Programação Orientada a Objetos

JavaFX - Controlos

Prof. Rui César das Neves, Prof. José Cordeiro

Departamento de Sistemas e Informática

Escola Superior de Tecnologia de Setúbal – Instituto Politécnico de Setúbal

2014/2015

Sumário

- Controlos
  - **`Button`** – criação e utilização
  - **`TextField`** – criação e utilização
  - **`Label`** - – criação e utilização
  - **`ListView`** – criação, utilização e preenchimento

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( //[…] );
```

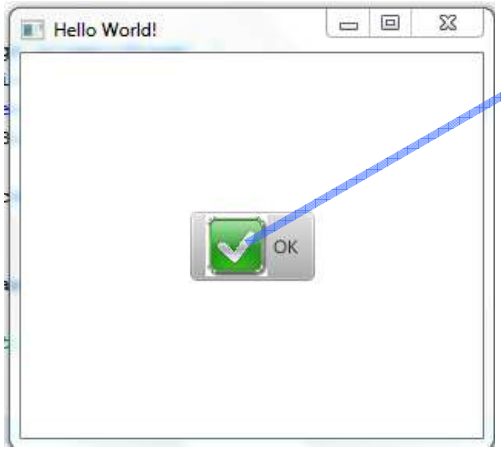
```
public void start(Stage primaryStage) {
  primaryStage.setTitle("Hello World!");
  Button btn = new Button();
  btn.setText("Say 'Hello World'");
  btn.setOnAction(
    new EventHandler<ActionEvent>(){
      @Override
      public void handle(ActionEvent event) {
        System.out.println("Hello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

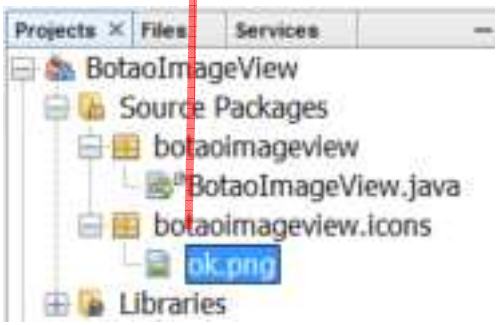
JavaFX- Controlos

- 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

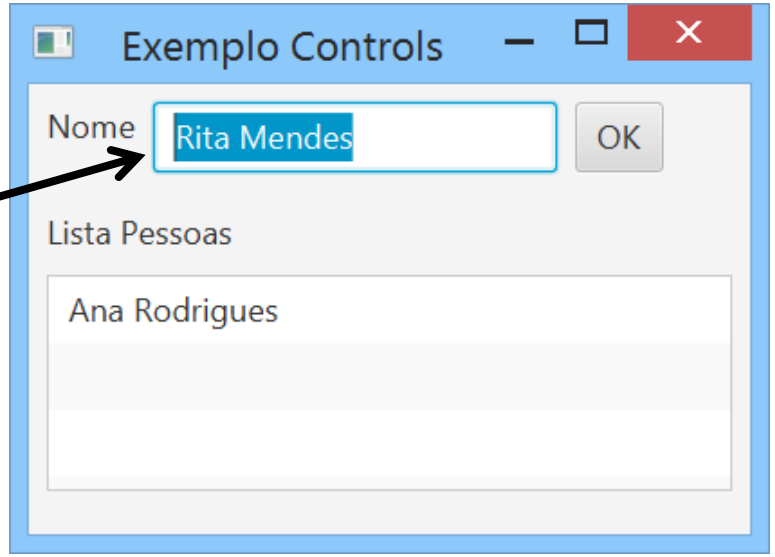
1. Criar um objeto do tipo **`TextField`**.

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

2. Para ler o conteúdo introduzido num **`TextField`** usa-se o método **`getText()`**;

```
String nome= textFieldNome.getText();
```

