AWT CARDLAYOUT CLASS

Introduction

The class **CardLayout** arranges each component in the container as a card. Only one card is visible at a time, and the container acts as a stack of cards.

Class declaration

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

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

Class constructors

S.N.	Constructor & Description
1	**CardLayout** Creates a new card layout with gaps of size zero.
2	**CardLayout***inthgap, intvgap* Creates a new card layout with the specified horizontal and vertical gaps.

Class methods

S.N.	Method & Description
1	**void addLayoutComponent***Componentcomp, Objectconstraints* Adds the specified component to this card layout's internal table of names.
2	**void addLayoutComponent***Stringname, Componentcomp* If the layout manager uses a per-component string, adds the component comp to the layout, associating it with the string specified by name.
3	**void first***Containerparent* Flips to the first card of the container.
4	**int getHgap** Gets the horizontal gap between components.

5

float getLayoutAlignmentX*Containerparent*

Returns the alignment along the x axis.

6

float getLayoutAlignmentY*Containerparent*

Returns the alignment along the y axis.

7

int getVgap

Gets the vertical gap between components.

8

void invalidateLayout*Containertarget*

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

9

void last*Containerparent*

Flips to the last card of the container.

10

void layoutContainer*Containerparent*

Lays out the specified container using this card layout.

11

Dimension maximumLayoutSize*Containertarget*

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

12

Dimension minimumLayoutSize*Containerparent*

Calculates the minimum size for the specified panel.

13

void next*Containerparent*

Flips to the next card of the specified container.

14

Dimension preferredLayoutSize*Containerparent*

Determines the preferred size of the container argument using this card layout.

15

void previous*Containerparent*

Flips to the previous card of the specified container.

16

void removeLayoutComponent*Componentcomp*

Removes the specified component from the layout.

17

void setHgap*inthgap*

Sets the horizontal gap between components.

18

void setVgap*intvgap*

Sets the vertical gap between components.

19

void show*Containerparent, Stringname*

Flips to the component that was added to this layout with the specified name, using addLayoutComponent.

20

String toString

Returns a string representation of the state of this card layout.

Methods inherited

This class inherits methods from the following classes:

- java.lang.Object

CardLayout 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.showCardLayoutDemo();
   }

   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 showCardLayoutDemo(){
      headerLabel.setText("Layout in action: CardLayout");

      final Panel panel = new Panel();
      panel.setBackground(Color.CYAN);
      panel.setSize(300,300);

      CardLayout layout = new CardLayout();
      layout.setHgap(10);
      layout.setVgap(10);
      panel.setLayout(layout);

      Panel buttonPanel = new Panel(new FlowLayout());

      buttonPanel.add(new Button("OK"));
      buttonPanel.add(new Button("Cancel"));

      Panel textBoxPanel = new Panel(new FlowLayout());

      textBoxPanel.add(new Label("Name:"));
      textBoxPanel.add(new TextField(20));

      panel.add("Button", buttonPanel);
      panel.add("Text", textBoxPanel);

      Choice choice = new Choice();
      choice.add("Button");
      choice.add("Text");

      choice.addItemListener(new ItemListener() {
         public void itemStateChanged(ItemEvent e) {
            CardLayout cardLayout = (CardLayout)(panel.getLayout());
            cardLayout.show(panel, (String)e.getItem());
         }
      });
      controlPanel.add(choice);
      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