Applets and Advanced Graphics

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

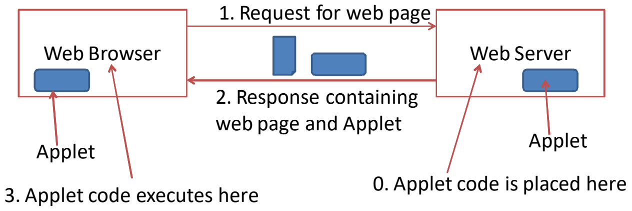
Dharmsinh Desai University
Prof. (Dr.) H B Prajapati

Updated: 13 September 2020

Applet

- Applet is a Java program that runs inside Web-browser.
- It is used to create dynamic GUI for web pages.
- Applets share many common programming features.
- However, in certain aspects, Applet and Application differ
  - Applications have main() method.
  - Applets do not have main() method.
- Generally, Applets are placed on server (e.g., web server) and are accessed from a client machine (e.g., using Web browser)

Location of Applet code and execution of Applet


1. Request for web page
Web Browser
Web Server
2. Response containing web page and Applet
Applet
Applet
3. Applet code executes here
0. Applet code is placed here

How to Embed Applet in a web-page?

- To run an applet the HTML page must contain applet tag.
- The <applet> tag and its basic attributes:
  - code: to specify the applet bytecode file
  - width and height: to specify applet viewing area
- Example:

```
<applet
code=classfilename.class
width=width-in-pixels
height=height-in-pixel>
</applet>
```

In applet tag, code, width, and height are compulsory attributes

Applet

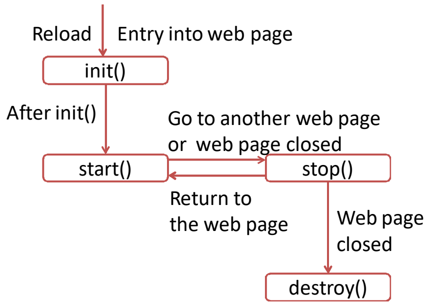
- Every applet is a subclass of java.applet.Applet
- The Applet class is an AWT class and is not designed to work with Swing components.
- To use Swing components in Java applets, you need to create a Java applet that extends javax.swing.JApplet, which is a subclass of java.applet.Applet.
- Applet provides the essential framework to enable applets to be run by a Web browser.

Structure of Applet

```
public class MyApplet extends Applet{
    …
/**Called by the browser when the page containing this applet is loaded
    */
    public void init() { … }
/**Called by the browser after init() and every time the web page is
    visited */
    public void start() { … }
/** Called by the browser when the user visits the other web page */
    public void stop() { … }
/** Called by the browser when the web browser exits*/
    public void destroy() { … }
}
```

Life-cycle of Applet

- The methods of Applet class do not do anything, but overridden methods do



Methods of Applet Life-cycle

- The init() method
  - The init() method is invoked when the applet is first loaded and again if it is reloaded.
  - This method should contain code related to initialization.
    - Creating new threads
    - Loading images
    - Setting-up user-interface components
    - Getting parameters from the <Applet> tag

Methods of Applet Life-cycle

- The start() method:
  - The start() method is invoked after the init() method.
  - It is also called whenever the applet becomes active again after a period of inactivity.
  - This method should contain code related to
    - Resuming paused activity, e.g., animation

Methods of Applet Life-cycle

- The stop() method:
  - The stop method is the opposite of the start method.
  - The start method is called when the user moves back to the page that contains the applet.
  - The stop method is invoked when the user leaves the page.
  - This method should contain code related to
    - Pausing the running threads so the applet does not take up system resources when it is not active

Methods of Applet Life-cycle

- The destroy() method:
  - The destroy() method is called when the browser exits normally to inform the applet that it is no longer needed.
  - The applet can release any resources it acquired.
  - The stop() method is always called before the destroy() method.
  - This method should contain code related to
    - Releasing resources (e.g., network connection)
    - Terminating the threads that applet created

Example: Applet

```
import java.applet.*;
import java.awt.*;
/*
<applet code="MyApplet" height="400" width="400">
</applet>
*/
public class MyApplet extends Applet{
    public void init(){
          System.out.println("init() is called");
    }
```

Example: Applet

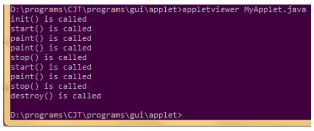
```
public void start(){
      System.out.println("start() is called");
}
public void stop(){
      System.out.println("stop() is called");
}
public void destroy(){
      System.out.println("destroy() is called");
}
```

Example: Applet

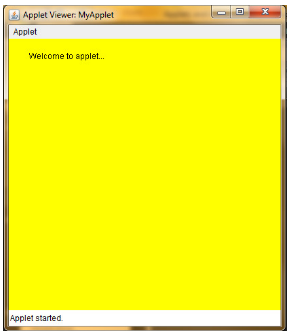
```
    public void paint(Graphics g){
          setBackground(Color.yellow);
          System.out.println("paint() is called");
          g.drawString("Welcome to applet...",30,30);
    }
}
```

Example: Applet

- JDK provides a utility called appletviewer to run applets
- For appletviewer to work, it needs <applet> tag embedded in .java file as comment.



```
D:\programs\CJT\programs\gui\applet>appletviewer MyApplet.java
init() is called
start() is called
paint() is called
paint() is called
stop() is called
start() is called
paint() is called
stop() is called
destroy() is called

D:\programs\CJT\programs\gui\applet>
```



Welcome to applet...