`TextField`

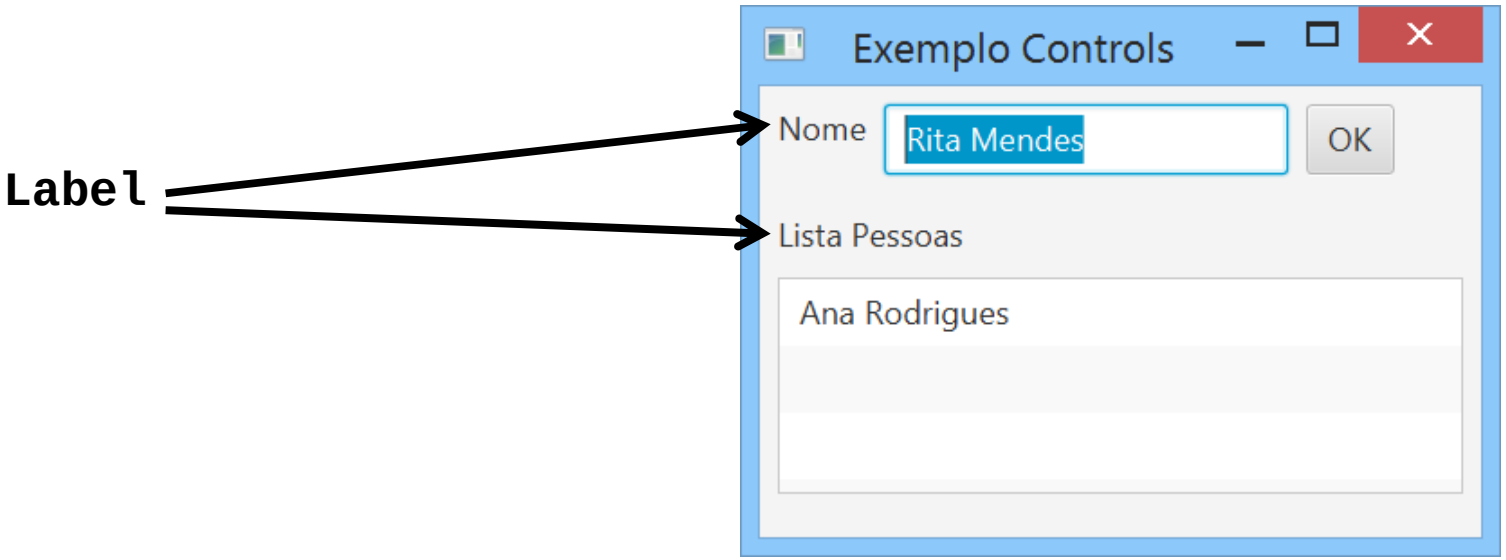


JavaFX- Controlos

- **`Label`** – Criação e Utilização

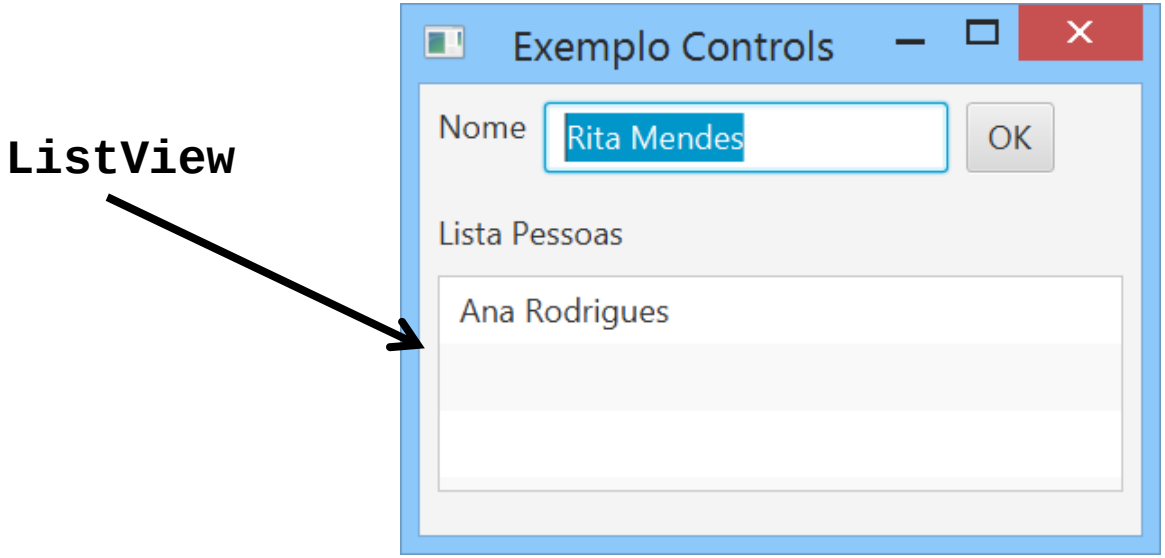
1. Criar um objeto do tipo **`Label`**.

```
Label labelNome = new Label("Nome");
Label labelLista = new Label("ListaPessoas");
```



