



Lecture 24: Graphical User Interfaces

CS 2110, Matt Eichhorn and Leah Perlmutter

November 18, 2025

Roadmap

Java, Complexity, OOP

- start– 9/30

ADTs I

- List, Stack, Queue, Iteration
 - 10/2 – 10/16

ADTs II

- Trees, Set, Map, Hash Table, Graph
 - Tues 10/21- 11/13

Beyond ADTs

- **Graphical User Interfaces & Event-Driven Programming**
 - 11/18, 11/20
- Parallel Programming
 - 11/25, 12/2
- Data Structures and Social Implications
 - 12/4

Overview of today

Introduction

- Motivation
- Ingredients of Graphical User Interfaces

TicTacToe (Tuesday)

- Review code for game logic
- Visual components

TicTacToe (Thursday)

- Interactive components



Motivation

History: How do we interact with computers?

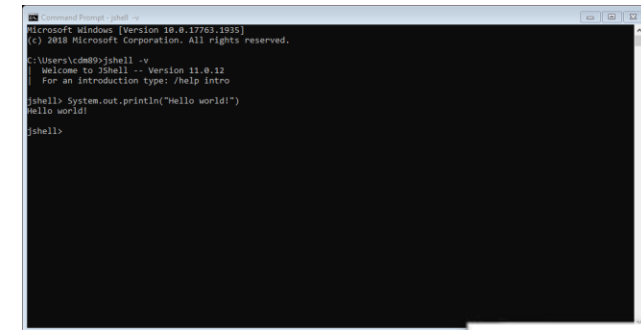
Phase 1: Batch applications

- Computers ran one program to completion
 - Input: Punchcards, files, program arguments
 - Output: Punchcards, files



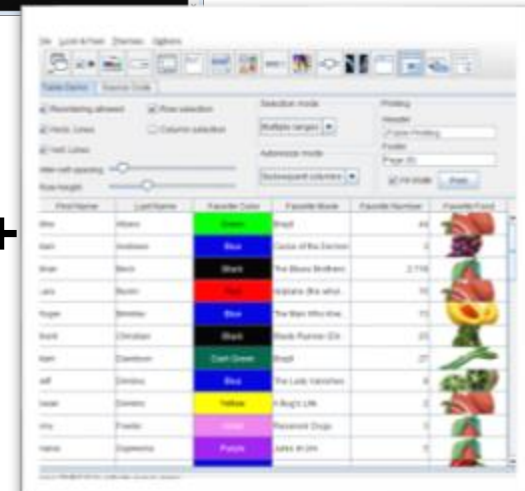
Phase 2: Command-line interface (CLI) 1960s+

- Interactive command-line application
 - Input: User typing (System.in)
 - Output: Console (System.out)



Phase 3: Graphical user interface (GUI) mainstream in 1980s+

- Rich, visual interactions with users
 - Input: mouse, keyboard, gamepad, ...
 - Output: formatted text, images, animations, ...



"The Mother of all Demos"

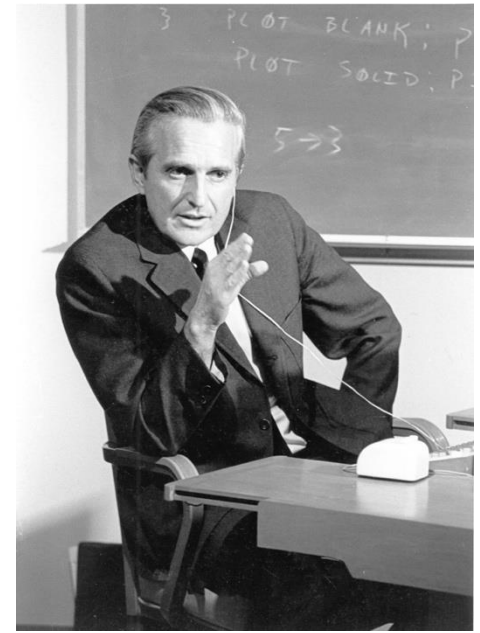
Douglas Engelbart, 1968

90-minute demo at ACM/IEEE Joint Computer Conference

Foundation of "Personal Computing"

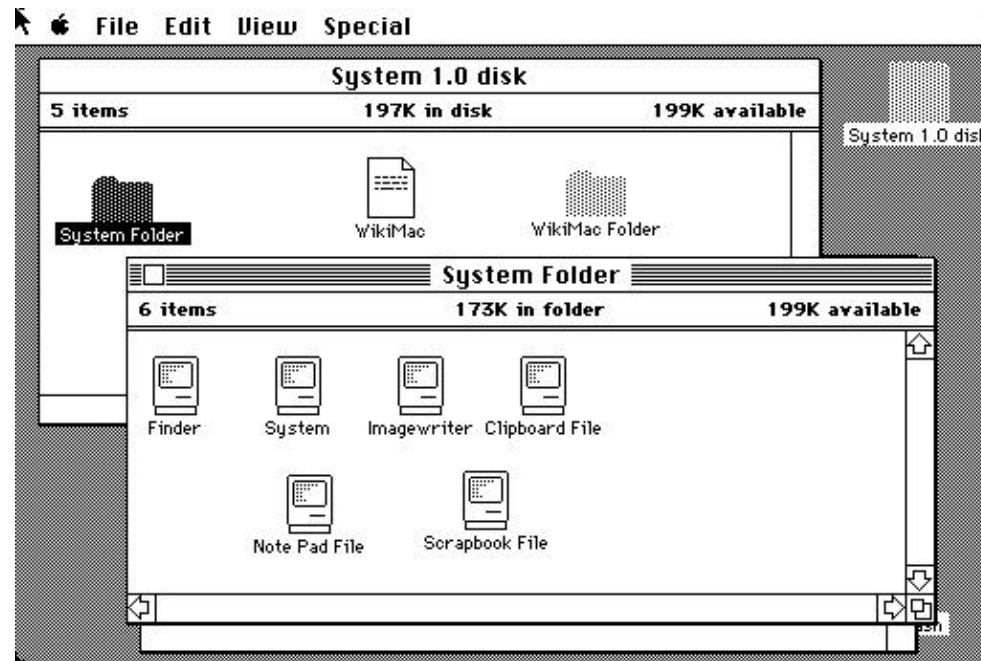
Computers can augment the capabilities of the human mind

- Computer mice
- Application windows
- Hyperlinks
- Word processors
- File menus
- Copy-paste
- Teleconferencing
- Document sharing



GUI Paradigm:

WIMP: Windows, Icons, Menus, Pointer



Macintosh System 1

1984



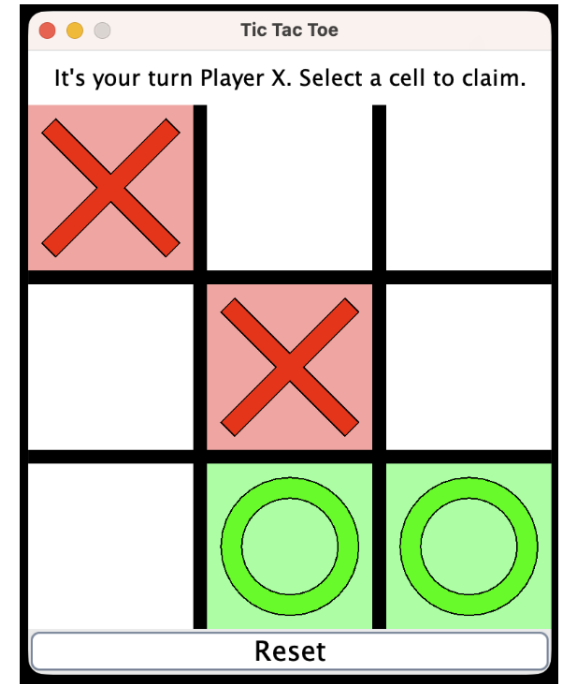


Ingredients for Graphical Applications

Ingredients for Graphical Applications

What's in a GUI?

- Accept user input, interaction - Controller
- Reset the game - Controller
- Display images in color - View
- Backend that stores state of game, connected to visual component - Model
- Display text - View



Model – objects & methods underlying application state

View – visuals

Controller – logic that responds to user input

Model View Controller (MVC)

Model (backend)

- objects underlying application state, including methods that access and modify state

View (frontend)

- stuff shown on the screen for the application

Controller

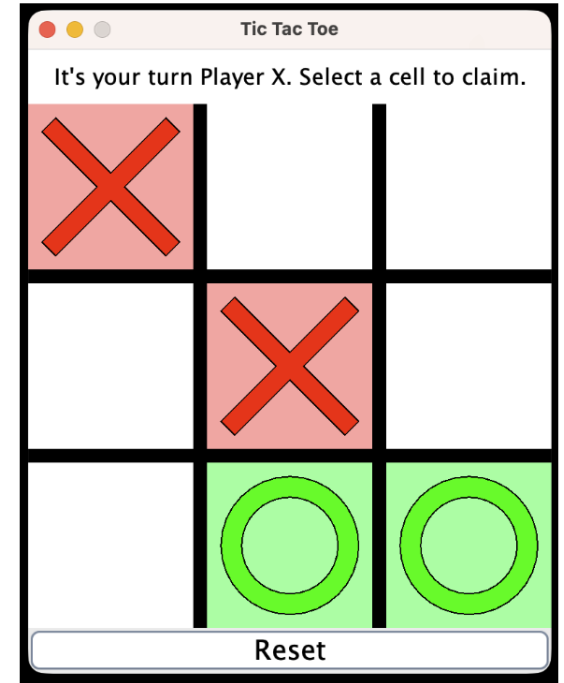
- logic that responds to user inputs (aka events) by updating model and view

The model, view, and controller must connect to each other for the application to work, but the more we can **minimize coupling**, making each component as self-contained as possible, the easier it is to maintain our program.