Applet started.

<applet> tag

```
<applet
    code = classfilename.class
    width = width-in-pixels
    height  = height-in-pixel
    [archive = archive-file-name-jar file]
    [codebase = applet_url]
    [vspace = vertical-margin]
    [hspace = horizontal-margin]
    [align = applet-alignment]
    [alt = alternative-text]
>
</applet>
```

Attributes of <applet> tag

- archive: Load an archive file that contains applet class and other required classes
  - How to create an archive file?
    $jar –cf myapplet.jar MyApplet.class OtherClass.class …
- codebase: It indicates directory for applets. If this attribute is not used, it is assumed applets are placed in the same directory in which html page is placed.
- vspace and hspace: margin around applet

Attributes of <applet> tag

- align: Indicates alignment of applet in the browser. Possible values: left, right, top, texttop, middle, absmiddle, baseline, bottom, and absbottom
- alt: It specifies the text to be displayed in case browser cannot run Java.

Passing Parameters to Applet

- For Java application, we can pass command line arguments, which are taken by the program through main() method.
- Applets do not have main() method and applet are not run via command line.
- How can we pass parameters to applet?
  - It is allowed via <param> child tag of <applet> tag.
- <param> tag has two attributes
  - name
  - value

Example: Passing parameters to applet

- Suppose, to our applet, we want to pass message, x-position, and y-position, which we can use in g.drawString()

```
<applet code=“AppletWithParameter" height="400"
    width="400">
    <param name=message value=“Passing parameters…”>
    <param name=x value=50>
    <param name=y value=50>
</applet>
```

- From applet, we can read the parameter using the following method:

```
public String getParameter(“parameter-name”);
```

Example: Passing parameters to applet

```
import java.applet.*;
import java.awt.*;
/*
<applet code="AppletWithParameter" height="400"
    width="400">
    <param name=message value="Passing parameters...">
    <param name=x value=50>
    <param name=y value=50>
</applet>
*/
```

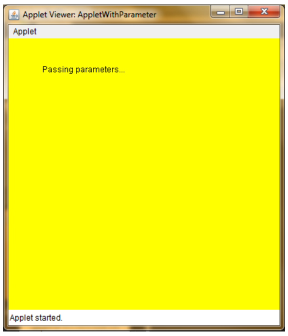
Example: Passing parameters to applet

```
public class AppletWithParameter extends Applet{
    String message;
    int x,y;
    public void init(){
        message=getParameter("message");
        x=Integer.parseInt(getParameter("x"));
        y=Integer.parseInt(getParameter("y"));
    }
```

Example: Passing parameters to applet

```
    public void paint(Graphics g){
        setBackground(Color.yellow);
        g.drawString(message,x,y);
    }
}
```

Example: Passing parameters to applet



```
Passing parameters...
```

Applet started.

Applet Security Restrictions

- Applets cannot read from, or write to, the file system of the computer.
  - Otherwise, they could damage the files and spread viruses.
- Applets cannot run programs on the browser's computer.
  - Otherwise, they might call destructive local programs and damage the local system on the user's computer.
- Applets can establish connections between the user's computer and the server where the applets are stored, not with any other machine.
  - This restriction prevents the applet from connecting the user's computer to another computer without the user's knowledge.

Conversions between Applications and Applets

- Frame class and Applet class have Container as a common superclass.
- UI components, layout managers, and event handling features are the same for both.
- We can convert an applet into an application.

Conversion of Applet into Application

- Steps of the conversion from applet to application:
  1. Remove import java.applet.* statement
  2. Eliminate HTML page or code.
  3. If applet takes parameters, we can pass these parameters to application as command line arguments.
  4. Make Frame class as parent class instead of Applet.
  5. Move code of init() and start() into constructor
  6. Move code of stop() and destroy() into windowClosing() event handler.
  7. Write main() method and create object of application, set its size, make its visible property true, and can add title (Applets do not have title)

Conversion of Application into Applet

- Steps of the conversion from application to applet :
  1. Add import java.applet.* statement
  2. Create HTML page with applet tag. If the application takes command line parameters, then add parameters in the <applet> tag.
  3. Derive the main class from Applet instead of Frame
  4. Replace application's constructor by init() method.
  5. Eliminate main() method, which usually contains code to crate and display frame. Applet is automatically displayed with the size indicated in the <applet> tag.
  6. Applet does not have title (remove setTitle() method). If application uses Menu, replace menu with buttons or other UI components.

