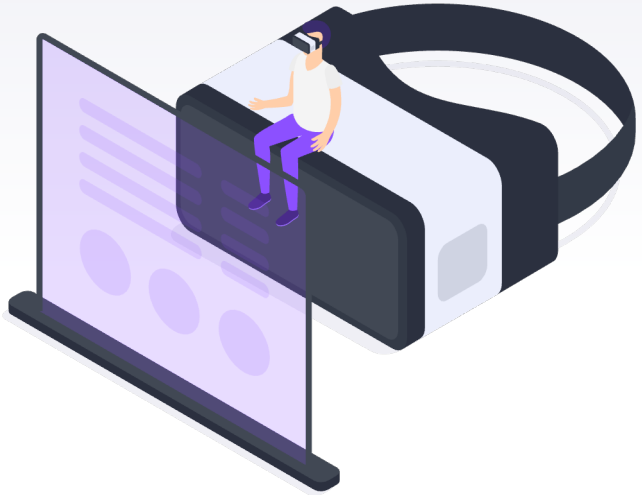
Programação Orientada por Objetos

Introdução ao JavaFX



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

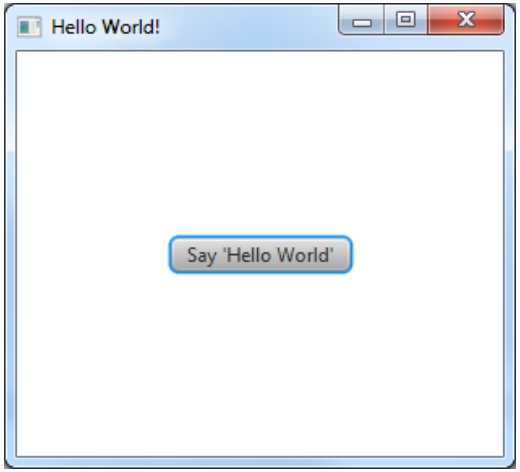
- Estrutura de um programa
  - Classes
    - `Application`
    - `Stage`
    - `Scene`
    - `Node` (com exemplos de Shape e UI Controls)
    - `Group`
- Textos e Fontes
  - A classe `javafx.scene.text.Text`
  - A classe `javafx.scene.text.Font`
- Figuras Geométricas
  - A classe `javafx.scene.shape.Shape` e suas classes derivadas

Estrutura de um Programa

- Introdução ao JavaFX

JavaFX- Uma Aplicação Básica

- Aplicação “Hello World:
  - Criada pelo Netbeans.



JavaFX- Uma Aplicação

- A classe **Application**

javafx.application.Application

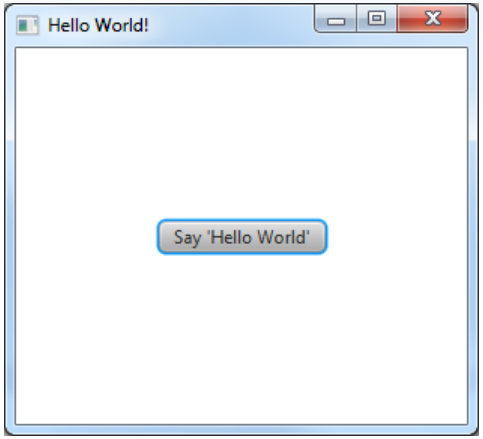
- Oferece as funcionalidades relativas ao ciclo de vida da aplicação tais como lançar e parar a aplicação durante o seu funcionamento.
- Lança os componentes da interface gráfica com o utilizador do JavaFX.
- A classe que contém o método **main** é subclasse de **Application**.
- No método **main** lançamos a aplicação em JavaFX passando os argumentos para o método **Application.launch()**
- Quando a aplicação tiver sido inicializada a infraestrutura do JavaFX invocará, ela própria, o método **Application.start()**.

```
import javafx.application.Application;
// outras clausulas "import" omitidas

public class HelloWorld extends Application {
  public static void main(String[] args) {
    launch(args);
  }

  @Override
  public void start(Stage primaryStage) {
   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);
   Scene scene = new Scene(root, 300, 250);
   primaryStage.setTitle("Hello World!");
   primaryStage.setScene(scene);
   primaryStage.show();
   }
}
```

