CS 1181
Week Four

Reese Hatfield

Review

- A good class is
  - Testable
  - Has Robust Exceptions
  - Encapsulated Behavior
  - etc.

Moving Forward

- That's cool I guess

- We have a lot of tools
    - But how do we build real applications?

Moving Forward

- Not like command line apps
- Real graphical applications

- CLI = Command Line Interface apps
- GUI = Graphical User Interfaces

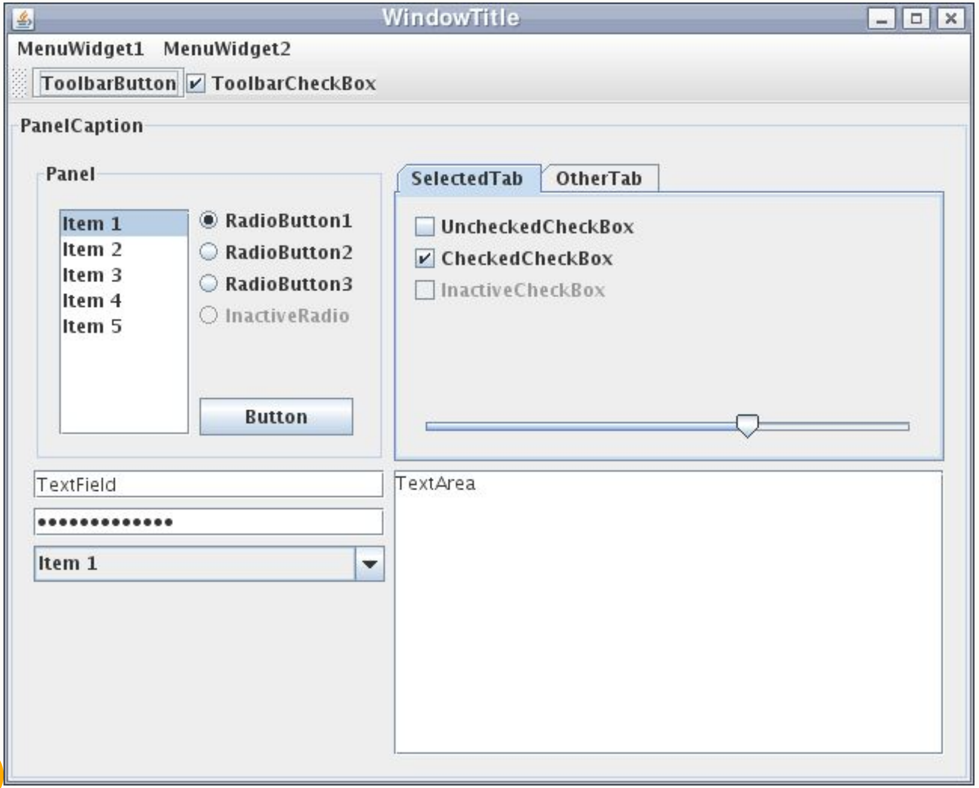
Moving Forward

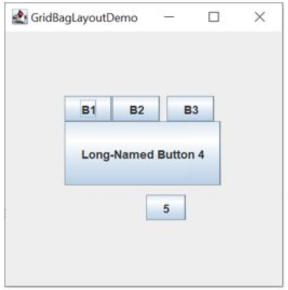
- CLI's are usually developer/power-user facing
- GUI's are more user focused

GUIs

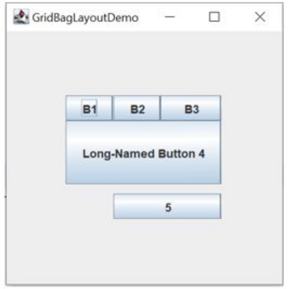
- Java's primary GUI library is *Swing*
- Built into java
- No external libraries needed
- Cross platform GUI framework

GUIs

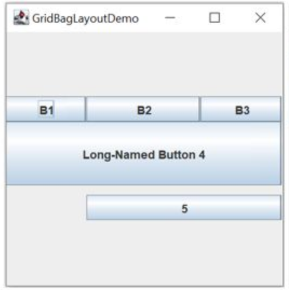




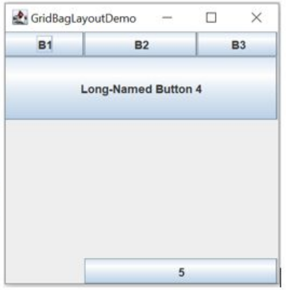
(a) Natural width and height



(b) `fill = HORIZONTAL`



(c)
Column 1: `weightx = 0.5`
Column 2: `weightx = 1.0`
Column 3: `weightx = 0.5`



