AWT GRIDBAGLAYOUT CLASS

 Copyright © tutorialspoint.com

Introduction

The class **GridBagLayout** arranges components in a horizontal and vertical manner.

Class declaration

Following is the declaration for **java.awt.GridBagLayout** class:

```
public class GridBagLayout
   extends Object
      implements LayoutManager2, Serializable
```

Field

Following are the fields for **java.awt.BorderLayout** class:

- **double[] columnWeights** -- This field holds the overrides to the column weights.
- **int[] columnWidths** -- This field holds the overrides to the column minimum width.
- **protected Hashtable comptable** -- This hashtable maintains the association between a component and its gridbag constraints.
- **protected GridBagConstraints defaultConstraints** -- This field holds a gridbag constraints instance containing the default values, so if a component does not have gridbag constraints associated with it, then the component will be assigned a copy of the defaultConstraints.
- **protected java.awt.GridBagLayoutInfo layoutInfo** -- This field holds the layout information for the gridbag.
- **protected static int MAXGRIDSIZE** -- The maximum number of grid positions *bothhorizontallyandvertically* that can be laid out by the grid bag layout.
- **protected static int MINSIZE** -- The smallest grid that can be laid out by the grid bag layout.
- **protected static int PREFERREDSIZE** -- The preferred grid size that can be laid out by the grid bag layout.
- **int[] rowHeights** -- This field holds the overrides to the row minimum heights.
- **double[] rowWeights** -- This field holds the overrides to the row weights.

Class constructors

S.N.	Constructor & Description
1	**GridBagLayout** Creates a grid bag layout manager.

Class methods

S.N.	Method & Description

1

void addLayoutComponent*Componentcomp, Objectconstraints*

Adds the specified component to the layout, using the specified constraints object.

2

void addLayoutComponent*Stringname, Componentcomp*

Adds the specified component with the specified name to the layout.

3

protected void adjustForGravity*GridBagConstraintsconstraints, Rectangler*

Adjusts the x, y, width, and height fields to the correct values depending on the constraint geometry and pads.

4

protected void AdjustForGravity*GridBagConstraintsconstraints, Rectangler*

This method is obsolete and supplied for backwards compatability only; new code should call adjustForGravity instead.

5

protected void arrangeGrid*Containerparent*

Lays out the grid.

6

protected void ArrangeGrid*Containerparent*

This method is obsolete and supplied for backwards compatability only; new code should call arrangeGrid instead.

7

GridBagConstraints getConstraints*Componentcomp*

Gets the constraints for the specified component.

8

float getLayoutAlignmentX*Containerparent*

Returns the alignment along the x axis.

9

float getLayoutAlignmentY*Containerparent*

Returns the alignment along the y axis.

10

int[][] getLayoutDimensions

Determines column widths and row heights for the layout grid.

11

protected java.awt.GridBagLayoutInfo getLayoutInfo*Containerparent, intsizeflag*

Fills in an instance of GridBagLayoutInfo for the current set of managed children.

12

protected java.awt.GridBagLayoutInfo GetLayoutInfo*Containerparent, intsizeflag*

This method is obsolete and supplied for backwards compatability only; new code should call getLayoutInfo instead.

13

Point getLayoutOrigin

Determines the origin of the layout area, in the graphics coordinate space of the target container.

14

double[][] getLayoutWeights

Determines the weights of the layout grid's columns and rows.

15

protected Dimension getMinSize*Containerparent, java. awt. GridBagLayoutInfoinfo*

Figures out the minimum size of the master based on the information from getLayoutInfo.

16

protected Dimension GetMinSize*Containerparent, java. awt. GridBagLayoutInfoinfo*

This method is obsolete and supplied for backwards compatability only; new code should call getMinSize instead.

17

void invalidateLayout*Containertarget*

Invalidates the layout, indicating that if the layout manager has cached information it should be discarded.

18

void layoutContainer*Containerparent*

Lays out the specified container using this grid bag layout.

19

Point location*intx, inty*

Determines which cell in the layout grid contains the point specified by *x, y*.

20

protected GridBagConstraints lookupConstraints*Componentcomp*

Retrieves the constraints for the specified component.

21

Dimension maximumLayoutSize*Containertarget*

Returns the maximum dimensions for this layout given the components in the specified target container.

22

Dimension minimumLayoutSize*Containerparent*

Determines the minimum size of the parent container using this grid bag layout.

23

Dimension preferredLayoutSize*Containerparent*

Determines the preferred size of the parent container using this grid bag layout.

24

void removeLayoutComponent*Componentcomp*

Removes the specified component from this layout.

25

void setConstraints*Componentcomp, GridBagConstraintsconstraints*

Sets the constraints for the specified component in this layout.

26

String toString

Returns a string representation of this grid bag layout's values.

