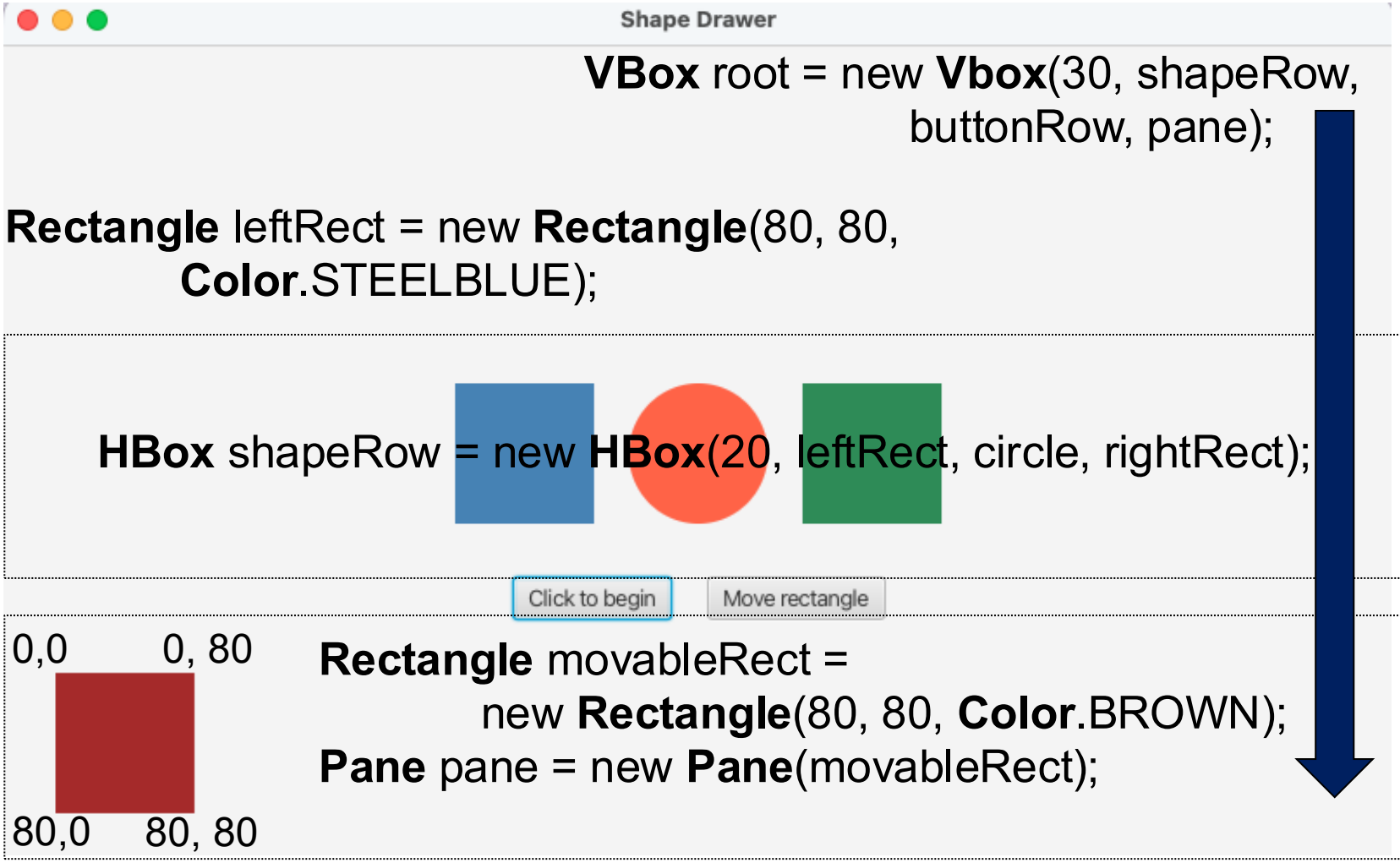
JavaFX Graphics, Event Handling, and Animation

2D shapes

- Two-dimensional shapes (rectangles, circles, etc) can be added to a parent node (such as a VBox)
- Layouts such as HBox, VBox, and GridPane position the shape the same way they position other nodes (such as buttons)
- Layouts such as *Pane* allow you to position the shape relative to its parent
- For types of shapes see tutorialspoint

