

The logo for Oracle Academy is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by a thin black border, with dark gray horizontal bars at the top and bottom.

ORACLE

Academy

Java Foundations

9-1

Introdução ao JavaFX

ORACLE
Academy



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Objetivos

- Esta lição abrange os seguintes objetivos:
 - Criar um projeto JavaFX
 - Explicar os componentes do projeto JavaFX padrão
 - Descrever os diferentes tipos de Nós e Painéis
 - Explicar os conceitos de Scene Graph, Root Node, Scenes e Stages



Está Quase Chegando a Hora dos Exames Finais!

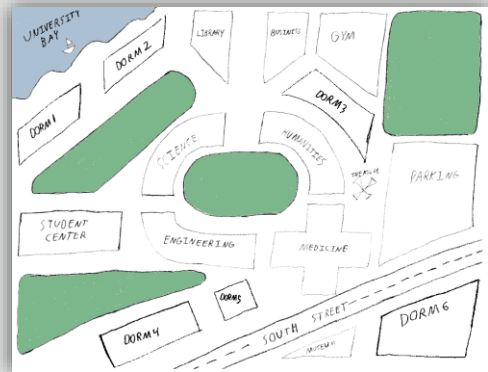
- É importante estudar
- Você gosta de estudar com um amigo?
 - Seus amigos ficam em outros alojamentos?
 - Qual é o melhor lugar para encontrar seus amigos?
 - Qual é o ponto localizado mais centralmente no campus?

Obrigado por me
lembrar...



O JavaFX Pode Ajudar

- O JavaFX é usado para criar aplicativos da interface gráfica do usuário (GUI)
- GUI: interface gráfica do usuário
- Uma interface gráfica do usuário permite ver a resposta em um mapa



Exercício 1



- O material de referência deste slide `CampusMap.jar` demonstra um aplicativo completo que foi criado usando o JavaFX
- Reproduza `CampusMap.jar`
- Cada quadrado é alinhado com o dormitório correto no mapa
- A população de cada dormitório é ajustada clicando e arrastando o texto abaixo de cada quadrado
- Observe as alterações nos seguintes pontos centrais:
 - Todos os estudantes em todos os dormitórios
 - Um grupo de estudo com três amigos que vivem nos Dormitórios 1, 2 e 4

Mas Este Não É o meu Campus!

- Você está certo
- Seria melhor se o programa usasse
 - O mapa do campus da escola
 - Os nomes dos alojamentos da escola
 - Os ocupantes dos alojamentos
 - E seu grupo de amigos
- Este é o conjunto de problemas desta seção
- A Seção 9 analisa tudo o que você precisará para criar novamente o programa

Java Puzzle Ball também é um aplicativo JavaFX, mas seria preciso muito tempo para criá-lo novamente.

Exercício 2

- Crie um projeto JavaFX
 - O material de referência desta lição forneceu instruções para criar um projeto JavaFX no NetBeans ou no Eclipse
 - Se você estiver usando outro IDE, consulte a documentação para saber como fazer isso
- Experimente o programa.
- Você consegue fazer estas alterações?
 - Altere o rótulo do botão
 - Altere o que é impresso quando o botão é clicado
 - Crie outro botão e exiba os dois botões
 - Altere o tamanho padrão da janela do aplicativo

O Projeto JavaFX Padrão

```
public class JavaFXMain extends Application {  
  
    @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);  
    }  
}
```

Continua no próximo slide...

O Projeto JavaFX Padrão

... continuação

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

primaryStage.setTitle("Hello World!");
primaryStage.setScene(scene);
primaryStage.show();
} // fim do método start

public static void main(String[] args) {
    launch(args);
} // fim do método main
} // fim da classe JavaFXMain
```

Dois Métodos: start() e main()

- start() é o ponto de entrada de todos os aplicativos JavaFX
 - Considere-o como o método main do JavaFX

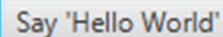
```
public void start(Stage primaryStage) {  
    ...  
} //fim do método start
```

- main() continua sendo obrigatório em seus programas
 - Ele inicializa o aplicativo JavaFX

```
public static void main(String[] args) {  
    launch(args);  
} //fim do método main
```

Os Botões São Objetos

- Os buttons são como qualquer outro objeto
 - Eles podem ser instanciados
 - Eles contêm campos
 - Eles contêm métodos



