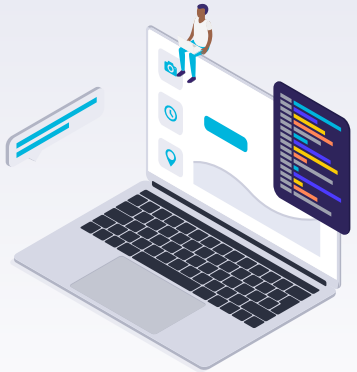
Programação Orientada por Objetos

JavaFX – Controlos

Prof. Cédric Grueau
Prof. José Sena Pereira
Departamento de Sistemas e Informática
Escola Superior de Tecnologia de Setúbal
Instituto Politécnico de Setúbal
2022/2023

Sumário

- Controlos do JavaFX
  - Controlos
    - **`Button`** – criação e utilização
    - **`TextField`** – criação e utilização
    - **`Label`** - – criação e utilização
    - **`ListView`** – criação, utilização e preenchimento
- Exemplo Controlos simples
- Exemplo: Desenhador de Formas



CoNtrolos do JavaFX

- JavaFX – Controlos

JavaFX – Controlos

- As classes para criar controlos de interface com o utilizador encontram-se no pacote **`javafx.scene.control`**

- Exemplos:
  - **`Button`**
  - **`TextField`**
  - **`Label`**
  - **`ListView`**

JavaFX – Controlos

- No exemplo inicial:
  - Definiu-se um botão!

```
Button btn = new Button();
btn.setText("Say 'Hello World'");
```

  - Associou-se ao botão um handler para o evento **Action**

```
btn.setOnAction( //[…] );
```

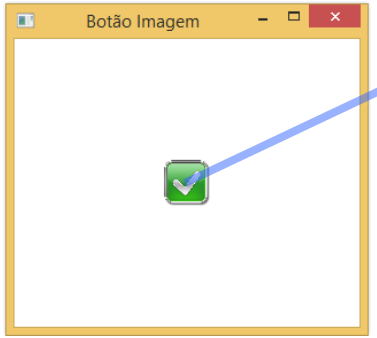
```
public void start(Stage primaryStage) {
  primaryStage.setTitle("Hello World!");
  Button btn = new Button();
  btn.setText("Say 'Hello World'");
  btn.setOnAction(
    new EventHandler<ActionEvent>(){
      @Override
      public void handle(ActionEvent event) {
        System.out.println("Hello World!");
      }
    });
  StackPane root = new StackPane();
  root.getChildren().add(btn);
  primaryStage.setScene(new Scene(root, 300, 250));
  primaryStage.show();
}
```

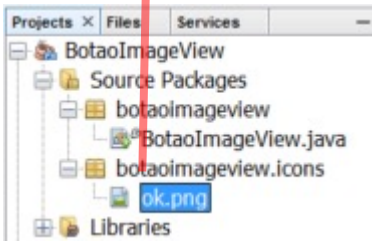
JavaFX – Controlos

- Colocar uma imagem num botão

1. Criar uma imagem, associada a um ficheiro.
2. Associar a imagem ao botão.

```
public void start(Stage primaryStage) {
  primaryStage.setTitle("Botão Imagem");
  Image imageOk = new Image(getClass().getResourceAsStream(
                  "icons/ok.png"), 40, 40, false, false);
  Button btn = new Button("OK", new ImageView(imageOk));

  btn.setOnMouseClicked(e -> System.out.println("OK"));

  StackPane root = new StackPane();
  root.getChildren().add(btn);

  Scene scene = new Scene(root, 300, 250);

  primaryStage.setScene(scene);
  primaryStage.show();
}
```





