Java Swing

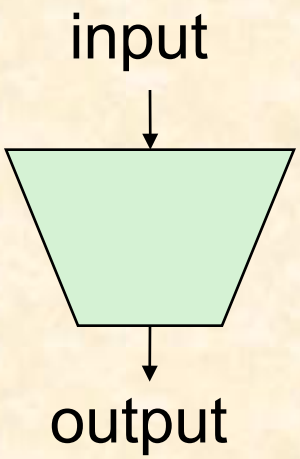
Chris North CS3724

&

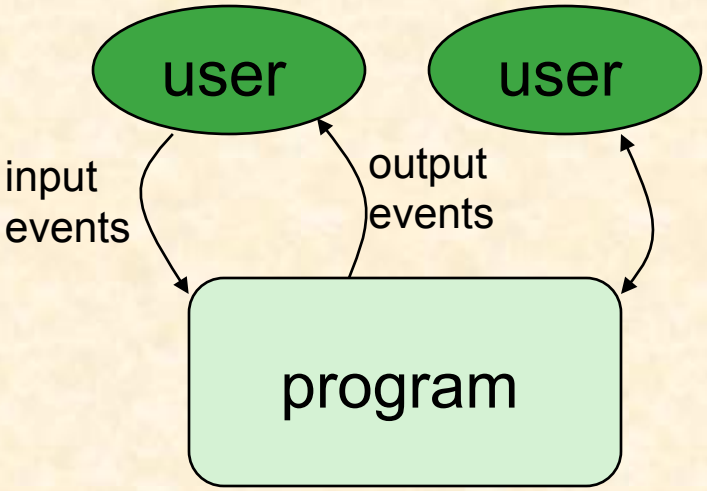
CS2110

Interactive Programs

- "Classic" view of computer programs: transform inputs to outputs, stop



- Event-driven programs: interactive, long-running
  - Servers interact with clients
  - Applications interact with user(s)



GUI Motivation

- Interacting with a program
  - Program-Driven = Proactive
    - Statements execute in sequential, predetermined order
    - Typically use keyboard or file I/O, but program determines when that happens
    - Usually *single-threaded*
  - Event-Driven = Reactive
    - Program waits for user input to activate certain statements
    - Typically uses a GUI (Graphical User Interface)
    - Often *multi-threaded*

Java Support for Building GUIs

- Our main focus: Swing
  - Building blocks of GUIs
  - Windows & components
  - User interactions
  - Built upon the AWT (Abstract Window Toolkit)
  - Java event model

Swing versus AWT

- AWT came first
- Swing builds on AWT
  - Strives for total portability
  - Basic architecture is pretty standard

Java Foundation Classes

- Classes for building GUIs
- Pluggable Look-and-Feel Support
  - Controls look-and-feel for particular windowing environment
  - E.g., Java, Windows, Mac
- Accessibility API
  - Supports assistive technologies such as screen readers and Braille
- Java 2D
  - Drawing
  - Includes rectangles, lines, circles, images, ...
- Drag-and-drop
  - Support for drag and drop between Java application and a native application
- Internationalization
  - Support for other languages

GUI Statics and GUI Dynamics

Statics: what's drawn on the screen

- Components
  - buttons, labels, lists, sliders, menus, ...
- Containers: components that contain other components
  - frames, panels, dialog boxes, ...
- Layout managers: control placement and sizing of components

Dynamics: user interactions

- Events
- button-press, mouse-click, key-press, ...
- Listeners: an object that responds to an event
- Helper classes
- Graphics, Color, Font, FontMetrics, Dimension, ...

