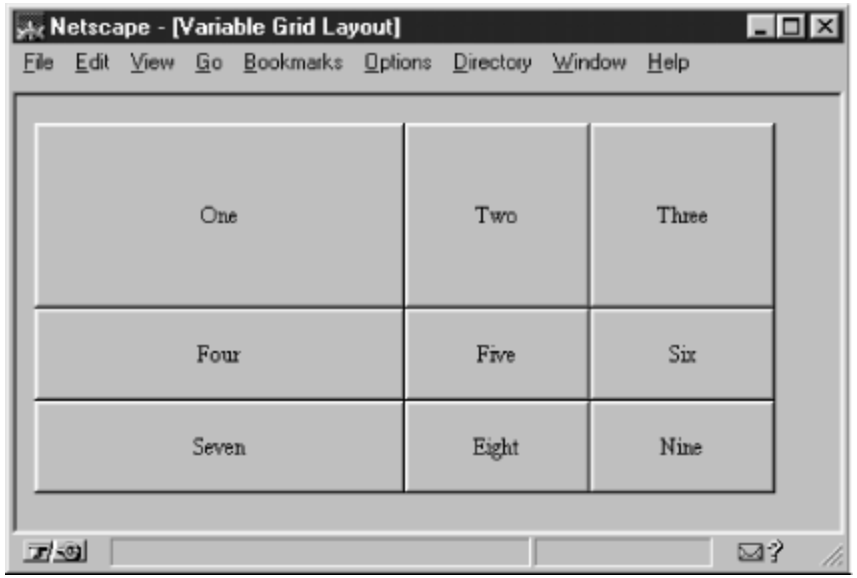
Lec 21

JavaFX

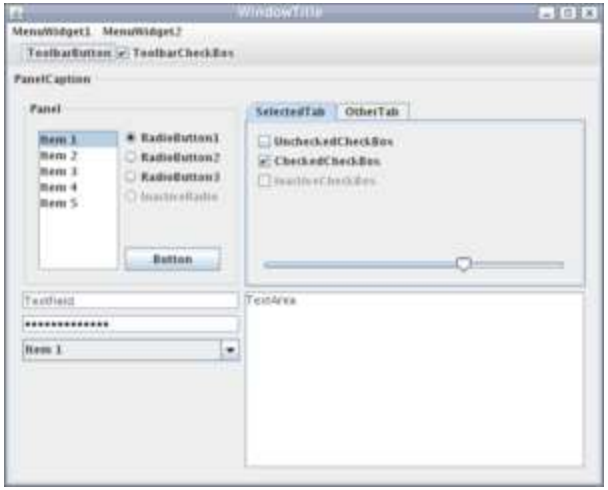
Graphical User Interface

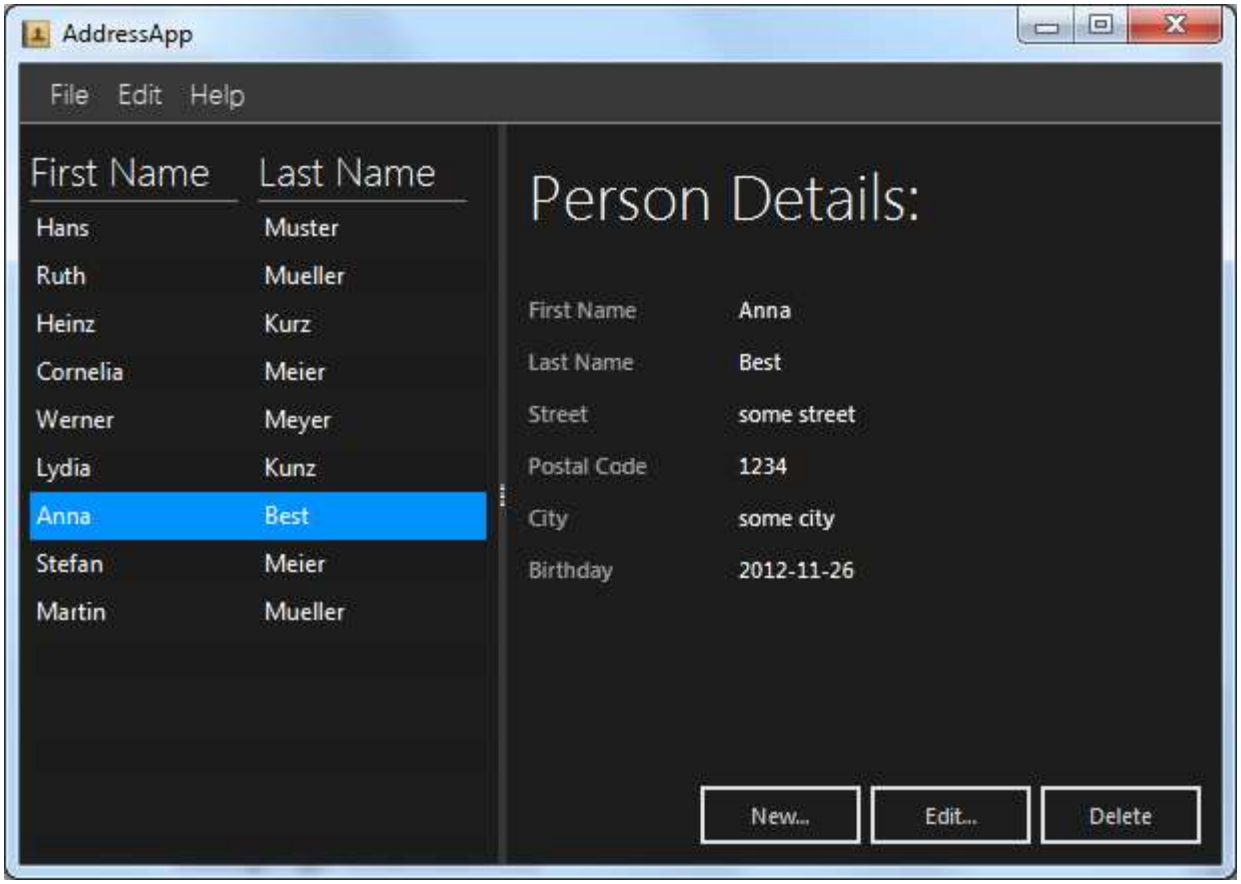
- https://www.youtube.com/watch?v=Q5763pPchvw



Java AWT

Java Swing




AddressApp
File Edit Help
First Name
Last Name
Hans
Muster
Ruth
Mueller
Heinz
Kurz
Cornelia
Meier
Werner
Meyer
Lydia
Kunz
Anna
Best
Stefan
Meier
Martin
Mueller
Person Details:
First Name
Anna
Last Name
Best
Street
some street
Postal Code
1234
City
some city
Birthday
2012-11-26
New...
Edit...
Delete

JavaFX

Java FX

- High-level support to make the following easy:
  - Shadowing
  - Blurring
  - Effects
  - 2D transforms
  - 3D transforms
  - Playing of media
  - Web application design