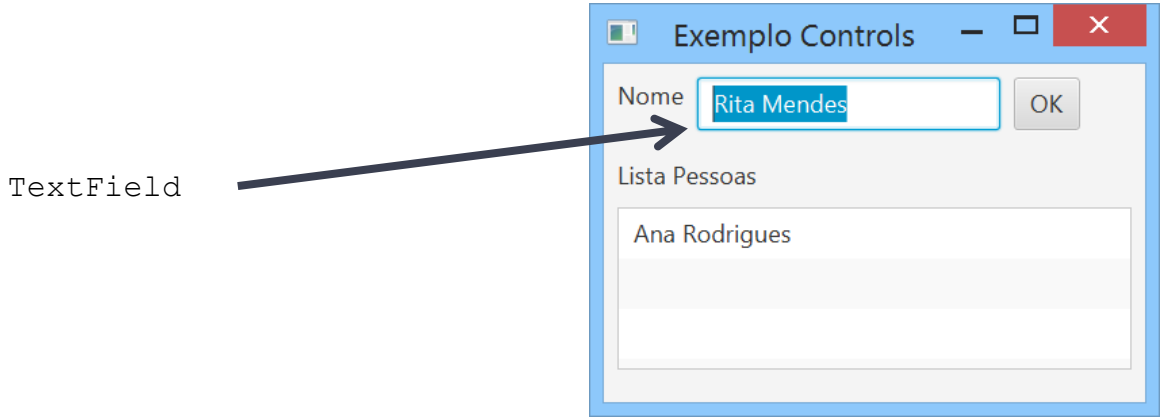
JavaFX – Controlos

- **`TextField`** – Criação e Utilização

1. Criar um objeto do tipo **`TextField`**.
2. Para ler o conteúdo introduzido num **`TextField`** usa-se o método **`getText()`**;

```
TextField textFieldName = new TextField();
textFieldName.setMinSize(12, 10);
```

```
String name= textFieldName.getText();
```

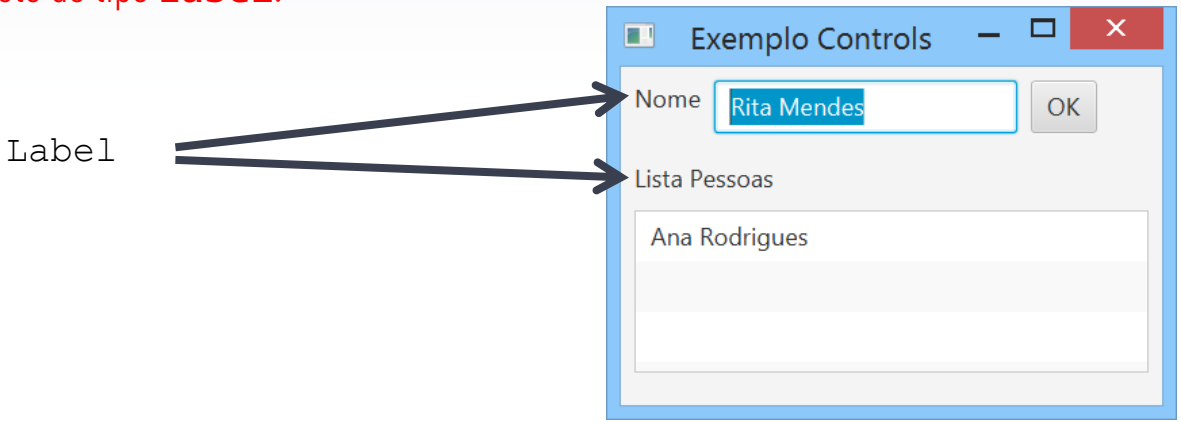


JavaFX – Controlos

- **`Label`** – Criação e Utilização

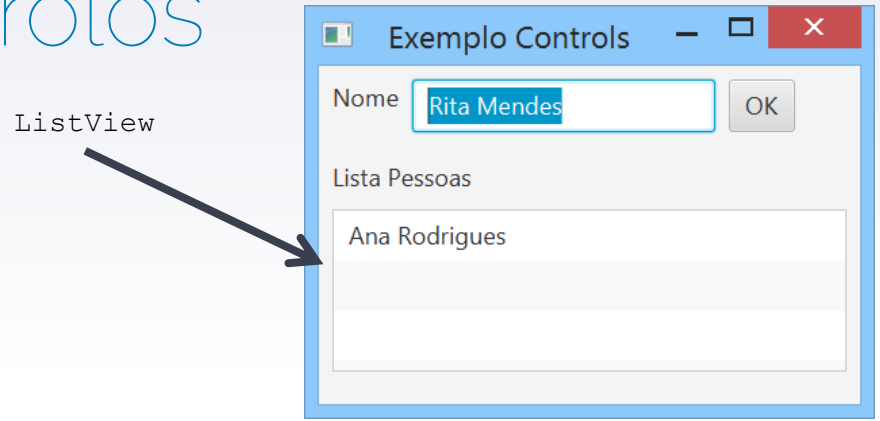
```
Label nameLabel = new Label("Nome");
Label listLabel = new Label("ListaPessoas");
```

