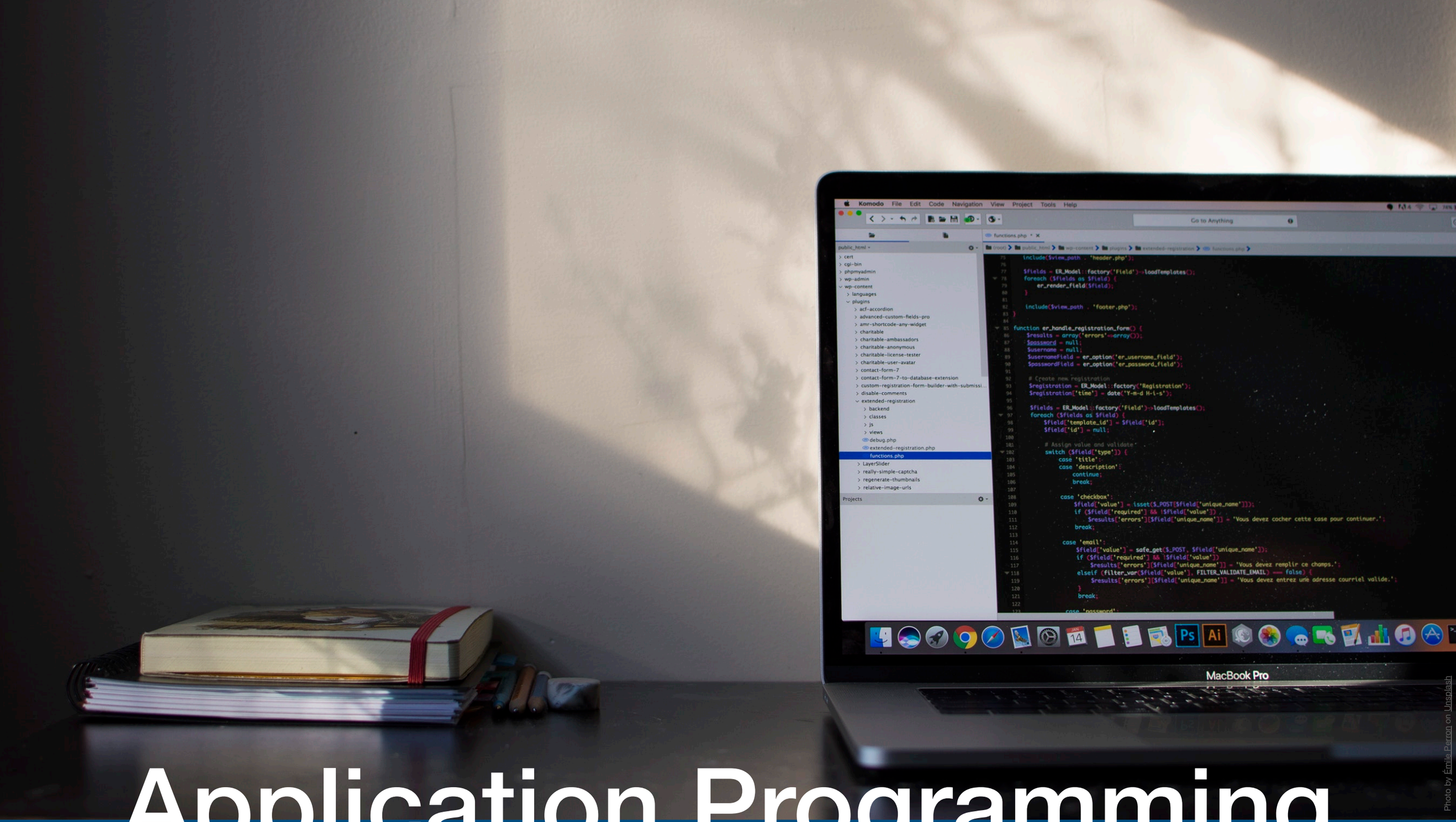


Application Programming

UTSA Computer Science 3443

Fall 2021

Dr. Amanda Fernandez

Week 9

- JavaFX applications
- Graphics
- Due Saturday: **Lab 4** and **Quiz 4**

Fall 2022*

Wee	Dates	Topic	Lab	Quiz	Chapter(s
1	8/23 - 8/27	Introductions, Syllabus, Java basics, Eclipse			1-6
2	8/30 - 9/3	Java & OOP Concepts, Javadoc	1	1	8,14
3	9/6 - 9/10	ArrayList, Strings			7,15
4	9/13 - 9/17	File I/O, UML, MVC	2	2	12,13
5	9/20 - 9/24	Introduction to JavaFX			
6	9/27 - 10/1	JavaFX, SOLID	3	3	
7	10/4 - 10/8	Introduction to Git, *Review*			
8	10/11 - 10/15	**Midterm Exam**			
9	10/18 - 10/22	JavaFX applications	4	4	13,20
10	10/25 - 10/29	Exception Handling			11
11	11/1 - 11/5	Collections & Generics	5	5	16, 19
12	11/8 - 11/12	JUnit Testing, logging, Application Design			
13	11/15 - 11/19	Concurrency & Multithreading + ***Review***	6	6	17
14	11/22 - 11/24*	Lambda Expressions (+*Thanksgiving*)			21
15	11/29 - 12/3	**Team project demos**			
-	12/6 - 12/10	**Final Exams**			

