universität wien

Conceptual Modeling and Programming in GIScience

Lecture 7: GUI I and Action Listeners

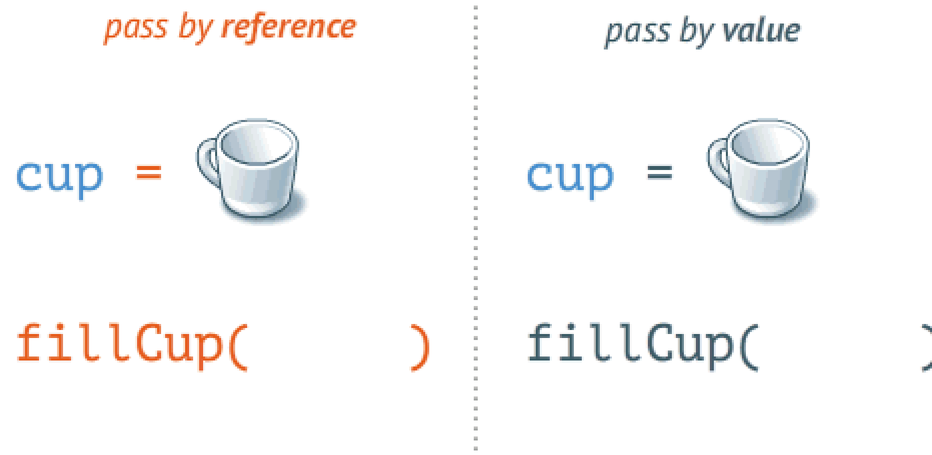
Yingjing Huang

yingjing.huang@univie.ac.at

2026S – June 1, 2026

Week 2 Recap: Pass-by-Value

- Primitive arguments pass **raw independent value copies** to methods
- Object arguments pass **reference copies** pointing to shared memory locations
- Modifications to referenced objects remain visible outside the method



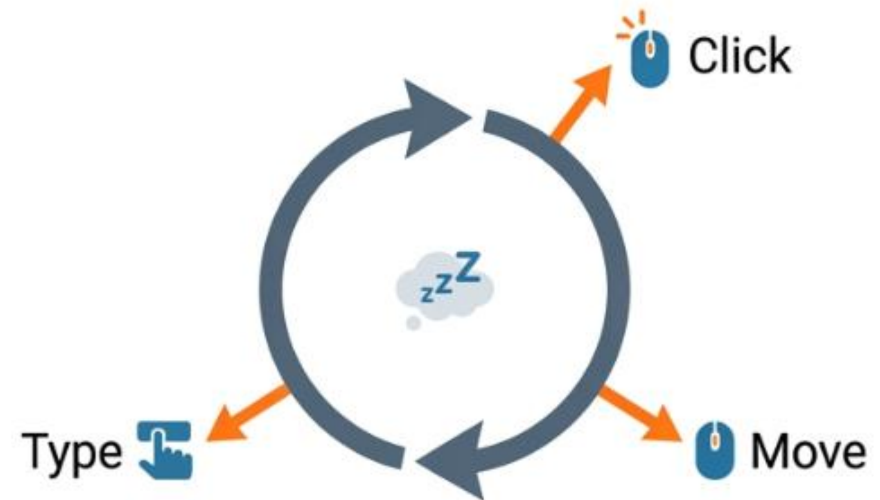
What You Will Learn in This Lecture

- Get a first impression of Graphical User Interfaces (GUI) and the Swing framework
- Understand the shift to Event-Driven Programming and the "Event Loop" mindset
- Learn about Containers, Components, and how to organize them with Layout Managers
- Understand what Listeners are and how to use them to handle user actions
- Learn about the difference between Geographic and Display coordinates
- Experiment with custom drawing and the "repaint" interactive loops

Sequential: Line by Line

- Execution order is fixed by the programmer.