JavaFX- Uma Aplicação




```
import javafx.application.Application;
// outras clausulas "import" omitidas

public class HelloWorld extends Application {
  public static void main(String[] args) {
    launch(args);
  }

  @Override
  public void start(Stage primaryStage) {
   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);
   Scene scene = new Scene(root, 300, 250);
   primaryStage.setTitle("Hello World!");
   primaryStage.setScene(scene);
   primaryStage.show();
   }
}
```

JavaFX- Um Palco

- A classe **Stage**

  javafx.stage.Stage

- Um objeto da classe **Stage** é equivalente a uma janela da aplicação e pode ser encarado como um painel capaz de conter vários objetos de diversos tipos (gráficos ou não).
- O nome provém de uma analogia com uma peça de Teatro apresentada num palco (*Stage*).
- Ao ser invocado pela infraestrutura do JavaFX, o método **start()** recebe um objeto da classe **Stage** em que podemos redefinir diferentes propriedades e no qual inserimos diferentes objetos.
- Uma vez construído o palco, "Abrimos a Cortina" com o método **show()**.

```
import javafx.application.Application;
// outras clausulas "import" omitidas

public class HelloWorld extends Application {
  public static void main(String[] args) {
    launch(args);
  }

  @Override
  public void start(Stage primaryStage) {
   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);
   Scene scene = new Scene(root, 300, 250);
   primaryStage.setTitle("Hello World!");
   primaryStage.setScene(scene);
   primaryStage.show();
   }
}
```

JavaFX- Um ou Mais Atos

- A classe **Scene**

  javafx.scene.Scene

- Um ato (**Scene**) contém os elementos gráficos que irão ser mostrados no palco (**Stage**).
- É num objeto da classe **Scene** (num ato) que o "enredo" terá lugar.
- Para isso temos de definir qual o ato (**Scene**) que tem lugar no palco (**Stage**) com o método **Stage.setScene()** que aceita como argumento um objeto da classe **Scene**.

```
import javafx.application.Application;
// outras clausulas "import" omitidas

public class HelloWorld extends Application {
  public static void main(String[] args) {
    launch(args);
  }

  @Override
  public void start(Stage primaryStage) {
   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);
   Scene scene = new Scene(root, 300, 250);
   primaryStage.setTitle("Hello World!");
   primaryStage.setScene(scene);
   primaryStage.show();
   }
}
```

JavaFX – Nodes (nós)

- A classe **Node**

javafx.scene.Node

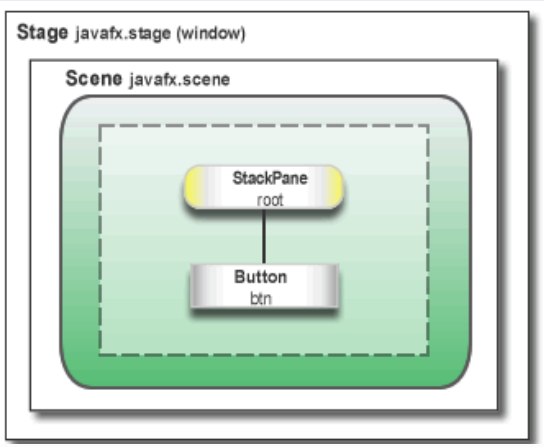
- Todos os elementos gráficos (botões, texto, imagens, etc.) de um ato são nós (**Node**).
- Os nós estão normalmente agrupados numa estrutura hierarquica em que alguns nós podem ter sub-nós (filhos) e outros não.
- Os nós devem ser adicionados à hierarquia

```
import javafx.application.Application;
// outras clausulas "import" omitidas

public class HelloWorld extends Application {
  public static void main(String[] args) {
    launch(args);
  }

  @Override
  public void start(Stage primaryStage) {
   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);
   Scene scene = new Scene(root, 300, 250);
   primaryStage.setTitle("Hello World!");
   primaryStage.setScene(scene);
   primaryStage.show();
   }
}
```