JavaFX- Controlos

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



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

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

2. Criar uma **`ObservableList`** de **`Strings`** e associá-la à **`ListView`**

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

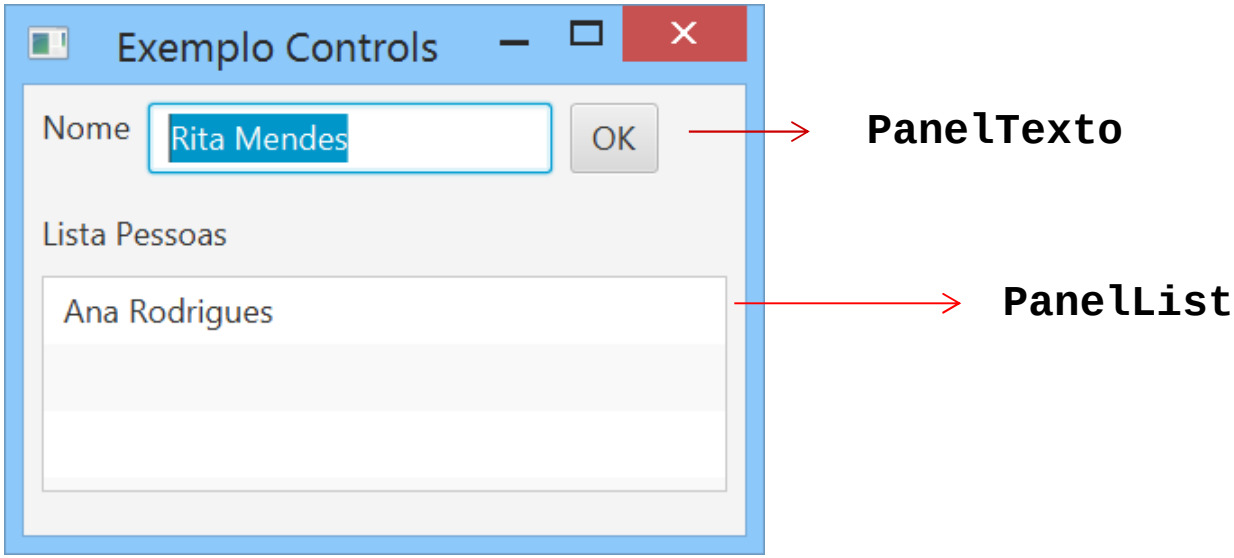
JavaFX- Exemplo Completo (ListView, TextField)

Objetivo:

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

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

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



JavaFX- Exemplo – adicionarPanelTexto

```
private TextField textFieldNome;
```

Atributo na Classe

```
public void adicionarPanelTexto(Pane painel) {

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

        Label labelNome = new Label("Nome");

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

        Button botaoOk = new Button("OK");
        botaoOk.setOnAction(e -> adicionarNomeLista());

        painelTexto.getChildren().add(labelNome);
        painelTexto.getChildren().add(this.textFieldNome);
        painelTexto.getChildren().add(botaoOk);
        painel.getChildren().add(painelTexto);
    }
```

Panel (HBox)

Label

TextField

Button

JavaFX- Exemplo – adicionarPanelList

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

⬅ **Atributos na Classe**

```
public void adicionarPanelList(Pane painel) {
      VBox painelList = new VBox();
      painelList.setPadding(new Insets(10));
      painelList.setSpacing(10);

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

      this.items= FXCollections.observableArrayList( );
      this.listaNomes = new ListView<>();
      this.listaNomes.setPrefWidth(100);
      this.listaNomes.setPrefHeight(120);
      this.listaNomes.setItems(this.items);

      painelList.getChildren().add(labelNomes);
      painelList.getChildren().add(this.listaNomes);
      painel.getChildren().add(painelList);
  }
```

Panel (VBox)