1. Criar um objeto do tipo **`Label`**.



JavaFX – Controlos

- **`ListView`** – Criação e Utilização



1. Criar um objeto do tipo **`ListView`**.
2. Criar uma **`ObservableList`** de **`Strings`** e associá-la à **`ListView`**

```
ListView<String> listOfNames= new ListView<>();


ObservableList<String> items=FXCollections.observableArrayList();
listOfNames.setItems(items);
```

3. **Adicionam-se linhas de texto à `ObservableList` para preencher a `ListView`**

```
items.add(nome);
```

JavaFX – Classe FXCollections

- A classe **`FXCollections`** disponibiliza um conjunto de métodos de classe (**`static`**) que permitem obter e manipular "coleções sincronizáveis":
  - `observableArrayList()`
  - `observableSet()`
  - `observableHashMap()`
  - `replaceAll`
  - `reverse`
  - `rotate`
  - `sort`
  - etc.
- As "coleções sincronizáveis" ao serem modificadas (adicionar ou remover elementos) tentam atualizar os objetos com que estão sincronizadas (normalmente elementos gráficos). Na próxima aula será explicado este mecanismo de sincronização.

Exemplo
Controlos simples

- JavaFX – Controlos

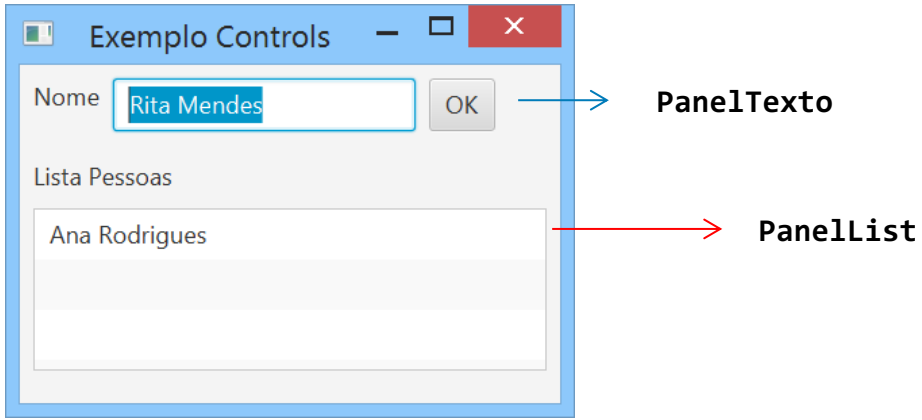
nplo Completo (ListView, TextField)

Objetivo:

Preencher a lista com os nomes introduzidos no **`TextField`**.

O nome introduzido é transferido para a lista após validado pelo utilizador através do botão OK!

```
public void start(Stage primaryStage) {
     primaryStage.setTitle("Exemplo Controlos");
     VBox root = new VBox();
     addTextPanel (root);
     addListPanel (root);
     primaryStage.setScene(new Scene(root, 400, 250));
     primaryStage.show();
 }
```



JavaFX – Exemplo: adicionarPanelTexto

```
private TextField textFieldName;
```

Atributo na Classe

```
public void addTextPanel (Pane root) {

        HBox textPanel = new HBox();
        textPanel.setPadding(new Insets(10));
       textPanel.setSpacing(10);

        Label nameLabel = new Label("Nome");

        this.textFieldName = new TextField();
         this.textFieldName.setMinSize(12, 10);

         Button okButton = new Button("OK");
         okButton.setOnAction(e -> addListName());

         textPanel.getChildren().addAll(nameLabel,this.textFieldName,okButton);
         root.getChildren().add(textPanel);
    }
```

Panel (HBox)

Label

TextField

Button

JavaFX – Exemplo: adicionarPanelList

```
private ObservableList<String> items;
private ListView<String> listOfNames;
```

Atributos na Classe

```
public void addListPanel (Pane root) {
      VBox listPanel = new VBox();
       listPanel.setPadding(new Insets(10));
       listPanel.setSpacing(10);

      Label listLabel = new Label("Lista Pessoas");

      this.items= FXCollections.observableArrayList();
      this.listOfNames = new ListView<>();
      this.listOfNames.setPrefSize(100,120);
      this.listIfNames.setItems(this.items);

      listPanel.getChildren().addAll(namesLabel,this.listOfNames);
      root.getChildren().add(listPanel);
  }
```

