Creating User Interfaces

B.Tech. (IT), Sem-5,
Core Java Technology (CJT)

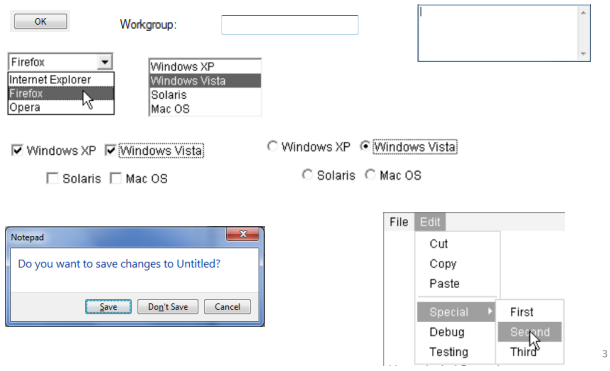
Dharmsinh Desai University
Prof. (Dr.) H B Prajapati

Updated: 13 August 2020

User Interface Components

- We will use various AWT user interface components to create GUI applications.
- Widely used Components are
  - Button
  - Label
  - TextField and TextArea
  - Choice and List
  - Checkbox and CheckboxGroup
  - Dialog
  - Menu

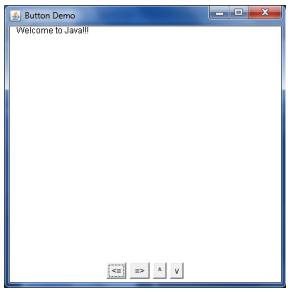
User Interface Components



Button

- Button component triggers an event when it is pressed or clicked.
- Constructors
  - public Button(): Creates an empty button with no label
  - public Button(String s): Creates a button with label
- Methods
  - public String getLabel(): Get the current label
  - public void setLabel(String s): Set the new label

Example: Use of Button



Example: Use of Button

```
import java.awt.*;
import java.awt.event.*;
class ButtonDemo extends Frame implements ActionListener{
    private MovingCanvas c;
    private Button left;
    private Button right;
    private Button up;
    private Button down;
```

Example: Use of Button

```
ButtonDemo(String title){
    super(title);
    c=new MovingCanvas("Welcome to Java!!!");
    left=new Button("<=");
    right=new Button("=>");
    up=new Button("^");
    down=new Button("v");
    add(c);
```


Example: Use of Button

```
    Panel p=new Panel();
    p.add(left);
    p.add(right);
    p.add(up);
    p.add(down);
    add("South",p);
```


Example: Use of Button

```
    setSize(400,400);
    setVisible(true);
    left.addActionListener(this);
    right.addActionListener(this);
    up.addActionListener(this);
    down.addActionListener(this);
}
public static void main(String [] args){
    new ButtonDemo("Button Demo");
}
```


Example: Use of Button

```
public void actionPerformed(ActionEvent e){
    String str=e.getActionCommand();
    if( e.getSource() instanceof Button)
        if(str.equals("<="))
            c.left();
        else if(str.equals("=>"))
            c.right();
        else if(str.equals("^"))
            c.up();
        else if(str.equals("v"))
            c.down();
}
}
```


Example: Use of Button

```
class MovingCanvas extends Canvas{
    String s;
    int x=10,y=10;
    MovingCanvas(String str){
        s=str;
    }
    public void paint(Graphics g){
        g.drawString(s,x,y);
    }
```

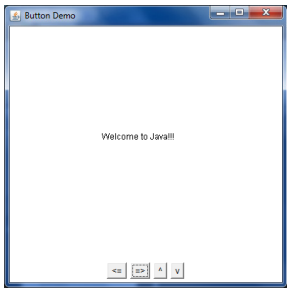

Example: Use of Button

```
public void left(){
    if(x>10)
        x-=10;
    repaint();
}
public void right(){
    if(x<250)
        x+=10;
    repaint();
}
```

Example: Use of Button

```
    public void up(){
        if(y>10)
                y-=10;
        repaint();
    }
    public void down(){
        if(y<250)
                y+=10;
        repaint();
    }
}
```


Example: Use of Button



Label

- Labels are simple strings used to label other components, e.g., TextField
- Constructors:
  - Label(String s): Label having string s
  - Label(): Creates an empty label
  - Label(String s, int alignment): Label having string s and alignment.
- Alignment
  - Label.LEFT, Label.RIGHT, Label.CENTER