```
public void start(Stage primaryStage) {  
    Button btn = new Button();  
    btn.setText("Say 'Hello World'");  
    ...  
} // fim do método start
```

- Com base nesse código, podemos afirmar...
 - Os buttons podem conter um campo de texto
 - Os buttons contêm um método para alterar o campo de texto

Os Botões São Nós

- Alguns desses campos e métodos são projetados para armazenar e manipular propriedades visuais:
 - `btn.getText()`
 - `btn.setMinHeight()`
 - `btn.setLayoutX()` **//definir a posição de x**
 - `btn.setLayoutY()` **//definir a posição de y**
 - `btn.isPressed()` **//ele é pressionado?**
- Objetos como esse são denominados Nós do JavaFX

Nós

- Existem muitos tipos de Nós do JavaFX:

Say 'Hello World'

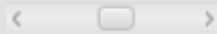
Button



Rectangle



PieChart



ScrollBar

Dorm 6:
200

Text



ImageView

- Os objetos visuais que você criará muito provavelmente...
 - Serão um Nó ou
 - Incluirão um Nó como um campo

Interação do Nó

- O código a seguir ajuda a tratar a interação do Botão:

```
public void start(Stage primaryStage) {  
    ...  
    btn.setOnAction(new EventHandler<ActionEvent>() {  
        @Override  
        public void handle(ActionEvent event) {  
            System.out.println("Hello World!");  
        } //fim do método handle  
    }); //fim setOnAction  
    ...  
} //fim do método start
```

- Isso denomina-se "**classe interna anônima**"

- A sintaxe parece bagunçada?
- Expressões Lambda do Java SE 8 são uma alternativa elegante
- Veremos as expressões Lambda posteriormente nesta seção

Criando Nós

- Os nós são instanciados como qualquer outro objeto Java:

```
public void start(Stage primaryStage) {  
    Button btn1 = new Button();  
    Button btn2 = new Button();  
    btn1.setText("Say 'Hello World'");  
    btn2.setText("222");  
    ...  
} //fim do método start
```

- Depois que você instancia um Nó:
 - Ele existe e a memória é alocada para armazenar o objeto
 - Seus campos podem ser manipulados, e os métodos podem ser chamados
 - Mas pode ser que não sejam exibidos...

Pelo menos ainda não...

Exibindo Nós

- Existem algumas etapas para a exibição de um nó

```
public void start(Stage primaryStage) {  
    Button btn1 = new Button();  
    Button btn2 = new Button();  
    btn.setText("Say 'Hello World'");  
    btn.setText("222");  
    StackPane root = new StackPane();  
    root.getChildren().add(btn1);  
    root.getChildren().add(btn2);  
    ...  
} // fim do método start
```

- Primeiro, adicione cada Nó ao Root Node
 - Normalmente, ele denomina-se root
 - É muito parecido com uma ArrayList de todos os Nós

Adicionando Nós ao Root Node

- Você poderia adicionar cada Nó separadamente:



```
root.getChildren().add(btn1);  
root.getChildren().add(btn2);  
root.getChildren().add(btn3);
```

- Ou poderia adicionar muitos Nós de uma só vez:



```
root.getChildren().addAll(btn1, btn2, btn3);
```

Adicionando Nós ao Root Node

- Mas não adicione o mesmo Nó mais de uma vez.
 - Isso causa um erro do compilador:



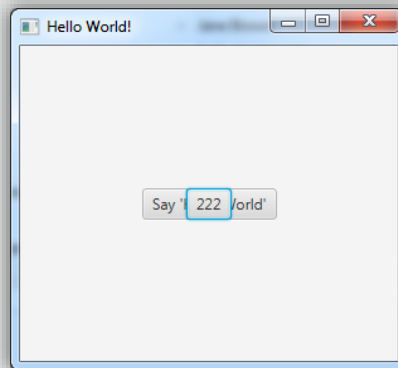
```
root.getChildren().add(btn1);  
root.getChildren().add(btn1);
```

Root Node StackPane

- O Root Node neste exemplo é um StackPane

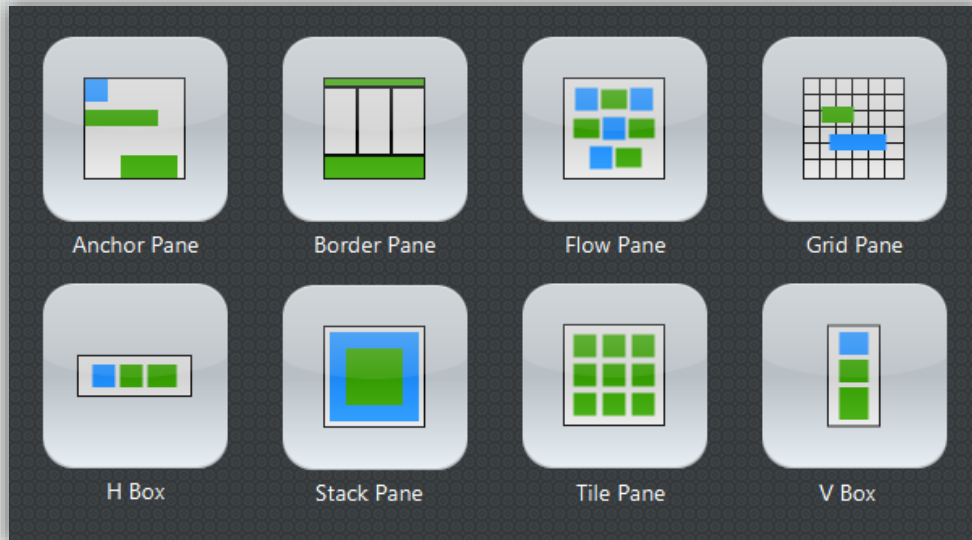
```
StackPane root = new StackPane();  
root.getChildren().addAll(btn1, btn2);
```

- O StackPane empilha Nós uns sobre os outros
- Mas os botões pequenos poderiam ficar muito embaixo e inacessíveis



Painéis como Nós Raiz

- Cada Painel determina o layout dos Nós



Programando Diferentes Painéis como Root Nodes

- É fácil projetar o nó raiz como um painel diferente
- Basta especificar um tipo de referência e um tipo de objeto diferentes

Altere isto

E isto