Running a program as an Applet and an Application

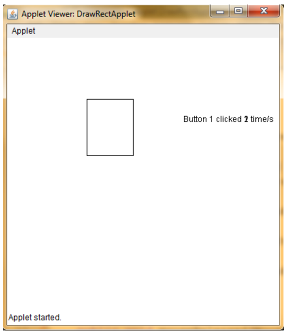
- Applet is derived from Panel.
- Therefore, we can add Panel (that means Applet also) into Frame and then we need to call methods of Applet at appropriate places in the frame object.
- We can do the above setup in main() method of our frame class.

Running a program as an Applet and an Application

- Suppose name of applet class is MyApplet
- Write main() method as following:

```
public static void main(String[] args){
    Frame f=new MyFrame(“My title”);
    MyApplet myApplet=new MyApplet();
    f.add(“center”, myApplet);
    f.setSize(300,300);
    f.setVisible(true);

    myApplet.init();
    myApplet.start();
}
```

Example: Event handling in applet



```
Button 1 clicked 2 time/s
Applet started.
```

Example: Event handling in applet

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
/*
<applet code="DrawRectApplet" height="400" width="400">
</applet>
*/
public class DrawRectApplet extends Applet implements
    MouseMotionListener,MouseListener{
    int sx=0,sy=0;
    int ex=0,ey=0;
```

Example: Event handling in applet

```
public void init(){
        addMouseMotionListener(this);
        addMouseListener(this);
}
public void mouseDragged(MouseEvent e){
        ex=e.getX();
        ey=e.getY();
        repaint();
}
public void mouseMoved(MouseEvent e){
}
```

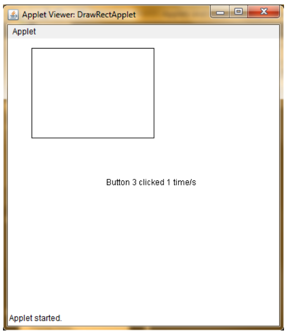
Example: Event handling in applet

```
public void paint(Graphics g){
        g.drawRect(sx,sy,ex-sx,ey-sy);
}
public void mouseEntered(MouseEvent e){

}
public void mouseExited(MouseEvent e){
}
public void mousePressed(MouseEvent e){
        sx=e.getX();
        sy=e.getY();
}
```

Example: Event handling in applet

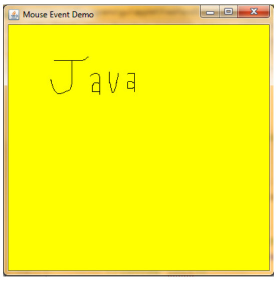
```
  public void mouseReleased(MouseEvent e){
          ex=e.getX();
          ey=e.getY();
  }
  public void mouseClicked(MouseEvent e){
          Graphics g=getGraphics();
          g.drawString("Button "+e.getButton()+"
  clicked                 "+e.getClickCount()+"
  time/s",e.getX(),e.getY());
  }
}
```

Example: Event handling in applet



```
Button 3 clicked 1 time/s
Applet started.
```

Example: Free-hand drawing in applet and application



Example: Free-hand drawing in applet and application

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
/*
<applet code="FreeHandDrawingDemo" height="400"
    width="400">
</applet>
*/
public class FreeHandDrawingDemo extends Applet{
    public static void main(String[] args){
```

Example: Free-hand drawing in applet and application

```
            //Create a frame
            Frame f=new Frame("Mouse Event Demo");
            //Create an instance of FreeHandDrawingDemo
            FreeHandDrawingDemo fhd=new FreeHandDrawingDemo();
            //invoke init()
            fhd.init();

            //add the applet (FreeHandDrawingDemo) to the frame
            f.add("Center",fhd);
            f.setSize(400,400);
            f.setVisible(true);
    }
```

Example: Free-hand drawing in applet and application

```
    public void init(){
        Canvas c = new PaintCanvas();
        c.setBackground(Color.yellow);

        //add canvas to applet
        setLayout(new BorderLayout());
        add("Center",c);
    }
}
```

Example: Free-hand drawing in applet and application

```
class PaintCanvas extends Canvas implements
    MouseMotionListener,MouseListener {
    final int CIRCLESIZE = 20;
    private Point lineStart = new Point(0,0);
    public PaintCanvas(){
        addMouseMotionListener(this);
        addMouseListener(this);
    }
```

Example: Free-hand drawing in applet and application

```
    public void mouseClicked(MouseEvent e){ }
    public void mouseReleased(MouseEvent e){ }
    public void mouseEntered(MouseEvent e){          }
    public void mouseExited(MouseEvent e){ }

    public void mousePressed(MouseEvent e){
        // Get new start point of the line
        lineStart.move(e.getX(),e.getY());
    }
```