Event-Driven: The Event Loop

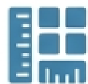


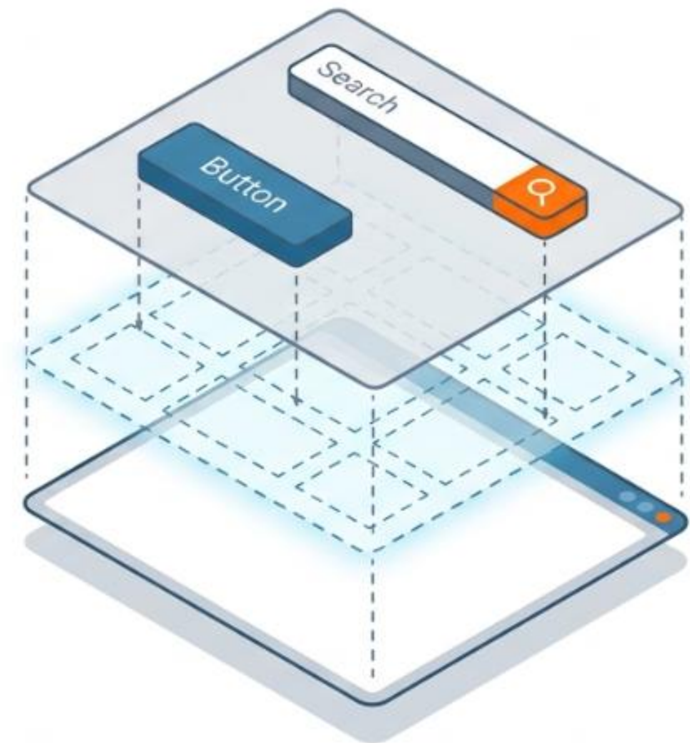
- Execution order is determined entirely by the user.

The Nesting Doll Structure

Containers: The blank canvas or outer box

Components: Interactive elements (Buttons, Menus)

 **Layout Managers**: Invisible rules arranging them.



Java's GUI Frameworks

universität wien

AWT

Abstract Window Toolkit

(1990s)

Swing

Built on AWT

The standard for decades

[无标题]

JavaFX

Modern replacement

Excellent for new application

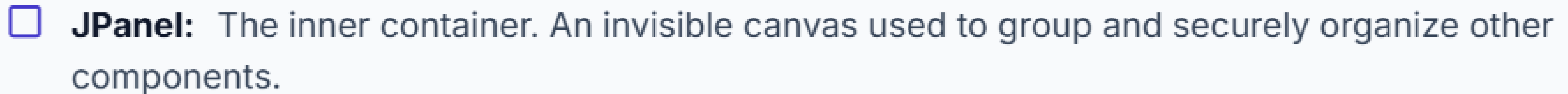
The oldest, minimal framework

Rarely used directly today but sits at the foundation

Ubiquitous, simple, and perfect for learning universal UI concepts

Easy to pick up once you understand Swing's core concepts

Universal Tip: All Swing class names start with the letter ‘J’