- JavaFX Scene Builder (we won’t use this!)
  - GUI design tool

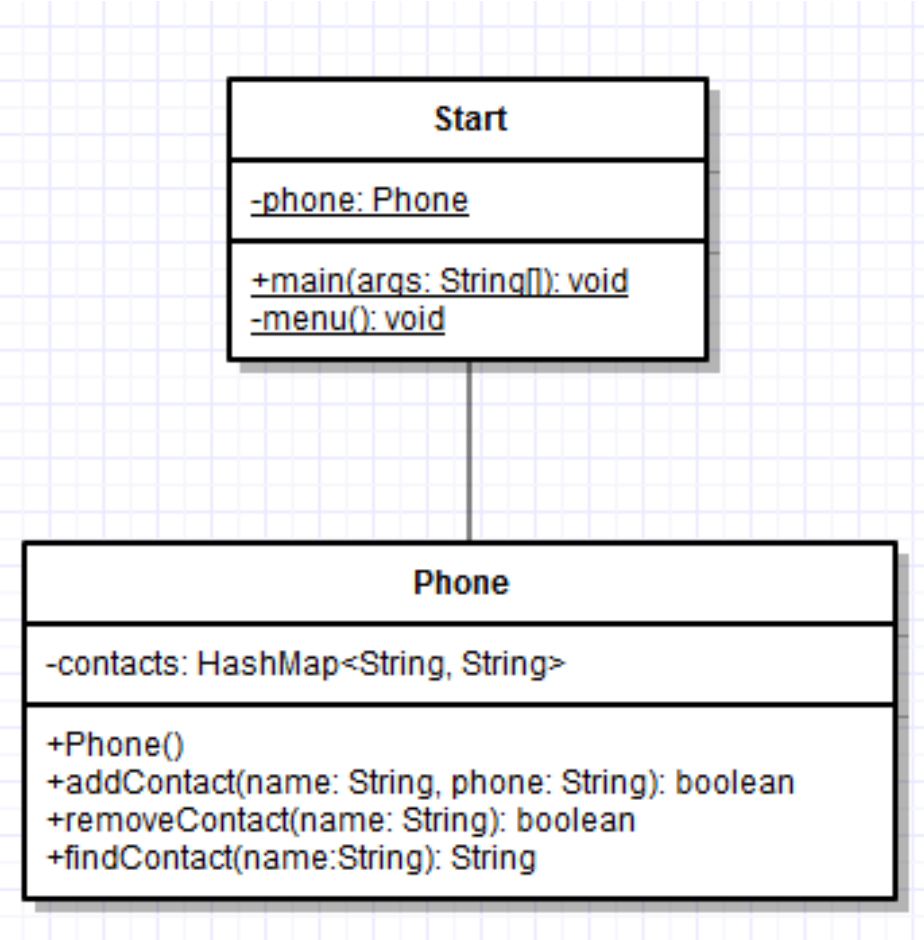
IDE

Phone

- Write a program to store phone contacts
  - Should be able to add contacts, remove contacts, and find a contact
- Ex. Usage

1) Add contact
2) Remove contact
3) Find contact
4) Quit
Section: 1
Contacts name: Adam Jundt
Contacts number: 123
Adam Jundt successfully added!
1) Add contact
2) Remove contact
3) Find contact
4) Quit
Section: 3
Contacts name: Adam Jundt
Adam Jundt: 123
1) Add contact
2) Remove contact
3) Find contact
4) Quit
Section: 2
Contacts name: Adam Jundt
Adam Jundt successfully removed

UML


Start
-phone: Phone
+main(args: String[]): void
-menu(): void
Phone
-contacts: HashMap<String, String>
+Phone()
+addContact(name: String, phone: String): boolean
+removeContact(name: String): boolean
+findContact(name:String): String

Base Code

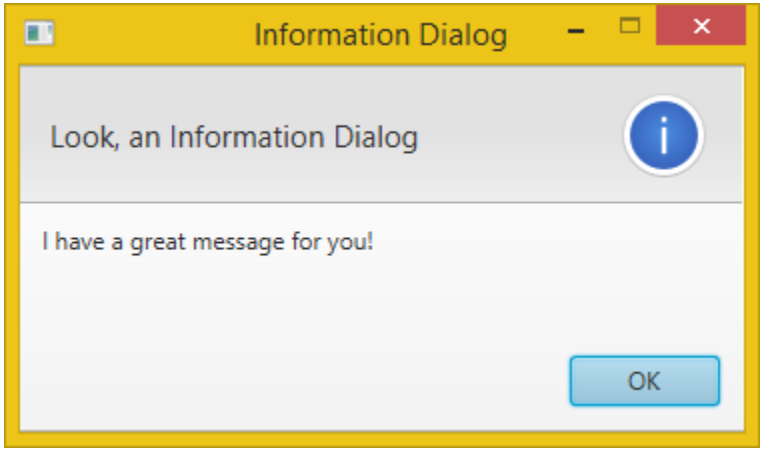
```
import javafx.application.Application;
import javafx.stage.Stage;

public class Base extends Application {

   public void start(Stage primaryStage) {
      //your code goes here
   }

   public static void main(String[] args) {
      launch(args);
   }
}
```