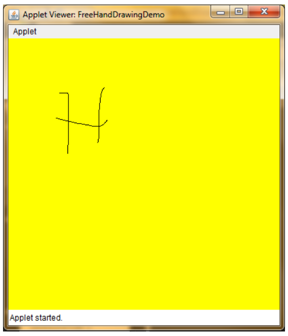
Example: Free-hand drawing in applet and application

```
public void mouseDragged(MouseEvent e){
            Graphics g=getGraphics();
            if(e.isMetaDown()){ //right mouse button
                    g.setColor(getBackground());
                    //erase drawing using oval
                   g.fillOval(e.getX()-(CIRCLESIZE/2),
                              e.getY()-(CIRCLESIZE/2), CIRCLESIZE, CIRCLESIZE);
                                  }
            else {
                  g.setColor(Color.black);
                  g.drawLine(lineStart.x, lineStart.y, e.getX(), e.getY());
            }
            lineStart.move(e.getX(), e.getY());
}
```

Example: Free-hand drawing in applet and application

```
    public void mouseMoved(MouseEvent e){
    }
    public void paint(Graphics g){
          System.out.println("Paint called");
    }
}
```

Example: Free-hand drawing in applet and application



Layout Managers: CardLayout

- It allows to use container to display one out of many possible component children (like flipping cards on a table).
- It can be used to show different child components to different users.
- We can also use card layout to let end user toggle among different interfaces and choose the one they prefer.
- It can be used to create a wizard based application.

CardLayout manager

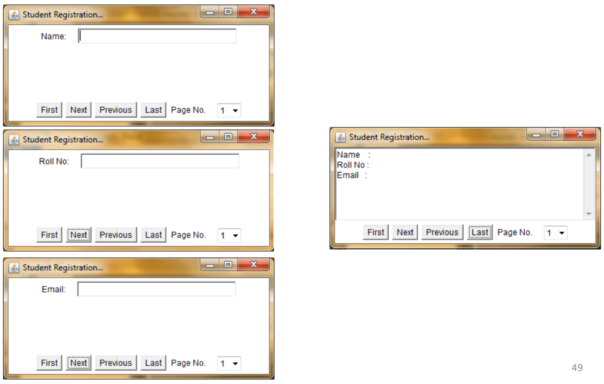
- Constructor:
  - CardLayout()
- Add Component in a container that uses the CardLayout
  - void add(Component c, String name)

  Component c is added having the name (or index) name.
- Methods to make a component visible
  - public void show(Container cn, String name)

CardLayout manager

- Other Methods to make a component visible
  - public void first(Container cn)
  - public void last(Container cn)
  - public void next(Container cn)
  - public void previous(Container cn)

Example: CardLayout



Example: CardLayout

```
import java.awt.*;
import java.awt.event.*;
class CardLayoutDemo extends Frame implements
   ActionListener, ItemListener{
   private CardLayout cardLayout=new CardLayout();
   private Panel cardPanel=new Panel();
   private TextField nameTF;
   private TextField rollNoTF;
   private TextField emailTF;
   private TextArea summaryTA;
```

Example: CardLayout

```
private Button firstBtn, nextBtn, previousBtn, lastBtn;
private Choice cardSelectChoice;
public CardLayoutDemo(){
     setTitle("Student Registration...");
     cardPanel.setLayout(cardLayout);

     Panel namePanel=new Panel();
     namePanel.add(new Label("Name:"));
     namePanel.add(nameTF=new TextField(30));
```

Example: CardLayout

```
     Panel rollNoPanel=new Panel();
     rollNoPanel.add(new Label("Roll No:"));
     rollNoPanel.add(rollNoTF=new TextField(30));

     Panel emailPanel=new Panel();
     emailPanel.add(new Label("Email:"));
     emailPanel.add(emailTF=new TextField(30));


     summaryTA=new TextArea(5,30);
```

Example: CardLayout

```
     cardPanel.add(namePanel,"1");
     cardPanel.add(rollNoPanel,"2");
     cardPanel.add(emailPanel,"3");
     cardPanel.add(summaryTA,"4");


     //Add buttons
     Panel buttonPanel=new  Panel();
     buttonPanel.setLayout(new  FlowLayout());
```

Example: CardLayout

```
     buttonPanel.add(firstBtn=new Button("First"));
     buttonPanel.add(nextBtn=new Button("Next"));
     buttonPanel.add(previousBtn=new
   Button("Previous"));
     buttonPanel.add(lastBtn=new Button("Last"));
     buttonPanel.add(new Label("Page No."));
     buttonPanel.add(cardSelectChoice=new
   Choice());
     for(int i=1; i<=4;i++)

     cardSelectChoice.addItem(String.valueOf(i));
```

