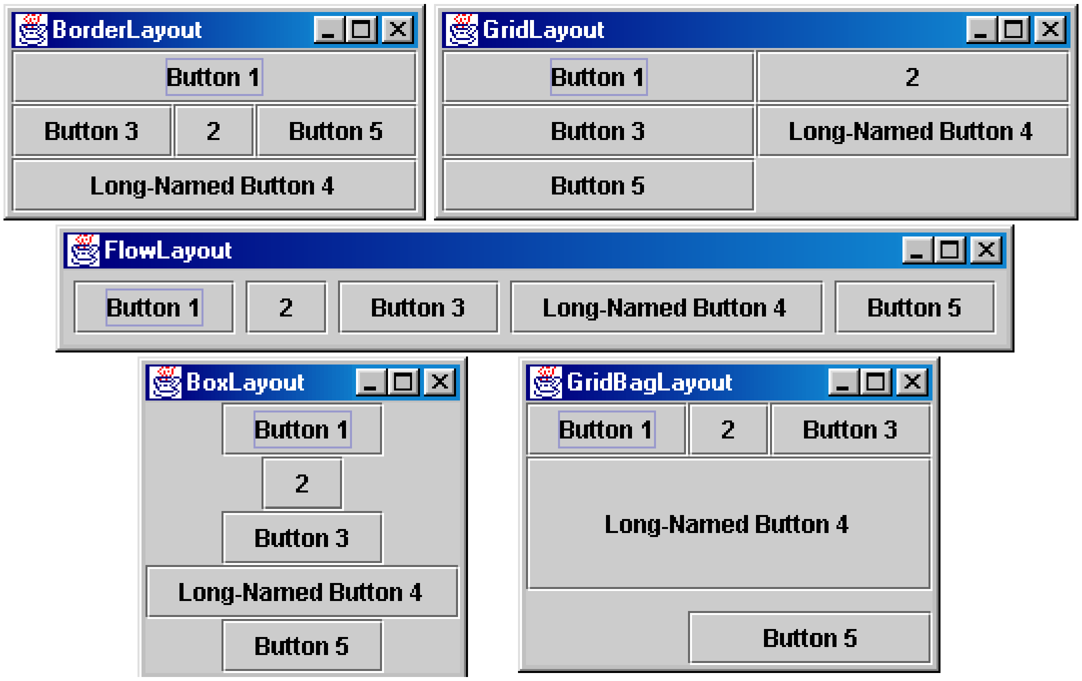
Layout Management

The following figure shows the GUIs of five programs, each of which displays five buttons. The buttons are identical, and the code for the programs is almost identical. So why do the GUIs look so different? Because they use different layout managers to control the size and the position of the buttons.



The Java platform supplies five commonly used layout managers: `BorderLayout`, `BoxLayout`, `FlowLayout`, `GridBagLayout`, and `GridLayout`.

Using Layout Managers

By default, every container has a layout manager. All `JPanel` objects use a `FlowLayout` by default, whereas content panes (the main containers in `JApplet`, `JDialog`, and `JFrame` objects) use `BorderLayout` by default. As a rule, the only time you have to think about layout managers is when you create a `JPanel` or add components to a content pane. If you don't like the default layout manager that a panel or content pane uses, you can change it to a different one. Just invoke the container's `setLayout` method. For example, here's the code that makes a panel use `BorderLayout`:

```
JPanel pane = new JPanel();
pane.setLayout(new BorderLayout());
```

When you add components to a panel or a content pane, the arguments you specify to the add method depend on the layout manager that the panel or content pane is using. So be sure to check the API documentation for the layout manager for details.

Here's a quick summary of the various layout managers and where to find more information in *The Java Tutorial* and API documentation.

`BorderLayout`

`BorderLayout` is the default layout manager for every content pane. The content pane is the main container in all frames, applets, and dialogs. A `BorderLayout` has five areas available to hold components: north, south, east, west, and center. All extra space is placed in the center area.