https://docs.oracle.com/javase/8/javafx/api/javafx/application/Application.html

Dialog Boxes

- http://code.makery.ch/blog/javafx-dialogs-official/

```
import javafx.scene.control.Alert;

Alert alert = new Alert(AlertType.INFORMATION);
alert.setTitle("Information Dialog");
alert.setHeaderText("Look, an Information Dialog");
alert.setContentText("I have a great message for you!");
alert.showAndWait();
```



Stages

- A stage in javafx is a window/frame/place that GUI elements reside
- Start() already comes with a default stage as a parameter
- Can create multiple stages
  - Same as creating multiple windows

Stages

```
import javafx.application.Application;
import javafx.stage.Stage;

public class MultiStage extends Application {

  public void start(Stage primaryStage) {
    primaryStage.setTitle(“Stage");
    primaryStage.show();
  }

  public static void main(String[] args) {
    launch(args);
  }
}
```

What happens

```
import javafx.application.Application;
import javafx.stage.Stage;

public class MultiStage extends Application {

  public void start(Stage primaryStage) {
    Stage secondaryStage = new Stage();
    primaryStage.setTitle("primary");
    secondaryStage.setTitle("secondary");
    primaryStage.show();
    secondaryStage.show();
  }

  public static void main(String[] args) {
    launch(args);
  }
}
```

A) Compiler error
B) Run time error
C) One window opens
D) Two windows open

Scenes

- Scenes in javafx contain everything that displays on the stage
- Only one scene per stage.
- There is a method, stage.setScene();
- Elements added to the scene are Nodes/Parent
- Scene scene = new Scene(Parent);

Scene

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.stage.Stage;

public class MyScene extends Application {

  public void start(Stage primaryStage) {
    Button button = new Button("Click");  //new, we'll cover next
    Scene scene = new Scene(button, 100, 100);
    primaryStage.setScene(scene);
    primaryStage.show();
}

  public static void main(String[] args) {
    launch(args);
  }
}
```

What get's displayed

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.stage.Stage;

public class MyScene extends Application {

  public void start(Stage primaryStage) {
    Button button1 = new Button("Click");
    Button button2 = new Button("Ok");
    Scene scene = new Scene(button1, 100, 100);
    scene = new Scene(button2, 100, 100);
    primaryStage.setScene(scene);
    primaryStage.show();
}

  public static void main(String[] args) {
    launch(args);
  }
}
```

A) A button with "Click"
B) A button with "Ok"
C) Both (A) and (B)
D) Compiler error

Layout Managers - HBox

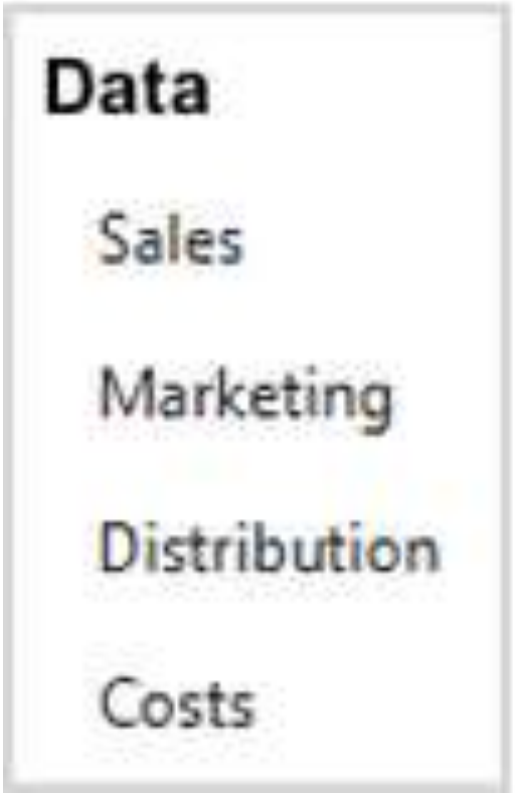
- Allows for you to place items side by side


```
HBox pane = new HBox();
pane.getChildren().add(button1);
pane.getChildren().add(button2);
  OR
pane.getChildren().addAll(button1, button2);
Scene scene = new Scene(pane);
Stage.setScene(scene);
```