Example: CardLayout

```
setLayout(new BorderLayout());
add("Center", cardPanel);
add("South", buttonPanel);

//Register listener
firstBtn.addActionListener(this);
nextBtn.addActionListener(this);
previousBtn.addActionListener(this);
lastBtn.addActionListener(this);
cardSelectChoice.addItemListener(this);
```

Example: CardLayout

```
        setSize(600,400);
        setVisible(true);
    }
    public static void main(String[] args){
        new CardLayoutDemo();
    }
```

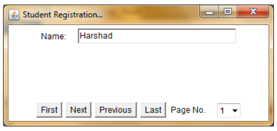
Example: CardLayout

```
public void actionPerformed(ActionEvent e){
            prepareSummary();
            String command=e.getActionCommand();
            if("First".equals(command))
                        cardLayout.first(cardPanel);
            else if("Last".equals(command))
                        cardLayout.last(cardPanel);
            else if("Previous".equals(command))
                        cardLayout.previous(cardPanel);
            else if("Next".equals(command))
                        cardLayout.next(cardPanel);

}
```

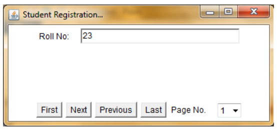
Example: CardLayout

```
    public void itemStateChanged(ItemEvent e){
        prepareSummary();
        if(e.getSource() instanceof Choice)
            cardLayout.show(cardPanel, (String)e.getItem());
    }
    private void prepareSummary(){
        summaryTA.setText("");
        summaryTA.append("Name    : "+nameTF.getText()+"\n");
        summaryTA.append("Roll No : "+rollNoTF.getText()+"\n");
        summaryTA.append("Email   : "+emailTF.getText()+"\n");
    }
}
```

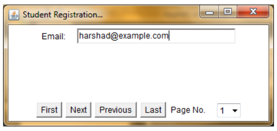
Example: CardLayout



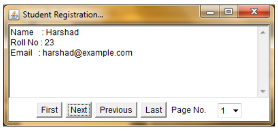
Name: Harshad



Roll No: 23



Email: harshad@example.com



```
Name    : Harshad
Roll No : 23
Email   : harshad@example.com
```

Layout Managers: GridBagLayout

- The Grid bag layout (like grid layout) arranges components into a grid of rows and columns, but allows us to fine-tune how the components are sized and positioned within the cells.
- Unlike the grid layout, the rows and columns are not required to be of uniform size.
- For example, a component can be set to span multiple rows or columns, or we can change its position on the grid.
- Various configurable parameters are available as a set of constraints that are represented by the GridBagConstraints object.

Layout Managers: GridBagLayout

- gridx/gridy
  - The gridx indicates the column in which the component will be placed (first column, gridx=0)
  - The gridy indicates the row in which the component will be placed (first row, gridy=0)
- gridwidth/gridheight
  - The gridheight indicates the number of cells in columns the component will use
  - The gridwidth indicates the number of cells in rows the component will use

Layout Managers: GridBagLayout

- weightx, weighty: Extra space to allocate to the component horizontally and vertically when the window is resized
- fill
  - Indicates how the component should be resized if the available area is larger than component's current size
  - Valid Values:
    - GridBagConstraints.NONE, GridBagConstraints.HORIZONTAL, GridBagConstraints.VERTICAL, GridBagConstraints.BOTH
  - Default value: GridBagConstraints.NONE

Layout Managers: GridBagLayout

- anchor: Where in the area the component is placed
  - GridBagConstraints.CENTER (default)
  - GridBagConstraints.NORTH
  - GridBagConstraints.EAST
  - GridBagConstraints.WEST
  - GridBagConstraints.SOUTH
  - GridBagConstraints.NORTHEAST
  - GridBagConstraints.SOUTHEAST
  - GridBagConstraints.NORTHWEST
  - GridBagConstraints.SOUTHWEST

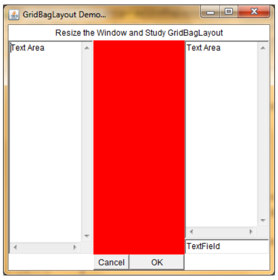
Steps to add a component in a GridBagLayout

- Create GridBagLayout object (gbl)
- Set GridBagLayout for a container (frame)
- Create a component (c)
- Create GridBagConstraints (gbc) and configure
- Using GridBagLayout (gbl) set GridBagConstraints (gbc) on the Component (c)

  gbl.setConstraints(c, gbc);
- Add component (c) in the container (f)