universität wien

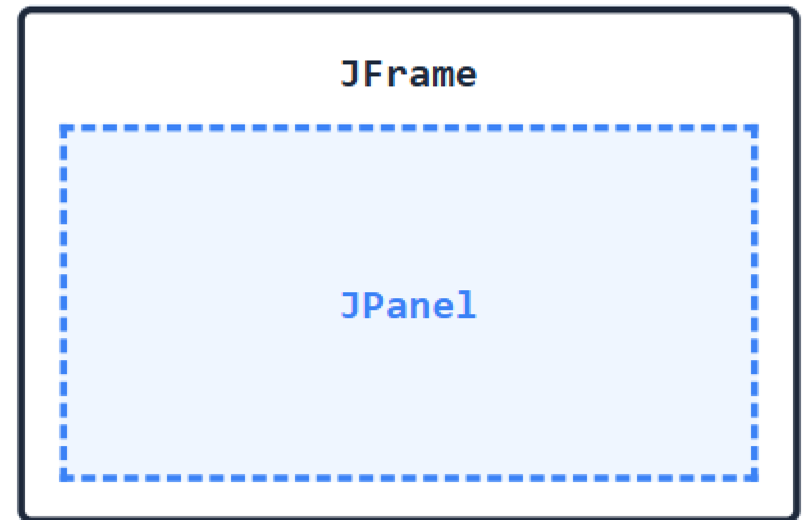
The 4-step Recipe

- By default, a frame is 0x0 pixels and invisible
- You must explicitly instantiate it and define its size
- Manage its memory closure logic to prevent background memory leaks
- Finally, explicitly set its visibility to *true*

```
// 1. Instantiate the window with a title
JFrame frame = new JFrame("My App");

// 2. Set dimensions (width, height in pixels)
frame.setSize(400, 300);

// 3. Terminate program when the window is closed
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

// 4. Make it visible (crucial step!)
frame.setVisible(true);
```

Never place components directly onto the JFrame.

- Create a panel, attach your components to it, and then attach the panel to the frame
- This securely protects your UI layouts from breaking during resizing




```
// 1. Create a blank canvas (container)
JPanel panel = new JPanel();

// 2. Attach the canvas to the main window
frame.add(panel);
```

universität wien

```java
// 1. Create the main window
JFrame frame = new JFrame("My App");
frame.setSize(400, 300);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

// 2. Create a panel (the container)
JPanel panel = new JPanel();

// 3. Create a button and add to panel
JButton btn = new JButton("Click Me!");
panel.add(btn);

// 4. Attach panel to frame and display
frame.add(panel);
frame.setVisible(true);
```

The Core Assembly Workflow

- Create a component object (e.g., the button)
- Pass the desired display text into the constructor
- Use the *add()* method to securely place the component into your designated container (the panel)

```
// 1. Static instruction text
JLabel lbl = new JLabel("Name:");

// 2. Input box (approx 15 chars wide)
JTextField txt = new JTextField(15);

// 3. Add all elements to the panel
panel.add(lbl);
panel.add(txt);
panel.add(btn);
```

Building a Basic Form

- JLabel: Uneditable instruction text
- JTextField: Single-line text input

add() all components to the JPanel

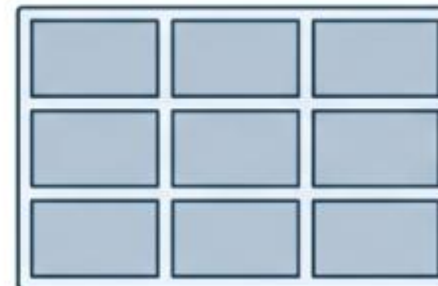
Organizing the Chaos: Layout Managers

Why not exact pixels?

- Avoid absolute positioning
- Hard-coded coordinates break on resize
- Layout Managers auto-adapt to window size

FlowLayout

BorderLayout



GridLayout

Putting it into a Grid

```
// 1. Change layout: 3 rows, 1 column
panel.setLayout(new GridLayout(3, 1));

// 2. Add elements (they will now stack
vertically)
panel.add(lbl);
panel.add(txt);
panel.add(btn);
```

- By invoking ***setLayout***, the manager actively calculates all component dimensions
- If the window resizes, components stretch and adapt perfectly
- Our visual UI is assembled, but the button still lacks a “brain”