(d)
Row 1: `weighty = 0` (not participating)
Row 2: `weighty = 0` (not participating)
Row 3: `weighty = 1.0` (take all)

JFrame

- A JFrame represents an individual window
- JFrame is a class that you can use yourself
- Let’s do it!

JFrame

- JFrame frame = new JFrame()
  - Nothing happened
- Useful methods:
  - setVisible(boolean)
  - setSize(int, int)
  - setTitle(String)
  - setDefaultCloseOperation(int)

Let’s add some more things

- Swing provides a series of pre-written components
- We can use these in our program to make our GUI
- All prefixed with “J”

JLabels

- Let's add some text
- JLabel class
- Nice constructor and setter

- Can we add more?

Let’s add some more things

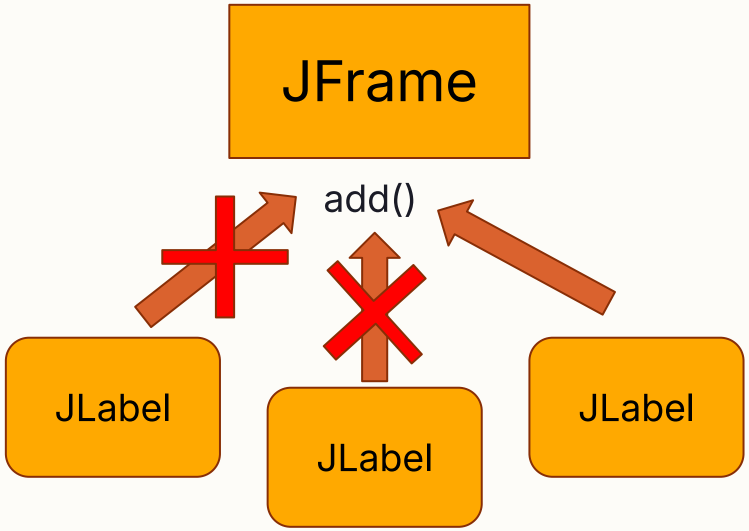
- What happened?
- 2nd label clobbered the 1st

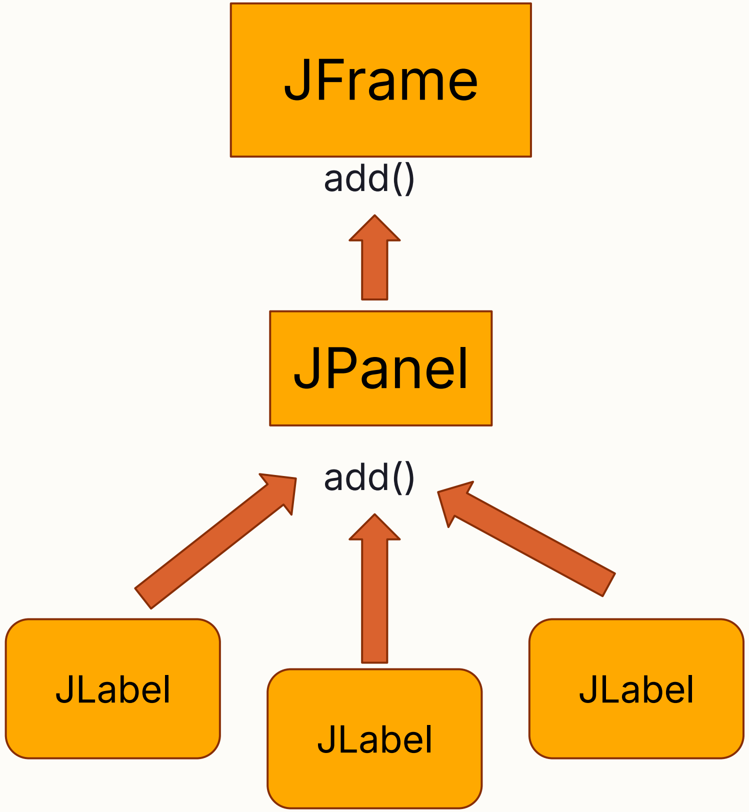
- JFrame can only hold a single swing component
- So how do we fix this?

