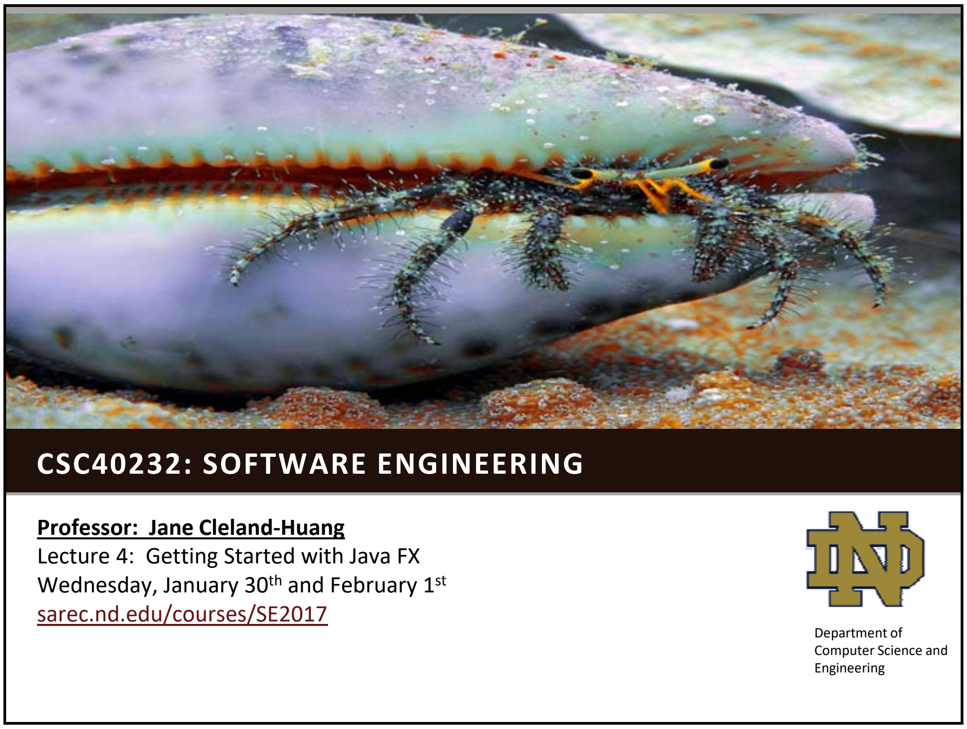


CSC40232: SOFTWARE ENGINEERING

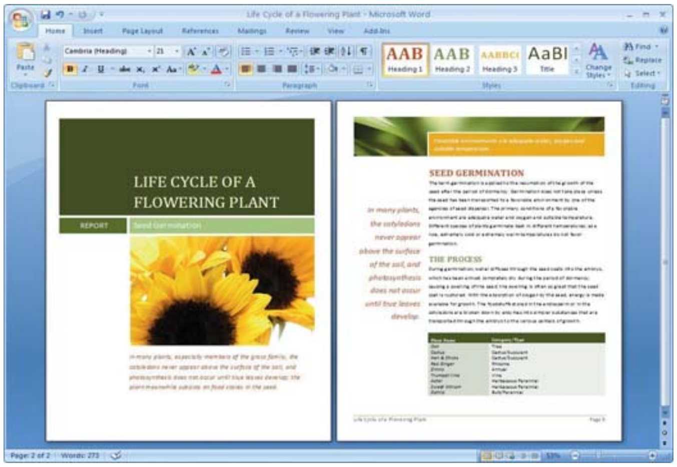
Professor: Jane Cleland-Huang
Lecture 4: Getting Started with Java FX
Wednesday, January 30th and February 1st
sarec.nd.edu/courses/SE2017

Department of Computer Science and Engineering

What are we covering today?