JavaFX – Nodes (Nós)

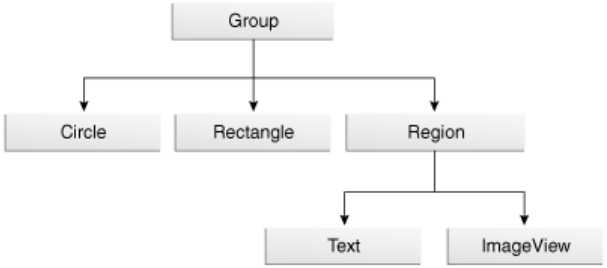


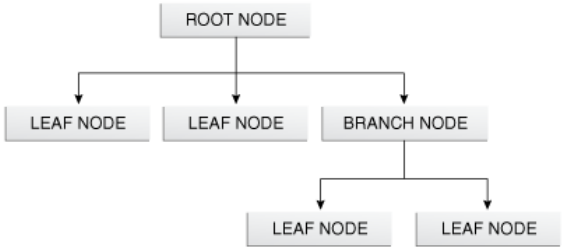
SceneGraph de Nós

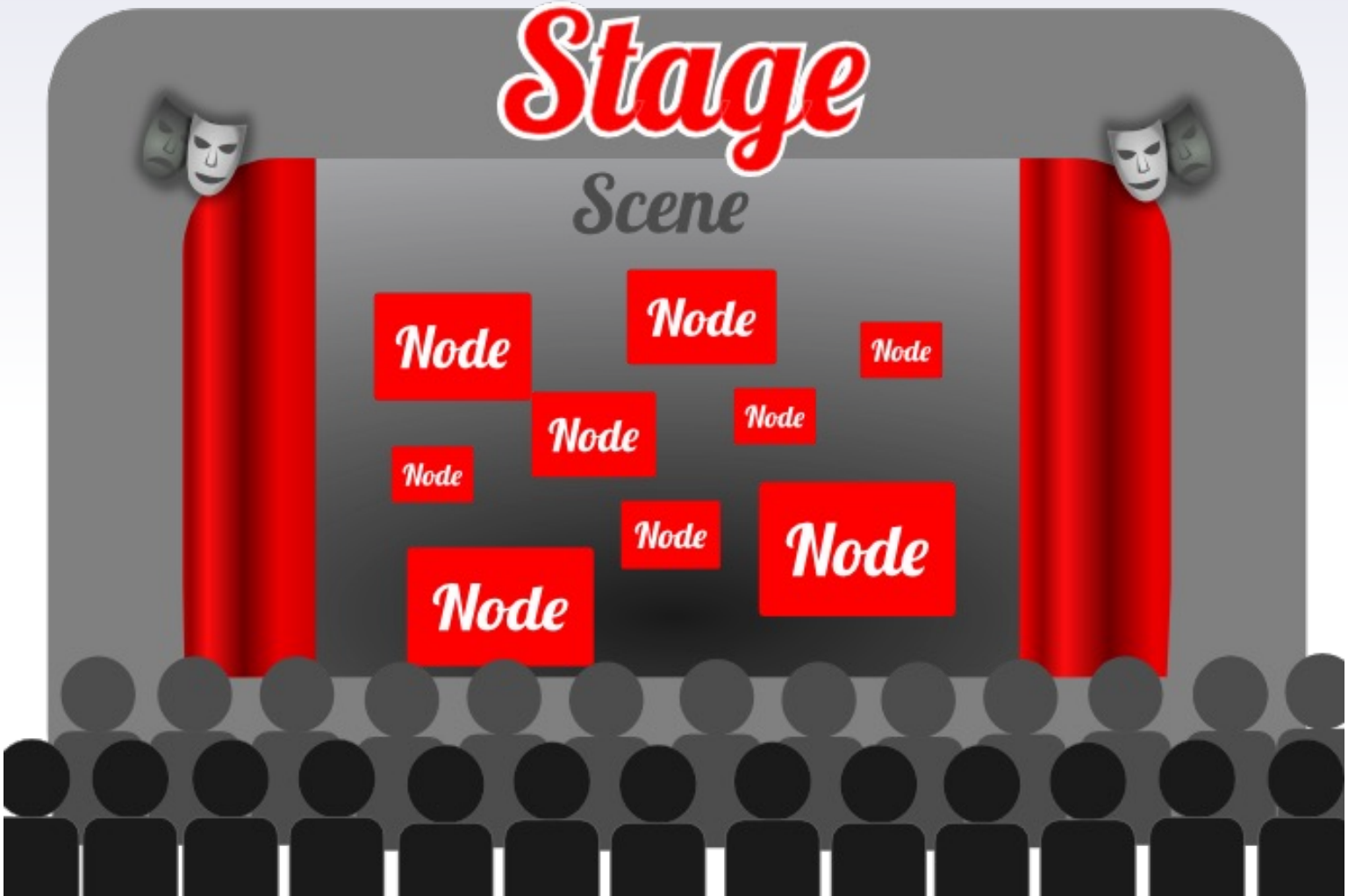
O **SceneGraph** (Grafo de Cena) é o ponto de partida para a construção de uma aplicação em JavaFX e é através dele que são geridas a introdução de dados e a apresentação de elementos gráficos.