Ingredients for Graphical Applications

What's in a GUI?

- ...
- ...



Model – objects & methods underlying application state

View – visuals

Controller – logic that responds to user input

GUI Frameworks

Framework – large software library that includes base implementations of many view components and utility classes for layout, rendering, and handling user input

Framework	Programming Language	Kind of application
Angular	TypeScript	web
Flutter	Dart	web, mobile, desktop
ReactNative	JavaScript	primarily mobile
Kotlin	Kotlin	mobile, web, desktop
Electron	any web programming language	desktop
Swing	Java	desktop

→ The concepts and principles of GUI programming that you will learn here are more important than the specifics of the library

Overview of today

Introduction

- Motivation
- Ingredients of Graphical User Interfaces

TicTacToe (Tuesday)

- Review code for game logic
- Visual components

TicTacToe (Thursday)

- Interactive components



Tic Tac Toe Game

Demo

Tic Tac Toe console game (lecture 8)

TicTacToe.java – model

TicTacToeConsole.java – view & controller

TicTacToe graphical game (new)

TicTacToeGraphical.java

Swing Components: JFrame

widget – A graphical element of a graphical application

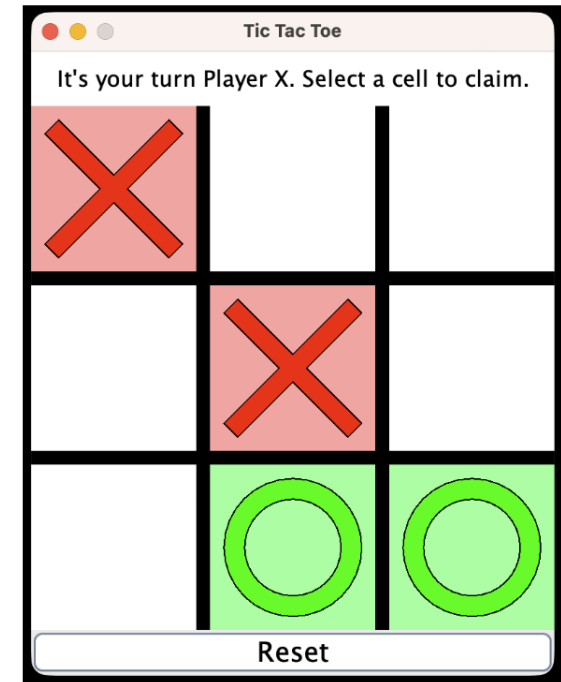
Widgets are typically used to

- a) display information about the state of the program or
- b) provide an interface for user interaction

Component – the Java Swing term for a widget

JFrame

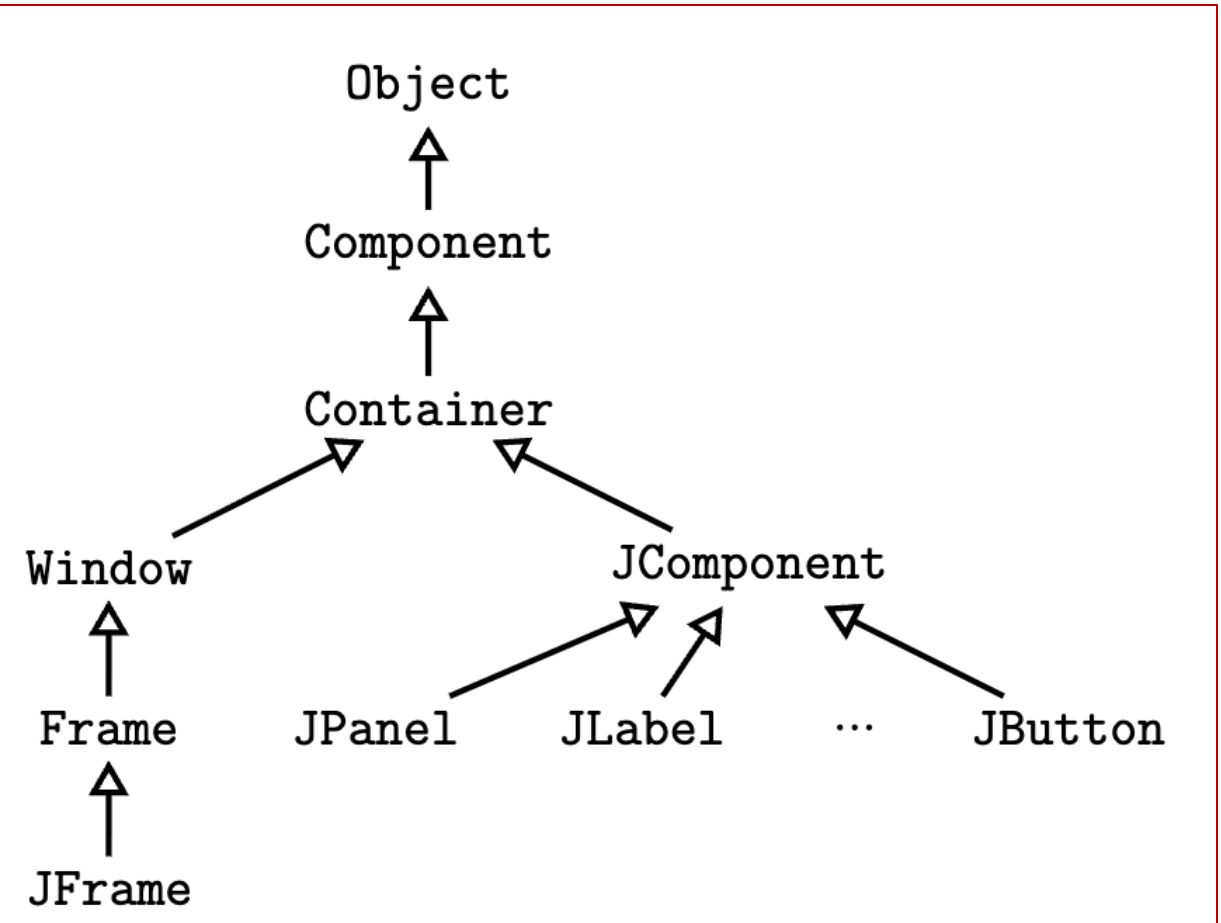
- an application window
- parent class of our Tic Tac Toe graphical interface
- can have title and window control buttons



Demo: TicTacToeGraphical constructor

Swing Widget Types

- JPanel
 - container that supports arrangement of other components inside it
- JLabel
 - for text
- JButton
 - can be clicked to invoke behavior
- Demo
 - the rest of the constructor



Inheritance diagram of Java Swing classes

Hierarchical design

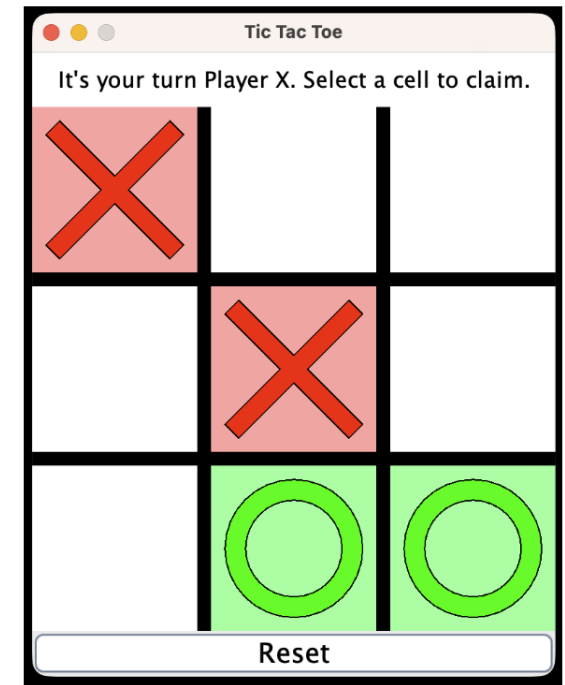
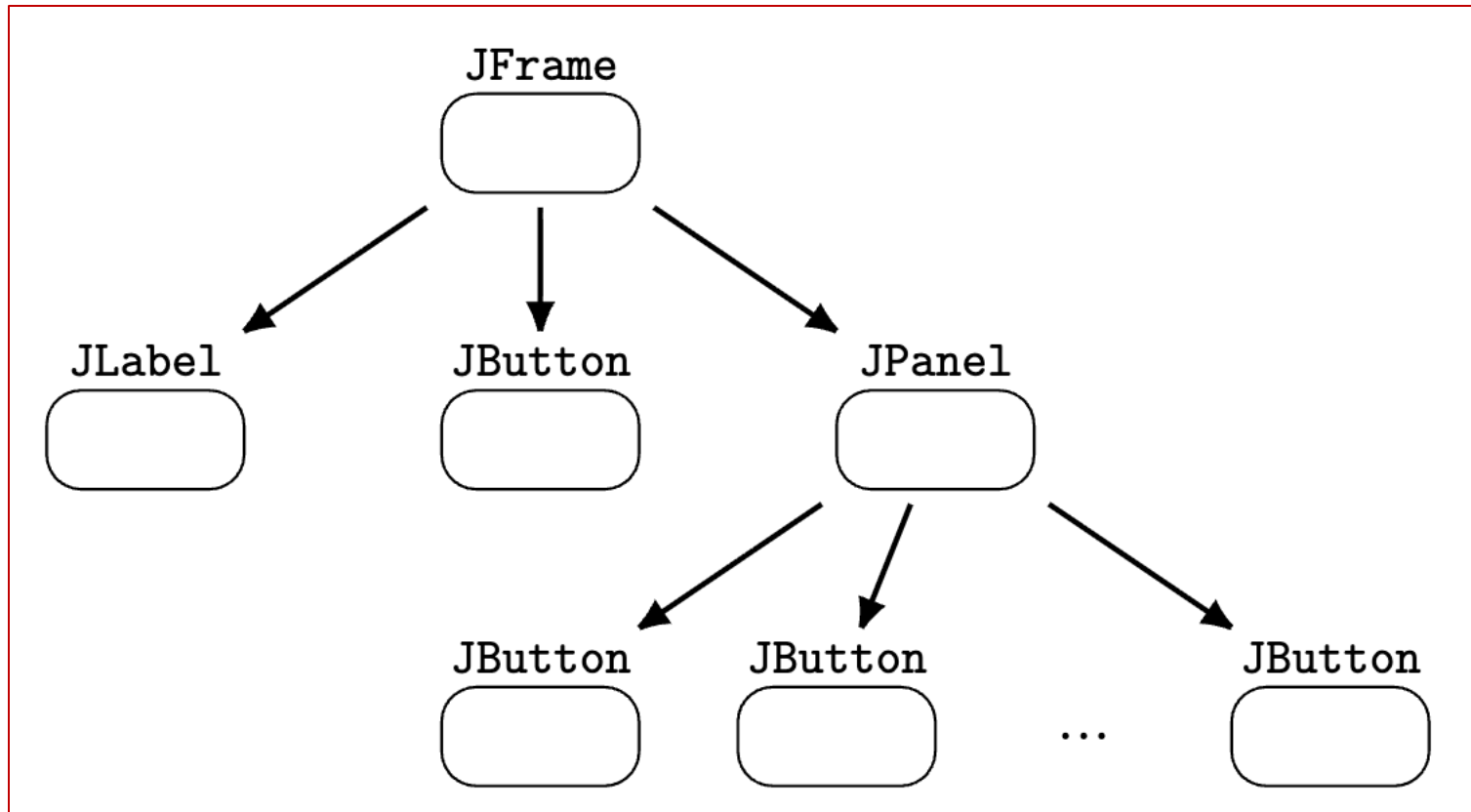