Panel (VBox)

Label

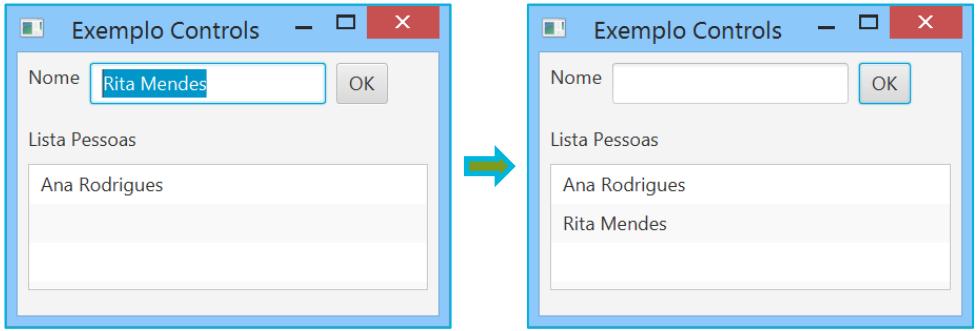
ListView

JavaFX – Exemplo: adicionarNomeLista

Método que é executado quando o utilizador clica no botão OK!

- O texto introduzido no **TextField (textFieldNome)**.
- é adicionado aos **items** associados à **ListView**
- O campo do **TextField** é colocado a vazio

```
public void addNameToList() {
    String name = this.textFieldName.getText();
    if (!name.isEmpty()) {
        this.items.add(name);
    }

    this.textFieldName.setText("");
}
```

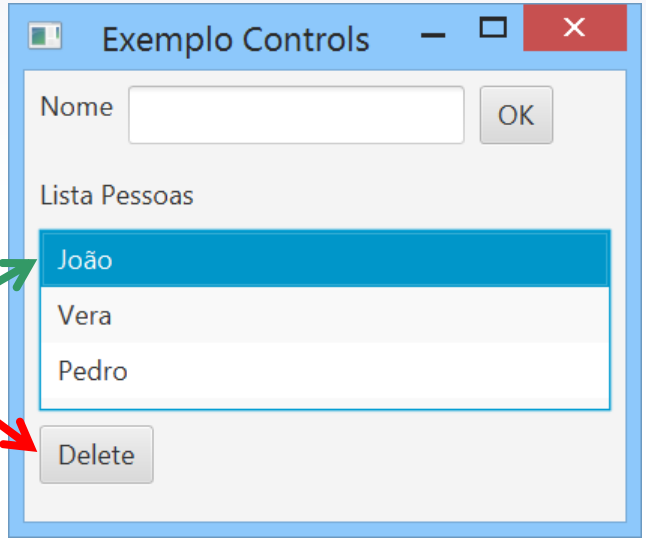


JavaFX – Exemplo: Apagar elementos da lista

Objetivo:

Selecionar um nome da lista e apagá-lo.

- Adicionar um botão para apagar
- Associar a acção de remover um elemento da lista
- Remover o elemento selecionado



JavaFX – Exemplo: Apagar elementos da lista

Objetivo:

Selecionar um nome da lista e apagá-lo.

- Adicionar um botão para apagar
- Associar a acção de remover um elemento da lista
- Remover o elemento selecionado da lista **`items`**.

```
public void addListPanel(Pane painel) {

...

Button deleteButton = new Button("Delete");
deleteButton.setOnAction(e -> removeNameFromList());

...
listPanel.getChildren().addAll(namesLabel, this.listOfNames,
deleteButton);
...
}
```

```
public void removeNameFromList() {
  int  index;
  index=this.listOfNames.getSelectionModel().getSelectedIndex();
  if (index != -1) {
    this.items.remove(index);
  }
}
```

JavaFX – Exemplo: Aceder aos elementos selecionados

Numa Lista podemos ter:

1. Selecção Singular

   podemos aceder ao:

   1. Item selecionado
   2. Índice do item selecionado

2. Selecção Múltipla

   podemos aceder aos:

   1. Itens selecionados
   2. Índices dos itens selecionados

Singular

- ```
  String item =
  list.getSelectionModel().getSelectedItem();
  ```

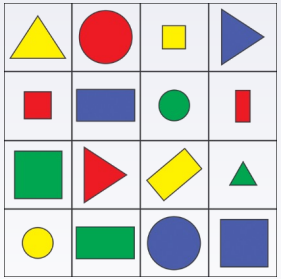
- ```
  int index =
  list.getSelectionModel().getSelectedIndex();
  ```