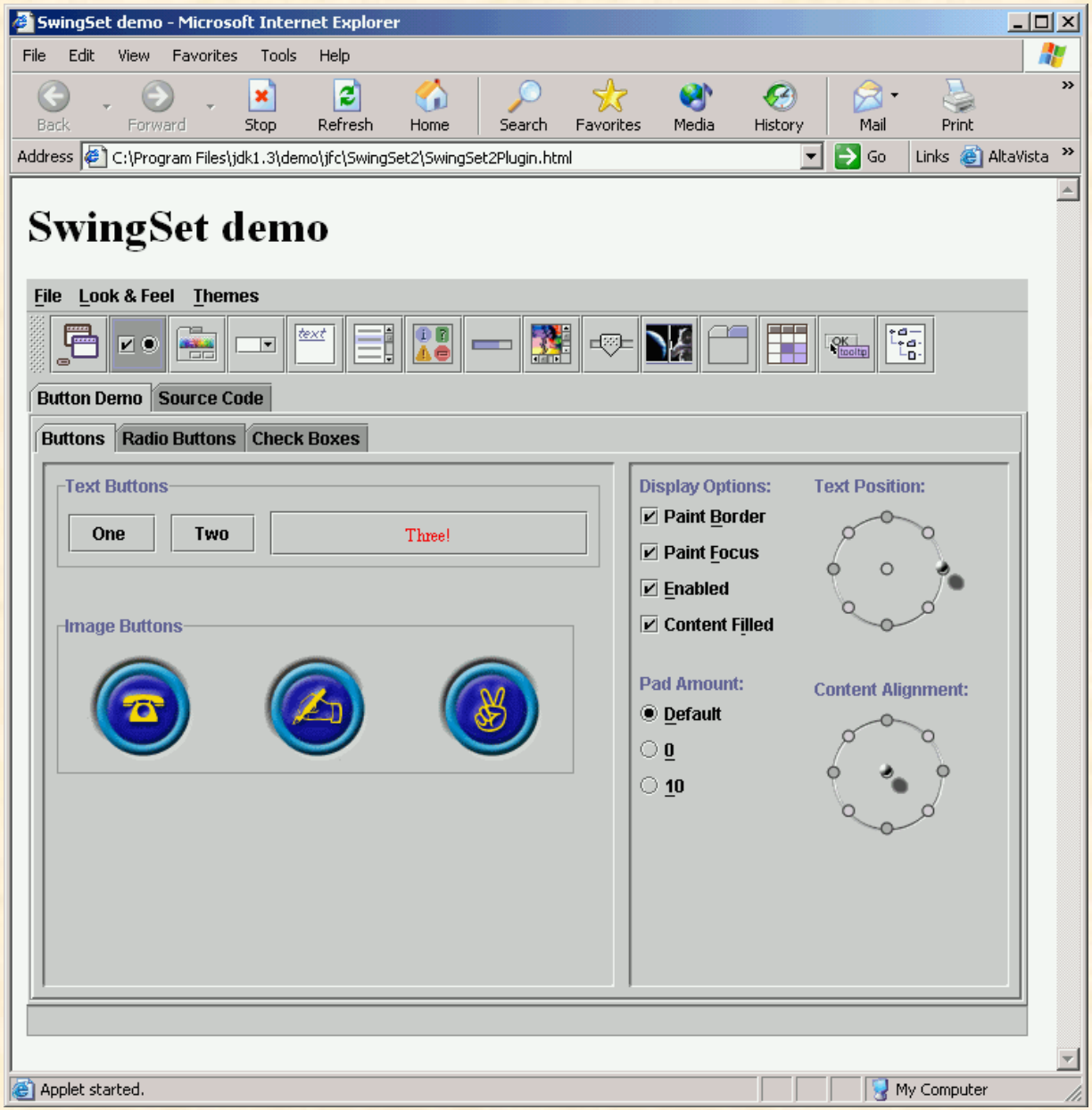
Swing

- import javax.swing.*
- Extends AWT
- Tons of new improved components
- Standard dialog boxes, tooltips, …
- Look-and-feel, skins
- Event listeners

https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html

Search for “Swing Set Demo”