Resume

Check resume templates on overleaf.com

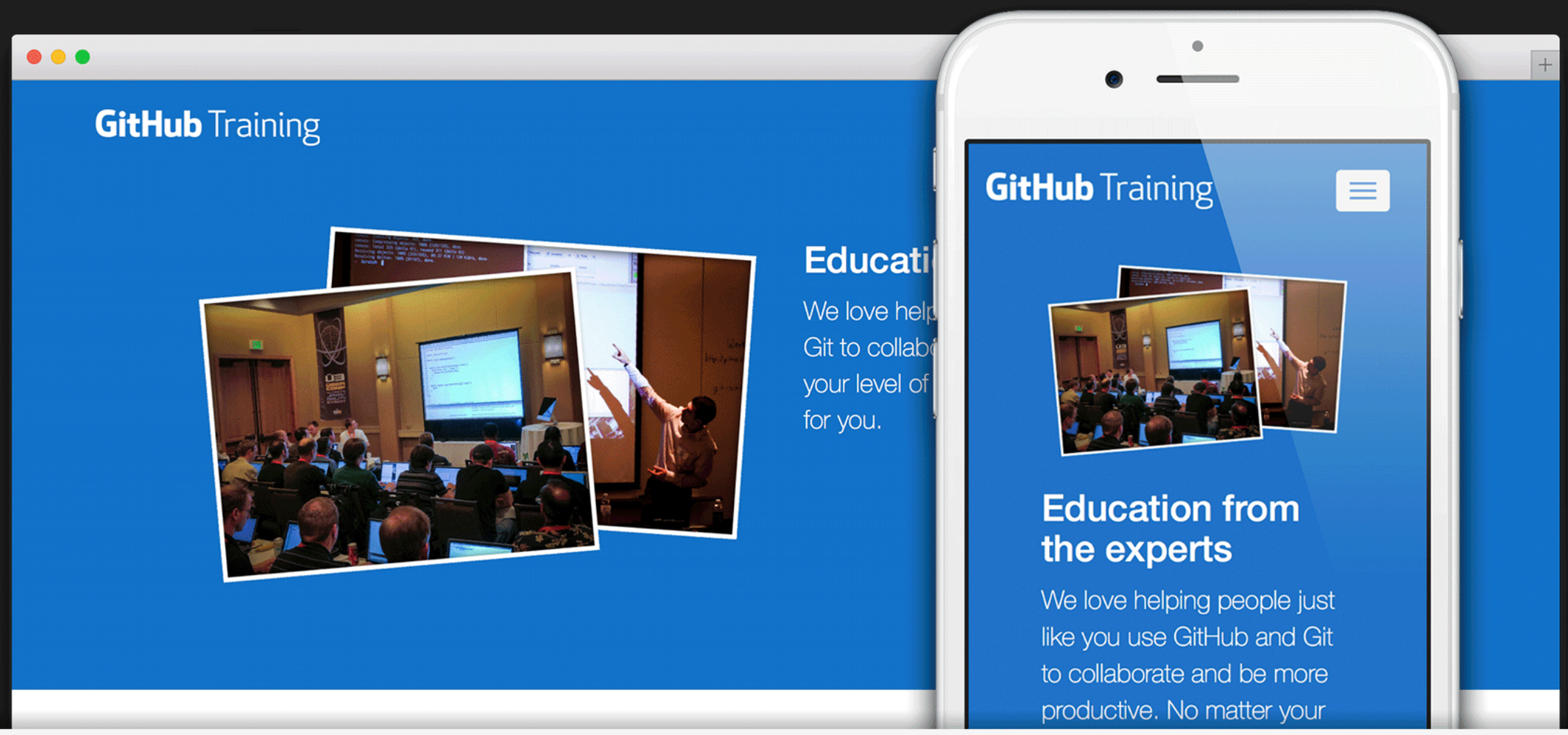
GitHub

Pages

GitHub Pages

Websites for you and your projects.

Hosted directly from your GitHub repository. Just edit, push, and your changes are live.



GITHUB

GitHub provides a way for users to create their own websites…

- Free hosting!
- Project websites:
  - You can create a website for each project you have.
- Personal website:
  - You can have one website for your GitHub account.
  - **http://*yourusername*.github.io**

GitHub pages

GitHub refers to these as **Pages**

- Check them out here: https://pages.github.com

Since you have a GitHub account, let's create a personal website!

1. Create the repository in GitHub. (Hello World!).
    - Follow the instructions in the link above - the repo needs to be named specifically to identify it as your website.
2. Add HTML code to add content to your website.
    - You can do this in the browser to start off! Just create a "hello world" HTML file and add it to your repo..
3. Add, commit, & push.
4. Visit http://username.github.io

Now spruce it up..

Once you have “Hello World” website working, it’s time to make it look nicer.

- Add HTML files for page content, and CSS for style.

Here are a few resources for free templates:

- HTML5 UP https://html5up.net
- Templated https://templated.co
- Pixelarity https://pixelarity.com
- Zerotheme http://www.zerotheme.com

Need some inspiration?

- http://tylercaro.github.io
- http://electron.atom.io
- http://www.sarahlichang.com
- http://www.callieschweitzer.com
- http://www.nathanielkoloc.com
- http://jimramsden.com/#
- http://deda.me

Need a blog?

Jekyll provides a convenient framework for blogging:

https://help.github.com/articles/using-jekyll-as-a-static-site-generator-with-github-pages/

JavaFX

Initializable

Initializable

- Need to load data to a view *before* the user interacts with it?
- The **Initializable** interface

```
public class MainController implements EventHandler<ActionEvent>, Initializable{


    @Override
    public void initialize(URL location, ResourceBundle resources) {

        // add code here to execute upon load

    }


    // Reminder: this class must include a handle method!
}
```

Activity

- Test out the **Initializable** interface..