É uma árvore de nós (**Node**) que representa todos os elementos visuais da interface gráfica com o utilizador e em que cada nó:

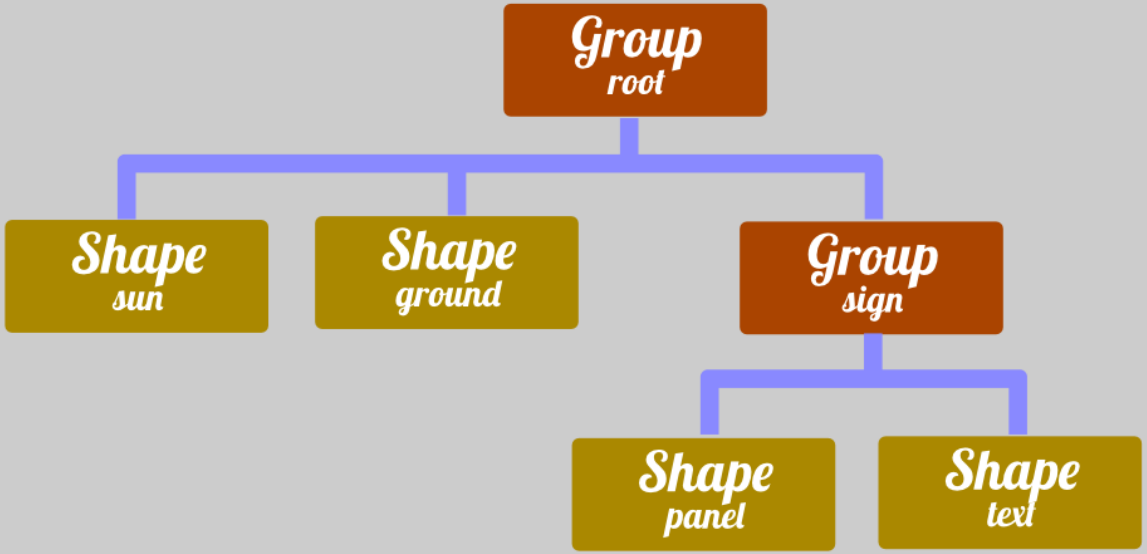
1. Tem zero ou um nó pai – **Em cada grafo de cena só há um nó de topo (sem pai) e é denominado "root" (raiz).**
2. Ou é folha (*leaf* - tem zero sub-nós) – São objetos de classes como **`Rectangle`**, **`Text`**, **`ImageView`**, ou outras que não possam ter nós "filho"
3. Ou é tronco (*branch* - tem um ou mais sub-nós) – São da classe abstrata **`Parent`** (subclasse de **`Node`**), cujas subclasses concretas são **`Group`**, **`Region`** e **`WebView`**, ou subclasses destas.