JPanels

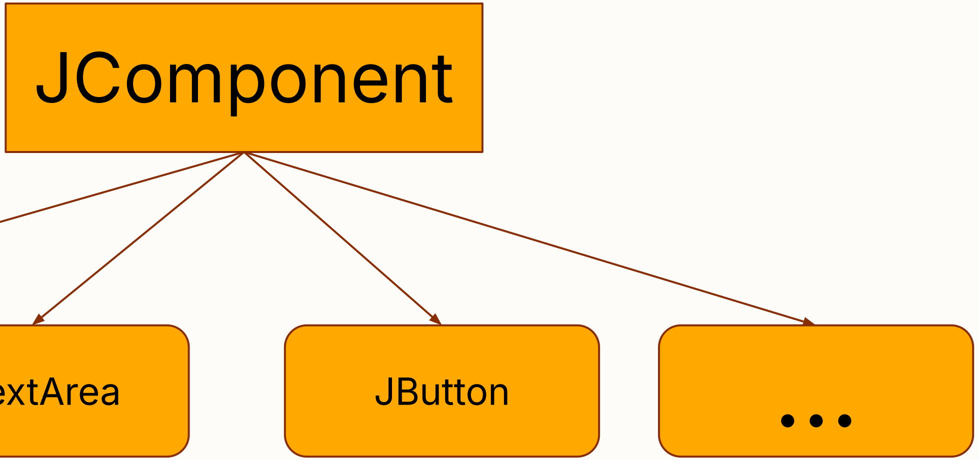
- Swing gives a “container” class
- JPanel
  - Let’s you add sub-components
  - Can add as many as you one

JPanels


JFrame
add()
JLabel
JLabel
JLabel


JFrame
add()
JPanel
add()
JLabel
JLabel
JLabel

Let’s add some more things



- Let’s play around with some of these

JButtons

- JButton btn = new JButton()
- btn.addActionListener(ActionListener)
- ActionListener documentation

JButtons

- JButtons let us make our GUIs do things
- Let’s play around with what they can do
- We can compose buttons in existing classes

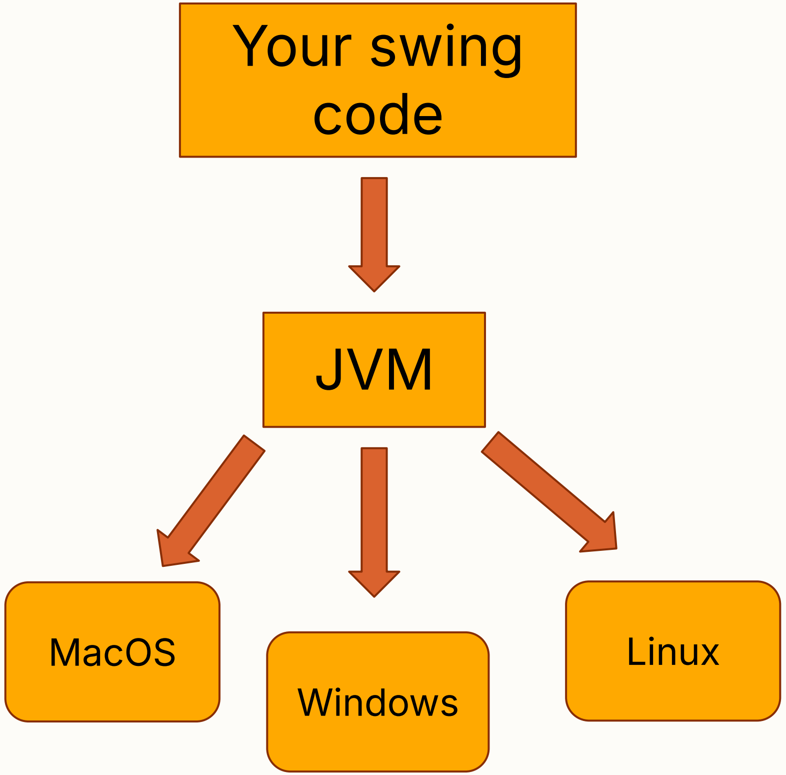
Aside: Why does it look ancient???