Label

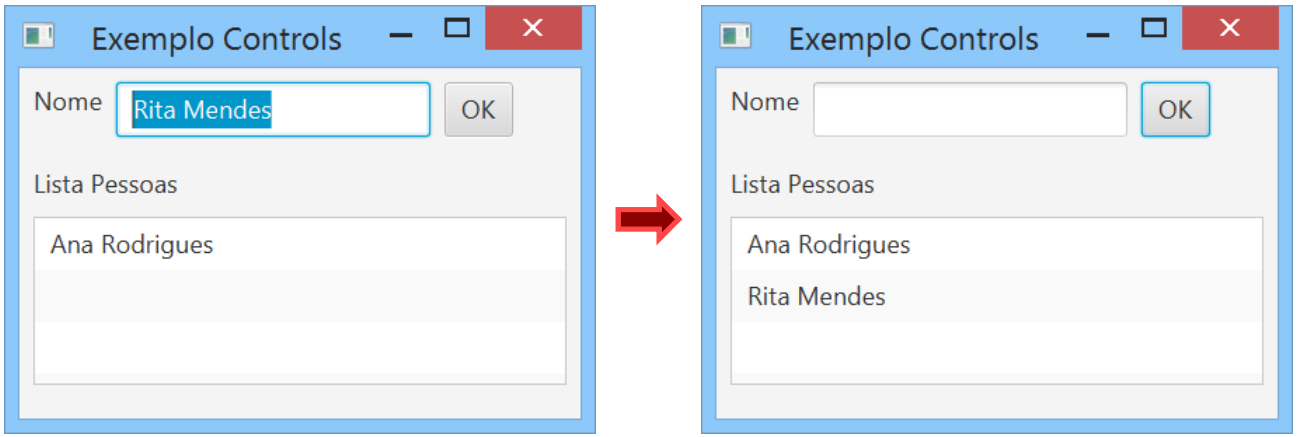
ListView

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 adicionarNomeLista() {
    String nome = this.textFieldNome.getText();
    if (!nome.isEmpty()) {
        this.items.add(nome);
    }

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

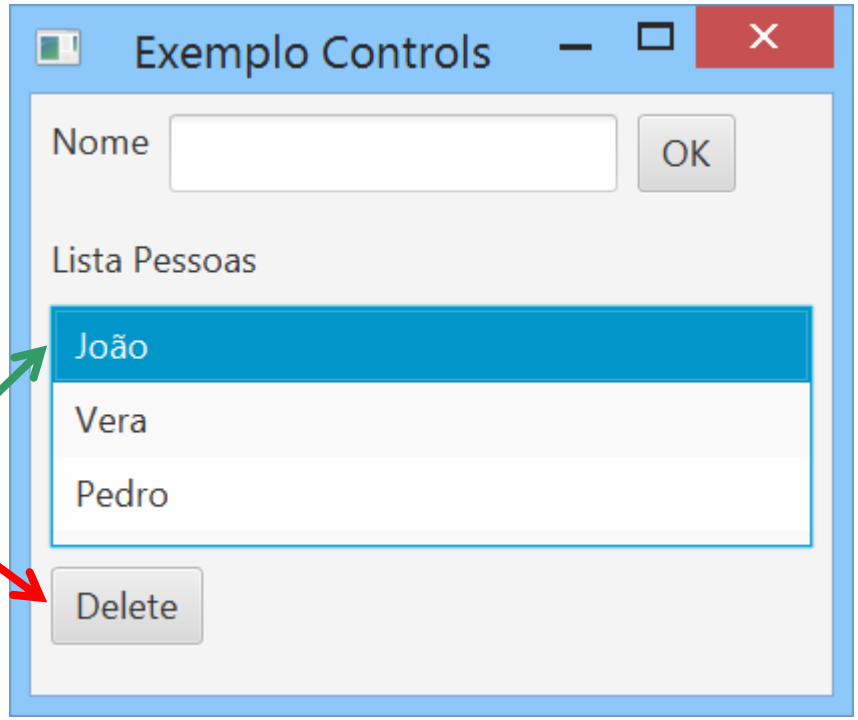


JavaFX- Exemplo – Apagar elementos da lista

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 adicionarPanelList(Pane painel) {

...

Button btnDelete = new Button("Delete");
btnDelete.setOnAction(e -> retirarNomeLista());

...
painelList.getChildren().add(btnDelete);
...