Label

- Methods:
  - public String getText(): Get current label
  - public void setText(String s): Set new label
  - public int getAlignment(): Get current alignment
  - public void setAlignment(int alignment): Set new alignment

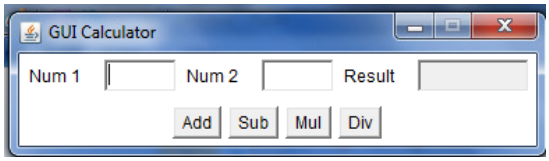
TextField

- TextField is a component in which a user can type characters. E.g., Name, Age, etc..
- Constructors
  - TextField(int width) Creates an empty text field with the specified number of columns
  - TextField(String s) Creates a text field with initial text s.
  - TextField(String s, int width) Creates a text field with initial text s and column width

TextField

- Methods
  - public String getText(): returns the string present in the text field
  - public setEditable(boolean editable): Enable or disable the text field to be edited. Default it is true.
  - public void setColumn(int column): Sets the number of columns in the text field.

TextField Example: GUI Calculator



TextField Example: GUI Calculator

```
import java.awt.*;
import java.awt.event.*;
class GUICal extends Frame implements
   ActionListener{
   TextField tfOp1;
   TextField tfOp2;
   TextField tfResult;
   Button add,sub,mul,div;
```


TextField Example: GUI Calculator

```
public static void main(String[] args){
      Frame f=new GUICal("GUI Calculator");
      // f.setSize(300,100);
      f.pack();
      f.setVisible(true);
}
GUICal(String title){
      super(title);
      tfOp1=new TextField(4);
      tfOp2=new TextField(4);
      tfResult=new TextField(8);
      tfResult.setEditable(false);
```


TextField Example: GUI Calculator

```
      Panel p=new Panel();
      p.add(new Label("Num 1"));
      p.add(tfOp1);
      p.add(new Label("Num 2"));
      p.add(tfOp2);
      p.add(new Label("Result"));
      p.add(tfResult);
```


TextField Example: GUI Calculator

```
      Panel pb=new Panel();
      pb.add(add=new Button("Add"));
      pb.add(sub=new Button("Sub"));
      pb.add(mul=new Button("Mul"));
      pb.add(div=new Button("Div"));

      add("North",p);
      add("South",pb);
```


TextField Example: GUI Calculator

```
      add.addActionListener(this);
      sub.addActionListener(this);
      mul.addActionListener(this);
      div.addActionListener(this);
      setResizable(false);
}
```

TextField Example: GUI Calculator

```
public void actionPerformed(ActionEvent e){
    String arg=e.getActionCommand();
    int no1,no2;
    no1=Integer.parseInt(tfOp1.getText().trim());
    no2=Integer.parseInt(tfOp2.getText().trim());
    if(arg.equals("Add")){
        int result=no1+no2;
        tfResult.setText(""+result);
    }
```

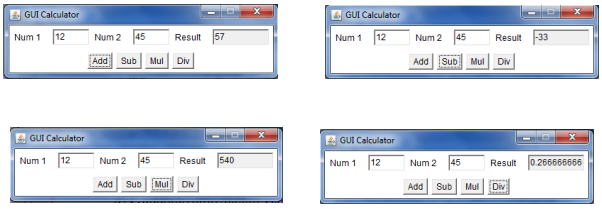

TextField Example: GUI Calculator

```
    else if(arg.equals("Sub")){
        int result=no1-no2;
        tfResult.setText(""+result);
    }
    else if(arg.equals("Mul")){
        int result=no1*no2;
        tfResult.setText(""+result);
     }
```


TextField Example: GUI Calculator

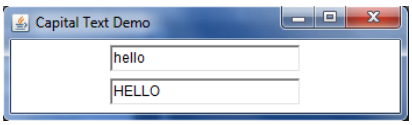
```
    else if(arg.equals("Div")){
        double result=((no1*1.0)/no2);
        tfResult.setText(""+result);
    }
  }
}
```