Layout Managers - VBox

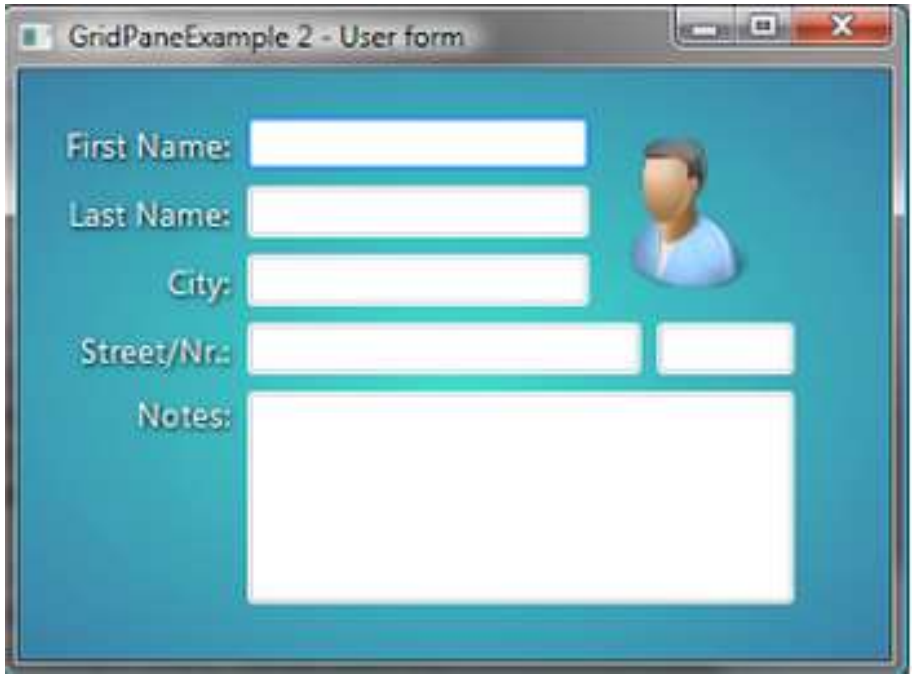
- Allows for you to place items top to bottom