1. A new design pattern (Composite)
2. Mouse Event handler

Lots of things are Hierarchical



LIFE CYCLE OF A FLOWERING PLANT

REPORT | Seed Germination

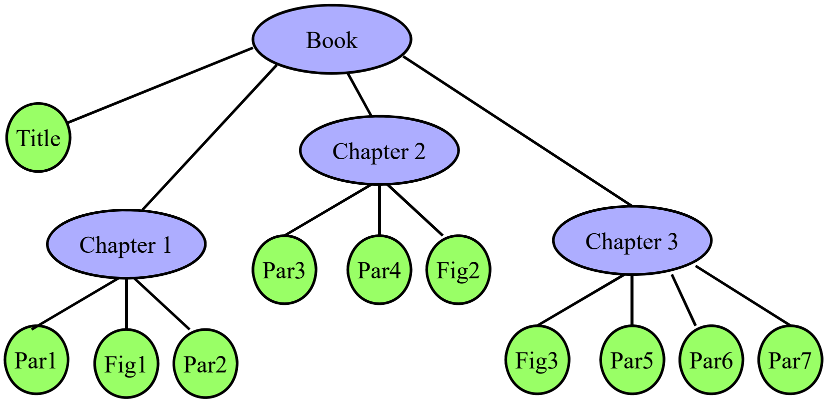
SEED GERMINATION

In many plants, the cotyledons never appear above the surface of the soil, and photosynthesis does not occur until true leaves develop.

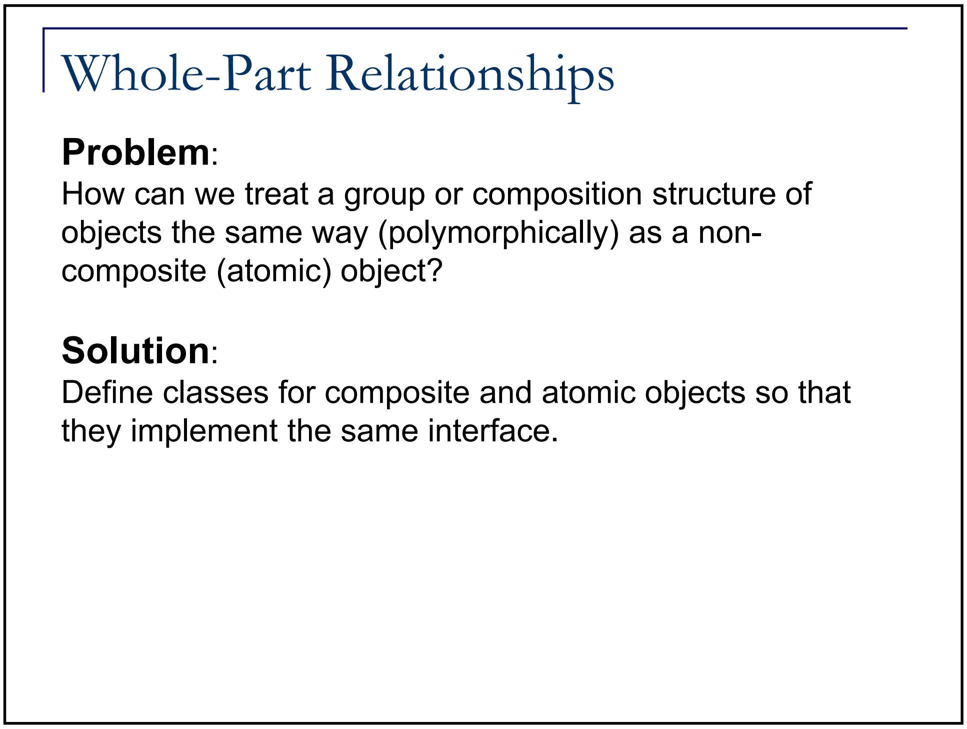
THE PROCESS

How could you manage the hierarchy in a document so that you could perform functions (such as resize) groups of objects together?