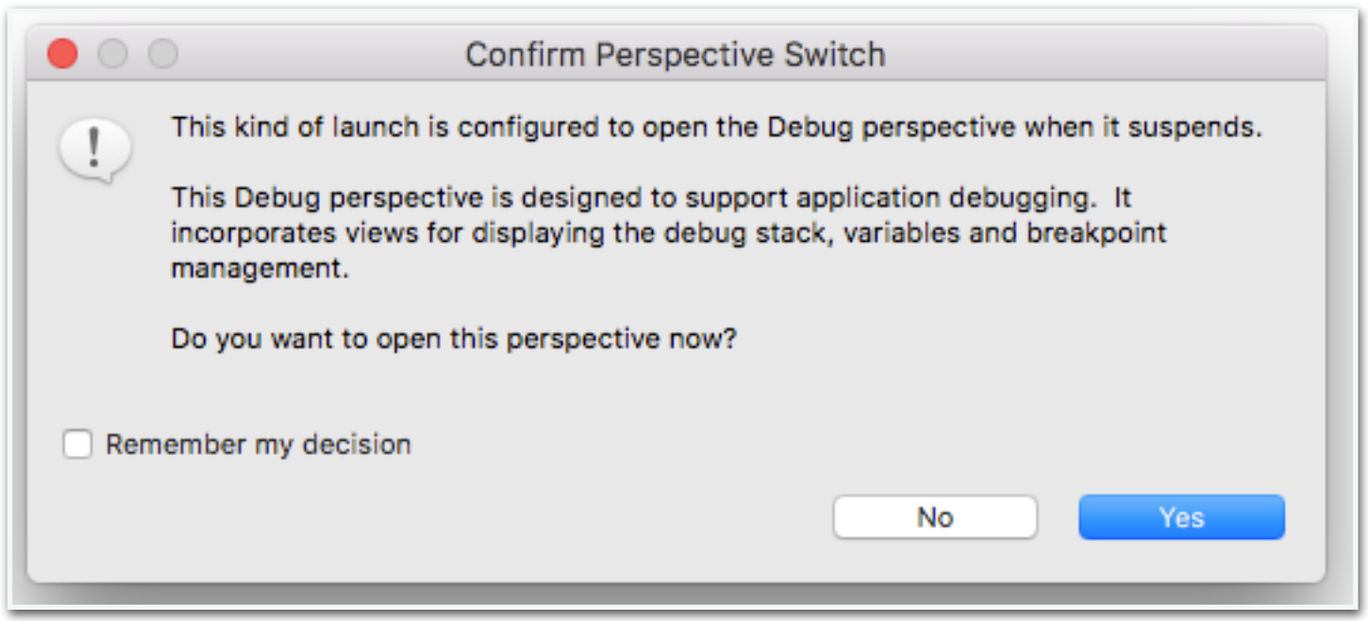
    1. Implement the interface
    2. Add the **initialize** method required
    3. Add a GUI component to load a message to the user before the view shows.

Eclipse help:

Debugging

Debugging in Eclipse

- To halt execution at a certain line of code, double-click on the line just to the left of the line number.
    - This adds a small blue dot.
- To run in **debug mode**, click the bug icon or "*Debug As Java Application*"



Debugging in Eclipse

- Eclipse will execute your program until the marked line(s):

```
16              cities[i] = new City(cityTest.cityNames[i], cityTest.stateN
17          }
18
19          Arrays.sort(cities);
20
21          Iterator<City> iterator = new ArrayReverse<City>(cities);
22          while (iterator.hasNext()) {
23              City city = iterator.next();
24              System.out.print(city + "; ");
25          }
26          System.out.println();
```

- This will **suspend** until you either:
  - ***Resume*** *- continues execution until next breakpoint*
  - ***Terminate*** *- halts execution completely*

Graphics

Some Resources

- https://www.materialpalette.com/deep-purple/amber
- *https://material.io/components/text-fields*