TextField Example: GUI Calculator



TextField Example: CapsText

- Show typed characters in upper case as they are typed



TextField Example: CapsText

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

class CapsText extends Frame implements TextListener{
    TextField tf1;
    TextField tf2;

    public CapsText(String title){
        super(title);
        setLayout(new FlowLayout());
        tf1=new TextField(20);
        tf2=new TextField(20);
```

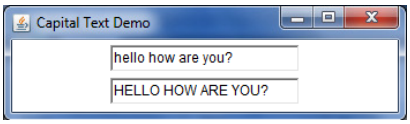
TextField Example: CapsText

```
        add(tf1);
        add(tf2);
        tf1.addTextListener(this);
        setVisible(true);
    }
    public static void main(String[] args){
        Frame f=new CapsText("Capital Text Demo");
        f.setSize(350,100);
    }
```


TextField Example: CapsText

```
    public void textValueChanged(TextEvent te){
        String s=tf1.getText();
        tf2.setText(s.toUpperCase());
    }
}
```


TextField Example: CapsText



TextArea

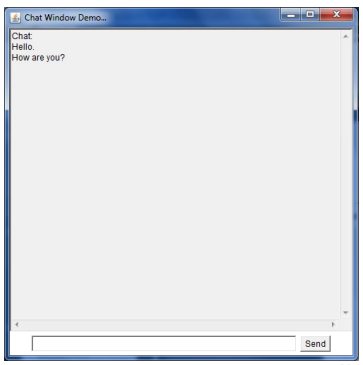
- If we want to enter multiple lines of text, we can use TextArea component.
- Constructors
  - public TextArea()
  - public TextArea(String str)
  - public TextArea(int rows, int columns): Creates text area with the specified number of rows and columns
  - public TextArea(String str, int rows, int columns): Creates text area with the initial text and the specified number of rows and columns

TextArea

- Methods:
  - public void insert(String s, int pos): Inserts the string at the specified position in the text area
  - public append(String s): Appends the string s to the end of the text in text area
  - public void replaceRange(String s, int start, int end): Replaces partial texts in the range from position start to position end with string s
  - public int getRows(): Returns the number of rows in the text area.

Example TextArea: Chat Window

- Action gets triggered using two ways:
  1. Fire Send button
  2. Hit enter key after typing text



Example TextArea: Chat Window

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

class ChatWindowDemo extends Frame implements
   ActionListener{
   TextArea ta;
   TextField tf;
   Button bt;
   public static void main(String []args){
        Frame f=new ChatWindowDemo("Chat Window
   Demo...");
   }
```


Example TextArea: Chat Window

```
ChatWindowDemo(String title){
      super(title);
      ta=new TextArea("Chat:");
      ta.setEditable(false);
      tf=new TextField(50);
      bt=new Button("Send");

      Panel p=new Panel();
      p.setLayout(new FlowLayout());
      p.add(tf);
      p.add(bt);
      add("Center",ta);
      add("South",p);
```


Example TextArea: Chat Window

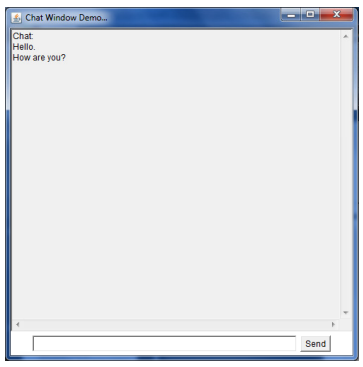
```
      bt.addActionListener(this);
      tf.addActionListener(this);

      setSize(500,500);
      setVisible(true);
      tf.requestFocus();
}
```


Example TextArea: Chat Window

```
public void actionPerformed(ActionEvent e){
      String arg=e.getActionCommand();
      if( (e.getSource() instanceof Button)||
            (e.getSource() instanceof TextField)
      ){
            ta.append("\n"+tf.getText());
            tf.setText("");
      }
      tf.requestFocus();
   }
}
```


Example TextArea: Chat Window



Choice

- Choice is a simple list of items from which a user can choose.
- A TextField cannot restrict value. But using Choice we can restrict possible values and no need to perform data validation.
- Constructors:
  - Choice(): It creates a choice component

Choice

- Methods
  - public void addItem(String s): Adds the item s into the choice
  - public String getItem(int index): Gets an item present at the specified index.
  - public int getSelectedIndex(): Gets the index of the selected item.
  - public String getSelectedItem(): Gets the selected item
  - public void select(int pos): Selects the item with the specified position
  - public void select(String str): Selects the item with the specified string str.