The Listener Pattern: Catching the Signal

The 2-Step Workflow

- Implement the Listener
- Attach it to the Component

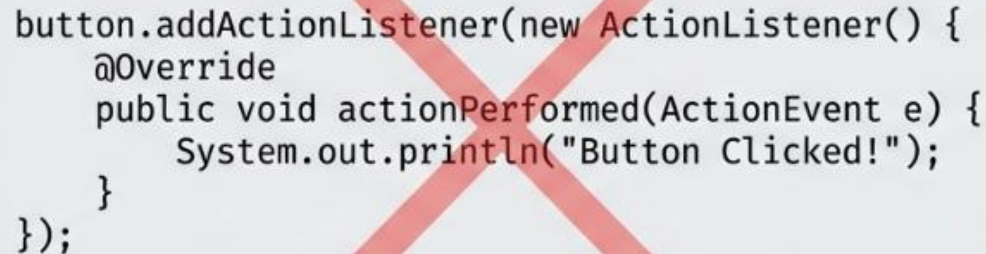
universität wien

The ActionListener Interface

- Clicking a button fires an ActionEvent
- Historically requires an Anonymous Inner Class

```
button.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        System.out.println("Button Clicked!");
    }
});
```

The Overgraded Past



The Modern 1-Line Standard

```
button.addActionListener(e -> System.out.println("Clicked!"));
```

(Event Object) (Arrow Operator) (Action Body)

Overriding

- Extend **Jpanel** to create your own custom canvas
- The **Graphics** object **g** acts as your digital paintbrush

GOLDEN RULE: NEVER call paintComponent() directly!
Java triggers it **automatically** on window open, resize, or refresh.

```
public class MyCanvas extends JPanel {
    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g); // Clears the canvas
        // Your drawing code here...
    }
}
```

Downcasting is Key

- The parameter is plain Graphics, but at runtime, it is actually a Graphics2D object
- Downcasting immediately unlocks anti-aliasing and spatial transforms
- Stateful Colors: Setting a color applies it to all subsequent drawing actions until changed again

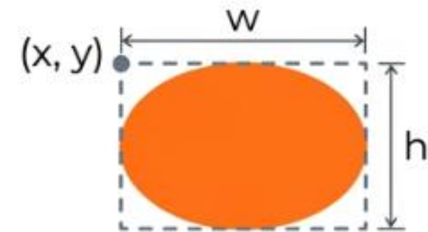
```
@Override
protected void paintComponent(Graphics g) {
    // 1. Clear the background properly
    super.paintComponent(g);

    // 2. Cast to Graphics2D for advanced drawing features
    Graphics2D g2d = (Graphics2D) g;

    // 3. Set the active brush color
    g2d.setColor(Color.BLUE);

    // Everything drawn below will be blue...
}
```

Drawing Primitives

```
g.drawLine(x1, y1, x2, y2);
```

```
g.drawRect(...) & g.fillRect(...)
```

```
g.fillOval(x, y, w, h);
```



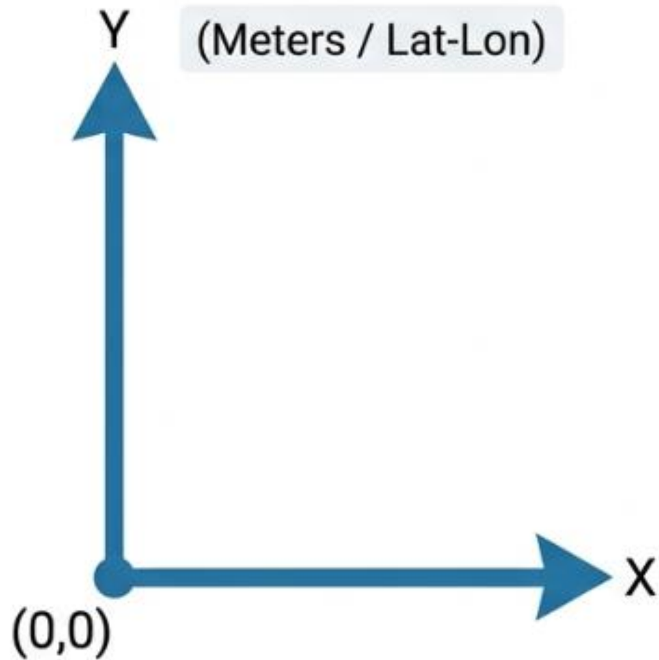

```
g.drawString("Text", x, y);
```

Hello GIS!