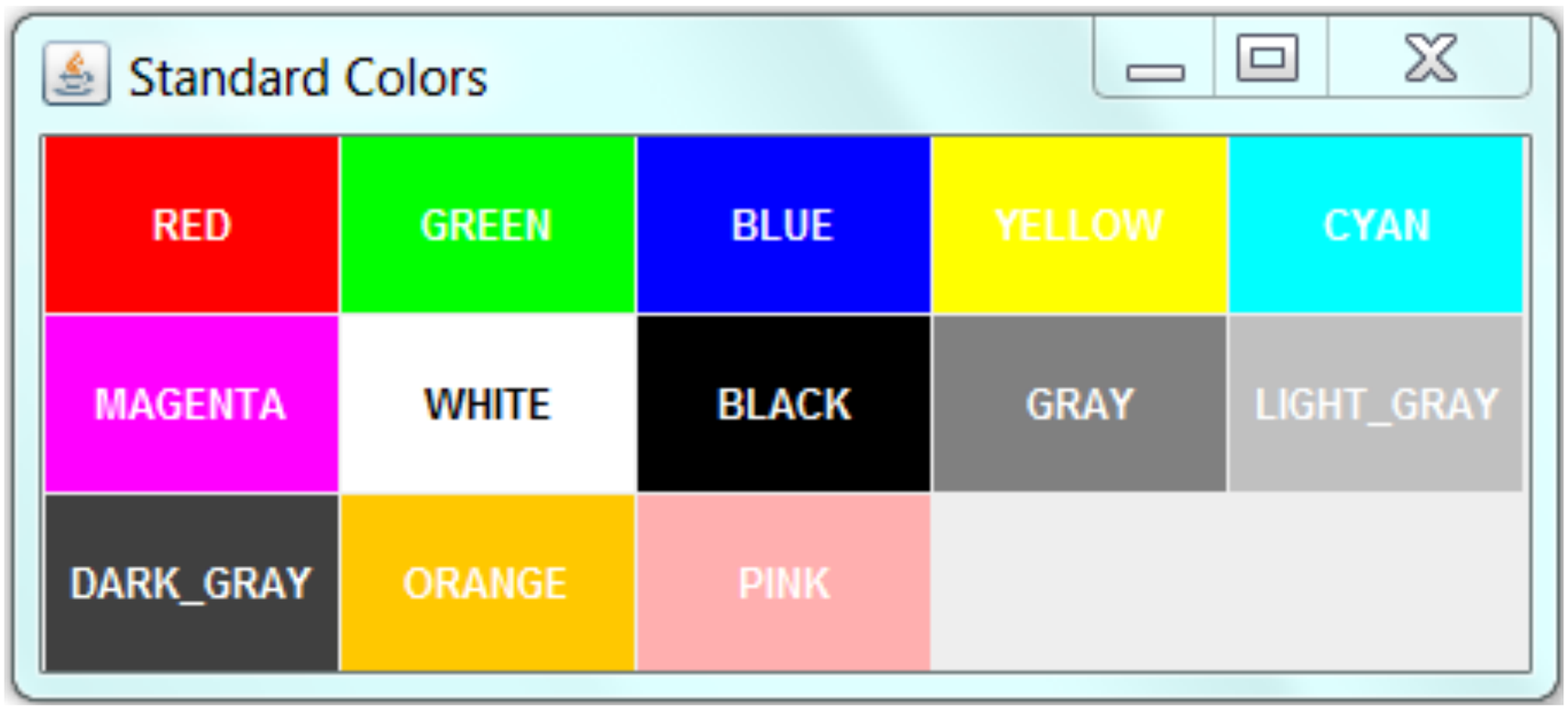
Graphics + JavaFx

- The **JavaFX** library provides scenes, layouts, and basic components for creating the GUI of your applications.
  - *ie: buttons, labels, checkboxes, text fields, etc.*
- There are also **graphics** packages
  - *for colors, shapes, fonts, and lots more…*

This is also provided by java.awt - be careful with your imports!

Colors

- **java.awt.Color** is a class with *static* declarations to handle colors, and useful methods to retrieve information about a color, for example:
  - `brighter(), darker()`
  - `getRed(), getGreen(), getBlue()`
  - `RGBtoHSB(…), HSBtoRGB(…)`



- `Color.RED`
- `Color.GREEN`
- `Color.BLUE`
- `Color.YELLOW`
- `Color.CYAN`
- …

Colors

- **javafx.scene.paint.Color** is a class with *static* declarations to handle colors, ***including more than*** *java.awt's Color class.*

- Additional useful methods:
  - `brighter(), darker(), grayscale(), saturate()`
  - `getRed(), getGreen(), getBlue()`
  - `getSaturation(), getOpacity(), getHue()`
  - `valueOf(..)`
  - `Color.color(double r, double g, double b)`

ColorPicker

Use **javafx.scene.control.ColorPicker** to prompt the user to select a color.

- *The color chosen is not in hexadecimal format, and will not apply to CSS styles. Here is a handy method for converting between them…..*

```
public String convertToHex( Color c ) {
      int red = (int) (255 * c.getRed());
      int green = (int) (255 * c.getGreen());
      int blue = (int) (255 * c.getBlue());
      String hex = String.format("#%02x%02x%02x", red, green, blue);
      System.out.println(hex);
      return hex;
   }
```