Múltipla

- ```
  ObservableList<String> items=
  list.getSelectionModel().getSelectedItems();
  ```

- ```
  ObservableList<Integer> indexes =
  list.getSelectionModel().getSelectedIndices();
  ```

JavaFX – Exemplo Controlos Simples

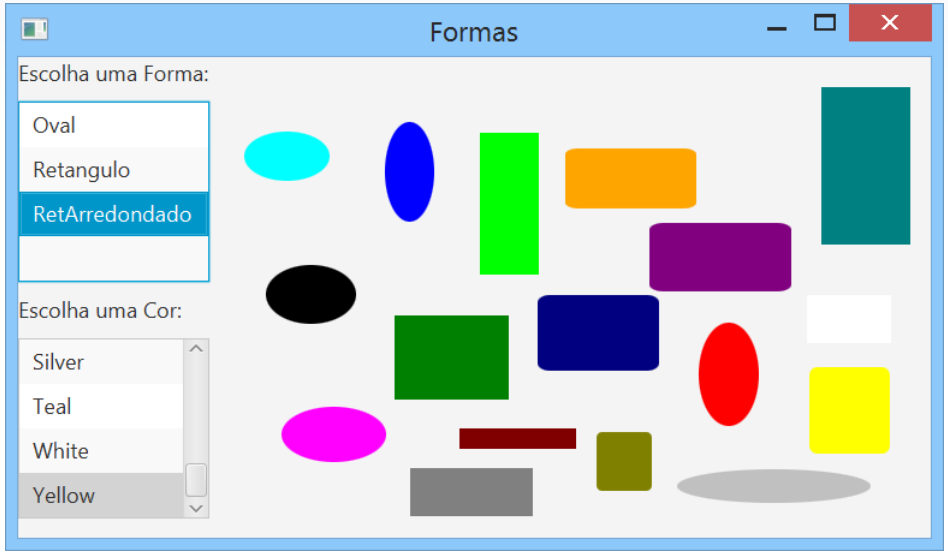


Desenhador de Formas

- JavaFX – Controlos

JavaFX Exemplo – Desenhador de formas

- Criar uma aplicação que permita o desenho de diversas formas geométricas, com diversas cores:



JavaFX Exemplo – Desenhador de formas

- Serão apresentadas duas **`ListView`** que permitem indicar a forma e a cor pretendidas. Estas serão colocadas num painel **VBox**;
- Através do uso do rato indica-se (carregando) o ponto inicial e (levantando) o ponto final;
- Em oposição à criação de objetos das subclasses de **Shape**, apresentados na aula de introdução, será criada uma zona de desenho (**Canvas**), e utilizam-se os métodos de desenho de **`GraphicsContext`** para criar todas as formas;
- **`GraphicsContext`** (http://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html) é uma classe que contém toda a informação necessária ao desenho (cores, espessuras, etc.) e os métodos necessários: **`setFill`**, **`getFill`**, **`setStroke`**, **`getStroke`**, **`fillArc`**, **`strokeArc`**, **`beginPath`**, **`closePath`**, **`moveTo`**, **`lineTo`**, **`fill`**, **`stroke`**, **`fillOval`**, **`strokeOval`**, **`fillRect`**, **`strokeRect`**, **`fillRoundRect`**, **`strokeRoundRect`**, **`strokeLine`**, **`fillText`**, **`strokeText`**, etc.

JavaFX Exemplo – Alternativas de formas

- Criar um **Enum** com as alternativas de formas: Ovais, Retângulos e Retângulos Arredondados

- É criado o método que desenha a forma, em função do valor atual

```
public enum Shape {
 Oval, Retangulo, RetArredondado;
//Foi quebrada a norma de escrita em MAIÚSCULAS para não
//rescrever o toString para ficar "bonitinho"

 private static final double ROUNDING = 5; //RetArredondado

 public void drawShape (GraphicsContext graphicsContext,
                        double x, double y,
                        double largura, double height) {
   switch (this) {
     case Oval:
         graphicsContext.fillOval(x, y, width, height);
         break;
     case Retangulo:
         graphicsContext.fillRect(x, y, width, height);
         break;
     case RetArredondado:
         graphicsContext.fillRoundRect(x,y, width, height,
width /Shape. ROUNDING,altura/ Shape. ROUNDING);
         break;
   }
 }
}
```