Illustration of how components are hierarchically nested in our TicTacToe application. (This is not an inheritance diagram.)

Hierarchical design

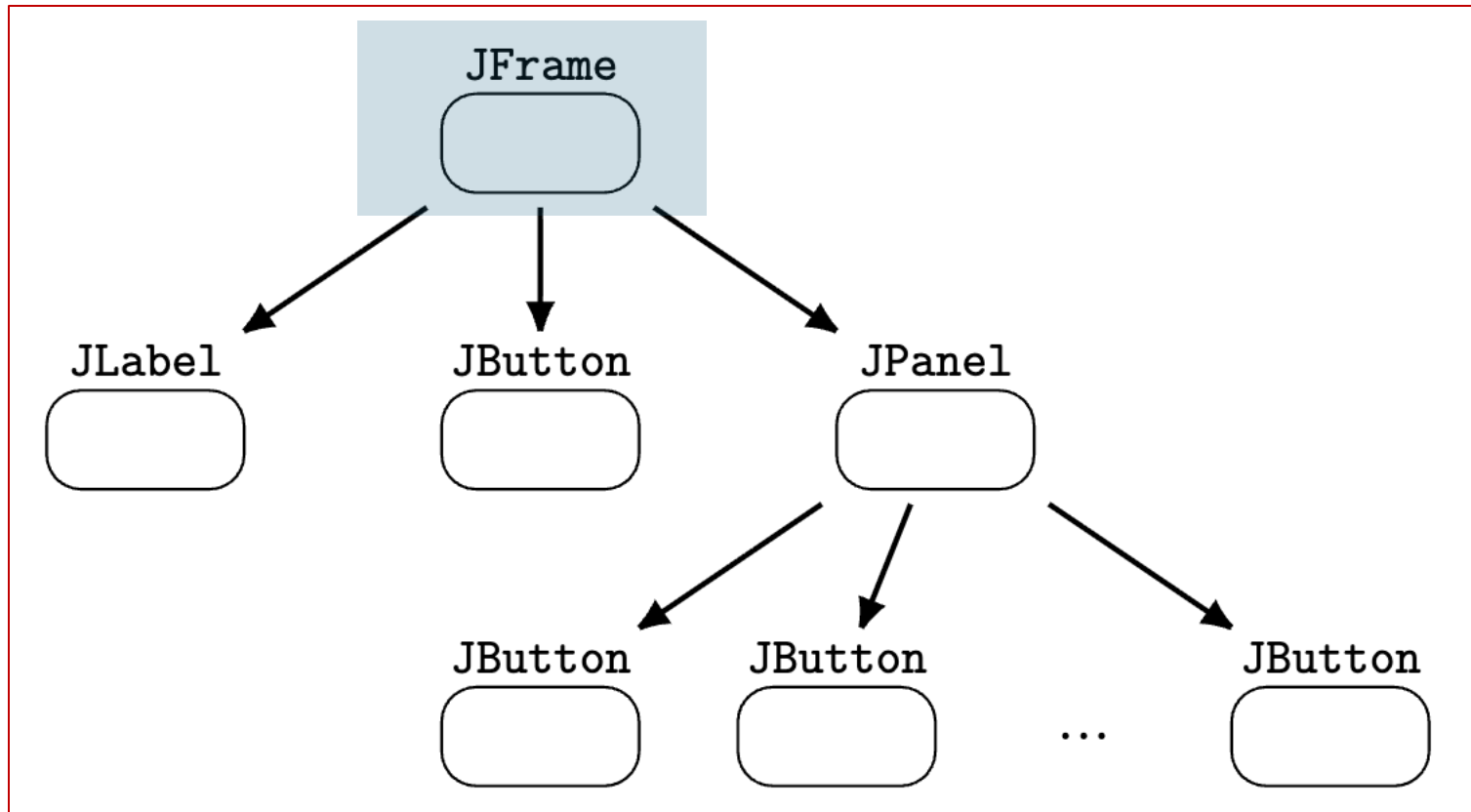
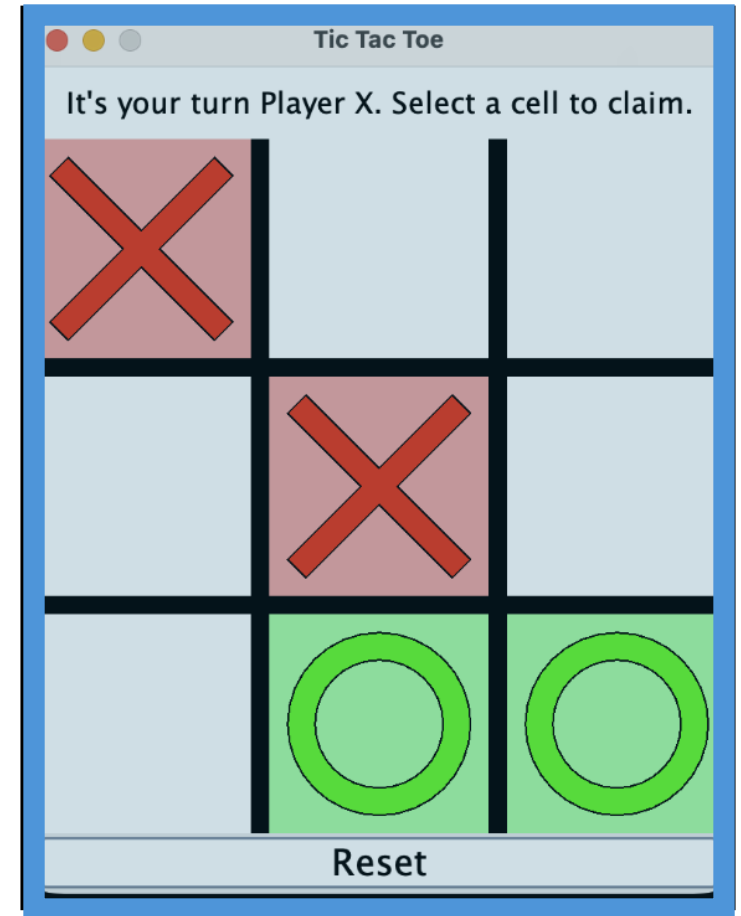


Illustration of how components are hierarchically nested in our TicTacToe application. (This is not an inheritance diagram.)