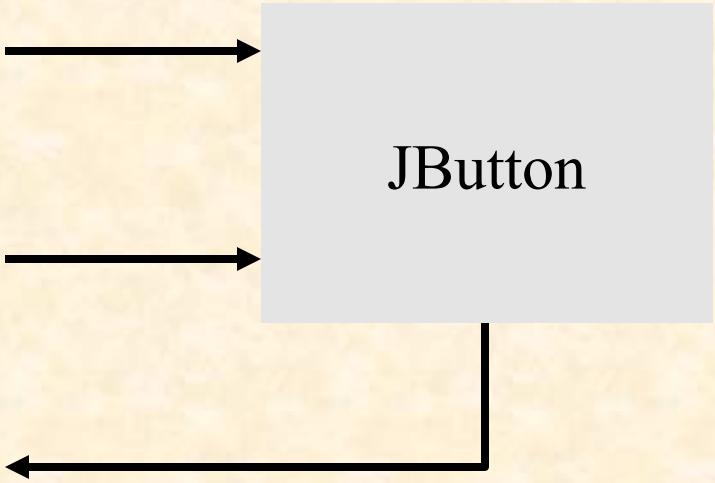
- Many predefined GUI components



GUI Component API

- Java: GUI component = class

- Properties
  -
- Methods
  -
- Events
  -



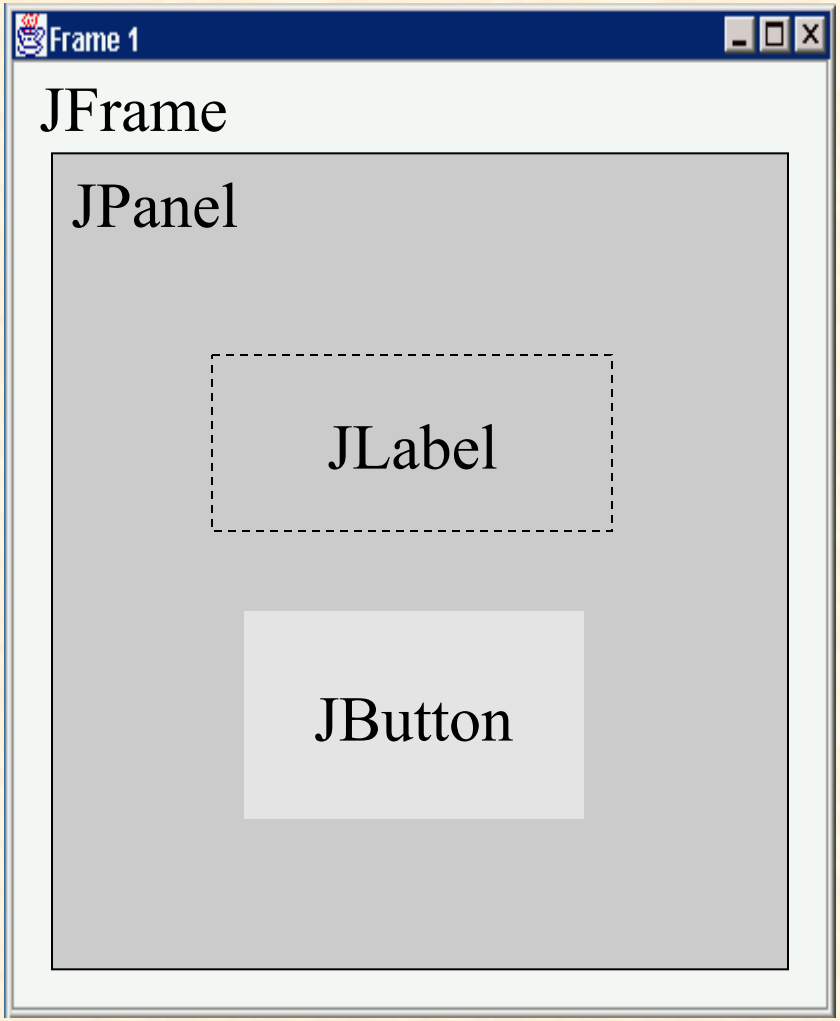
Using a GUI Component 1

1. Create it
   - Instantiate object: b = new JButton("press me");
2. Configure it
   - Properties: ~~b.text = "press me"~~; [avoided in java]
   - Methods: b.setText("press me");
3. Add it
   - panel.add(b);
4. Listen to it
   - Events: Listeners