JavaFX Exemplo – Seletor de Cores e Formas

- O seletor de formas e cores será um **VBox** com duas **ListView** (atributos da classe) e dois **Label**
- São definidas constantes com as dimensões dos controlos
- A constante **NAMES_colors** contém as **String** com os nomes de algumas cores standard (as originais em HTML)

```
public class ShapeColorSelector extends VBox {

  private static final double SPACING = 10.0;
  private static final double LARGURA_LISTA = 130.0;
  private static final double ALTURA_LISTA = 150.0;
  private static final String[] NAMES_colors =
       {"Aqua", "Black", "Blue", "Fuchsia", "Gray",
        "Green", "Lime", "Maroon", "Navy", "Olive",
        "Orange", "Purple", "Red", "Silver", "Teal",
        "White", "Yellow"};

  private ListView<Shape> shapes;
  private ListView<String> colors;

  public ShapeColorSelector() {
    Label shapeLabel = new Label("Escolha uma Forma:");
    Label colorsLabel = new Label("Escolha uma Cor:");
    shapes = criarListView(Shapes.values());
    colors =
criarListView(ShapeColorSelector.NAMES_Colors);
    getChildren().addAll(
      shapeLabel,shapes, colorsLabel, colors);
    setSpacing(SeletorFormaCor SPACING);
  }
```

JavaFX Exemplo – Seletor de Cores e Formas

- Criar um método genérico (tipo de elementos), que recebe um **`array`** com os elementos (de diferentes tipos) a serem colocados na **`ListView`**

- Criar os dois métodos inspetores que devolvem as escolhas da forma e cor

```
  private <T> ListView<T> criarListView(T[] valores)
  {
    ListView<T> result = new ListView<>();
    result.setItems(FXCollections.observableArrayList(val
ores));
    result.getSelectionModel().select(0);
    result.setPrefWidth(SeletorFormaCor.LARGURA_LISTA);
    result.setPrefHeight(SeletorFormaCor.ALTURA_LISTA);
    return result;
  }

  public Forma formaEscolhida() {
    return formas.getSelectionModel().getSelectedItem();
  }

  public Color corEscolhida() {
    String nomeCor = cores.getSelectionModel().getSelected
Item();
    return Color.valueOf(nomeCor);
  }

}
```

JavaFX Exemplo – Área de desenho

Criar um **Canvas** que tem definido os seus atributos (**shapeColorSelector**, recebido como argumento no Construtor, e as coordenadas iniciais).
Criar, ainda, os seus **EventHandler** de rato carregado (início da figura: **setOnMouse Pressed**) ou levanta- do (fim da figura: **setOnMouseReleased**)

```
public class DrawingArea extends Canvas {

  private static final double WIDTH = 600;
  private static final double EIGHT = 400;

  private SeletorFormaCor shapeColorSelector;
  private double xInicial;
  private double yInicial;

  public DrawingArea(SeletorFormaCor shapeColorSelector) {
    super(DrawingArea.WIDTH, DrawingArea.EIGHT);
    this.shapeColorSelector = shapeColorSelector;
    setOnMousePressed(this:: fixStart);
    setOnMouseReleased(this:: drawShape);
  }

  private void fixStart(MouseEvent event) {
     xInicial = event.getX();
     yInicial = event.getY();
  }
```

- O método associado ao **setOnMousePressed (fixarInicio)** regista, nos atributos, as coordenadas do ponto onde começou a forma a desenhar-se

Expressões lamba versus referenciação de métodos

- Quando uma *expressão lambda* se limita a executar um método:

  `e -> fixStart(e)` ou **`e -> drawShape(e)`**

- E esse método tem uma assinatura semelhante à do **`EventHandler`**:

  `void handle(MouseEvent event)`

- Então pode-se indicar apenas o nome do método que deverá ser executado (antecedendo-o com a indicação do objeto onde o método será executado):