Hierarchical design

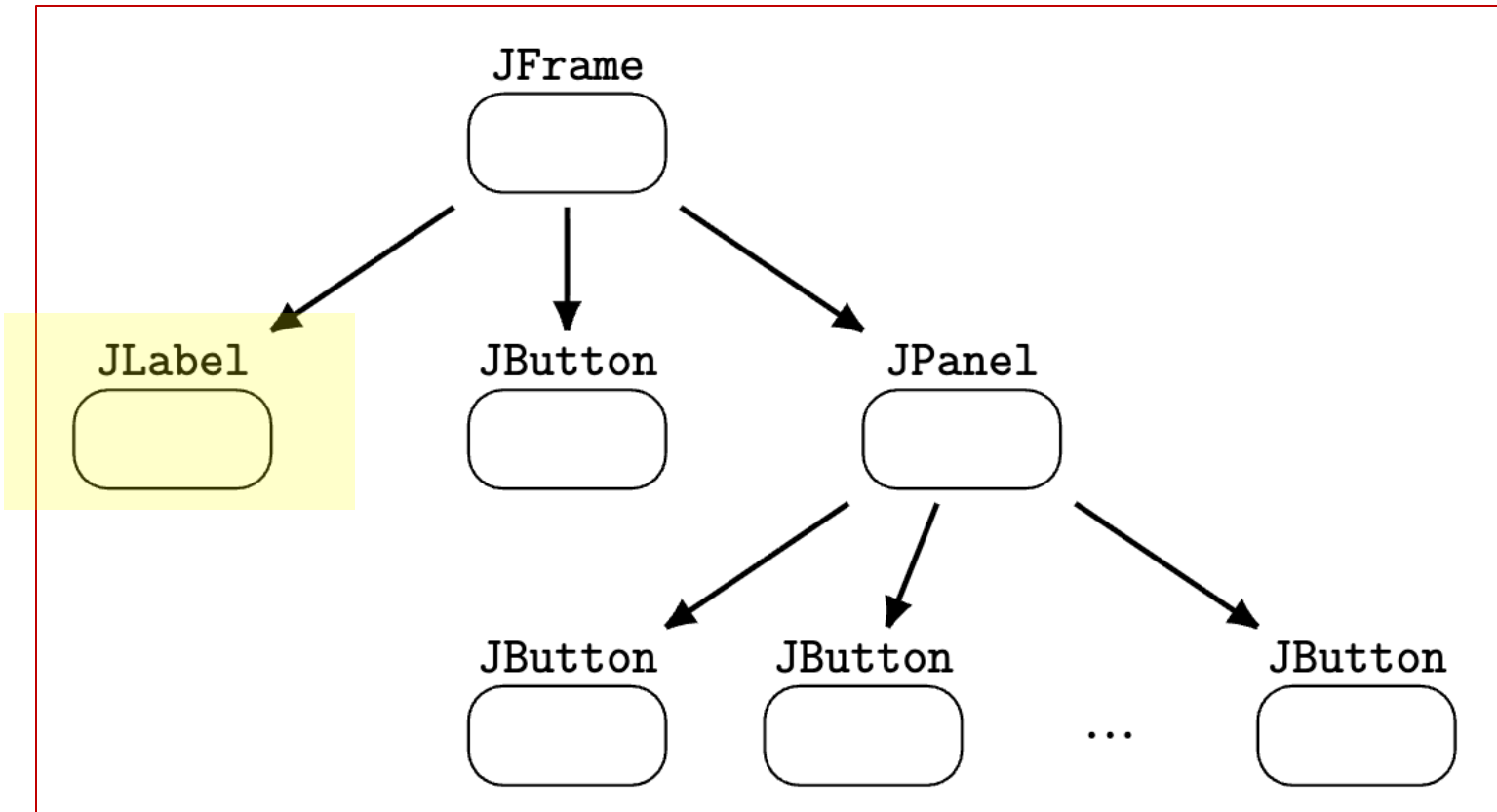
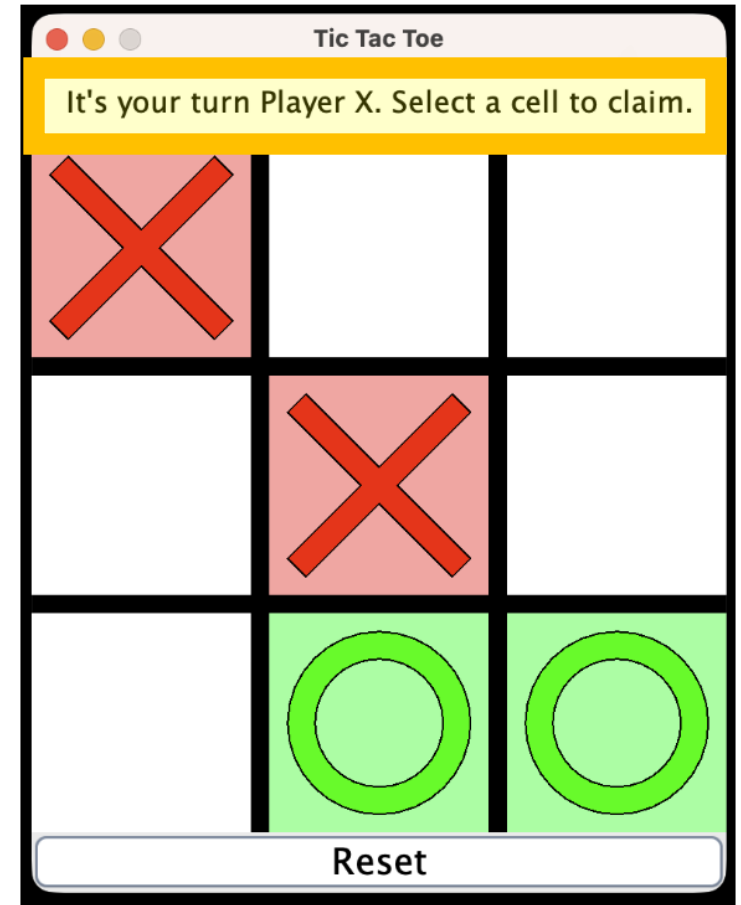


Illustration of how components are hierarchically nested in our TicTacToe application. (This is not an inheritance diagram.)



Hierarchical design

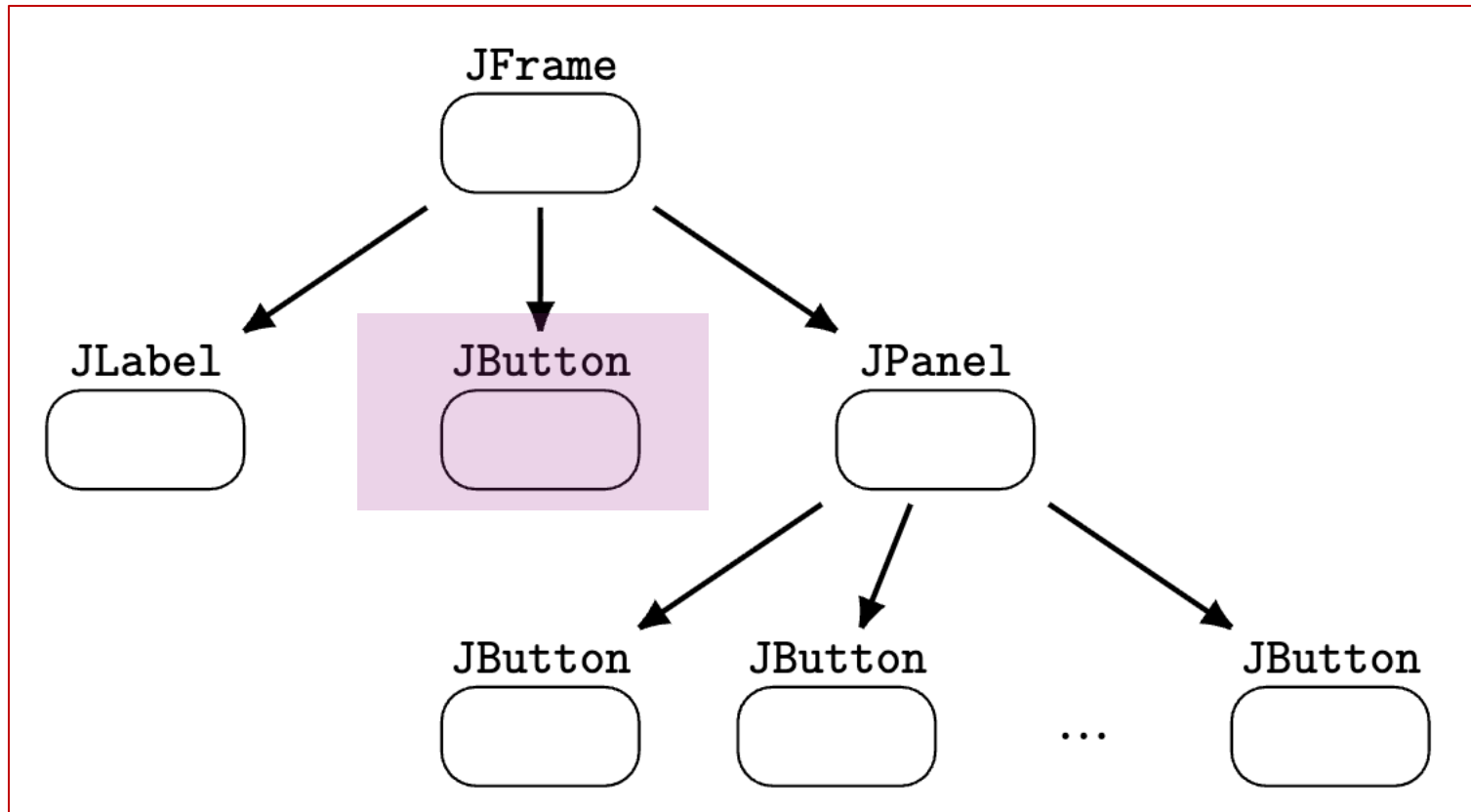
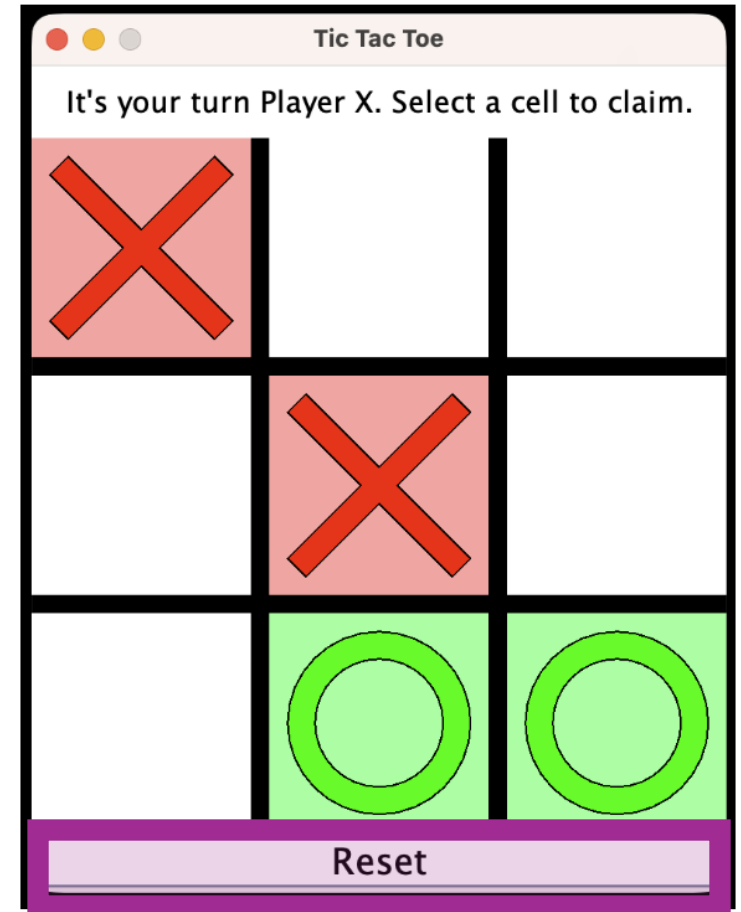


Illustration of how components are hierarchically nested in our TicTacToe application. (This is not an inheritance diagram.)