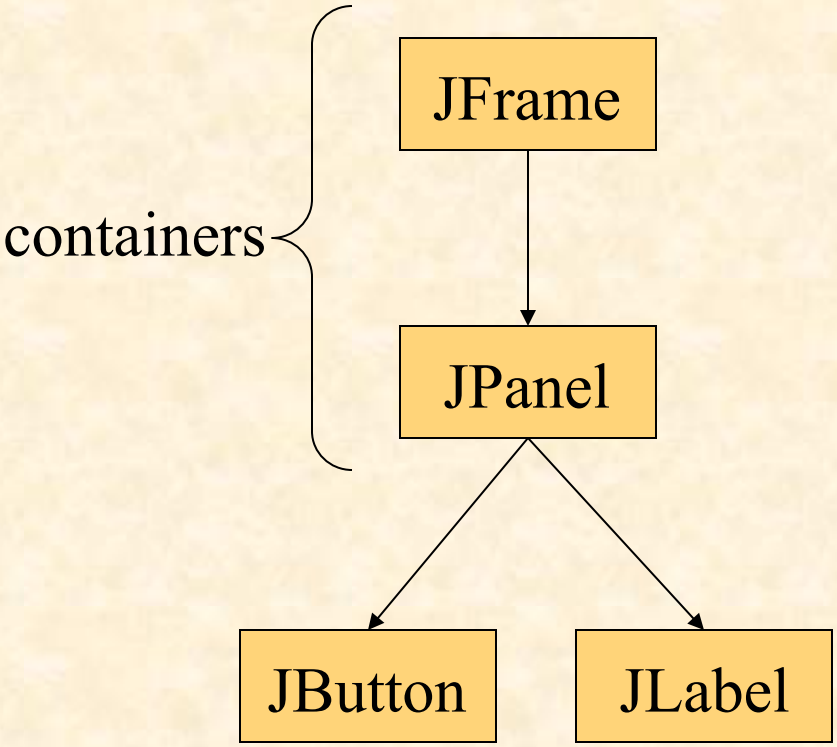
JButton

Anatomy of an Application GUI

GUI



Internal structure



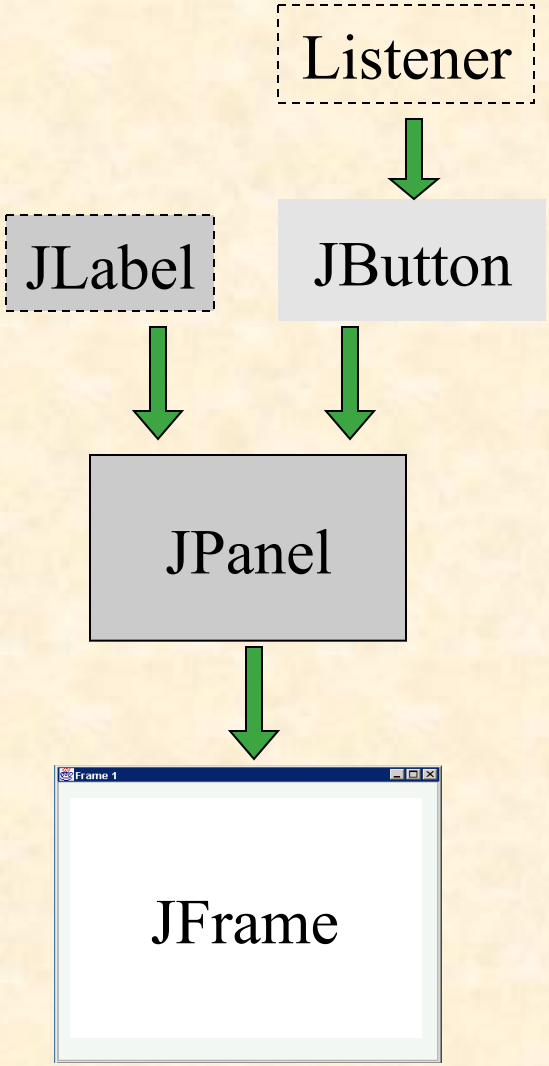
Using a GUI Component 2

1. Create it
2. Configure it
3. Add children (if container)
4. Add to parent (if not JFrame)
5. Listen to it

order important

Build from bottom up

- Create:
  - Frame
  - Panel
  - Components
  - Listeners
- Add: (bottom up)
  - listeners into components
  - components into panel
  - panel into frame