}
```

```
public void retirarNomeLista() {
  int  index;
  index=this.listaNomes.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();
  ```

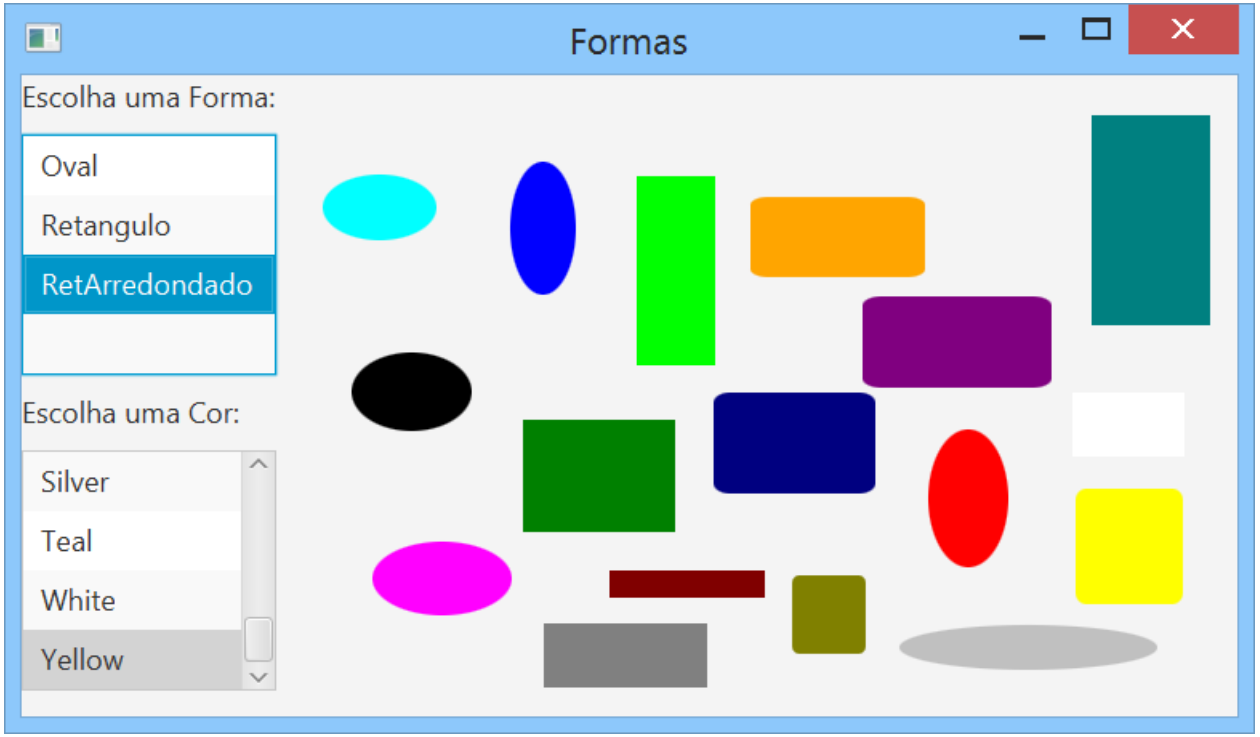
Múltipla

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

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

JavaFX Exemplo II- Criar um desenhador de formas

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



JavaFX Exemplo II- Criar um 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`**, **`fillText`**, **`strokeText`**, etc.

JavaFX Exemplo II- 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 Forma {
 Oval, Retangulo, RetArredondado;
//Foi quebrada a norma de escrita em MAIÚSCULAS para não
//termos de rescrever o toString para ficar "bonitinho"

 public void desenharForma(GraphicsContext graphicsContext,
                           double x, double y,
                           double largura, double altura,
                           double arredondamento) {
   switch (this) {
     case Oval:
         graphicsContext.fillOval(x, y, largura, altura);
         break;
     case Retangulo:
         graphicsContext.fillRect(x, y, largura, altura);
         break;
     case RetArredondado:
         graphicsContext.fillRoundRect(x, y, largura,
altura, largura/arredondamento, altura/arredondamento);
         break;
   }
 }

}
```

JavaFX Exemplo II- 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 **NOMES_CORES** contém as **String** com os nomes de algumas cores standard (as originais em HTML)

```
public class SeletorFormaCor extends VBox {

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

  private ListView<Forma> formas;
  private ListView<String> cores;

  public SeletorFormaCor() {
    Label rotuloFormas = new Label("Escolha uma Forma:");
    Label rotuloCores = new Label("Escolha uma Cor:");
    formas = criarListView(Forma.values());
    cores = criarListView(SeletorFormaCor.NOMES_CORES);
    getChildren().addAll(rotuloFormas,formas,rotuloCores,
cores);
    setSpacing(SeletorFormaCor.ESPACAMENTO);
  }
```

JavaFX Exemplo II- Seletor de Cores e Formas

- Criar um método genérico (tipo de elementos), que recebe um **`array`** com os elementos 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 II- Área de desenho

Criar um **`Canvas`** que tem definido os seus atributos (**`SeletorFormaCor`**, recebido como argumento, e as coordenadas iniciais).

