Cartoon Help Session
Topics

- Shape package
- Design
- Layout Panes
- Graphical and Logical Containment
- EventHandler
- Timeline
- Step by Step
- Helpful Documentation
To use the shape package type `import javafx.scene.shape.*;`
  ○ Import specific shapes: `import javafx.scene.shape.Rectangle;`

Ex. instantiating a new rectangle:
`Rectangle rectangle = new Rectangle(100, 200, Color.BLACK);`

Look at [Graphics II lecture](#) for reference on how to set properties of shapes

Also, check out the [JavaFX Shapes Documentation](#) on our website
Design (Part 1)

- What do you want your Cartoon to do?
- Draw a picture of what it will look like
- Think about the classes you’ll need
  - Which can just be JavaFX components?
  - Which will you need to create?
- Consider what warrants `public` vs. `private` methods and classes
  - Remember - split up your constructor into helper methods. Should these be `public` or `private`?
Design (Part 2)

- How will you contain your classes graphically?
  - Remember the scene graph

- Determine what each of your classes should do

- Determine which classes need to know about other classes
  - This should impact your *logical* containment. More later...
BorderPane borderPane = new BorderPane;

- arranges components in five regions
  - Top, Left, Center, Right, Bottom
  - unused regions are collapsed - don’t show up!
Layout Panes (Part 2)

- HBox hBox = new HBox();
  - horizontally aligns nodes

- VBox vBox = new VBox();
  - vertically aligns nodes

combined HBox and VBox
Layout Panes (Part 3)

- GridPane gridPane = new GridPane();
  - create a flexible grid of rows and columns in which to lay out nodes
You Can Contain Multiple Panes

Note: Unused regions of the BorderPane are not shown; they are automatically hidden!
Graphical and Logical Containment

- What’s the graphical containment of this program? How does this differ from logical containment? Consider scene graph!

- Reminder: Logical containment is like diagrams you’ve been making all along

- ex. what classes instantiate other classes (containment), and what classes know about other classes (association).
Logical Containment Diagram
EventHandler

- How do we make a Button print out something? (from DQ Question 5)
  - Create a new Button
  - Create one class that implements javafx.event.EventHandler<ActionEvent>
    - What method must this class define? What would the body be?
  - What should this class have in its parameters?
We don’t need the full generality, and can do simple animation using a single KeyFrame that is repeated a fixed number of times or an indefinite number of times; each time the EventHandler is called, it makes incremental changes to the time-varying variables (e.g., (x, y) position of a character).
Timeline (part 2)

1. Instantiate an instance of `KeyFrame`
   ```java
   KeyFrame kf = new KeyFrame(Duration.millis(1),
                               new ShapeHandler());
   ```

2. Write a “ShapeHandler”, a class that implements `EventHandler<ActionEvent>`
   a. Note that you don’t *have* to name it `ShapeHandler`
   b. what method must this class have?
Timeline (part 3)

3. Then we instantiate a `Timeline` and pass in our `KeyFrame` and set the CycleCount to `INDEFINITE`.

```java
Timeline timeline = new Timeline(kf);
timeline.setCycleCount(Animation.INDEFINITE);
```
Timeline (part 4)

4. Start the Timeline!

timeline.play();
For More on Timelines

- See the [Graphics I lecture](#)
- Check out the [Timeline](#) section of the [JavaFX Guide](#)
- See the [Javadoc](#) for [Timeline](#)
Step by Step (Part 1)

● Fill in your App to set the title of the Stage

● Instantiate an instance of PaneOrganizer

● Create a Scene with a reference to the root Pane, which is created in PaneOrganizer

● Don’t forget to set the Stage’s Scene!

● Show the Stage!
Step by Step (Part 2)

- **PaneOrganizer** should have **BorderPane** as the root
- Add other components:
  - Other **Panes**, Your Composite Shape, other **Shapes** (optional), **Labels**, and **Button**
- Add **EventHandler** and **Timeline**
- Do a little dance cuz **u earned it!!**
Helpful Documentation

- We have prepared a lot of useful documentation for you!
- [JavaFX Shape Docs](#)
- [JavaFX Guide](#)
- Javadocs also contain lots of helpful information
  - You will learn more about these in lab next week!
- Whenever you want to find Javadocs of any class, we recommend to Google "<class> Javafx Javadocs"
Good luck!

- Happy painting! Be creative, and as always…

Start early
Start today
Start yesterday!