Methods inherited

This class inherits methods from the following classes:

- java.lang.Object

GridBagLayout Example

Create the following java program using any editor of your choice in say **D:/ > AWT > com > tutorialspoint > gui >**

AwtLayoutDemo.java

```
package com.tutorialspoint.gui;

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

public class AwtLayoutDemo {
   private Frame mainFrame;
   private Label headerLabel;
   private Label statusLabel;
   private Panel controlPanel;
   private Label msglabel;

   public AwtLayoutDemo(){
      prepareGUI();
   }

   public static void main(String[] args){
      AwtLayoutDemo  awtLayoutDemo = new AwtLayoutDemo();
      awtLayoutDemo.showGridBagLayoutDemo();
   }

   private void prepareGUI(){
      mainFrame = new Frame("Java AWT Examples");
      mainFrame.setSize(400,400);
      mainFrame.setLayout(new GridLayout(3, 1));
      mainFrame.addWindowListener(new WindowAdapter() {
         public void windowClosing(WindowEvent windowEvent){
            System.exit(0);
         }
      });
      headerLabel = new Label();
      headerLabel.setAlignment(Label.CENTER);
      statusLabel = new Label();
```

```
      statusLabel.setAlignment(Label.CENTER);
      statusLabel.setSize(350,100);

      msglabel = new Label();
      msglabel.setAlignment(Label.CENTER);
      msglabel.setText("Welcome to TutorialsPoint AWT Tutorial.");

      controlPanel = new Panel();
      controlPanel.setLayout(new FlowLayout());

      mainFrame.add(headerLabel);
      mainFrame.add(controlPanel);
      mainFrame.add(statusLabel);
      mainFrame.setVisible(true);
   }

   private void showGridBagLayoutDemo(){
      headerLabel.setText("Layout in action: GridBagLayout");

      Panel panel = new Panel();
      panel.setBackground(Color.darkGray);
      panel.setSize(300,300);
      GridBagLayout layout = new GridBagLayout();

      panel.setLayout(layout);
      GridBagConstraints gbc = new GridBagConstraints();

      gbc.fill = GridBagConstraints.HORIZONTAL;
      gbc.gridx = 0;
      gbc.gridy = 0;
      panel.add(new Button("Button 1"),gbc);

      gbc.gridx = 1;
      gbc.gridy = 0;
      panel.add(new Button("Button 2"),gbc);

      gbc.fill = GridBagConstraints.HORIZONTAL;
      gbc.ipady = 20;
      gbc.gridx = 0;
      gbc.gridy = 1;
      panel.add(new Button("Button 3"),gbc);

      gbc.gridx = 1;
      gbc.gridy = 1;
      panel.add(new Button("Button 4"),gbc);

      gbc.gridx = 0;
      gbc.gridy = 2;
      gbc.fill = GridBagConstraints.HORIZONTAL;
      gbc.gridwidth = 2;
      panel.add(new Button("Button 5"),gbc);

      controlPanel.add(panel);

      mainFrame.setVisible(true);
   }
}
```

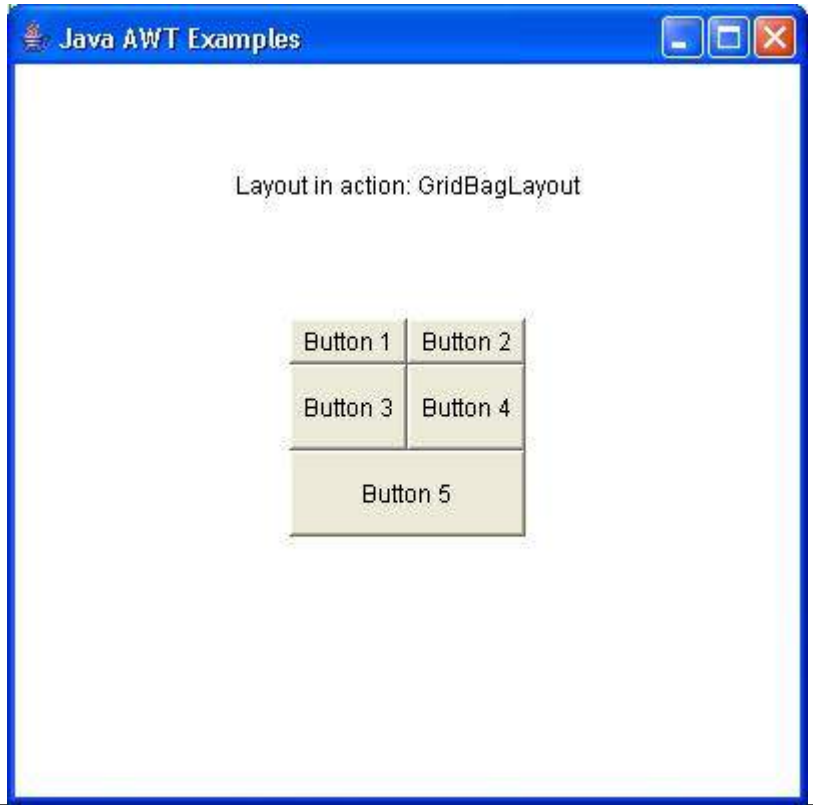
Compile the program using command prompt. Go to **D:/ > AWT** and type the following command.

```
D:\AWT>javac com\tutorialspoint\gui\AwtlayoutDemo.java
```

If no error comes that means compilation is successful. Run the program using following command.

```
D:\AWT>java com.tutorialspoint.gui.AwtlayoutDemo
```

Verify the following output



Processing math: 100%