(x, y)

Coordinate Systems: The Mismatch

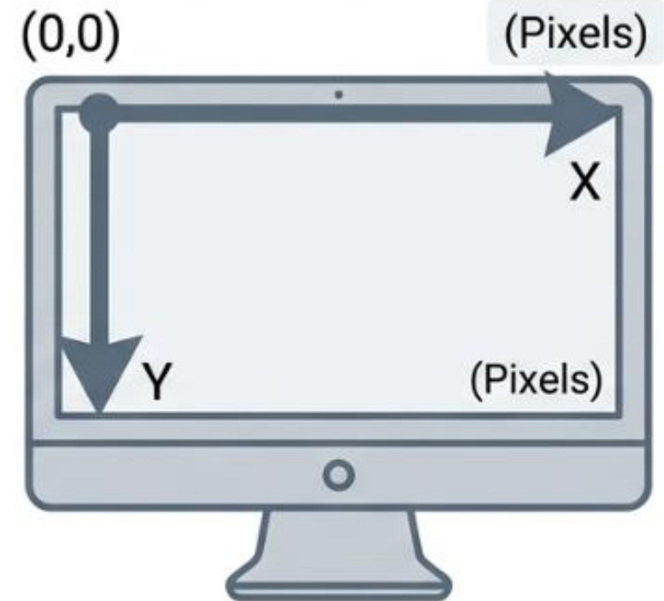
World Data



THE MISMATCH!

- Wrong Position
- Wrong Scale
- Upside Down!

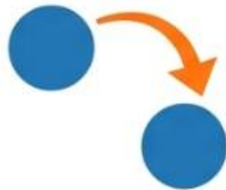
Screen Data



The 3 Steps

Scale
(Fit to screen)

Mirror
(Flip Y-axis)



Translate
(Move origin)

```
// 1. Initialize a new transformation matrix
AffineTransform transform = new AffineTransform();

// 2. Move origin to the bottom and flip Y-axis
// (Screen -> Cartesian)
transform.translate(0, panelHeight);
transform.scale(scaleFactor, -scaleFactor);

// 3. Apply the transformation to our brush
g2d.setTransform(transform);

// 4. Draw using real-world "logical" coordinates!
g2d.drawLine(worldX1, worldY1, worldX2, worldY2);
```

MouseListener

(Discrete Events)

- mousePressed()
- mouseReleased()
- mouseClicked()

