**http://www.jchq.net/tutorial/08\_01Tut.htm**

**Oddities of the BorderLayout manager**

If you add multiple components to a Container that uses the BorderLayout but do not pass a Constraint parameter (North, South, etc), you may get unexpected results. Here is a sample that illustrates this.

import java.awt.\*;

public class FlowAp extends Frame{

public static void main(String argv[]){

FlowAp fa=new FlowAp();

// fa.setLayout(new FlowLayout());

fa.setSize(400,300);

fa.setVisible(true);

}

FlowAp(){

add(new Button("One"));

add(new Button("Two"));

add(new Button("Three"));

add(new Button("Four"));

add(new Button("Five"));

add(new Button("Six"));

add(new Button("Seven"));

add(new Button("Eight"));

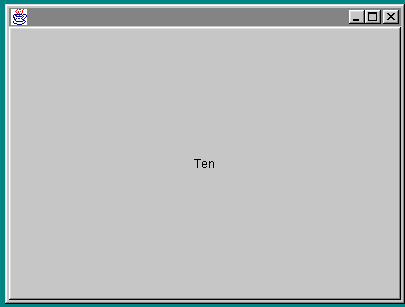
add(new Button("Nine"));

add(new Button("Ten"));

}//End of constructor

}//End of Appl

**Using the default BorderLayout**



The reason you get this unexpected big button in the center is that the BorderLayout uses a set of coordinates when arranging components. It divides its surface area up into

* North
* South
* East
* West
* Center

You might guess that the default when laying out components would be for them to be placed clockwise around the points of the compass or some such arrangement. Instead the designers decided to make the default the center of the layout area. Thus in this example every button has been laid out on the previous button, taking up the entire available area. As a result it appears that you only have one button, the last one added.

Because the BorderLayout only divides the area up into the five mentioned coordinates it is not the most useful of Layout Managers. However you need to be aware of it for the exam and you need to be aware of the way it defaults to placing all components in the center.

I have created a simple demonstration applet with source that shows how nothing much happens unless you play with the GridBagConstraints class..

<http://www.jchq.net/tutorial/08_01Tut.htm>

The GridBagLayout acts a little more like the GridLayout if you use the GridBagConstraints class and use the gridx and gridy fields to assign a position in a "virtual" grid to each component as you add it.s

The fields for the GridBagConstraints class are

* gridx gridy
* gridwidth and gridheight
* weightx and weighty
* fill
* ipadx and ipady
* insets
* anchor

**Table 8.1. CardLayout methods.**

|  |  |  |
| --- | --- | --- |
| **Layout Manager Method** | **Description** | **Parameters** |
| first(Container) | Show the first card | The parent container |
| last(Container) | Show the last card | The parent container |
| next(Container) | Show the next card | The parent container |
| previous(Container) | Show the previous card | The parent container |
| show(Container, String) | Show a named card | The parent container and the name of the card |

|  |  |  |
| --- | --- | --- |
| **Layout Manager Method** | **Description** | **Parameters** |
| setConstraints(Component, GridBagConstraints) | Associate constraints with a component | The component andthe constraints |