An Entire Book



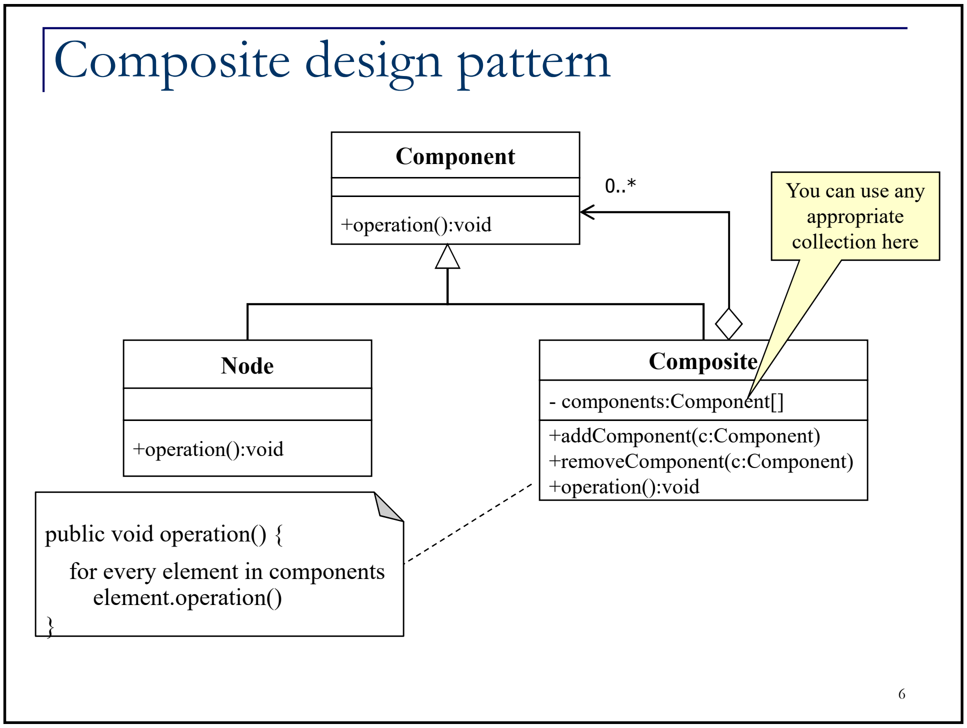
In the real world and in computer systems we often see examples of compositional hierarchies.



Whole-Part Relationships

Problem:
How can we treat a group or composition structure of objects the same way (polymorphically) as a non-composite (atomic) object?

Solution:
Define classes for composite and atomic objects so that they implement the same interface.



Composite design pattern

```
public void operation() {
   for every element in components
      element.operation()
}
```

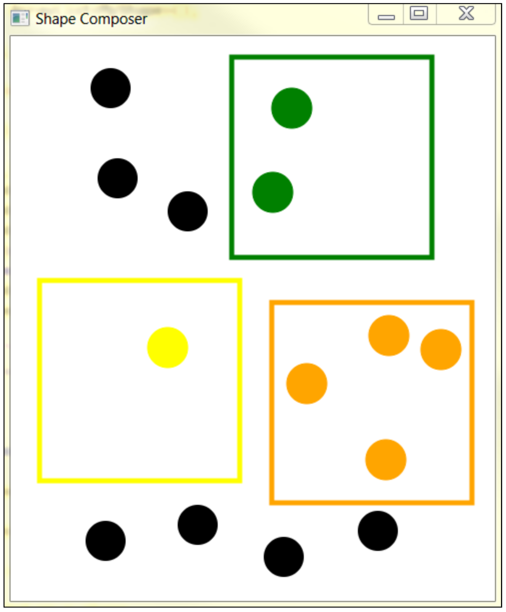
6

A Code Example

```
// Leaf node
class Block implements Group {
    String blockName;
    public Block(String blockName){
        this.blockName = blockName;
    }

    public void assemble(int level) {□
}

// Composite node
class Structure implements Group {
    // Collection of child groups.
    private List<Group> childGroup = new ArrayList<Group>();
    String blockName;
    public Structure(String blockName){
        this.blockName = blockName;
    }

    public void add(Group group) {□
    public void remove(Group group) {□
    public void assemble(int level) {□
}

public class ImplementComposite {
     public static void main(String[] args) {□
}
```

```
interface Group {
    public void assemble(int level);
}
```

Another Example



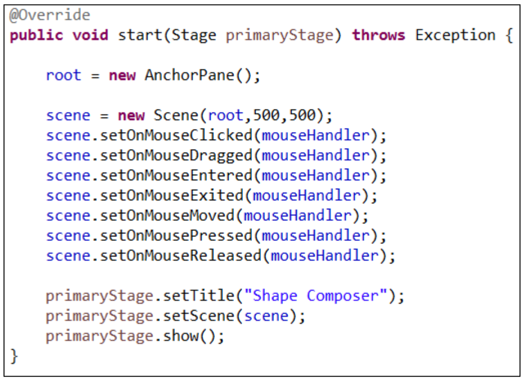
Uses the mouse to add black circles and colored rectangles onto the canvas.