Stage
Scene
Node
Node
Node
Node
Node
Node
Node
Node
Node

Scene Graph


Group
root
Shape
sun
Shape
ground
Group
sign
Shape
panel
Shape
text

JavaFX – Agrupar Nós (Group)

- A classe **Group**

javafx.scene.Group

- A classe **Group** é uma classe de coleção destinada a agrupar objetos das subclasses de **Node** (são armazenados sem qualquer tipo de posicionamento, como acontece nos painéis, que iremos estudar na próxima aula).
- Podemos trabalhar com objetos da classe **Group** *como se de Listas ou Coleções se tratassem.*
- A sua utilização é simples:

1. Criamos um objeto da classe **Group** (**root**).
2. Adicionamos-lhe os nós que queremos mostrar na cena.
3. Criamos um ato (objeto da classe **Scene**) com o objeto da classe **Group**.

```
import javafx.scene.Group
// outras clausulas "import" omitidas

public class HelloWorld extends Application {
  public static void main(String[] args) {
    launch(args);
  }

  @Override
  public void start(Stage primaryStage) {
    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!");
      }
    });
    btn.setTranslateX(100);
    btn.setTranslateY(100);
    Group root = new Group();
    root.getChildren().add(btn);
    Scene scene = new Scene(root, 300, 250);
    primaryStage.setTitle("Hello World!");
    primaryStage.setScene(scene); primaryStage.show();
  }
}
```

Textos e Fontes

- Introdução ao JavaFX

JavaFX – Como Escrever

- A classe **Text**

javafx.scene.text.Text

- A classe **Text** define um nó para apresentação de texto.
- Os parágrafos são separados por **'\n'** e o texto expandido para ocupar os limites do texto.
- A sua utilização é simples:

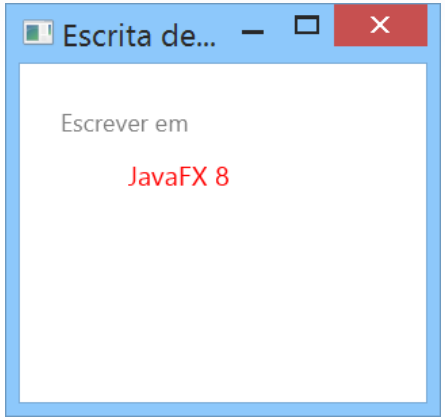
1. Criar um objeto/nó da classe **Text**, eventualmente com a sua posição e o texto.
2. Adicionar o nó ao **SceneGraph** (eventualmente um objeto da classe **Group** ou **Stage**)

```
import javafx.scene.text.Text;
import javafx.scene.text.Font;
// outras clausulas "import" omitidas

public class DrawingText extends Application {
  public static void main(String[] args) {
    launch(args);
    }

  @Override
  public void start(Stage primaryStage) {
    Group root = new Group();
    Text text1 = new Text(30, 50, "Escrever em");
    Text text2 = new Text(80, 90, "JavaFX 8");
    text1.setFill(Color.rgb(0, 0, 0, 0.5));
    text2.setFill(Color.rgb(255, 0, 0, 1.0));
    text2.setFont(new Font(20));
    root.getChildren().add(text1);
    root.getChildren().add(text2);
    Scene scene = new Scene(root, 300, 250);
    primaryStage.setTitle("Escrita de texto");
    primaryStage.setScene(scene);
    primaryStage.show();
   }
}
```