Hierarchical design

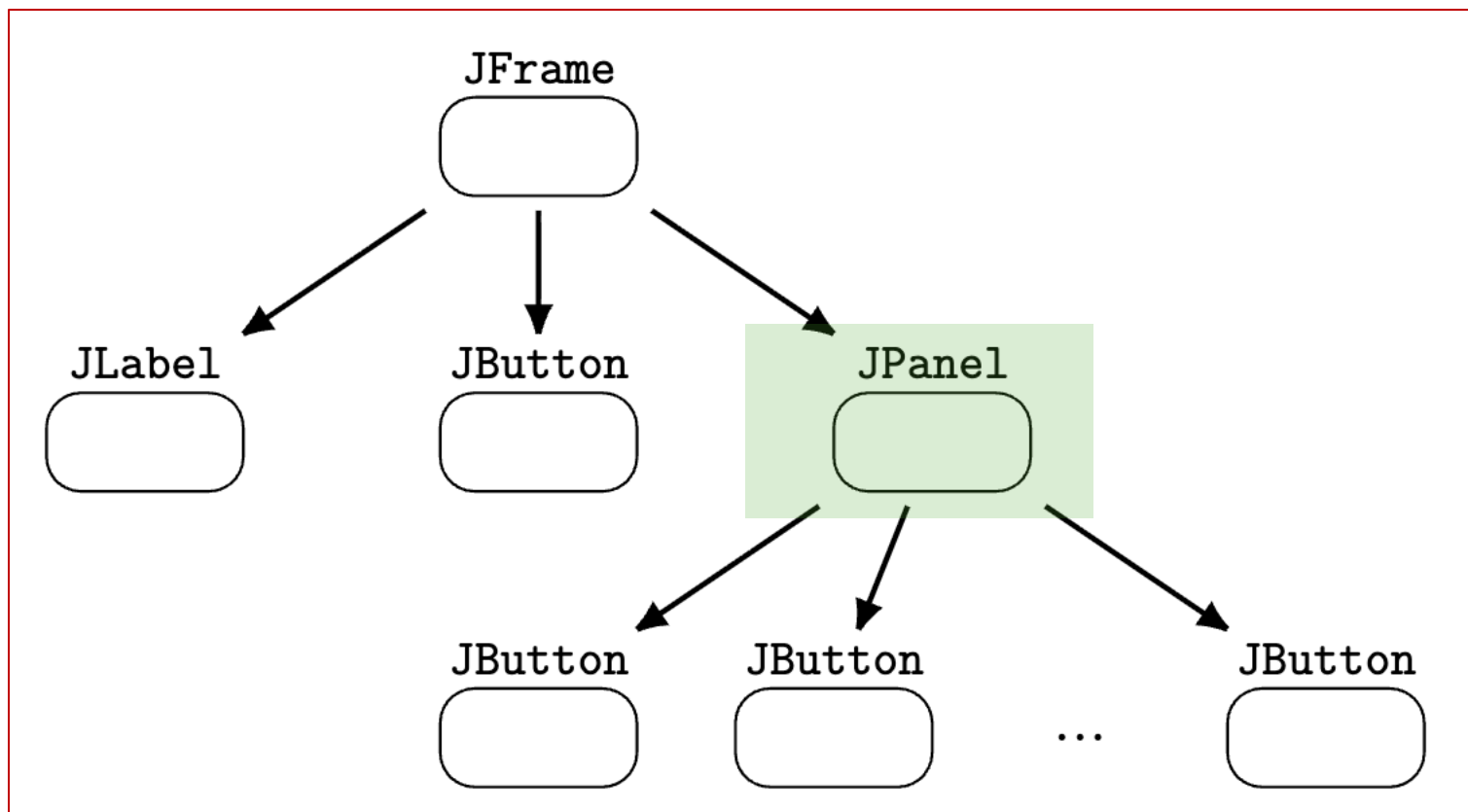
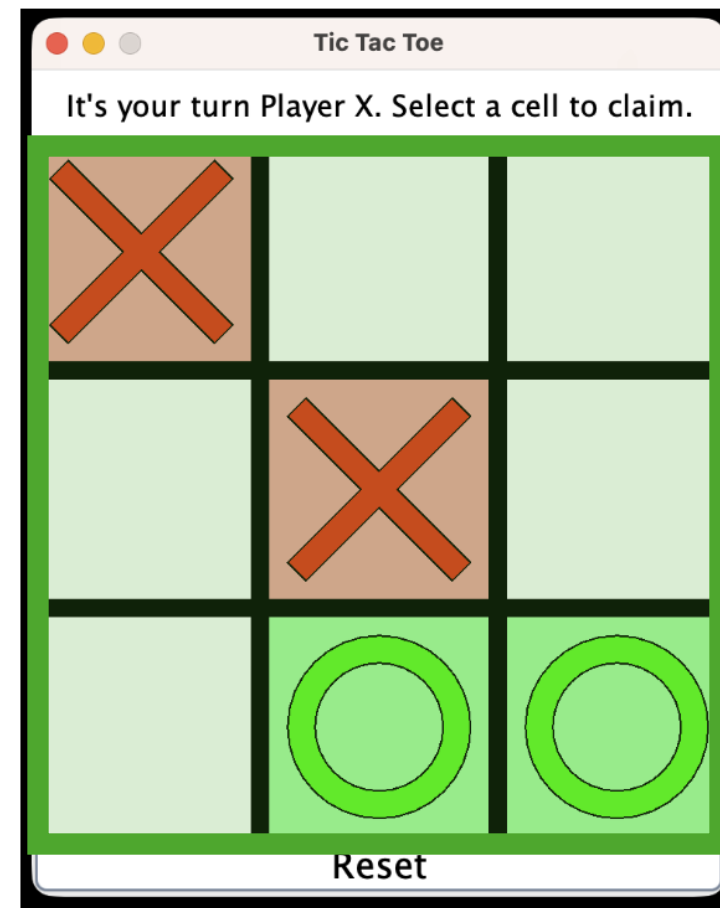


Illustration of how components are hierarchically nested in our TicTacToe application. (This is not an inheritance diagram.)