Uses the mouse to drag a black circle into a rectangular container.

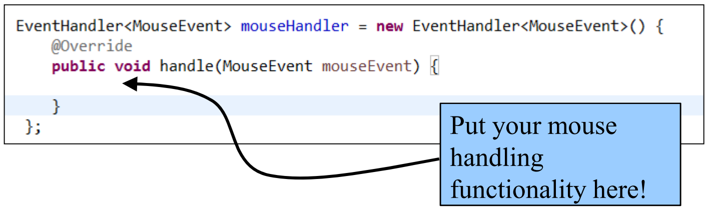
Once the circle is placed into the container it takes on the container’s color.

If we drag a container, all circles within it are also dragged.

Creating a Mouse Event Handler (JavaFX)



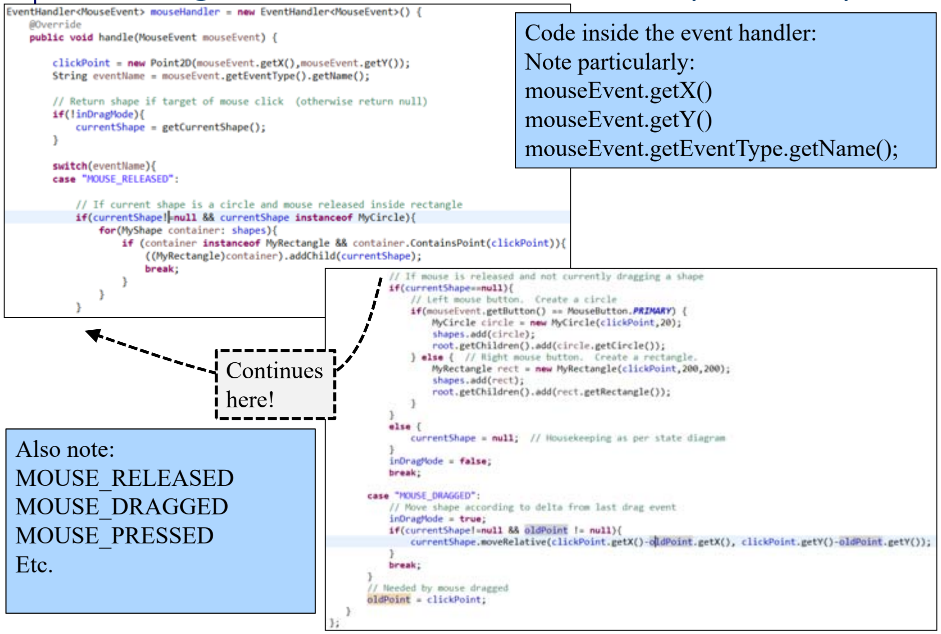
```
@Override
public void start(Stage primaryStage) throws Exception {

    root = new AnchorPane();

    scene = new Scene(root,500,500);
    scene.setOnMouseClicked(mouseHandler);
    scene.setOnMouseDragged(mouseHandler);
    scene.setOnMouseEntered(mouseHandler);
    scene.setOnMouseExited(mouseHandler);
    scene.setOnMouseMoved(mouseHandler);
    scene.setOnMousePressed(mouseHandler);
    scene.setOnMouseReleased(mouseHandler);

    primaryStage.setTitle("Shape Composer");
    primaryStage.setScene(scene);
    primaryStage.show();
}
```



```
EventHandler<MouseEvent> mouseHandler = new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent mouseEvent) {

    }
};
```

Put your mouse handling functionality here!