JavaFX – Como Escrever




```
import javafx.scene.text.Text;
import javafx.scene.text.Font;
// outras clausulas "import" omitidas

public class DrawingText extends Application {
  public static void main(String[] args) {
    launch(args);
    }

  @Override
  public void start(Stage primaryStage) {
    Group root = new Group();
    Text text1 = new Text(30, 50, "Escrever em");
    Text text2 = new Text(80, 90, "JavaFX 8");
    text1.setFill(Color.rgb(0, 0, 0, 0.5));
    text2.setFill(Color.rgb(255, 0, 0, 1.0));
    text2.setFont(new Font(20));
    root.getChildren().add(text1);
    root.getChildren().add(text2);
    Scene scene = new Scene(root, 300, 250);
    primaryStage.setTitle("Escrita de texto");
    primaryStage.setScene(scene);
    primaryStage.show();
   }
}
```

JavaFX – Como Escrever

- A classe **Font**

javafx.scene.text.Font

- A classe **Font** permite definir o aspeto do texto.
- No exemplo ao lado criamos três tipos de fontes: com serifa, sem serifa, monoespaçadas usando o método **font()**. Este método procura, entre as fontes disponíveis no sistema a que melhor se adapte às características passadas nos argumentos.
- Note-se que o método **Font.font()** não garante qual o tipo de fonte retornado.

```
import javafx.scene.text.Font;
// outras clausulas "import" omitidas

public class ChangingTextFonts extends Application {
// método main() omitido

  @Override
  public void start(Stage primaryStage) {
    Group root = new Group();
    Scene scene = new Scene(root, 700, 250);
    primaryStage.setTitle("Escrita de texto");
    primaryStage.setScene(scene);
    primaryStage.show();
    Text text1 = new Text(50, 50, "Font Serif RED");
    Font serif = Font.font("Serif", 30);
    text1.setFont(serif);
    text1.setFill(Color.RED);
    root.getChildren().add(text1);
    Text text2 = new Text(50, 100, "Font SanSerif BLUE");
    Font sanSerif = Font.font("SanSerif", 50);
    text2.setFont(sanSerif);
    text2.setFill(Color.BLUE);
    root.getChildren().add(text2);
    Text text3 = new Text(50, 200, "Font Monospaced BLACK");
    Font monoFont = Font.font("Monospaced", 70);
    text3.setFont(monoFont);
    text3.setFill(Color.BLACK);
    root.getChildren().add(text3);
  }
}
```

JavaFX – Como Escrever

JavaFX – Propriedades dos Nós

Propriedades

- Nos exemplos dos anteriores slides deparámo-nos com diversas instruções do tipo **`setXxx()`.**
- Os diferentes tipos de nós de um objeto das classes **`Group`** ou **`Stage`** possuem diferentes propriedades que nos permitem alterar o aspeto e o comportamento dos diferentes nós.

```
primaryStage.setTitle("Hello World!");
primaryStage.setScene(scene);
Button btn = new Button();
btn.setText("Say 'Hello World'");
btn.setTranslateX(100);
btn.setTranslateY(100);
Text text1 = new Text(30, 50, "Escrever em");
text1.setFill(Color.rgb(0, 0, 0, 0.5));
text1.setFont(new Font(20));
Font serif = Font.font("Serif", 30);
text1.setFont(serif);
text1.setFill(Color.RED);
```

As propriedades comuns a todas as subclasses de Node estão descritas em:
http://docs.oracle.com/javase/8/javafx/api/javafx/scene/Node.html

Mas obviamente, cada uma das subclasses de Node acrescenta a esta lista uma série de outras propriedades relevantes para a definição do aspeto e comportamento de cada tipo de nó.