- Sun
  - Developed Java Swing
  - Switched to JavaFX
- Oracle
  - Removed FX from JDK (Why?)

- OpenJDK now maintains Swing
- Third party + open source maintains FX

Look and Feel

- Appearance of application
- OS Defaults



Layout Managers

Reese Hatfield

Structure

- How can we organize our components
- So far, everything kind of flows together

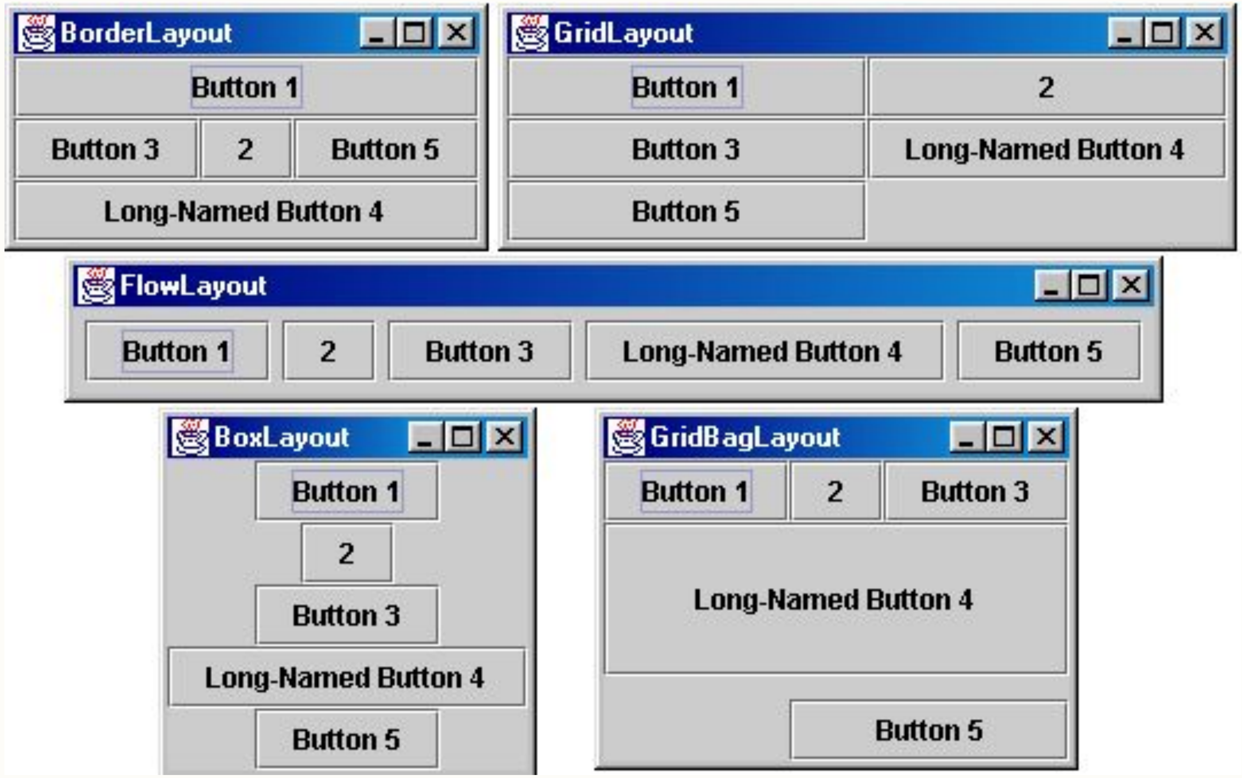
- FlowLayout = Default Layout
- There are many others

Layout Managers

- Usually placed onto JPanels
- Tell Swing how to organize components
  - As they get added

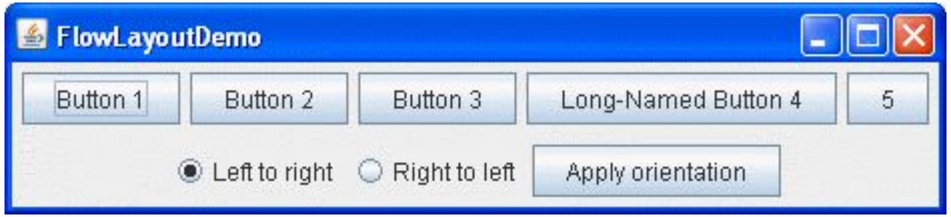
- We have already seen how things get added without changing anything

