

GUI COMPONENTS AND LAYOUT

Another way of using a *JPanel* is as a **container** to hold other components. Java has many classes that define GUI components. Before these components can appear on the screen, they must be **added** to a container. In this program, the variable named `content` refers to a *JPanel* that is used as a container, and two other components are added to that container. This is done in the statements:

```
content.add(displayPanel, BorderLayout.CENTER);  
content.add(okButton, BorderLayout.SOUTH);
```

Here, `content` refers to an object of type *JPanel*; later in the program, this panel becomes the content pane of the window. The first component that is added to `content` is `displayPanel` which, as discussed above, displays the message, "Hello World!". The second is `okButton` which represents the button that the user clicks to close the window. The variable `okButton` is of type *JButton*, the Java class that represents push buttons.

The "BorderLayout" stuff in these statements has to do with how the two components are arranged in the container. When components are added to a container, there has to be some way of deciding how those components are arranged inside the container. This is called "laying out" the components in the container, and the most common technique for laying out components is to use a **layout manager**. A layout manager is

an object that implements some policy for how to arrange the components in a container; different types of layout manager implement different policies. One type of layout manager is defined by the *BorderLayout* class. In the program, the statement

```
content.setLayout(new BorderLayout());
```

creates a new *BorderLayout* object and tells the `content` panel to use the new object as its layout manager. Essentially, this line determines how components that are added to the content panel will be arranged inside the panel. We will cover layout managers in much more detail later, but for now all you need to know is that adding `okButton` in the `BorderLayout.SOUTH` position puts the button at the bottom of the panel, and putting `displayPanel` in the `BorderLayout.CENTER` position makes it fill any space that is not taken up by the button.

This example shows a general technique for setting up a GUI: Create a container and assign a layout manager to it, create components and add them to the container, and use the container as the content pane of a window or applet. A container is itself a component, so it is possible that some of the components that are added to the top-level container are themselves containers, with their own layout managers and components. This makes it possible to build up complex user interfaces in a hierarchical fashion, with containers inside containers inside containers...

Source : <http://math.hws.edu/javanotes/c6/s1.html>