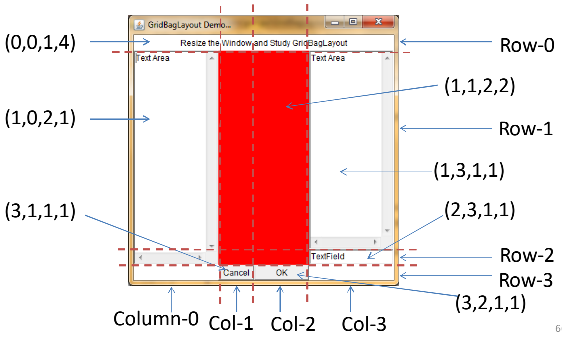
  f.add(c);

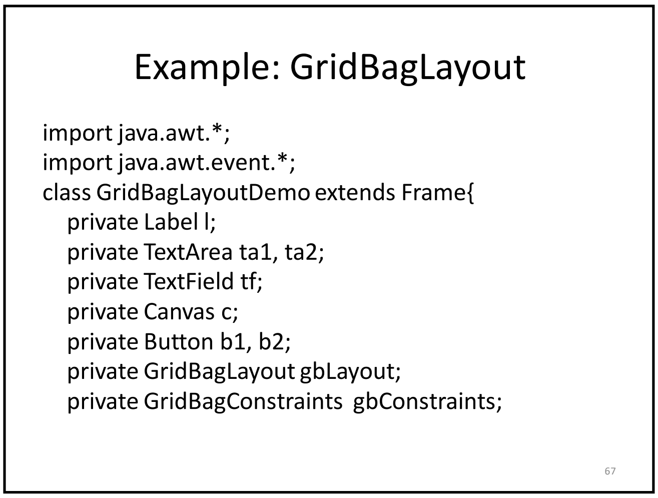
Example: GridBagLayout



Example: GridBagLayout

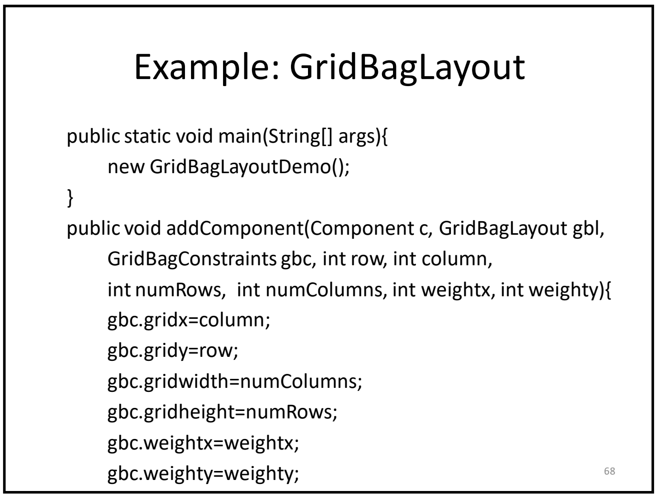
- For each component, we show (rowNo, ColNo, noRows, noCols)





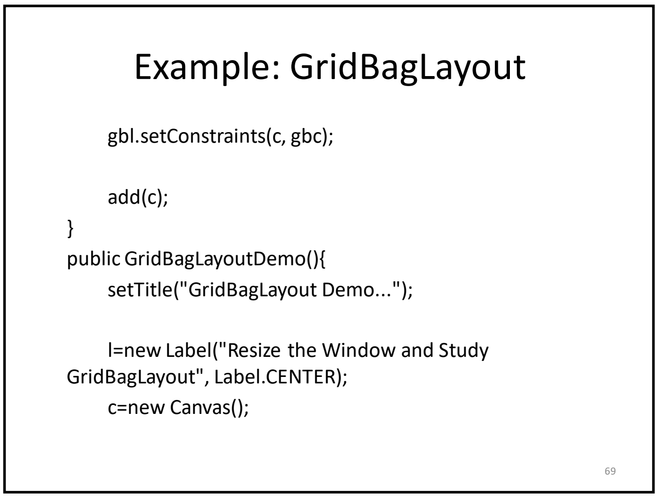
Example: GridBagLayout

```
import java.awt.*;
import java.awt.event.*;
class GridBagLayoutDemo extends Frame{
   private Label l;
   private TextArea ta1, ta2;
   private TextField tf;
   private Canvas c;
   private Button b1, b2;
   private GridBagLayout gbLayout;
   private GridBagConstraints  gbConstraints;
```