MouseMotionListener

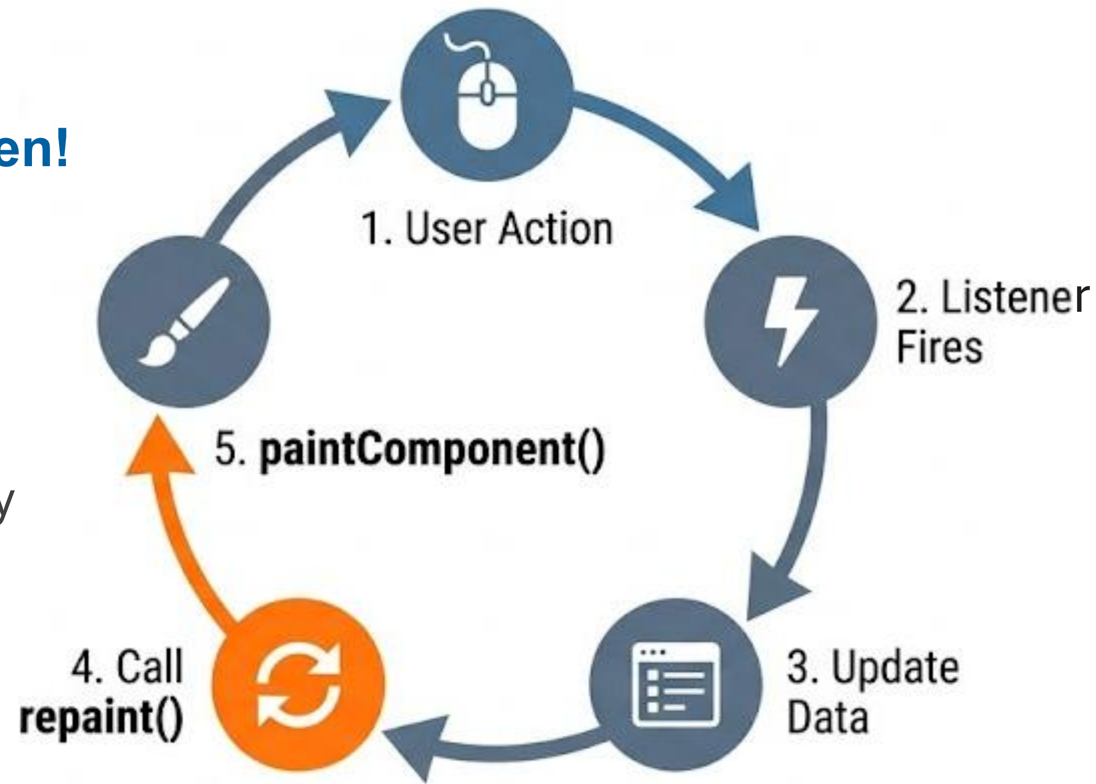
(Continuous Events)

- mouseMoved()
- mouseDragged()

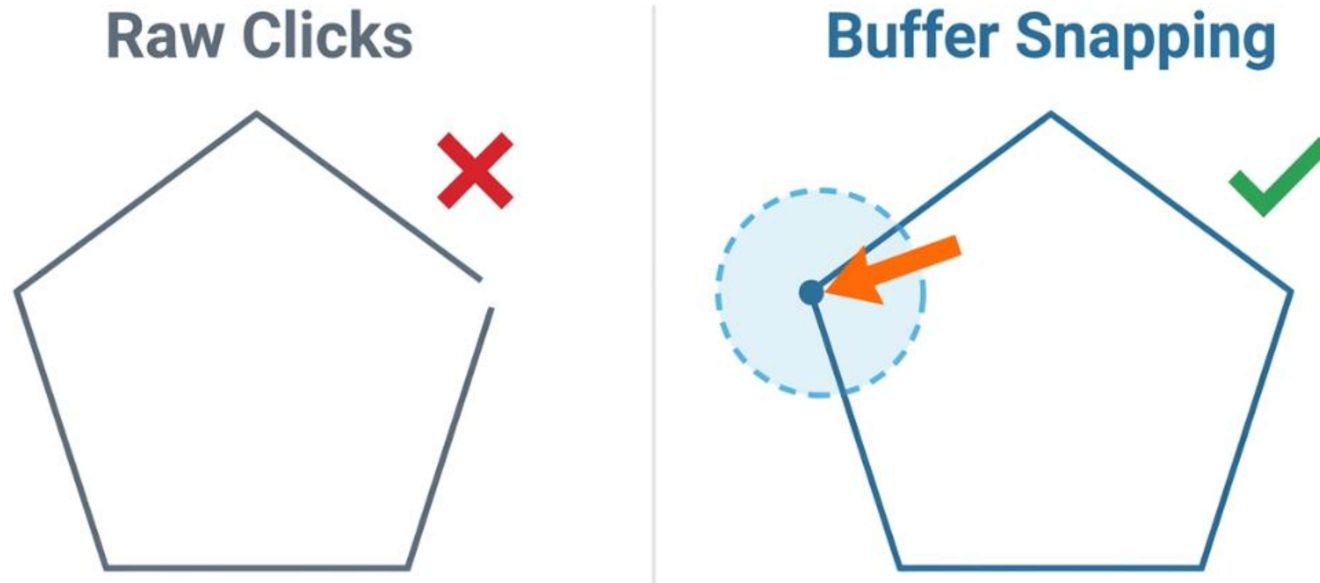
Note:
No Lambdas here!
(Multiple methods require full implementation)

Data Changed? Tell the Screen!