  `this::fixStart` ou **`this::drawShape`**

- Se o método fosse **`static`** então seria indicado o nome da classe antes do método:
  `NomeDaClasse::NomeDoMétodo`

http://docs.oracle.com/javase/tutorial/java/java00/methodreferences.html

JavaFX Exemplo – Área de desenho

- O método associado ao **`setOnMouseReleased`** é responsável por armazenar as coordenadas do ponto onde termina a forma a desenhar-se, por calcular os limites da forma, por preparar o ambiente de desenho e, finalmente, por mandar desenhar a forma:

```
private void drawShape(MouseEvent event) {
  double xFinal = event.getX();
  double yFinal = event.getY();

  double x = Math.min(xInicial, xFinal);
  double y = Math.min(yInicial, yFinal);
  double width = Math.abs(xFinal - xInicial);
  double eight = Math.abs(yFinal - yInicial);

  GraphicsContext graphicsContext = getGraphicsContext2D();
  Color chosenColor = shapeColorSelector.chosenColor();
  graphicsContext.setFill(chosenColor);
  graphicsContext.setStroke(chosenColor);
  shapeColorSelector.chosenShape().drawShape(graphicsContext, x, y, width, eight);
 }
}
```

JavaFX Exemplo – desenharForma

- O método **drawShape**, associado ao **setOnMouseReleased** , começa por registar as coordenadas do ponto onde termina a forma a desenhar-se;
- Calcula, em seguida, as coordenadas do canto superior esquerdo (menores coordenadas x e y) e a largura e altura para a respetiva forma (diferença entre as coordenadas);
- Fixa a cor a utilizar em função da cor escolhida (**chosencolor**);
- Executa o método **drawShape** da forma escolhida, obtida com o método **chosencolor**, passando o **GraphicsContext** (obtido por **getGraphicsContext2D**) da **Canvas** e os valores calculados previamente.

JavaFX Exemplo – método start

- A aplicação principal é apenas constituída por um **HBox** que conterá um **ShapeColorSelector** e um **DrawingArea** (que recebe o **ShapeColorSelector** como argumento):