Creating a Mouse Event Handler (JavaFX)



```
EventHandler<MouseEvent> mouseHandler = new EventHandler<MouseEvent>() {
    @Override
    public void handle(MouseEvent mouseEvent) {

        clickPoint = new Point2D(mouseEvent.getX(),mouseEvent.getY());
        String eventName = mouseEvent.getEventType().getName();

        // Return shape if target of mouse click  (otherwise return null)
        if(!inDragMode){
            currentShape = getCurrentShape();
        }

        switch(eventName){
        case "MOUSE_RELEASED":

            // If current shape is a circle and mouse released inside rectangle
            if(currentShape!=null && currentShape instanceof MyCircle){
                for(MyShape container: shapes){
                    if (container instanceof MyRectangle && container.ContainsPoint(clickPoint)){
                        ((MyRectangle)container).addChild(currentShape);
                        break;
                    }
                }
            }
```

Continues here!

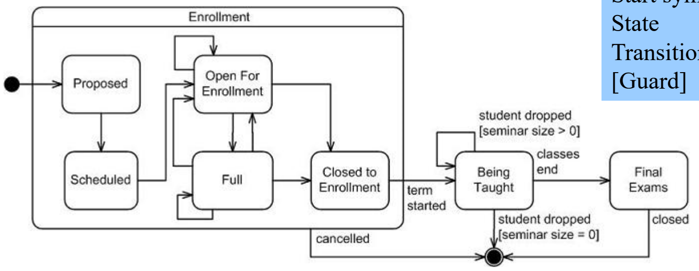
```
            // If mouse is released and not currently dragging a shape
            if(currentShape==null){
                // Left mouse button.  Create a circle
                if(mouseEvent.getButton() == MouseButton.PRIMARY) {
                    MyCircle circle = new MyCircle(clickPoint,20);
                    shapes.add(circle);
                    root.getChildren().add(circle.getCircle());
                } else {  // Right mouse button.  Create a rectangle.
                    MyRectangle rect = new MyRectangle(clickPoint,200,200);
                    shapes.add(rect);
                    root.getChildren().add(rect.getRectangle());
                }
            }
            else {
                currentShape = null;  // Housekeeping as per state diagram
            }
            inDragMode = false;
            break;

        case "MOUSE_DRAGGED":
            // Move shape according to delta from last drag event
            inDragMode = true;
            if(currentShape!=null && oldPoint != null){
                currentShape.moveRelative(clickPoint.getX()-oldPoint.getX(), clickPoint.getY()-oldPoint.getY());
            }
            break;
        }
        // Needed by mouse dragged
        oldPoint = clickPoint;
    }
};
```

Code inside the event handler:
Note particularly:
mouseEvent.getX()
mouseEvent.getY()
mouseEvent.getEventType.getName();

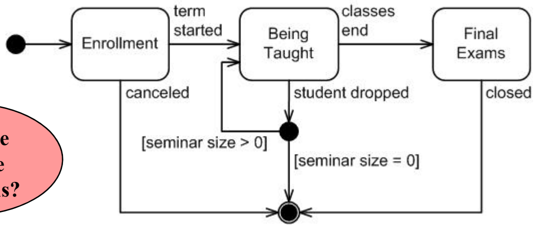
Also note:
MOUSE_RELEASED
MOUSE_DRAGGED
MOUSE_PRESSED
Etc.

Think State Transitions!!

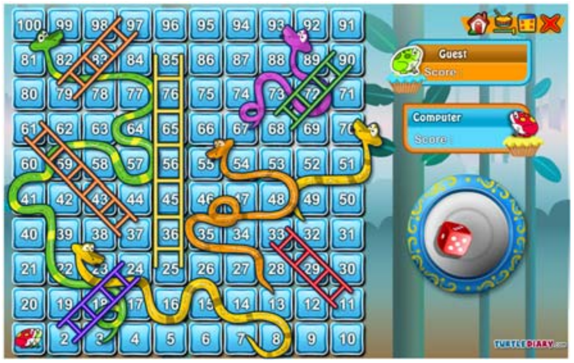
Start symbol
State
Transitions
[Guard]



What are my state transitions?



Design Time



http://www.turtlediary.com/game/snakes-and-ladders.html

In groups of 3-4 think about the classes that you might need to implement a snakes and ladders game. (Don’t worry about the GUI components for this exercise).

Use the CRC cards to identify classes and responsibilities.
Sketch out the finished UML design on paper and we’ll do some show and tell..