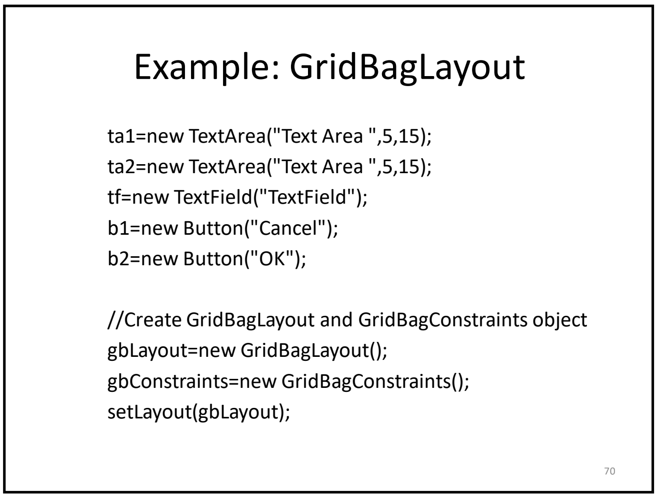
Example: GridBagLayout

```
public static void main(String[] args){
      new GridBagLayoutDemo();
}
public void addComponent(Component c, GridBagLayout gbl,
      GridBagConstraints gbc, int row, int column,
      int numRows,  int numColumns, int weightx, int weighty){
      gbc.gridx=column;
      gbc.gridy=row;
      gbc.gridwidth=numColumns;
      gbc.gridheight=numRows;
      gbc.weightx=weightx;
      gbc.weighty=weighty;
```



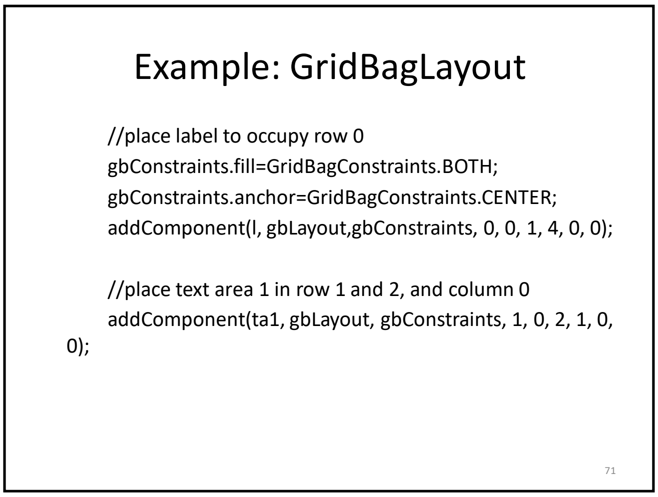
Example: GridBagLayout

```
      gbl.setConstraints(c, gbc);

      add(c);
}
public GridBagLayoutDemo(){
      setTitle("GridBagLayout Demo...");

      l=new Label("Resize the Window and Study
GridBagLayout", Label.CENTER);
      c=new Canvas();
```



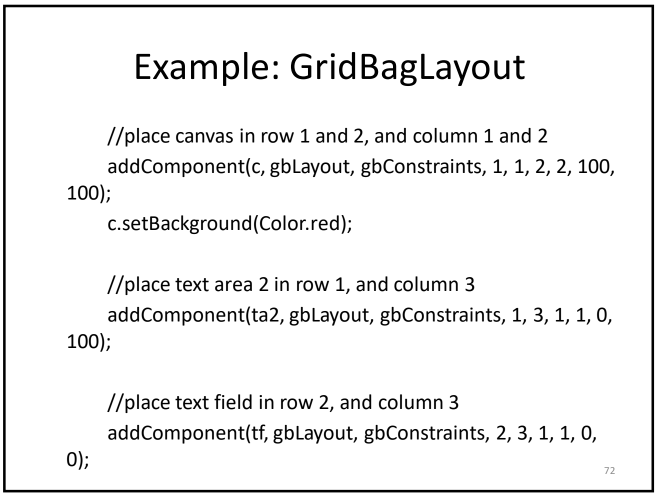
Example: GridBagLayout

```
      ta1=new TextArea("Text Area ",5,15);
      ta2=new TextArea("Text Area ",5,15);
      tf=new TextField("TextField");
      b1=new Button("Cancel");
      b2=new Button("OK");

      //Create GridBagLayout and GridBagConstraints object
      gbLayout=new GridBagLayout();
      gbConstraints=new GridBagConstraints();
      setLayout(gbLayout);
```



Example: GridBagLayout

```
      //place label to occupy row 0
      gbConstraints.fill=GridBagConstraints.BOTH;
      gbConstraints.anchor=GridBagConstraints.CENTER;
      addComponent(l, gbLayout,gbConstraints, 0, 0, 1, 4, 0, 0);

      //place text area 1 in row 1 and 2, and column 0
      addComponent(ta1, gbLayout, gbConstraints, 1, 0, 2, 1, 0,
0);
```



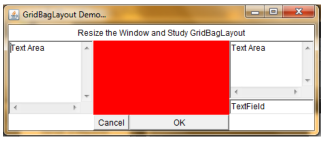
Example: GridBagLayout

```
      //place canvas in row 1 and 2, and column 1 and 2
      addComponent(c, gbLayout, gbConstraints, 1, 1, 2, 2, 100,
100);
      c.setBackground(Color.red);

      //place text area 2 in row 1, and column 3
      addComponent(ta2, gbLayout, gbConstraints, 1, 3, 1, 1, 0,
100);

      //place text field in row 2, and column 3
      addComponent(tf, gbLayout, gbConstraints, 2, 3, 1, 1, 0,
0);
```

