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 today.


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
2. Listening/responding to events

Lecture notes page of course website, rows for GUI lectures: Contains guiDemo.zip. It's filled with short demos of GUI features including demos for today and next lecture. Download it and look at demos in DrJava or Eclipse.

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)

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)
getX() getY() (coordinates of top-left point)
getWidth() getHeight() setLocation(int, int)
getTitle() setTitle(String)
getLocation() setLocation(int, int)

Over 100 methods in a JFrame object!

Class JFrame is in package javax.swing

Placing components in a JFrame

Layout manager: Instance controls placement of components.
JFrame layout manager default: BorderLayout.
BorderLayout layout manager: Can place 5 components:

```
public class C extends JFrame {
    public C() {
        Container cp= getContentPane();
        JButton jb= new JButton("Click here");
        JLabel jl= new String("label 2");
        cp.add(jb, BorderLayout.EAST);
        cp.add(jl, BorderLayout.WEST);
        setLocation(100, 100);
        setVisible(true);
    }
}
```

Putting components in a JFrame

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

public class ComponentExample extends JFrame {

    ** Constructor: a window with title t and 5 components */
    public ComponentExample(String t) {
        super(t);
        Container cp= getContentPane();
        cp.add(new JButton("click me"), BorderLayout.EAST);
        cp.add(new JTextField("type here"), BorderLayout.NORTH);
        cp.add(new JButton("type here"), BorderLayout.NORTH);
        cp.add(new JLabel("label 2"), BorderLayout.WEST);
        cp.add(new JTextArea("type here", 4, 10), BorderLayout.CENTER);
        pack();
    }
}
```

ComponentExample.java
Components that can contain other components

Component
Box
Container
JComponent
JPanel
Panel
Applet
Window
Frame
JFrame

Java.awt is the old GUI package.
Java.swing is the new GUI package.
When they wanted to use an old name, they put \texttt{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.

Basic Components

Component
Button, Canvas
Checkbox, Choice
Label, List, ScrollBar
TextComponent
TextField, TextArea
Container
JComponent
AbstractButton
JButton
JComponent
JCheckButtom
JComboBox
JComponent
JLabel
JList
JPopupMenu, JScrollPane, JScrollBar, JSlider
JTextField
JTextComponent
JTextField, JTextArea

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

import java.awt.*; 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"));
    Container cp= getContentPane();
    cp.add(b, BorderLayout.CENTER);
    pack();
  }
}

class Box:

Component has two subclasses.

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

```java
JFrame jf = new JFrame("title");
Box b = new Box(BoxLayout.X_AXIS);
Add components to b;
jf.add(b, BorderLayout.CENTER);
```

1. Start developing a GUI by changing an already existing one. A lot of details. Hard to get all details right when one starts from scratch and has little idea about the Java GUI package.

2. Showed how to place components in a GUI. Next time: how to "listen" to things like button clicks in a GUI.