Layout Managers



Flow Layout

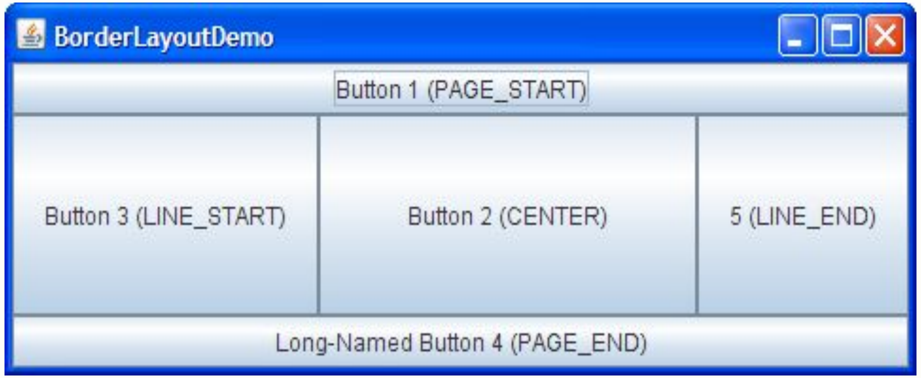
- Default Layout Manager
- Positioning:
  - Horizontal Center
  - Vertical Top → Bottom



[✓ Left to right] Right to left

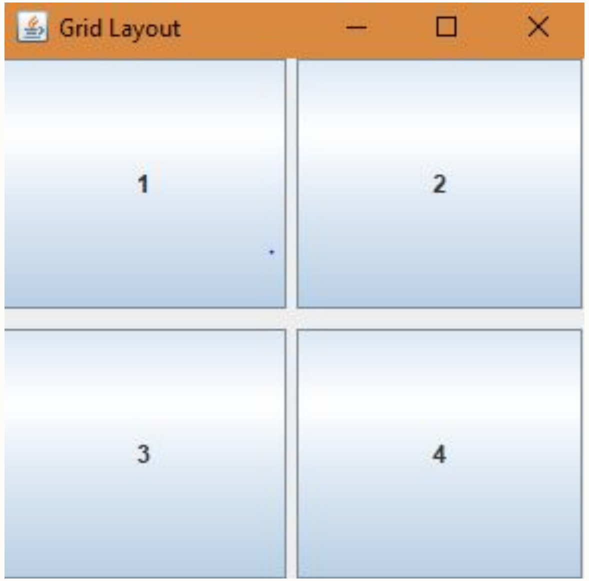
Border Layout

- Cardinal Organization
  - NORTH
  - SOUTH
  - EAST
  - WEST
  - CENTER
- target.add(component, position)



Grid Layout

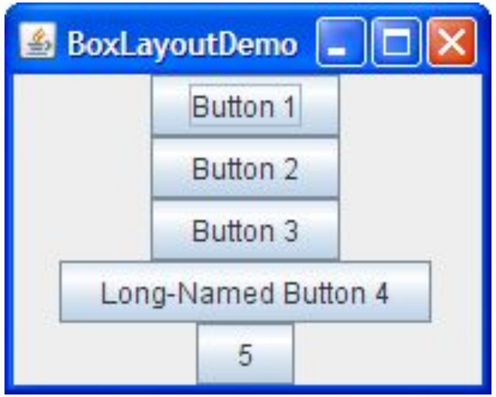
- Grid Organization
- Set rows and columns
    - Can also sets gaps
    - Left → Right
    - Top → Bottom



- New GridLayout(rows, cols)

Box Layout

- Provides different axis:
    - X_AXIS
    - Y_AXIS
    - etc.*
- Different syntax



- new BoxLayout(target, BoxLayout.Y_AXIS)