Figuras Geométricas

- Introdução ao JavaFX

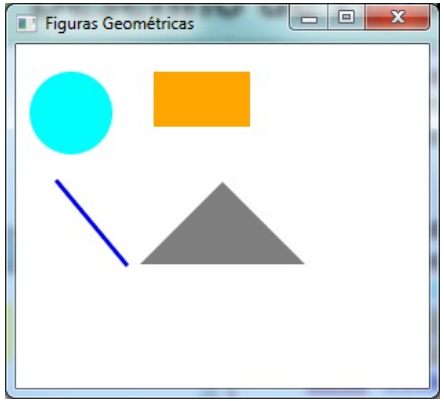
Desenho de Figuras Geométricas

- A classe abstrata **`javafx.scene.shape.Shape`** (que estende **`Node`**) é a super classe de diversas classes que implementam representações gráficas de figuras geométricas: **`Arc`**, **`Circle`**, **`CubicCurve`**, **`Ellipse`**, **`Line`**, **`Path`**, **`Polygon`**, **`Polyline`**, **`QuadCurve`**, **`Rectangle`**, **`SVGPath`**, **`Text`**.
- Esta classe implementa diversos métodos, dos quais se destacam:
  - **`getFill()`** e **`setFill()`** – para lidar com o aspeto do preenchimento da figura
  - **`getStroke()`** e **`setStroke()`** – para lidar com o aspeto da linha envolvente
  - **`getStrokeWidth()`** e **`setStrokeWidth()`** – para lidar com a dimensão da linha envolvente
- Mais informação sobre esta classe e as suas derivadas pode ser obtida em:
  http://docs.oracle.com/javase/8/javafx/api/javafx/scene/shape/Shape.html

Desenho de Figuras Geométricas

Construtores utilizados:

- **Circle(x,y,r,Paint)**
- **Rectangle(x,y,l,a)**
- **Line(x1,y1,x2,y2)**
- **Polygon(x1,y1,x2,y2,…,xn,yn)**




```
public class Figuras extends Application {

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

  @Override
  public void start(Stage primaryStage) {
    Group root = new Group();
    Circle circulo = new Circle(40, 50, 30, Color.AQUA);
    Rectangle retangulo = new Rectangle(100, 20, 70, 40);
    retangulo.setFill(Color.rgb(255, 255, 0));
    Line linha = new Line(30, 100, 80, 160);
    linha.setStroke(Color.BLUE);
    linha.setStrokeWidth(3);
    Polygon poligono = new Polygon(150, 100, 90, 160, 210, 160);
    poligono.setFill(Color.gray(0.5));
    root.getChildren().addAll(circulo, retangulo, linha, poligono);
    Scene scene = new Scene(root, 300, 250);
    primaryStage.setTitle("Figuras Geométricas");
    primaryStage.setScene(scene);
    primaryStage.show();
  }
}
```

Resumindo

- Estrutura de um programa em JavaFX e suas classes base:
  - **`Application`**
    - Lança os componentes da interface gráfica com o utilizador do JavaFX numa "thread" protegida.
  - **`Stage`** (Palco)
    - "Espaço" onde se desenrola a peça de teatro.
    - Equivale a uma janela da aplicação.
  - **`Scene`**
    - Define os vários cenários que queremos apresentar numa janela.
    - Em cada momento definimos o cenário a apresentar com o método **`Stage.setScene()`**.
  - **`Node`**
    - A classe abstrata e base dos nós de um grafo de cena.
  - **`Group`**
    - É uma classe de coleção destinada a agrupar objetos das subclasses de **`Node`**.
- Escrita de texto
  - O texto escreve-se com a classe **`javafx.scene.text.Text`**
  - E define-se a sua apresentação com a classe **`javafx.scene.font.Font`**
- Propriedades
  - Todos os nós possuem "propriedades" que permitem alterar o seu aspeto e comportamento
- Desenho de Figuras Geométricas
  - A classe **`javafx.scene.shape.Shape`** e suas classes derivadas

Leitura Complementar

- Chapter 1 e 2
  - Páginas 1 a 60
- Sobre o JavaFX 8:

http://docs.oracle.com/javase/8/javase-clienttechnologies.htm

  - O que é o JavaFX 8: http://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-overview.htm#JFXST784
  - Iniciação ao JavaFX 8: http://docs.oracle.com/javase/8/javafx/get-started-tutorial/get_start_apps.htm#JFXST804
  - Sobre a arquitetura do JavaFX 8: http://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-architecture.htm#JFXST788
- Tutorial sobre o JavaFX (Java 8): http://docs.oracle.com/javase/8/javafx/JFXST.pdf