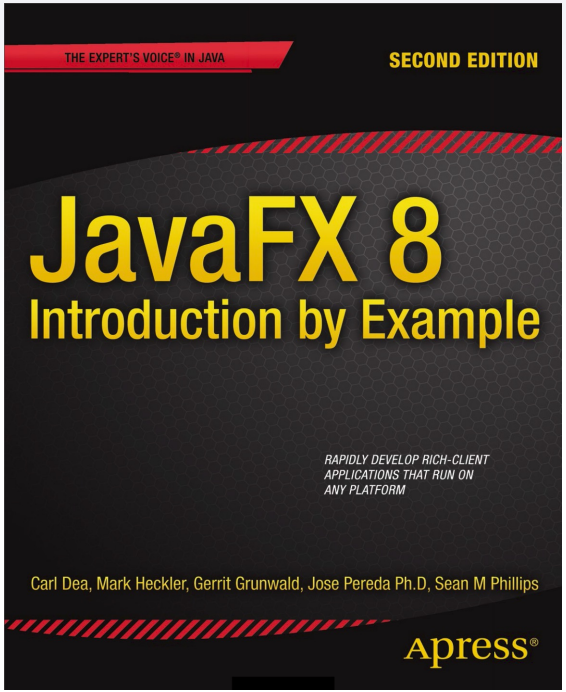
```
public class CanvasDemo extends Application {

  @Override
  public void start(Stage primaryStage) {
    HBox root = new HBox();
    ShapeColorSelector shapeColorSelector = new ShapeColorSelector();
    root.getChildren().addAll(shapeColorSelector,
                              new DrawingArea(shapeColorSelector));
    primaryStage.setTitle("Formas");
    primaryStage.setScene(new Scene(root));
    primaryStage.show();
  }

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

Resumindo

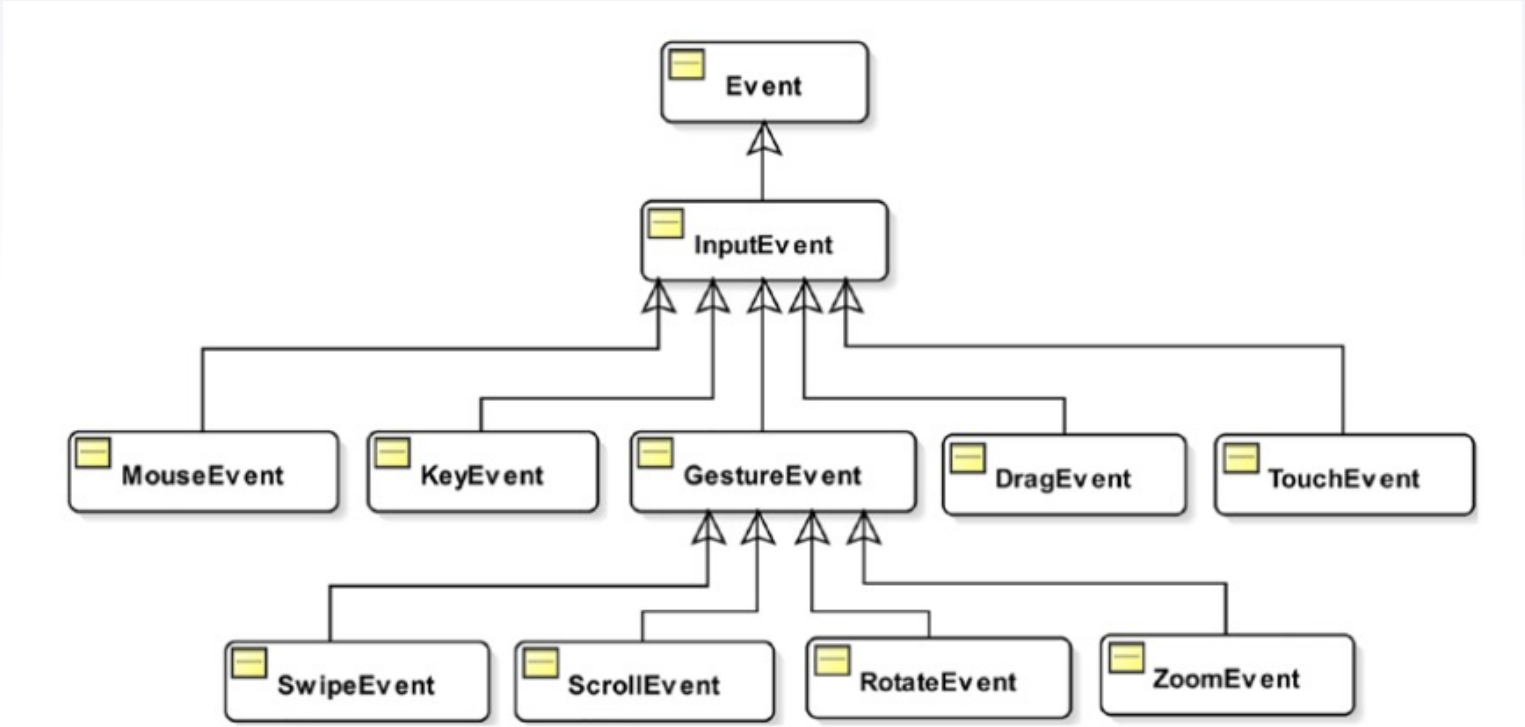
- Controlos
  - **`Button`** – inserir imagem num botão
    1. Criar uma imagem associada a um ficheiro (**`Image`**).
    2. Associar a imagem ao botão (**`ImageView`**).
  - **`TextField`** – criação e utilização
    1. Criar um objeto do tipo **`TextField`**.
    2. Usar **`getText()`** para ler conteúdo e o **`setText()`** para o alterar.
  - **`Label – criação e utilização`**
    1. Criar um objeto do tipo **`Label`**.
  - **`ListView – criação, utilização e preenchimento`**
    1. Criar um objeto do tipo **`ListView`**.
    2. Criar uma **`ObservableList`** e associá-la à **`ListView`** (será utilizado o **`toString`**)
    3. Adicionar ou remover linhas à **`ObservableList`** para preencher a **`ListView`**
  - Uso de **`Canvas`** e **`GraphicsContext`**

Leitura Complementar

- Chapter 2 – JavaFX Fundamentals Pgs 31 a 51
- Chapter 4 – Layouts and UI Controls Pgs 108 a 111
- Chapter 5 – Graphics with JavaFX Pgs 123 a 139
- Documentação:
  - http://docs.oracle.com/javase/8/javafx/api/toc.htm
  - Controlos:
    - http://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/package-frame.html
  - http://docs.oracle.com/javase/8/javafx/user-interface-tutorial/ui_controls.htm#JFXUI336
  - Desenhar no Canvas:
    - http://docs.oracle.com/javase/8/javafx/graphics-tutorial/canvas.htm#JFXGR214



InputEvent


Event
InputEvent
MouseEvent
KeyEvent
GestureEvent
DragEvent
TouchEvent
SwipeEvent
ScrollEvent
RotateEvent
ZoomEvent