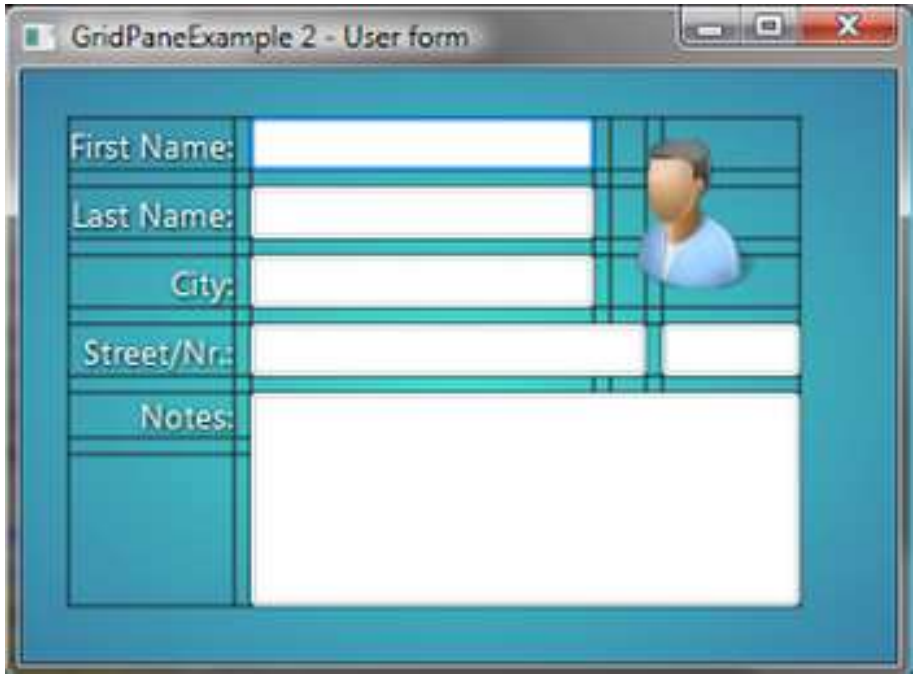
```
VBox pane = new VBox();
pane.getChildren().add(label1);
pane.getChildren().add(label2);
   OR
pane.getChildren().addAll(label1, label2);
Scene scene = new Scene(pane);
Stage.setScene(scene);
```

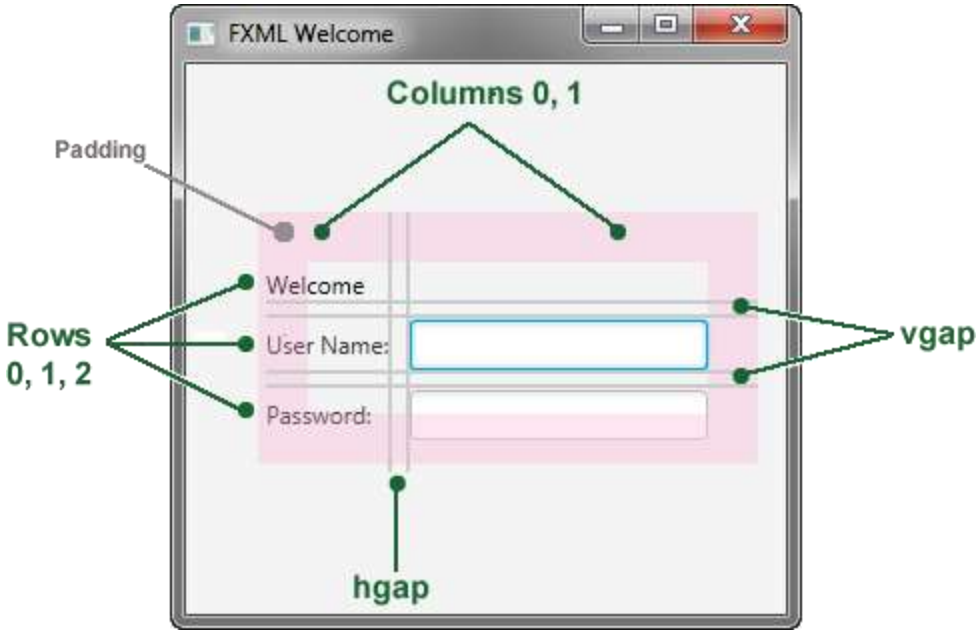


Layout Managers - Grid Pane

- Allows for placement of items in a grid
- Likely useful for the next assignment



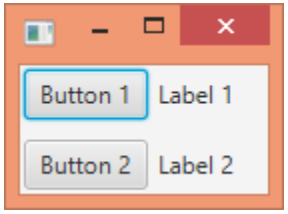



FXML Welcome
Columns 0, 1
Padding
Welcome
User Name:
Password:
Rows
0, 1, 2
vgap
hgap

