GUIs: Graphical User Interfaces

Their mouse had a mean time between failure of ... a week ... it would jam up irreparably, or ... jam up on the table--.. It had a flimsy cord whose wires would break. Steve Jobs: ... Xerox says it can't be built for < $400, I want a $10 mouse that will never fail and can be mass produced, because it's going to be the primary interface of the computer ...”

... Dean Hovey ... came back, “I've got some good and some bad news. Good news: we've got a new project with Apple. Bad news: I told Steve we'd design a mouse for 10 bucks.”

... year later ... we ... filed ... and were granted a patent, on the electro-mechanical-optical mouse of today; ... we ended up ... [making] the mouse as invisible to people as it is to today.


---

No recitation next week: Fall break

A4: Due Saturday, 14 October.

Wednesday TAs will be in their recitation rooms ready to help those who need help on A4.

Look carefully at Piazza note A4 FAQs!

We will attempt to keep up with Piazza questions over the break.

Last day to ask for prelim 1 grade: Friday

Sign up for lunches with instructors. Pinned Piazza note.

---

GUI (Graphical User Interface)

- Provides a friendly interface between user and program
- Allows event-driven or reactive programming: The program reacts to events such as button clicks, mouse movement, keyboard input
- Often is multi-threaded: Different threads of execution can be executing simultaneously. We study concurrency and threads in April.

Two aspects to making a GUI:

1. Placing components (buttons, text, etc.) in it. TODAY
2. Listening/responding to events

Class JFrame

- JFrame object: associated with a window on your monitor.
- Generally, a GUI is a JFrame object with various components placed in it.
- Some methods in a JFrame object
  - hide() show() setVisible(boolean)
  - getLocation() setLocation(int, int)
  - getWidth() getHeight()

Class JFrame is in package java.awt

---

GUI (Graphical User Interface)

There are three GUI packages in Java:

- **AWT** (Abstract or Awful Window Toolkit) — first one. Some parts are implemented not in Java but in code that depends on the platform. Came with first Java.

- **Swing** — a newer one, which builds on AWT as much as possible. It is “lightweight”: all code written as Java classes/interfaces. Released in 97-98.

- **JavaFX** — completely new! Much more functionality, flexibility, but far too complicated to teach in CS2110. (Released first in 2008)

We use Swing (and parts of AWT)
Placing components in a JFrame

**Layout manager:** Instance controls placement of components.

**JFrame layout manager default:** BorderLayout.

**BorderLayout** layout manager: Can place 5 components:

```java
class JFrameDemo {
    public static void main(String[] args) {
        JFrame frame = new JFrame("JFrameDemo.java");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JFrameDemo jd = new JFrameDemo();
        frame.add(jd);
        frame.pack();
        frame.setVisible(true);
        frame.setVisible(true);
    }
    JFrameDemo() {
        super("JFrameDemo.java");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JButton jb = new JButton("Click here");
        JLabel jl = new JLabel("west");
        add(jb, BorderLayout.EAST);
        add(jl, BorderLayout.NORTH);
        add(new JLabel("South"), BorderLayout.SOUTH);
        add(new JLabel("center"), BorderLayout.CENTER);
        add(new JLabel("North"), BorderLayout.NORTH);
        pack();
    }
}
```

### Packages -- Components

Packages that contain classes that deal with GUIs:

**java.awt:** Old package.  **javax.swing:** New package.

Java Swing has a better way of listening to buttons, text fields, etc. Components are more flexible.

**Component:** Something that can be placed in a GUI window. They are instances of certain classes, e.g.

- **JButton, Button:** Clickable button
- **JLabel, Label:** Text that cannot be clicked
- **JTextField, TextArea:** Field into which the user can type
- **JPanel, Panel:** Used for graphics; to contain other components
- **JCheckBox, Checkbox:** Checkable box with a title
- **JComboBox, Choice:** Menu of items, one of which can be checked
- **JRadioButton, RadioButton:** Checkable box with a title
- **JLabel:** A label in a GUI window

Note the use of subclasses to provide structure and efficiency. For example, there are two kinds of JToggleButton, so that class has two subclasses.

### Basic Components

- **Component:** Something that can be placed in a GUI window. These are the basic ones used in GUIs

- **Container:** A component that can contain other components

- **Note:** java.awt is the old GUI package. java.swing is the newer GUI package. When they wanted to use an old name, they put J in front of it. (e.g. Frame and JFrame)

When constructing java.swing, the attempt was made to rely on the old package as much as possible. So, JFrame is a subclass of Frame. But they couldn’t do this with JPanel.

### Components that can contain other components

- **JFrame, Window, Frame, JPanel:** Can contain other components

- **Button, Canvas, TextComponent:** Text fields, etc. Components are more flexible.
public class PanelDemo extends JFrame {
    JPanel p = new JPanel();
    public PanelDemo() {
        super("Panel demo");
        p.add(new JButton("0"));
        p.add(new JButton("1"));
        p.add(new JButton("2"));
        p.add(new JButton("3"));
        add(new JLabel("east"), BorderLayout.EAST);
        add(new JLabel("west"), BorderLayout.WEST);
        add(new JLabel("    "), BorderLayout.SOUTH);
        add(p, BorderLayout.CENTER);
        pack();
    }
}

Class Box:

To simulate using a BoxLayout manager for a JFrame, create a Box and place it as the sole component of the JFrame:

```
import javax.swing.*;
import java.awt.*;

public class BoxDemo extends JFrame {
    public BoxDemo() {
        super("Box demo");
        Box b = new Box(BoxLayout.X_AXIS);
        b.add(new JButton("0"));
        b.add(new JButton("1"));
        b.add(new JButton("2"));
        b.add(new JButton("3"));
        BoxLayout CENTER;
        pack();
    }
}
```

Simulate BoxLayout Manager in a JFrame

To simulate using a BoxLayout manager for a JFrame, create a Box and place it as the sole component of the JFrame:

```
import javax.swing.*;
import java.awt.*;

public class BoxDemo2 extends JFrame {
    public BoxDemo2(String title, int n) {
        super(title);
        // Create Box b1 with n buttons.
        Box b1 = new Box(BoxLayout.Y_AXIS);
        for (int i = 0; i < n; i++)
            b1.add(new JButton("1" + i));
        // Create Box b2 with n+1 buttons.
        Box b2 = new Box(BoxLayout.Y_AXIS);
        int k = n + 1
        for (int i = 0; i < b2;
            b.add(new JButton("1" + i));
        // Create Box b3 with n+2 buttons.
        Box b3 = new Box(BoxLayout.Y_AXIS);
        for (int i = 0; i < n+2; i++)
            b3.add(new JButton("1" + i));
        // Create horizontal box b containing b1, b2, b3
        Box b = new Box(BoxLayout.X_AXIS);
        b.add(b1);
        b.add(b2);
        b.add(b3);
        add(b, BorderLayout.CENTER);
        pack();
    }
}
```