Hierarchical design

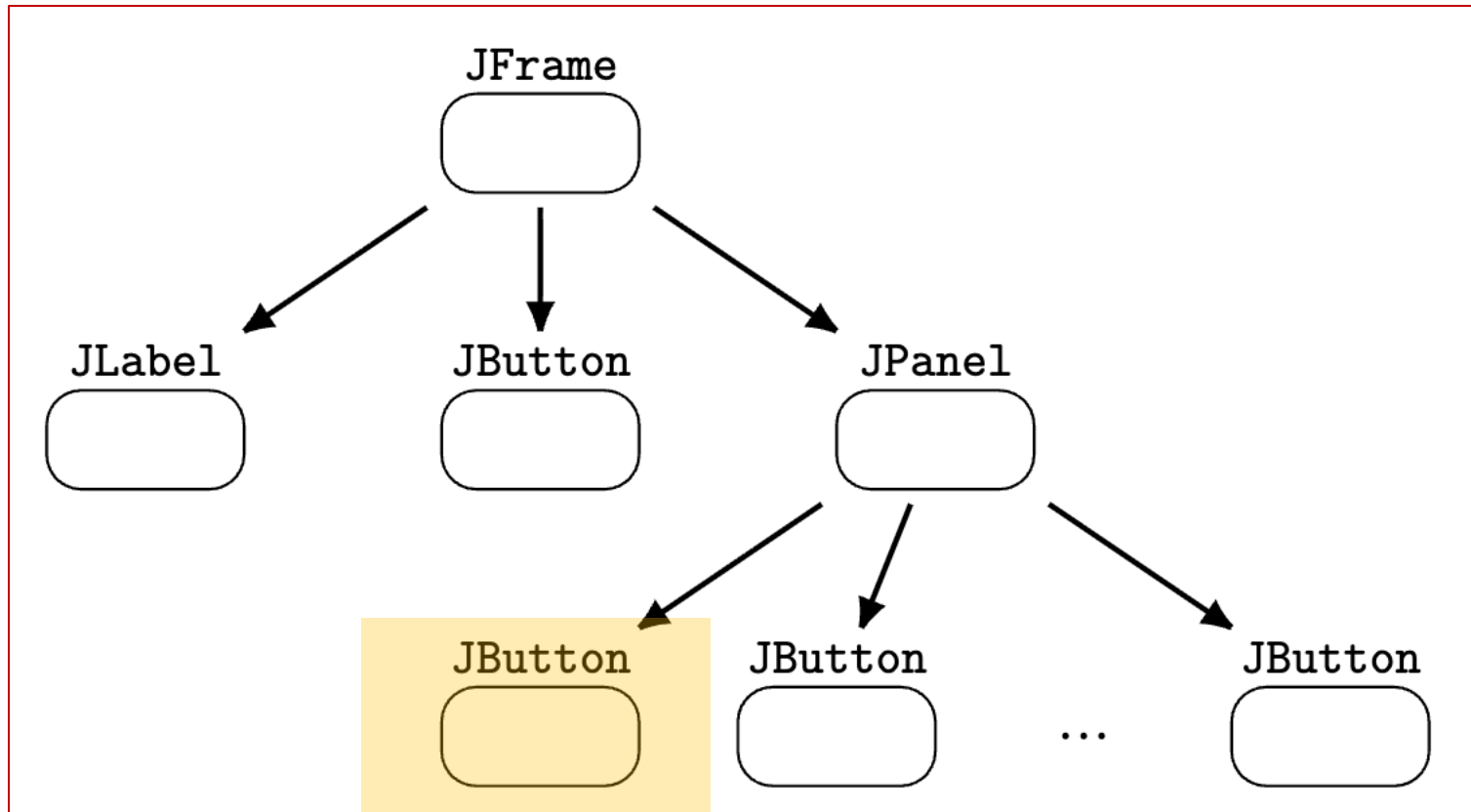
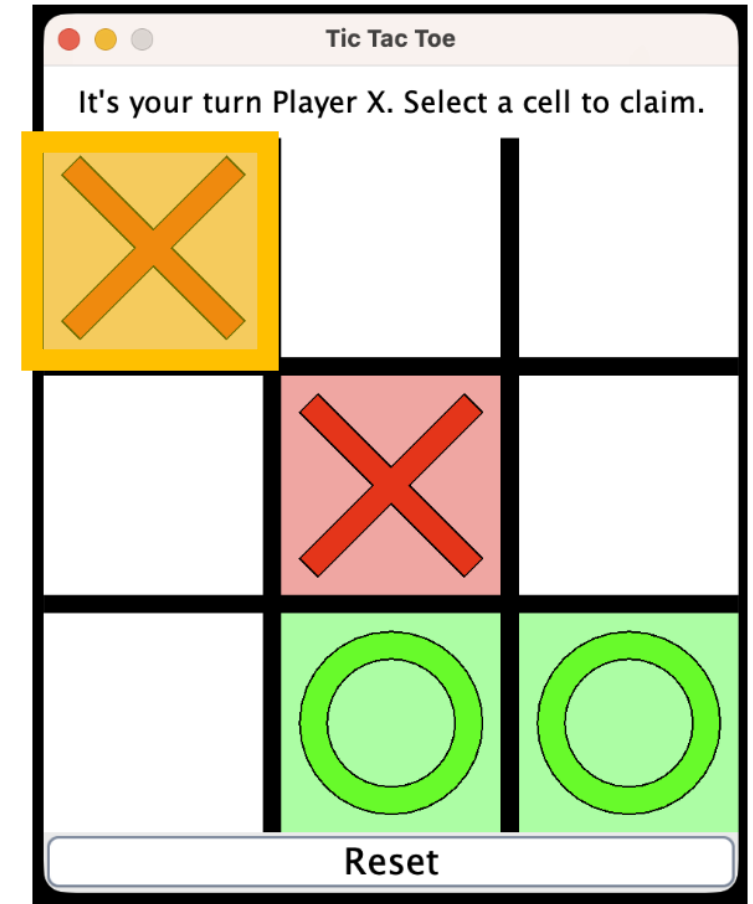


Illustration of how components are hierarchically nested in our TicTacToe application. (This is not an inheritance diagram.)



Hierarchical design

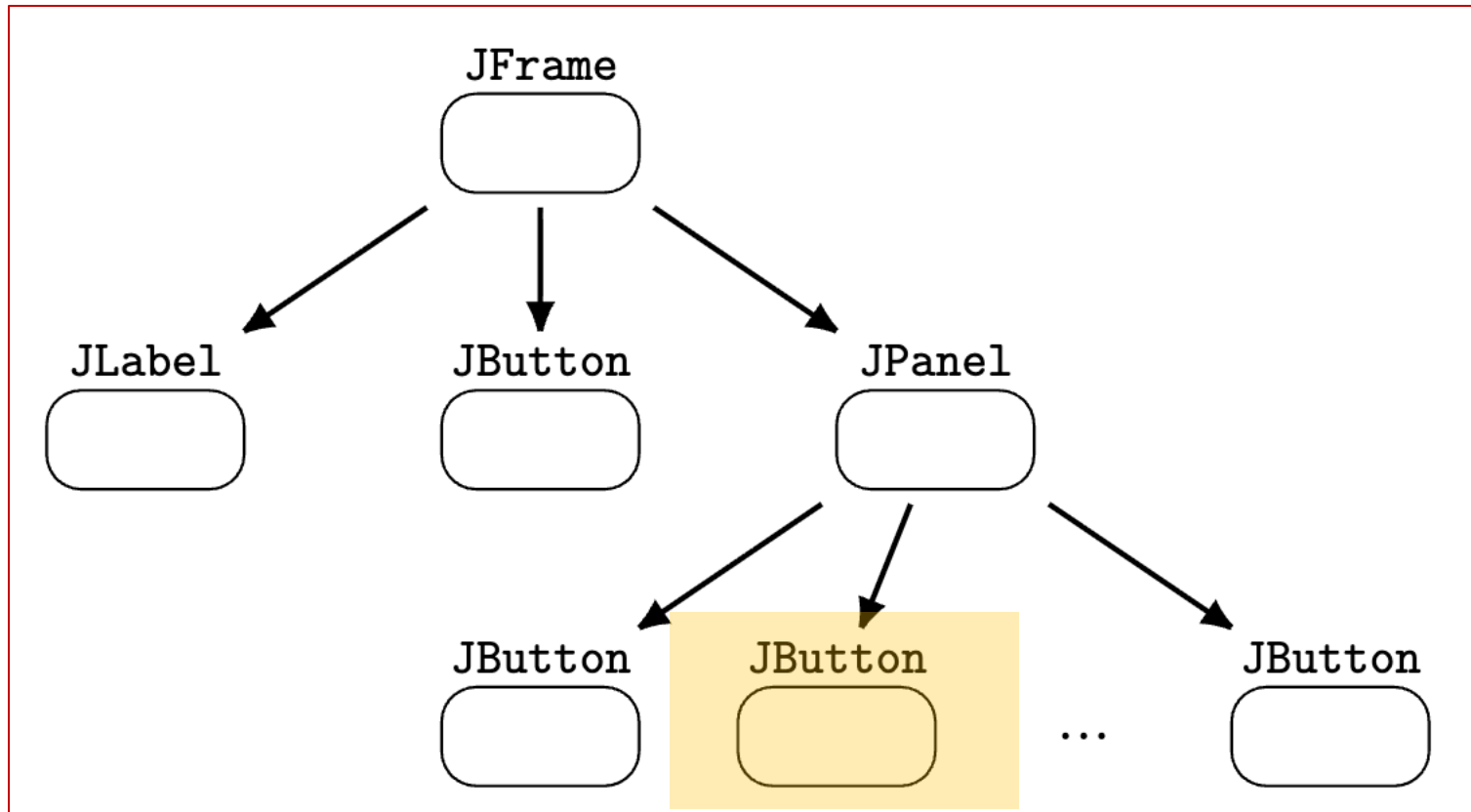
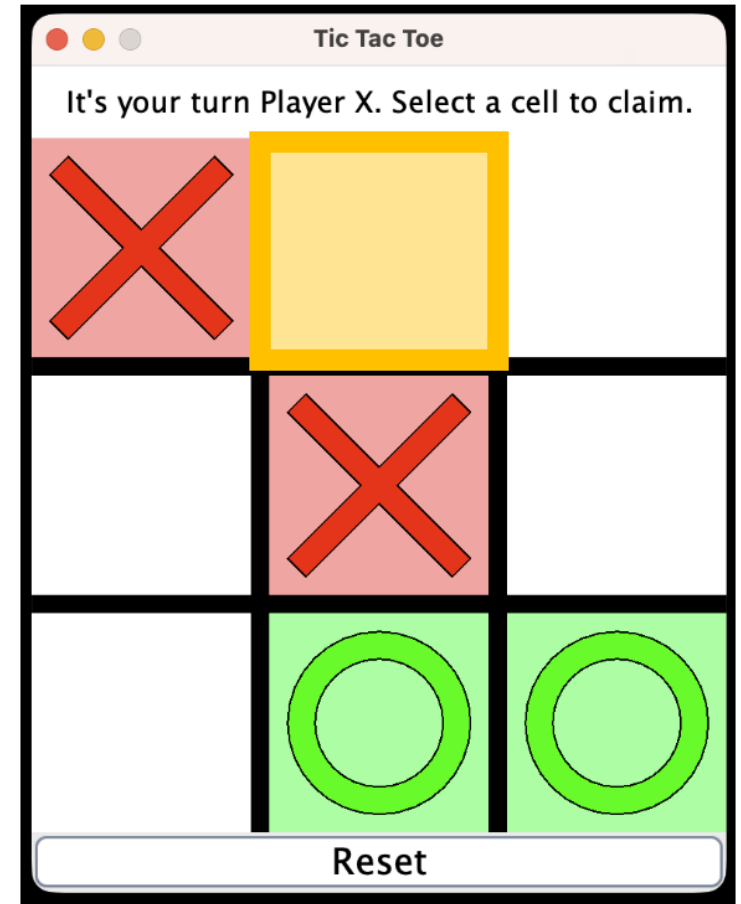


Illustration of how components are hierarchically nested in our TicTacToe application. (This is not an inheritance diagram.)