- Screen does NOT update automatically
- You must call **repaint()** explicitly
- Java will then trigger ***paintComponent()***



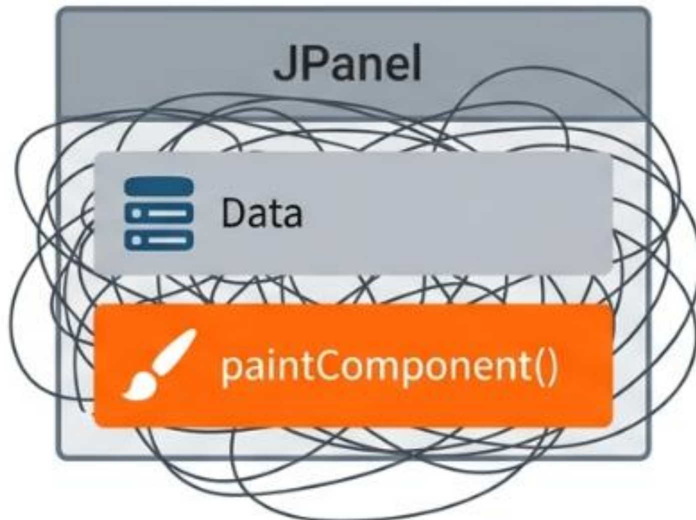
Buffer Snapping: Closing the Gap



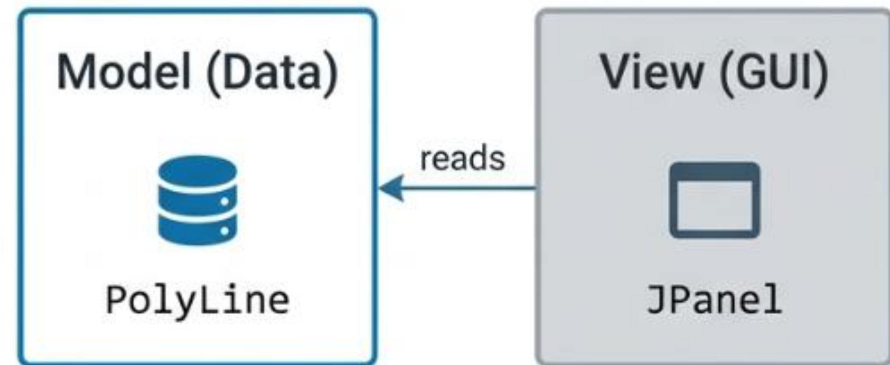
- **Problem**: When manually digitizing a polygon, human hands cannot click the exact original pixel to close it properly. Topologies break
- **Buffer Snapping**: Check if the click falls within a mathematical “snap tolerance” radius of the first point
- **Action**: If yes, ignore raw click coordinates and *snap* mathematically to the exact first point

A Design Preview: Keep Model and View Separate

Tangled



Separated



Note:
No import javax. swing.* in your core domain classes!

The Famous Pitfall: Null and NullPointerException

universität wien

```
// A reference pointing to nothing (null)
Point lastPoint = null;

// 💥 CRASH! Throws NullPointerException at runtime
// double x = lastPoint.getX();
```

The trap (Uninitialized)

```
if (lastPoint != null) {
    // Safe execution: only runs if lastPoint
actually exists
    double x = lastPoint.getX();
}
```

The denfense (Null check)

Note:
Primitives default to 0. Objects default to NOTHING (null).

Code Examples

1. Basic UI
(Layouts & Buttons)

2. Custom Canvas
(paintComponent & Mouse)

3. Geo-snapping
(Transform & Buffer)

Lab Time