Example Application



JavaFX Canvas

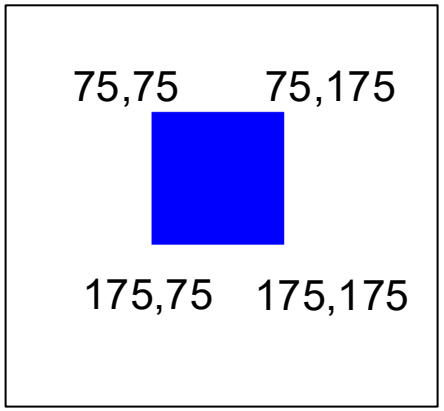
- A **canvas** is a UI component that can be drawn on using a set of commands provided by a GraphicsContext object

```
final Canvas canvas =
      new Canvas(250,250);
GraphicsContext gc = canvas.getGraphicsContext2D();

gc.setFill(Color.BLUE);

//          (x,y,width,height)
gc.fillRect(75,75,100,100);

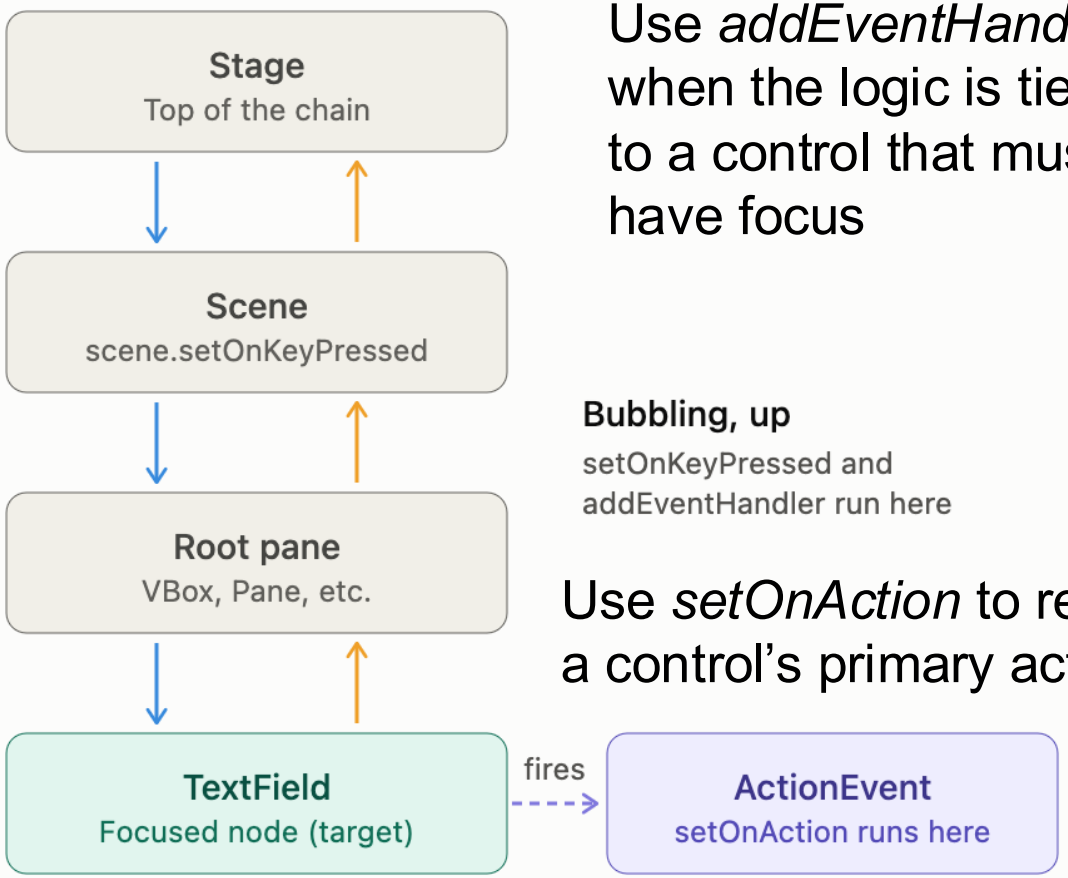
// Group is a a lightweight layout with
// size based on its children
Group group = new Group(canvas);
scene.setRoot(group)
```



Handling events

Use *event.consume()* to end propagation of the event

Use *addEventHandler* when the logic is tied to a control that must have focus



Use *addEventFilter* when you need to see or control an event before the target node does

Use *setOnAction* to react to a control's primary action

A TextField fires when it has focus and [Enter] is pressed

Example

```
// in JavaFX_shapes we use an event filter so that
// this works even if a specific control has the
// focus
sceneDraw.addEventFilter(KeyEvent.KEY_PRESSED, e -> {

        // if space bar is pressed
        if (e.getCode() == KeyCode.SPACE) {
            clearCanvas(); // clear the canvas
            e.consume(); // stop propagation
        }
});
```

EventType

EventHandler

For key events, use **KeyEvent** to specify the event, e.**getCode()** to identify the key pressed/released, and **KeyCode** to refer to specific keys

Other events include a **MouseEvent**, **ActionEvent**, and **ScrollEvent**

Animation Timer (abstract class)

```
AnimationTimer timer = new AnimationTimer() {
    private long lastNanos = -1;

    @Override
    public void handle(long now) {
        if (lastNanos < 0) { // first frame: just record the time
            lastNanos = now;
            return;
        }
        // how many seconds have passed?
        double elapsedSeconds = (now - lastNanos) / 1_000_000_000.0;
        lastNanos = now; // update last timesamp

        // Move 10 px per second, scaled by the time since the last frame
        rect.setX(rect.getX() + 10 * elapsedSeconds;);
        }
};
```

An AnimationTimer is used for smooth rendering of objects

Example to move a rectangle right by 10 pixels each second

The *handle* method is called every frame the timer is active (targeting 60 frames / second), with *long* the timestamp in nanoseconds