Hierarchical design

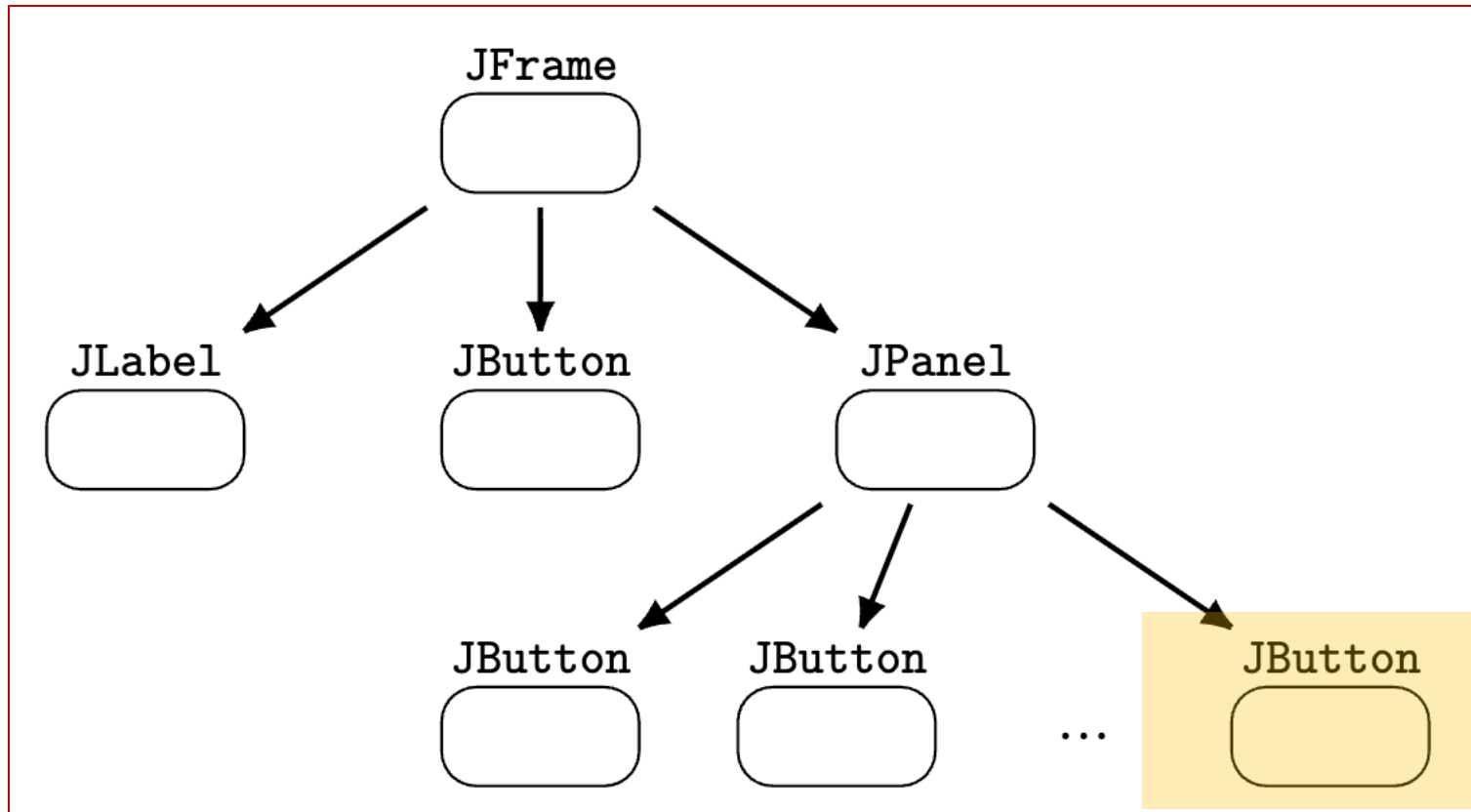
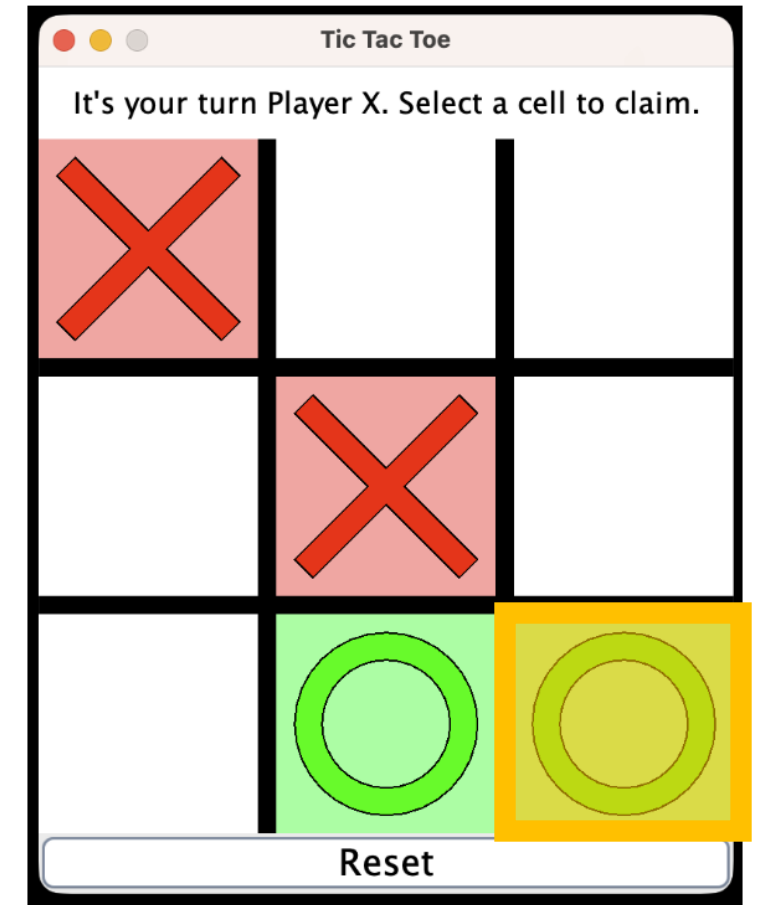
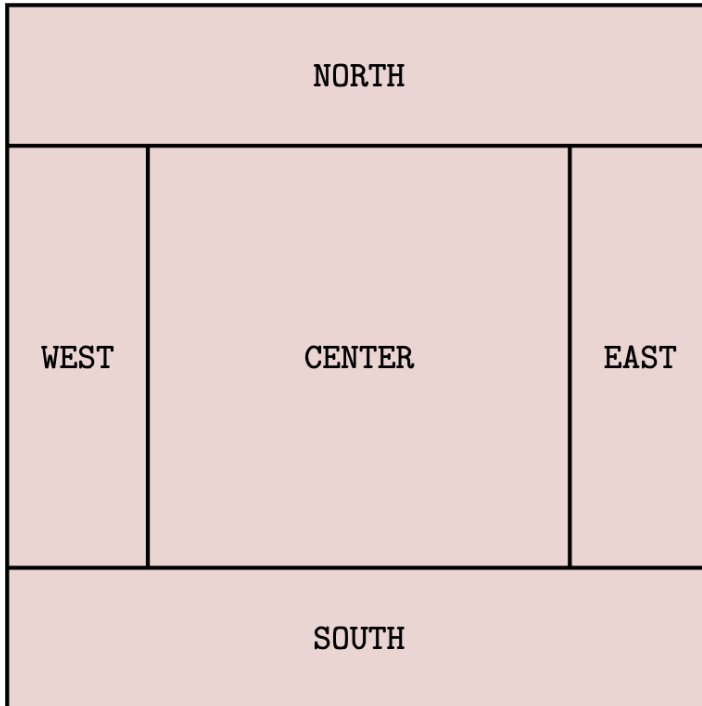