Shapes

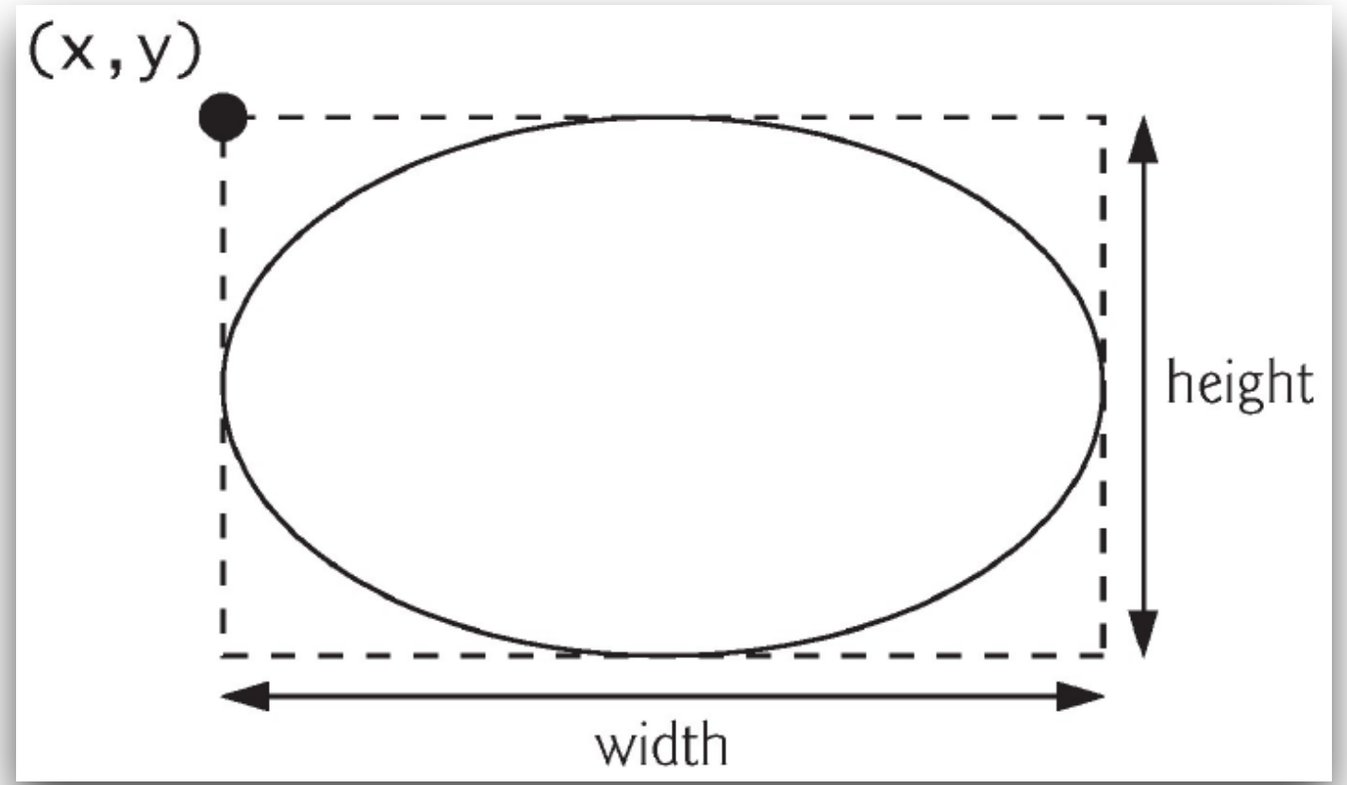
javafx.scene.shape provides classes to create shapes for display.

```
// draw a rectangle at (20,10)
//     width = 50, height = 25
```

```
Rectangle r = new Rectangle( 20, 10, 50, 25 );
```