More Layouts

- There are more layout managers
- Provides additional flexibility

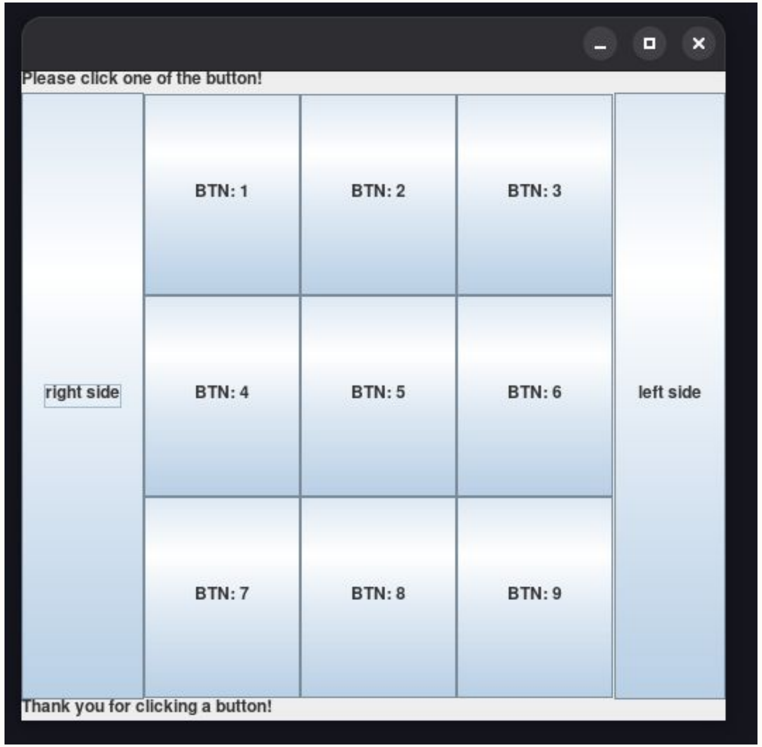
- “A Visual Guide to Layout Managers”
- Let’s take a look

Nesting Layouts

- We already saw how we can nest JPanels
- We can use this to next Layouts
- Can use this to design more complex applications

Nesting Layouts

- What if we wanted to make something like this?
- What layouts would we want?



Nesting Layouts

- Composition of layouts
- This idea transfers beyond what you’ll do in swing

- Every UI framework has this same idea

More JFrames

- Right now our buttons do something simple
- Let’s do something more complicated
  - More (custom) JFrames
  - Dispose current JFrame

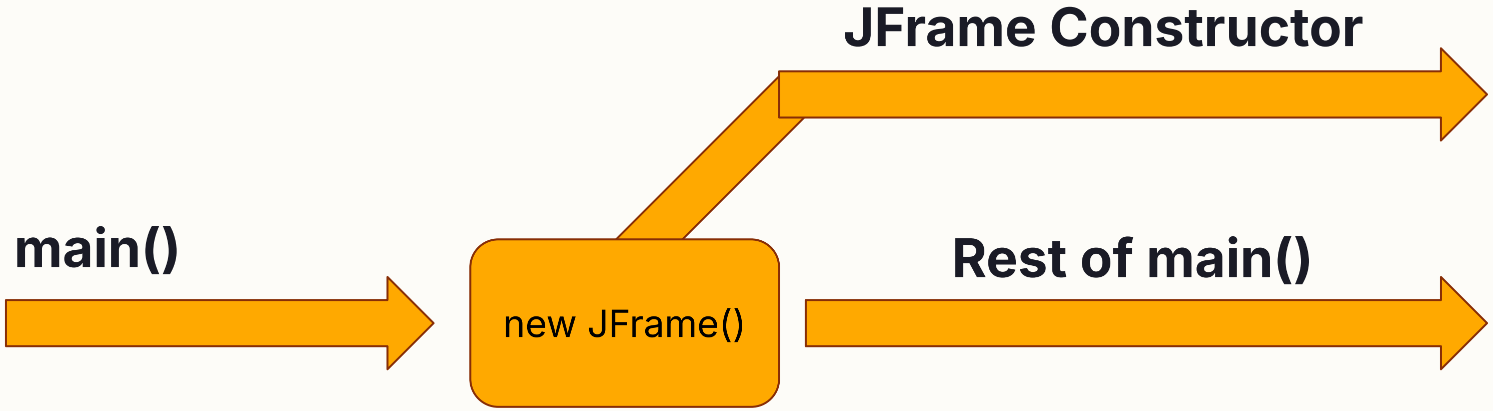
Work

- How did java make a new JFrame?
- Our code continued after dispose()

- This same thing happens in main()
- What is going on here?