Event Handling for Choice

- Choice component responds to itemStateChanged(ItemEvent e) handler, which is present in ItemListener interface.
- We can get selected item inside itemStateChanged() handler using e.getItem()
- We can also get selected item using getSelectedItem() method of choice component.

List

- A List is a component that performs the same functions as a Choice.
- A List can allow selection of one or multiple values.
- Constructors:
  - public List(int rows, boolean multipleSelection): The rows indicates how many rows are visible in a scrolling list and multipleSelection indicates whether multiple items can be selected or not.
  - public List(int rows): Creates a List that allows only single selection
  - public List(): Creates a List with no visible lines or multiple selection.

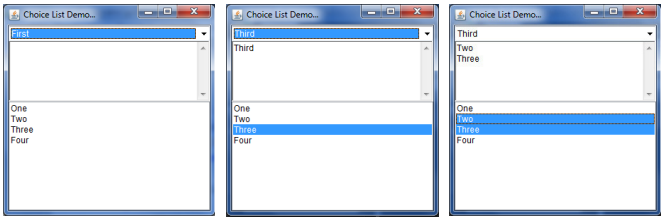
List

- Methods
  - public void addItem(String s): Adds the item s into the list
  - public String getItem(int row): Gets an item present at the specified row.
  - public int getSelectedIndex(): Gets the index of the selected item.
  - public String getSelectedItem(): Gets the selected item
  - public String[] getSelectedItems(): Gets the selected item

Event Handling for List

- List may generate ActionEvent and ItemEvent
  - Single click generates ItemEvent
  - Double click generates ActionEvent

Example: Choice List Demo



Example: Choice List Demo

```
import java.awt.*;
import java.awt.event.*;
class ChoiceListDemo extends Frame implements
    ItemListener{
    Choice c;
    List l;
    TextArea ta;
    public static void main(String[] args){
        Frame f=new ChoiceListDemo("Choice List Demo...");
        f.setSize(300,300);
        f.setVisible(true);
    }
```


Example: Choice List Demo

```
ChoiceListDemo(String title){
        super(title);
        c=new Choice();
        c.add("First");
        c.add("Second");
        c.add("Third");
        c.add("Fourth");

        l=new List(10,true);
        l.add("One");
        l.add("Two");
        l.add("Three");
        l.add("Four");
```


Example: Choice List Demo

```
        ta=new TextArea();

        add("North",c);
        add("South",l);
        add("Center",ta);
        c.addItemListener(this);
        l.addItemListener(this);
    }
```

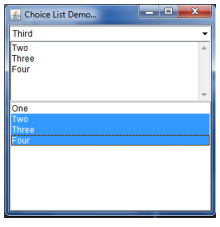

Example: Choice List Demo

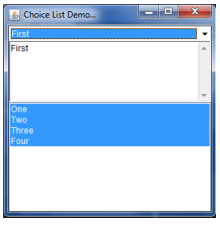
```
public void itemStateChanged(ItemEvent e){
    if(e.getSource().equals(c)){
            String s=c.getSelectedItem();
            System.out.println("Selected item: "+s);
            l.select(c.getSelectedIndex());
            ta.setText(s);
    }
```


Example: Choice List Demo

```
        else if(e.getSource().equals(l)){
                String[]ls =l.getSelectedItems();
                StringBuffer sb=new StringBuffer();
                for(int i=0;i<ls.length;i++){
                                sb.append(ls[i]+"\n");
                                ta.setText(sb.toString());
                }
        }
    }
}
```


Example: Choice List Demo





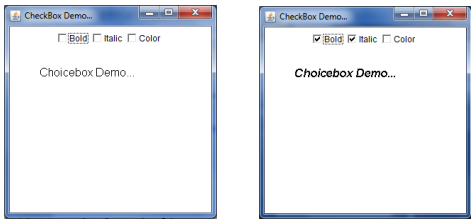
Checkbox

- A check box enables the user to toggle a choice on or off.
- It is used when we have possible choices for input and ask the user to select relevant choices.
- Constructors:
  - public Checkbox(String label): Creates a check box with the specified label
- Methods:
  - public boolean getState(): Returns whether the Checkbox is selected or not.