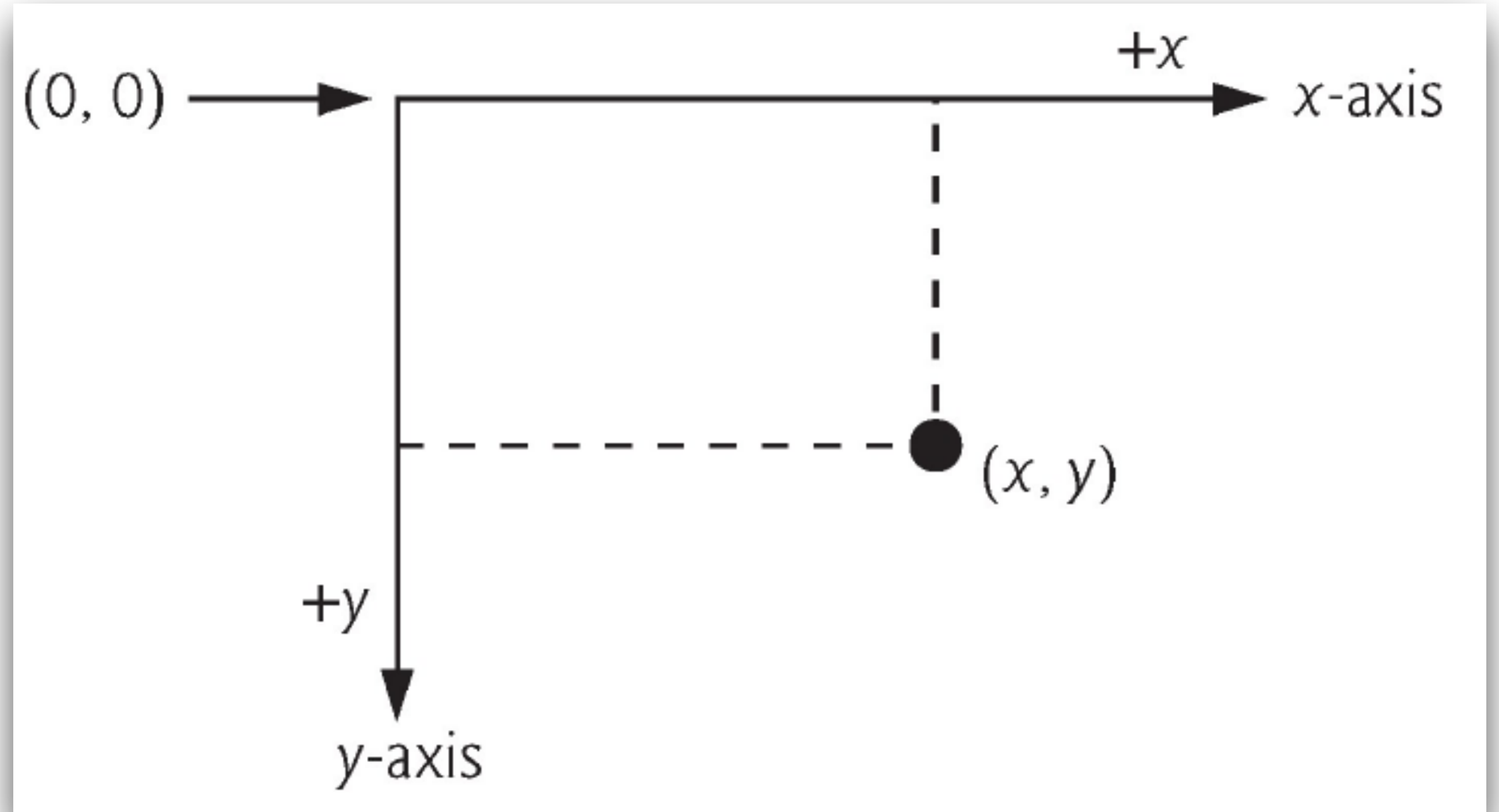
Shapes

- All shapes will require:
    - The size of the shape to be created (width, height)
    - The coordinate location to begin drawing