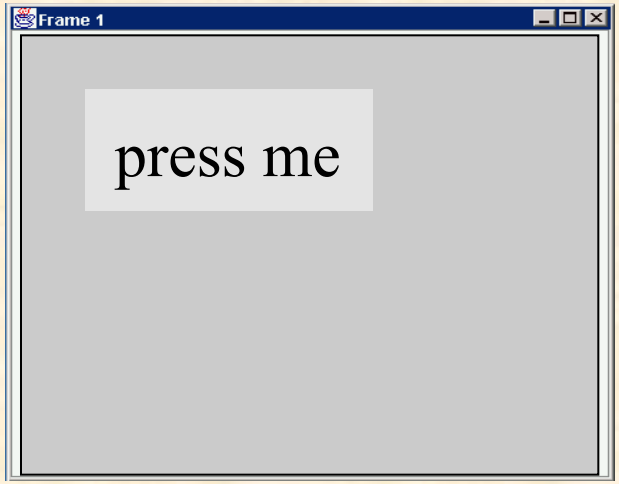
Code

```
import javax.swing.*; // don't forget

JFrame frame = new JFrame("title");
JPanel panel = new JPanel( );
JButton button = new JButton("press me");

panel.add(button);  // add button to panel
frame.setContentPane(panel); //add panel to frame

frame.show();
```



Layout Managers

- Automatically control placement of components in a panel
- Why?

