COMPOSING GUI IN JAVA

We don't yet have the ability to build complex windows: We can create components, and we can place them on the north, south, east, or west edge of the window; or we can place it in the window's center. That limits us to five components, and they're not necessarily placed how we want them.

The JPanel class

Building more complex interfaces requires using the `JPanel` class. A `JPanel` object is an empty graphical component, but it is a subclass of `Container`, so you can add other components into it.

There are four JPanel methods that are particularly important.

`JPanel()`

> (Constructor) Creates an empty JPanel.

`void add(Component what)`

> Inserts `what` into this panel, using the default placement.

`void add(Component what, Object info)`

> Inserts `what` into this panel, using `info` as information about where the object should be placed.

`void setLayout(LayoutManager manager)`

> Configures this container to use the indicated technique for laying out its components. We'll see how this works soon.

Layouts

There are several categories of classes involved in creating GUIs using Swing, and we've covered all but the last.

- the `JFrame` class
- component classes (`JButton`, `JTextField`, `JPanel`)
- interfaces for listeners (`ActionListener`)
- event classes (`ActionEvent`)
- container classes (`Container`, `JPanel`)

- classes for managing layouts (`FlowLayout`, `BorderLayout`, `GridLayout`)

This last category consists of classes implementing Java's `LayoutManager` interface. The layout classes tell a container class how it should arrange the components it contains. We can use the `setLayout` method to configure a container, such as `Container` or `JPanel`, to use a different approach for arranging its components.

`BorderLayout`

We've actually already seen the `BorderLayout` class in passing: It's the default layout for a `JFrame`'s container. Creating a `BorderLayout` object is simple:

```
BorderLayout()
```

(Constructor) Creates a BorderLayout object.

When you add something to a container that's using the BorderLayout, you will generally use the `add` method that takes an `info` parameter, and you will pass something like`BorderLayout.NORTH` saying on which border to place the component (or `BorderLayout.CENTER` to place the component in the middle).

FlowLayout

The FlowLayout class is even simpler: It places the components in left-to-right order. Each component gets sized to its preferred size. When the components fill the row, they start being placed in the next row.

```
FlowLayout()
```

(Constructor) Creates a FlowLayout object.

With a container using FlowLayout, you'll generally use the `add` that *doesn't* take an `info` parameter.

FlowLayout is the default layout for a JPanel.

GridLayout

In the GridLayout class, components are placed left-right, top-down in a strict grid. Each component is resized to fill its grid space.

```
GridLayout(int rows, int columns)
```

(Constructor) Creates a GridLayout object, represent the layout strategy with `rows` rows and `columns` columns.

Source : http://www.toves.org/books/java/ch24-swing/index.html