Coordinate Systems

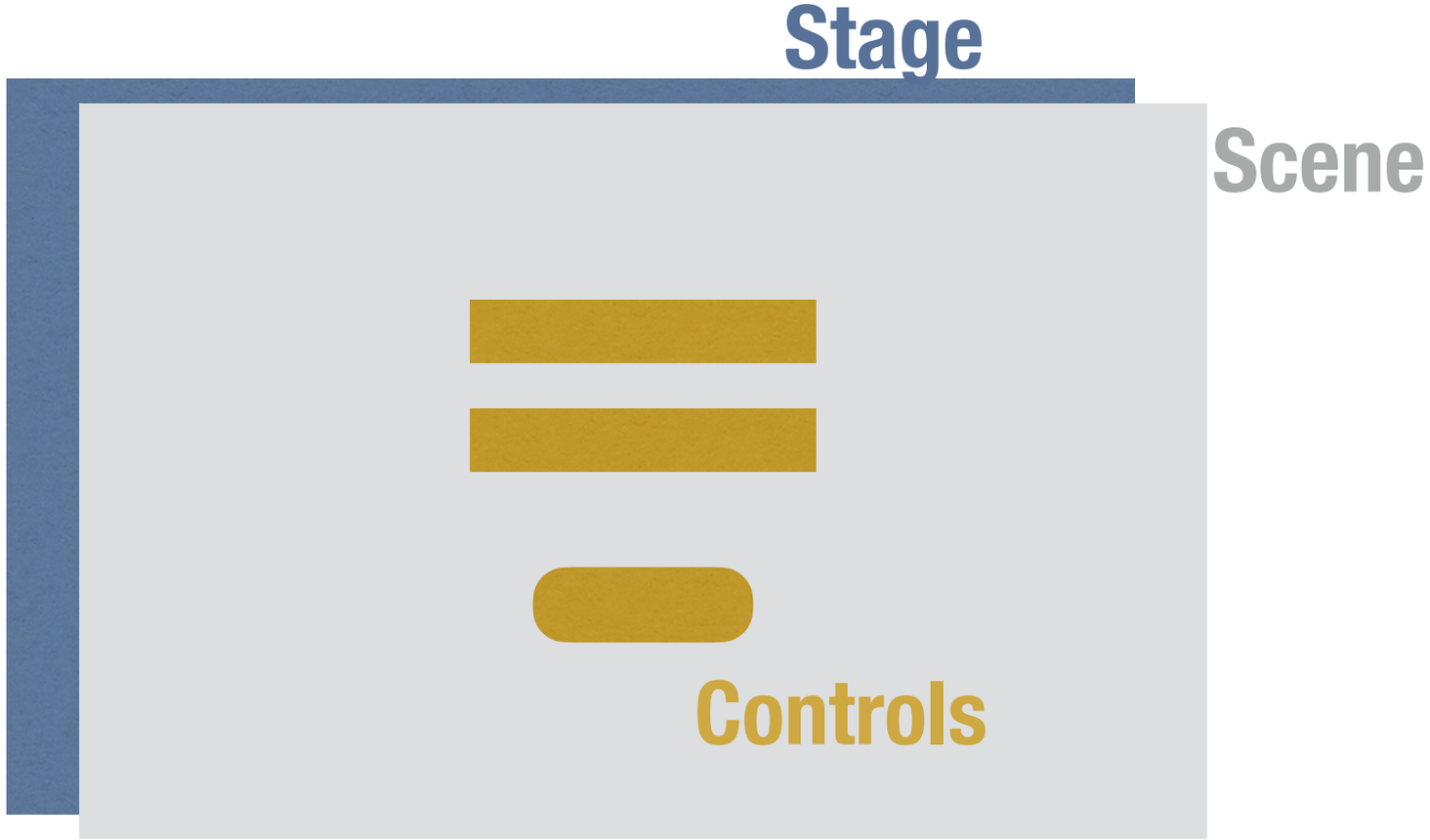
- Placing components within an application is facilitated by a coordinate scheme, which helps identify points on the screen.



JavaFX UI Controls

javafx.scene.control contains classes which are specialized Nodes in the JavaFX Scenegraph

- *So that you don't need to "reinvent the wheel" in order to implement simple UI components such as checkboxes, buttons, textfields, menus, and much more!*



Translating Shapes

```
public class MainController implements EventHandler<ActionEvent> {

    @FXML
    Circle blueCircle;

    @Override
    public void handle(ActionEvent event) {

        double location = blueCircle.getCenterX();

        Translate translate = new Translate();
        translate.setX( location + 5 );
        blueCircle.getTransforms().addAll( translate );

        // change the circle color
        blueCircle.setFill(Color.color(0.1, 0.2, 0.3));



    }

}
```