Layout Manager Heuristics

null

none,
programmer
sets x,y,w,h

FlowLayout

Left to right,
Top to bottom

GridLayout

BorderLayout

n

w c e

s

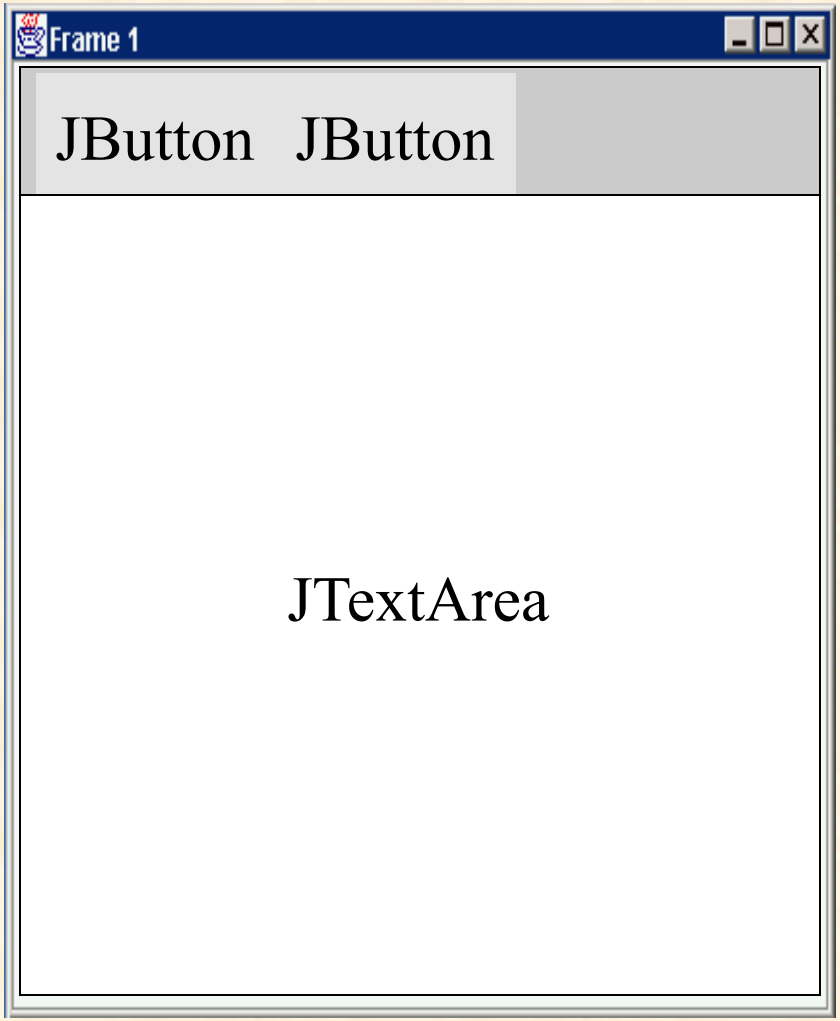
CardLayout

One at a time

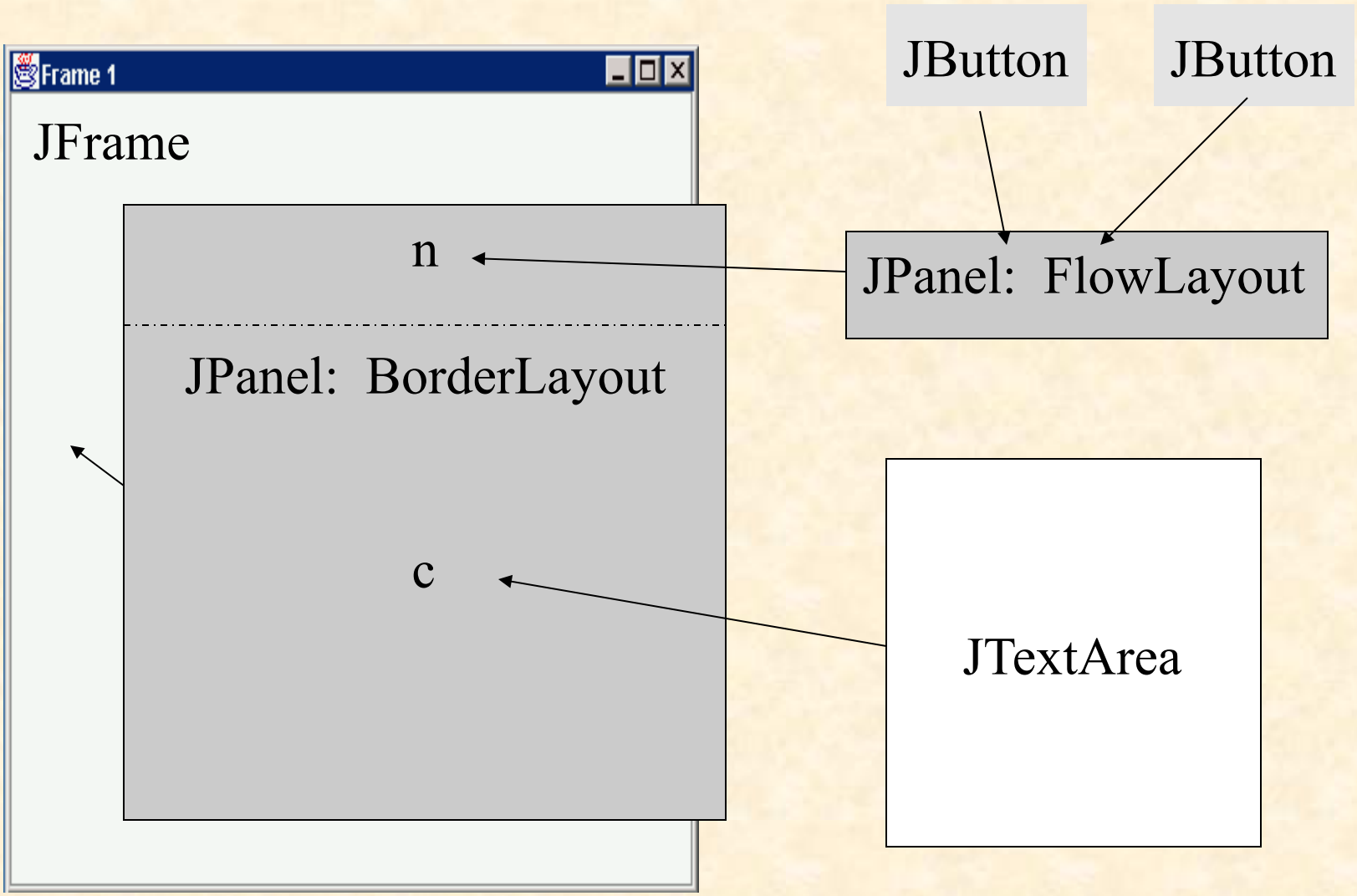
GridBagLayout

JButton

Combinations



Combinations



Swing Tutorial

https://docs.oracle.com/javase/tutorial/uiswing/