Criar, ainda, os seus **`EventHandler`** de rato carregado (início da figura) ou levantado (fim da figura): **`setOnMousePressed`** e **`setOnMouseReleased`**

```
public class AreaDesenho extends Canvas {

  private static final double LARGURA = 600;
  private static final double ALTURA = 400;
  private static final double ARREDONDAMENTO = 5;

  private SeletorFormaCor seletorFormaCor;
  private double xInicial;
  private double yInicial;

  public AreaDesenho(SeletorFormaCor seletorFormaCor) {
    super(AreaDesenho.LARGURA, AreaDesenho.ALTURA);
    this.seletorFormaCor = seletorFormaCor;
    setOnMousePressed(this::fixarInicio);
    setOnMouseReleased(this::desenharForma);
  }
```

Expressões lamba versus referenciação de métodos

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

  `e -> fixarInicio(e)` ou **`e -> desenharForma(e)`**

- Que tem uma assinatura semelhante à esperada:

  `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::fixarInicio` ou **`this::desenharForma`**

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

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

JavaFX Exemplo II- Área de desenho

- O método associado ao **`setOnMousePressed`** armazena as coordenadas do ponto onde começou a forma a desenhar-se

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

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

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

  GraphicsContext graphicsContext = getGraphicsContext2D();
  graphicsContext.setFill(seletorFormaCor.corEscolhida());
  Forma formaEscolhida = seletorFormaCor.formaEscolhida();
  formaEscolhida.desenharForma(graphicsContext, x, y, largura, altura,
AreaDesenho.ARREDONDAMENTO);
 }
}
```

JavaFX Exemplo II- desenharForma

- O método **`desenharForma`**, 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 (**`corEscolhida`**);
- Executa o método **`desenharForma`** da forma escolhida, obtida com o método **`formaEscolhida`**, passando o **`GraphicsContext`** (obtido por **`getGraphicsContext2D`**) da **`Canvas`** e os valores calculados previamente.

JavaFX Exemplo II- método start

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

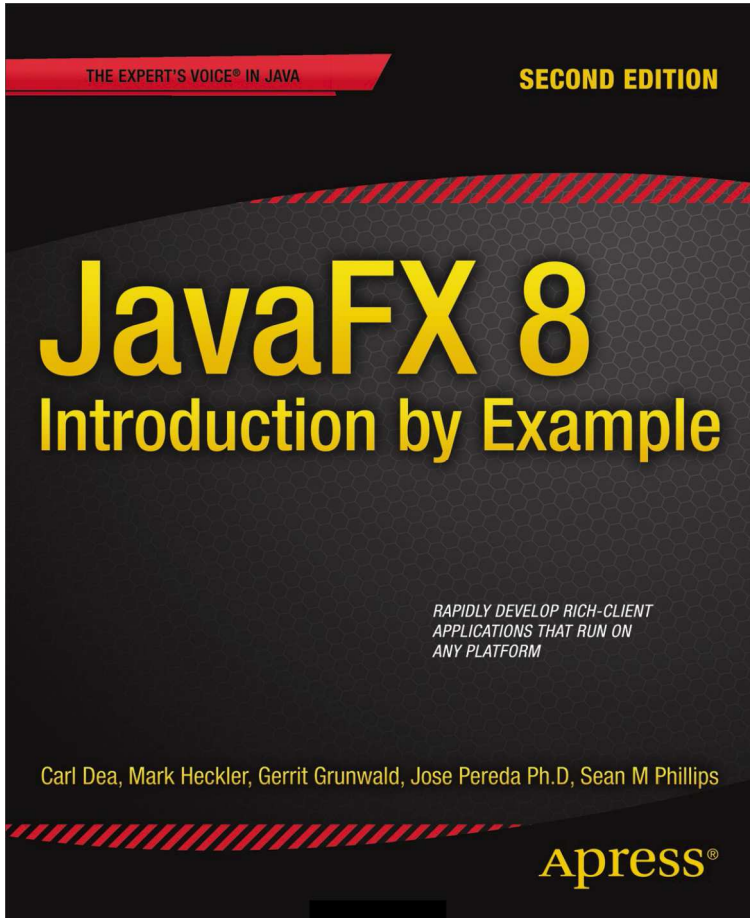
```
public class ExemploCanvas extends Application {

  @Override
  public void start(Stage primaryStage) {
    SeletorFormaCor seletorFormaCor = new SeletorFormaCor();
    HBox root = new HBox();
    root.getChildren().add(seletorFormaCor);
    root.getChildren().add(new AreaDesenho(seletorFormaCor));
    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