GUI: Event-driven programming

GUI = Graphical User Interface

* Windows, icons, and mouse pointing

GUI programs are all about responding to user events. For example, in MS Word,

* Typing a key on the keyboard results in the display of that letter
* Clicking the mouse on a menu item executes that menu item
* Double clicking, special keystrokes, etc.

You don’t write a GUI program that starts at the beginning, runs stepwise through all the code, and then terminates. Instead, you “register” certain methods to be executed in response to events. Events are handled by “listeners” that listen for the event, and then execute the relevant method.

The fundamental unit of windowing in Java is the JFrame. In order to create a window on the screen, we need to create a JFrame object.

When constructing a JFrame, consider:

* Title bar text
* Close operation: what do you want to happen when the “Close” button is clicked?
* Size: How big? (if you use constants, you should define them as static final)
* Components: What goes into the window? (JLabel)
* When you have more than one component, you need to add a layout manager e.g. FlowLayout

Our design pattern:

* Extend JFrame
* Put all window behavior, appearance, and component stuff in the constructor
* Events will be handled within the extended class
* To display the window, create an instance of the extended class

Listeners and Event Handlers

* Each action is accompanied by a listener
* You can “register” a method to be executed when that event happens
* For example, clicking on a button
  + Add an object of type ActionListener to the button
  + ActionListener is an interface that contains one method: actionPerformed
  + When the event occurs, the listener calls actionPerformed