Checkbox

- Event handling
  - Checkbox can generate ItemEvent
  - The handler is itemStateChanged() declared in ItemListener.

Example: Choicebox



Example: Choicebox

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

class CheckBoxDemo extends Frame implements
    ItemListener{
    Checkbox cbBold;
    Checkbox cbItalic;
    Checkbox cbColor;

    boolean bold;
    boolean italic;
    boolean color;
```


Example: Choicebox

```
String message="Choicebox Demo...";

public static void main(String[] args){
      Frame f=new CheckBoxDemo("CheckBox Demo...");
      f.setSize(300,300);
      f.setVisible(true);
}
public CheckBoxDemo(String title){
      super(title);
      //Panel p1=new Panel();
      cbBold=new Checkbox("Bold");
      cbItalic=new Checkbox("Italic");
      cbColor=new Checkbox("Color");
```


Example: Choicebox

```
      setLayout(new FlowLayout());
      add(cbBold);
      add(cbItalic);
      add(cbColor);

      cbBold.addItemListener(this);
      cbItalic.addItemListener(this);
      cbColor.addItemListener(this);
}
```

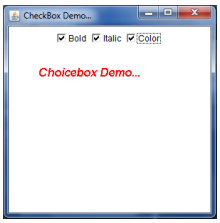
Example: Choicebox

```
public void itemStateChanged(ItemEvent e){
        bold=italic=color=false;
        if(cbBold.getState()){
                bold=true;
        }
        if(cbItalic.getState()){
                italic=true;
        }
        if(cbColor.getState()){
                color=true;
        }
        repaint();
}
```


Example: Choicebox

```
    public void paint(Graphics g){
        int fontStyle=0;
        if(bold)
                fontStyle+=Font.BOLD;
        if(italic)
                fontStyle+=Font.ITALIC;
        if(color)
                g.setColor(new Color(255,0,0));
        Font f=new Font("Arial",fontStyle,16);
        g.setFont(f);
        g.drawString(message,50,100);
    }
}
```

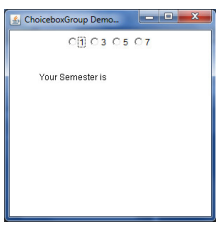

Example: Choicebox



CheckboxGroup

- CheckboxGroup is known as option buttons (select any one out of many choices).
- If one choice is selected, the other gets unselected automatically.
- We create Checkbox objects and add them in an instance of CheckboxGroup to make Checkboxes as options/radio buttons.
- Constructor:
  - Checkbox(String choiceName, CheckboxGroup cbg, boolean initialState)

Example: CheckboxGroup



Example: CheckboxGroup

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

class CheckBoxGroupDemo extends Frame implements
    ItemListener{
    Checkbox c1;
    Checkbox c2;
    Checkbox c3;
    Checkbox c4;
    CheckboxGroup cbg;
    String s="Your Semester is ";
```

Example: CheckboxGroup

```
public static void main(String[] args){
    Frame f=new CheckBoxGroupDemo("ChoiceboxGroup
Demo...");
    f.setSize(300,300);
    f.setVisible(true);
}
public CheckBoxGroupDemo(String title){
    super(title);
    setLayout(new FlowLayout());
    cbg=new CheckboxGroup();
```


Example: CheckboxGroup

```
    c1=new Checkbox("1",cbg,false);
    c2=new Checkbox("3",cbg,false);
    c3=new Checkbox("5",cbg,false);
    c4=new Checkbox("7",cbg,false);
    add(c1);
    add(c2);
    add(c3);
    add(c4);
    c1.addItemListener(this);
    c2.addItemListener(this);
    c3.addItemListener(this);
    c4.addItemListener(this);
}
```


Example: CheckboxGroup

```
public void itemStateChanged(ItemEvent e){
    s="Your Semester is ";
    if(c1.getState()){
        s=s+c1.getLabel();
    }
    else if(c2.getState()){
        s=s+c2.getLabel();
    }
    else if(c3.getState()){
        s=s+c3.getLabel();
    }
```


Example: CheckboxGroup

```
    else if(c4.getState()){
        s=s+c4.getLabel();
    }
    repaint();
}
public void paint(Graphics g){
        g.drawString(s,50,100);
}
}
```


Example: CheckboxGroup