Example: GridBagLayout

```
        //place button 1 in row 3, and column 1
        addComponent(b1, gbLayout, gbConstraints, 3, 1, 1, 1, 0,
0);

        //place button 2 in row 3, and column 2
        addComponent(b2, gbLayout, gbConstraints, 3, 2, 1, 1, 0,
0);

        setSize(400,400);
        setVisible(true);
    }
}
```

Example: GridBagLayout



Layout Manager: no layout

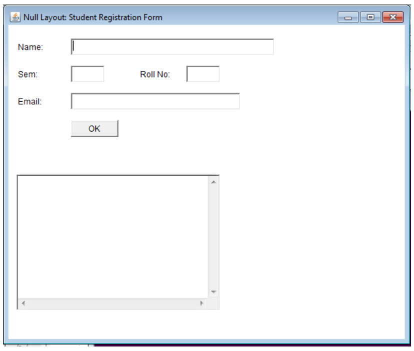
- Java provides to create GUI without using any Layout manager.
- It is enabled by calling setLayout(null) on the container object.
- We need to specify absolute position and size of component in terms of pixels. i.e.,
- We have to set bounds for each component that we want to add into the container
  - void setBounds(int x, int y, int width, int height)

  x and y indicates position of the component and width and height indicates size of the component

Steps to add a component using no-layout manager

1. Set no layout manager on a container (f)
   - f.setLayout(null)
2. Add the component (c) to the container
   - f.add(c);
3. Specify the location of the component and size of the component using setBounds() method
   - c.setBounds(10, 10, 40, 20);

Example: null layout



Example: null layout

```
import java.awt.*;
import java.awt.event.*;
class NullLayoutDemo extends Frame implements ActionListener{
    private Label nameLbl;
    private TextField nameTF;
    private Label semLbl;
    private TextField semTF;
    private Label rollNoLbl;
    private TextField rollNoTF;
    private Label emailLbl;
    private TextField emailTF;
    private Button okBtn;
    private TextArea summaryTA;
```

Example: null layout

```
public NullLayoutDemo(){
    setTitle("Null Layout: Student Registration Form");
    setLayout(null);

    nameLbl=new Label("Name:");
    nameTF=new TextField();
    add(nameLbl);
    add(nameTF);
    nameLbl.setBounds(20,50,50,25);
    nameTF.setBounds(100,50,300,25);
```

Example: null layout

```
semLbl=new Label("Sem:");
semTF=new TextField();
add(semLbl);
add(semTF);
semLbl.setBounds(20,90,50,25);
semTF.setBounds(100,90,50,25);

rollNoLbl=new Label("Roll No:");
rollNoTF=new TextField();
add(rollNoLbl);
add(rollNoTF);
rollNoLbl.setBounds(200,90,50,25);
rollNoTF.setBounds(270,90,50,25);
```

Example: null layout

```
emailLbl=new Label("Email:");
emailTF=new TextField();
add(emailLbl);
add(emailTF);
emailLbl.setBounds(20,130,50,25);
emailTF.setBounds(100,130,250,25);

okBtn=new Button("OK");
add(okBtn);
okBtn.setBounds(100,170,70,25);
```

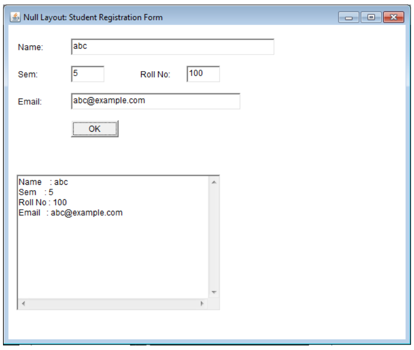
Example: null layout

```
        summaryTA=new TextArea();
        add(summaryTA);
        summaryTA.setBounds(20,250,300,200);

        //Register listener
        okBtn.addActionListener(this);

        setSize(600,500);
        setVisible(true);
    }
    public static void main(String[] args){
            new NullLayoutDemo();
    }
```

Example: null layout

```
    public void actionPerformed(ActionEvent e){
        String command=e.getActionCommand();
        if("OK".equals(command))
            prepareSummary();
    }
    private void prepareSummary(){
        summaryTA.setText("");
        summaryTA.append("Name    : "+nameTF.getText()+"\n");
        summaryTA.append("Sem    : "+semTF.getText()+"\n");
        summaryTA.append("Roll  No : "+rollNoTF.getText()+"\n");
        summaryTA.append("Email   : "+emailTF.getText()+"\n");
    }
}
```

Example: null layout



Name: abc

Sem: 5 Roll No: 100

Email: abc@example.com

```
Name   : abc
Sem   : 5
Roll No : 100
Email  : abc@example.com
```