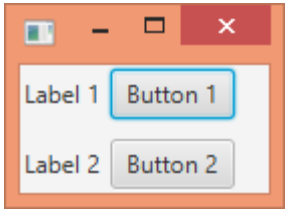
What happens

```
public void start(Stage primaryStage) {
  Button button1 = new Button("Button 1");
  Button button2 = new Button("Button 2");
  Label label1 = new Label("Label 1");
  Label label2 = new Label("Label 2");

  GridPane pane = new GridPane();
  pane.setHgap(5);
  pane.setVgap(10);
  pane.setPadding(new Insets(2));

  pane.add(button1,0, 0);  // (node, col, row);
  pane.add(button2,0, 1);
  pane.add(label1,2, 0);
  pane.add(label2,1, 1);

  Scene scene = new Scene(pane);
  primaryStage.setScene(scene);
  primaryStage.show();
}
```

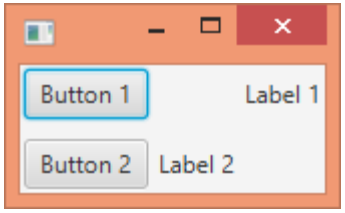
A



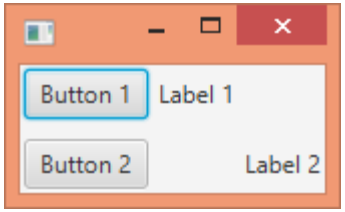
B



C



D



Items to add to a Layout Manager

- Button

- Label

- TextField

- CheckBox

Items to add to a Layout Manager

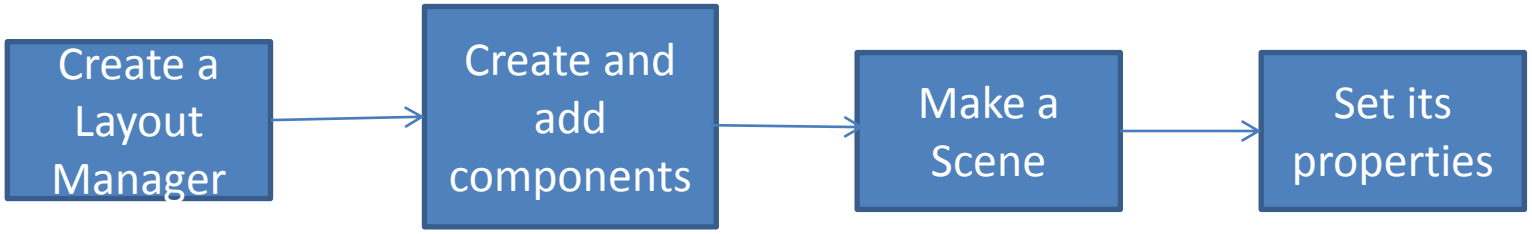
- TextArea – larger/multi-line text boxes
- ImageViewer – To view Images

- Shapes:
  - Line, Circle, Ellipse, Rectangle, Path, Polygon, Polyline, Text

JavaFX - GUIs

- Stage: Top level Java FX Component
- Scene: A container for all the items in the scene
- Border, Hbox, Vbox, FlowPane, GridPane: Helper classes that govern where components appear in the Scene

A very rough guide to creating a simple GUI:



You don't always have to do this in this order. Some of the steps can have substeps (e.g., creating a Layout Manager to organize other components)

What's needed for this program

- Program that asks user for their name and, when they hit a "run" button an alert shows up with their name.

A) Stage, Scene, Layout Manager
B) Stage, Scene, Layout Manager, button, textField, Alert,
C) Stage, Scene, Layout Manager, label, button, textField, Alert
D) Stage, Scene, Layout Manager, button, Alert
E) B or C

CSS + JavaFX

http://docs.oracle.com/javafx/2/api/javafx/scene/doc-files/cssref.html

http://code.makery.ch/library/javafx-8-tutorial/part4/

```
scene.getStylesheets().add(getClass()
      .getResource("cssfile.css").toExternalForm());
```