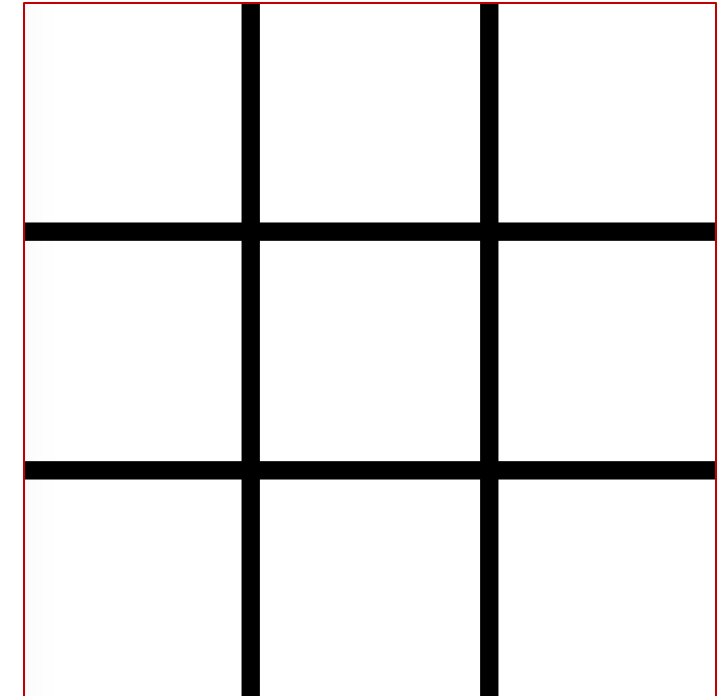
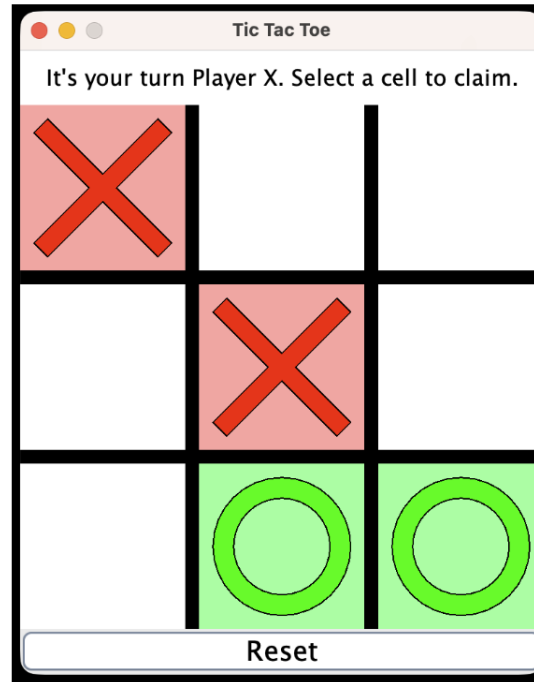
Illustration of how components are hierarchically nested in our TicTacToe application. (This is not an inheritance diagram.)



Layout Managers



Border Layout
Default location is Center



Grid layout (can have any dimensions)
Items are added in row-major order

Demo: looking at layouts

TicTacToeGraphical – border layout

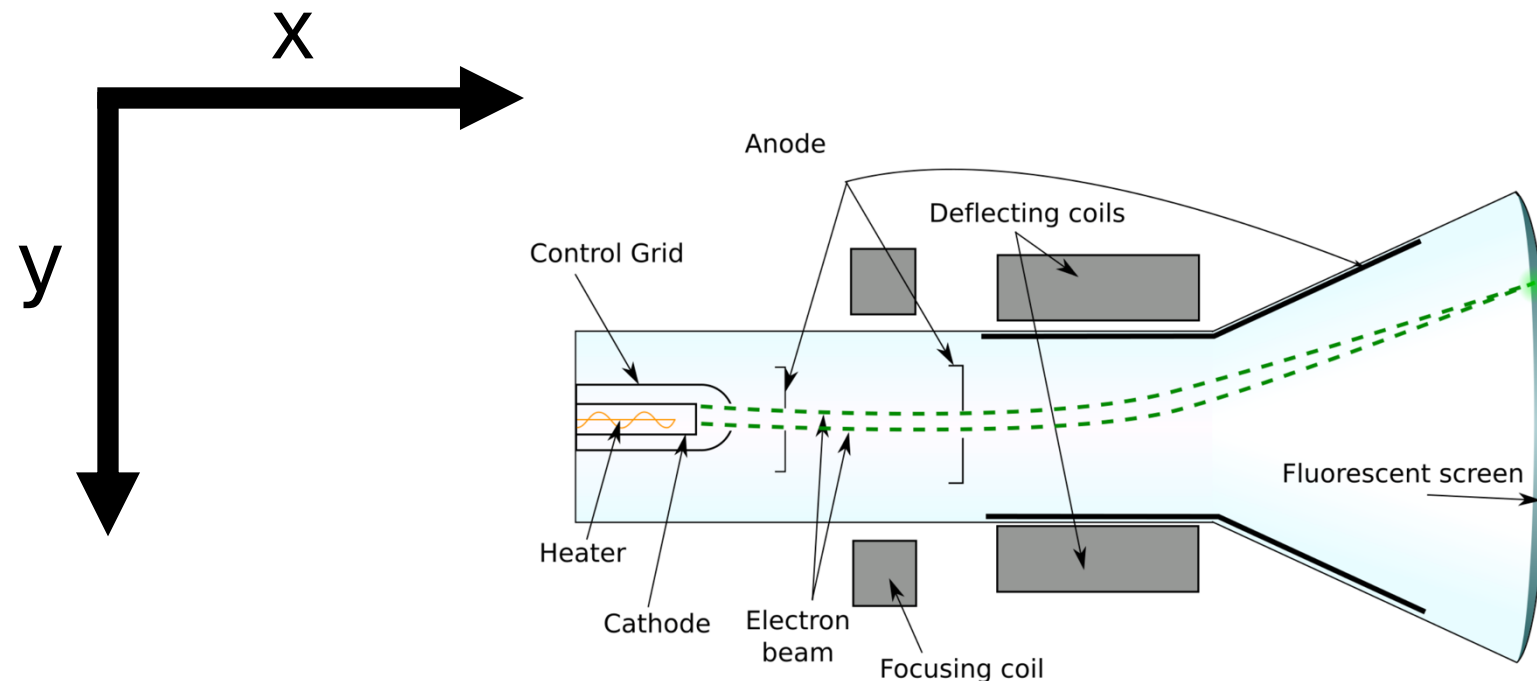
TicTacToeGrid – grid layout

Demo: Model & Controller

TicTacToeGrid – has a field for the model (to be continued...)

TicTacToeCell – addSymbol and reset methods

Computer graphics coordinates



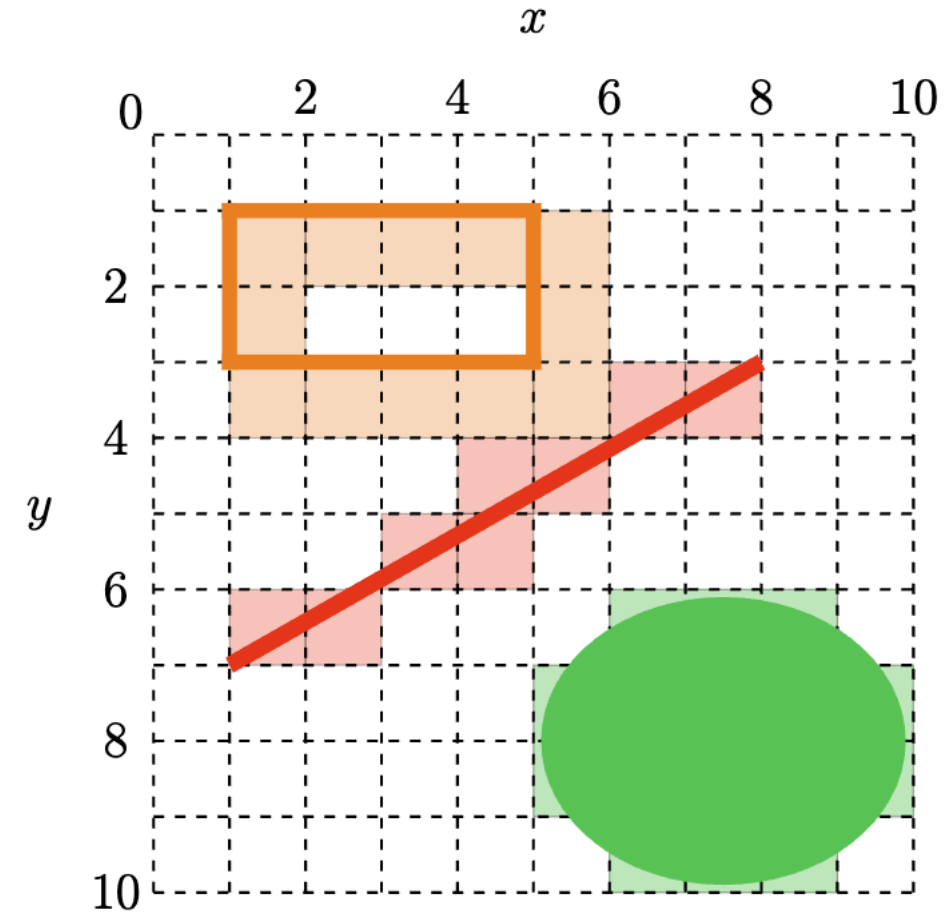
For historical reasons, the coordinate system we use for computer graphics has the origin in the upper left. The x axis points right and y axis points down.

java.awt.Graphics and paintComponent()

@Override

```
public void paintComponent(Graphics g) {  
    super.paintComponent(g);  
  
    g.setColor(Color.ORANGE);  
    g.drawRect(1,1,4,2);  
  
    g.setColor(Color.RED);  
    g.drawLine(1,7,8,3);  
  
    g.setColor(Color.GREEN);  
    g.fillOval(5,6,5,4);  
}
```

[Demo: paint component code in Tic Tac Toe]



Do not call paintComponent() yourself, you can only request that swing call it using setVisible() and repaint()

Computer graphics & color choices

Poll: Do you have color blindness or color vision deficiency?

- A) Yes
- B) No
- C) Not sure or would rather not say

About 10% of the class responded "yes". If your app's colors did not have sufficient contrast, and you did not mitigate the issue in any way, then up to 1 in 10 users might find it hard to use! That's a significant portion of the user base.



PollEv.com/leahp
text **leahp** to 22333

Demo: Accessible colors

Color Blind Simulator

In our app, the shapes (X and O) make it usable to colorblind users even if the color contrast is low. However, to achieve the visual satisfaction of contrasting colors for more users, it is meaningful to adjust our color palette!

Overview of today

Introduction

- Motivation
- Ingredients of Graphical User Interfaces

TicTacToe (Tuesday)

- Review code for game logic
- Visual components

TicTacToe (Thursday)

- Interactive components

Metacognition

- Take 1 minute to write down a brief summary of what you have learned today

Thanks and have a great day!