all platforms available
  - more features (tooltips, icons, table- & tree-control)
  - requires Java plug-in on pre 1.2 JVMs (http://java.sun.com/products/plugin)
- **SWT**
  - simple windowing toolkit, OTI

Event Driven Programming

- **Events = User Interactions**
  - mouse click
  - key pressed
  - menu selection
- **Event Driven Programming**
  - definition of actions to be performed upon events



Event Handling in AWT/Swing

- **Events**
  - class **XXXEvent** extends EventObject { … }
- **Listener Interface**
  - interface **XXXListener** {
    void **handleEvent**(**XXXEvent** e);
    }
- **Registration in a GUI control**
  - **addXXXListener(**XXXListener l);
  - **removeXXXListener**(XXXListener l);

Example:

ActionEvent

interface **ActionListener** {
  void **actionPerformed**(
    ActionEvent e);
}

addActionListener(
  ActionListener l);
removeActionListener(
  ActionListener l);

Example: ActionEvent

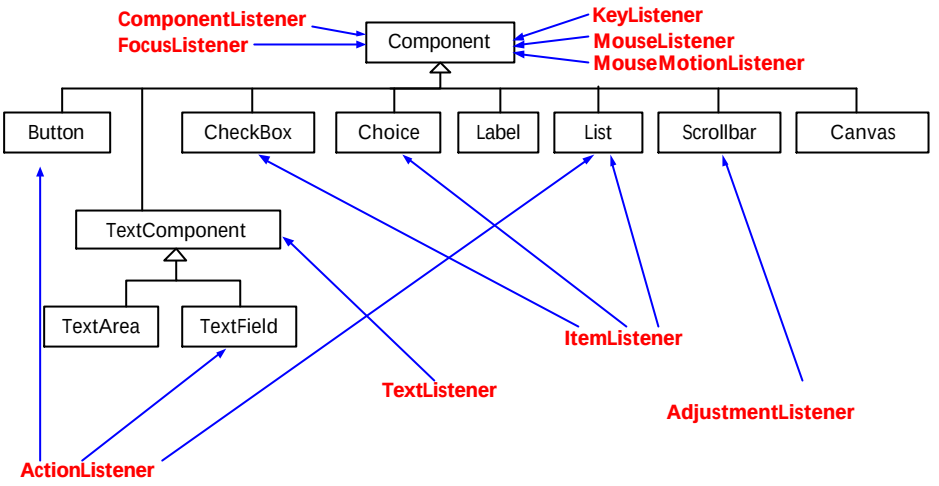
```
import java.awt.*;
import java.awt.event.*;

public class ButtonApp extends Frame {
    public static void main(String[] args){
        Frame f  = new Frame();
        Button b = new Button("Beep");
        b.addActionListener(new ButtonActionListener());

        f.add(b); f.pack(); f.setVisible(true);
    }
}

class ButtonActionListener implements ActionListener {
    public void actionPerformed(ActionEvent e){
        System.out.println("Beep pressed");
    }
}
```


Component

- **Component**
  - abstract base class of the nonmenu-related AWT components
  - class **Component** {
    - getWidth **1.2**
    - getHeight **1.2**
    - get/setSize
    - get/setLocation
    - get/setBounds
    - getPreferredSize
    - is/setEnabled
    - is/setVisible
    - get/setCursor

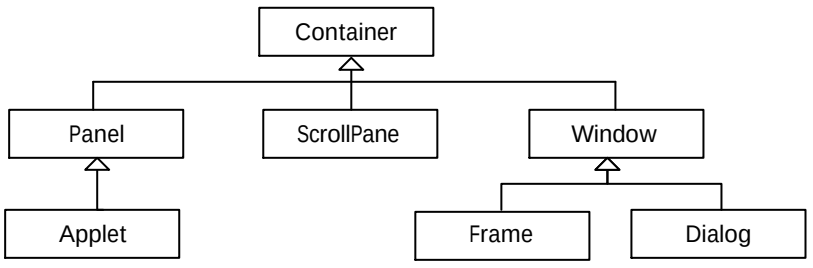
    }

Components and their Listeners



Container

- **Container = object which can contain other components**



Container

- class **Container**
  - void add(Component c)
    void add(String name, Conponent c)
    void add(Component c, Object constraint)
  - void remove(Component c)
  - int getComponentCount()
  - Component[] getComponents()
  - Component getComponentAt(int x, int y)
  - add/removeContainerListener
    - ContainerListener:
      - componentAdded(ContainerEvent e)
      - componentRemoved(ContainerEvent e)

Layout

- **How are components placed in a container?**
  - absolute placing

```
c.setLayout(null);
c.add(button);
button.setBounds(new Rectangle(10,10,40,20))
```

  - automatic placing, with a layout manager

```
c.setLayout(new FlowLayout());
c.add(button)
```

    - BorderLayout
    - FlowLayout
    - GridLayout
    - GridBagLayout

Window

- class **Window**
  - pack() // computes the size of the window
  - setSize(int w, int h) // sets the size of the window, called by pack
  - show() / hide()
  - setVisible(boolean visible) // shows window on screen
  - toFront() / toBack()
  - dispose() // releases OS resources

  - add/removeWindowListener
    - WindowListener
      - windowActivated
      - windowDeactivated
      - windowClosing
      - windowClosed
      - windowIconified
      - windowDeiconified

Frame & Dialog

- **Frame extends Window**
  - get/setTitle
  - get/setMenubar
  - is/setResizable
  - get/setState (NORMAL | ICONIFIED)

- **Dialog extends Window**
  - dialog is associated to another window, no entry in task bar
    - Dialog(Frame owner, String title, boolean modal);
    - Dialog(Dialog owner, String title, boolean modal);

Menus

- **MenuBar**
  - MenuBar()
  - add(Menu)
- **Menu**
  - Menu(String label)
  - Menu(String label, boolean tearoff)
  - add(MenuItem)
  - add(String)
  - addSeparator()
- **MenuItem**
  - MenuItem(String label)
  - MenuItem(String label, MenuShortcut s)
  - get/setLabel
  - is/setEnabled
  - addActionListener

Simple Application (1/3)

```
import java.awt.*;
import java.awt.event.*;

public class SimpleApp extends Frame implements ActionListener {
    public static void main(String[] args){
        new SimpleApp();
    }

    TextField text   = new TextField(20);
    Button    button = new Button("Submit");

    public void actionPerformed(ActionEvent e){
        System.out.println("Submit: " + text.getText());
    }
```

Simple Application (2/3)

```
SimpleApp(){
        super("Simple Application");          // set title
        setLayout(new FlowLayout());
        add(text);
        add(button);  button.addActionListener(this);

        MenuBar mbar = new MenuBar();    setMenuBar(mbar);
        Menu    m    = new Menu("File");  mbar.add(m);
        m.add("New");
        m.addSeparator();
        m.add(new MenuItem("Exit"));

        addWindowListener(new CloseListener());
        pack();
        setVisible(true);
    }
}
```


Simple Application (3/3)

```
class CloseListener extends WindowAdapter {
    public void windowClosing(WindowEvent e){
        System.exit(0);
    }
}
```