Work

- These pieces of code ran simultaneously



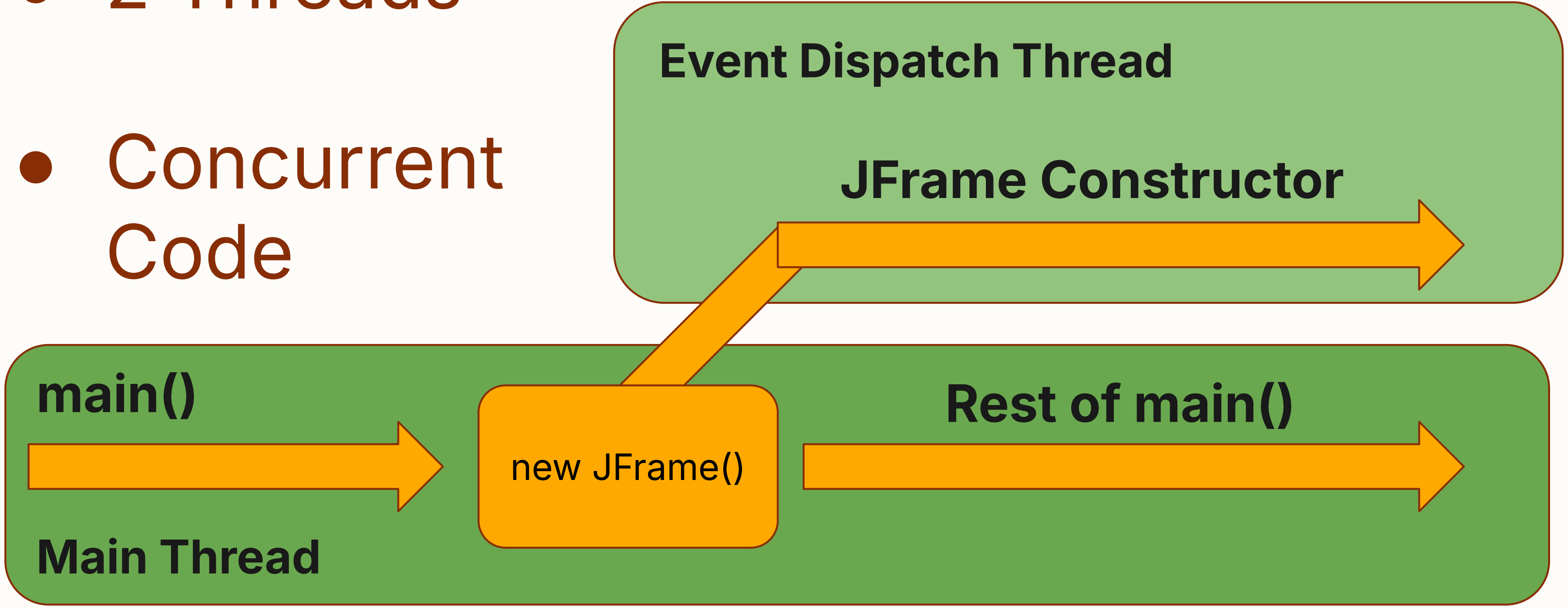
Event Dispatch Thread

- Thread = “Line” of java code execution

- All your code so far has run on the “main” thread
- Swing code runs on the **“Event Dispatch Thread”**

EDT

- 2 Threads
- Concurrent Code



Event Dispatch Thread

- This can get us into trouble
- By default, *all* our Swing code will run on the EDT

- EDT is responsible for all swing events (movement, graphics, etc)
- What if we did a lot of work?

Practice with Swing

- Let’s build something actually useful
- To-Do app
    - Common example
    - Design:
        - How should it look?
    - Data Model
        - How should we code it?

Design

- How do we want it to look?
  - Let’s draw it
  - Think about layouts

Data Modeling

- How do we want to code it?
    - Think about our design
    - How can we link those with our current tools?

Custom Swing Graphics

Reese Hatfield

Separation of Concern

Reese Hatfield