```
StackPane root = new StackPane();  
root.getChildren().addAll(btn1, btn2);
```

```
TilePane root = new TilePane();  
root.getChildren().addAll(btn1, btn2);
```

```
VBox root = new VBox();  
root.getChildren().addAll(btn1, btn2);
```

Exercício 3

- Edite seu projeto JavaFX atual
 - Vamos fazer uma pequena experiência
- Depois de adicionar um botão ao Root Node, tente alterar sua posição
 - `btn1.setLayoutY(100);`
- A posição de um botão mudaria se o Root Node não fosse um StackPane?
- Tente estas alternativas:
 - TilePane
 - VBox
 - Group

Root Node Group

- Um Group permite que você coloque Nós em qualquer local

```
Group root = new Group();  
root.getChildren().addAll(btn1, btn2);  
btn1.setLayoutY(100);
```

- Um painel pode restringir o local em que os Nós são posicionados
 - Você não poderia movê-los mesmo que desejasse
 - Você não poderia clicar e arrastar um nó que estivesse bloqueado em um painel

```
StackPane root = new StackPane();  
root.getChildren().addAll(btn1, btn2);  
btn1.setLayoutY(100); //Não tem efeito
```


Um Group pode Conter um Painel

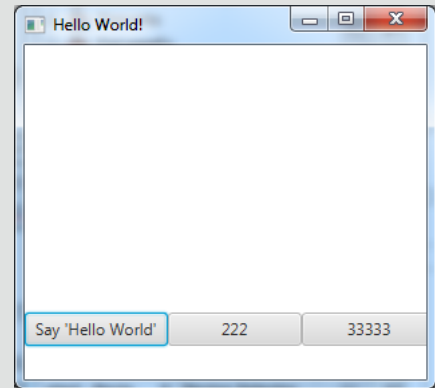
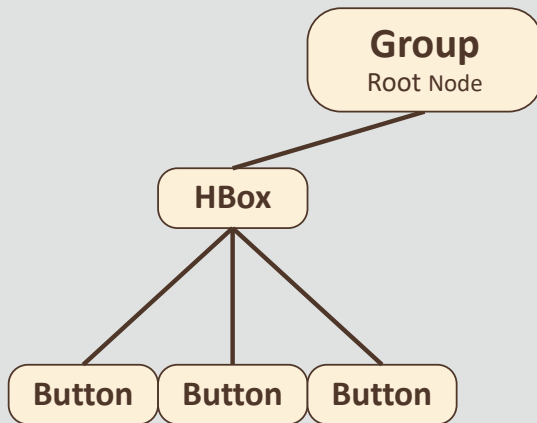
- Os Painéis também são Nós
 - É possível adicionar qualquer nó ao Root Node
- Um Painel pode ser uma boa opção para armazenar botões, caixas de diálogo de entrada de texto e outros elementos da interface gráfico do usuário
 - Você praticamente não pode mover Nós individuais em um Painel
 - Mas você pode mover todo o Painel em um Group
 - Mova o Painel como faria com qualquer outro Nó

Exercício 4

- Edite seu projeto JavaFX atual
 - É hora de experimentar
- Você consegue imaginar como fazer o seguinte?
 - Crie um painel HBox e adicione vários botões a ele
 - Adicione um painel HBox a um Root Node Group
 - Posicione o HBox próximo à parte inferior da janela

O Scene Graph do JavaFX

- A forma como você decide adicionar nós pode ser ilustrada como um Scene Graph
 - O Root Node contém um Hbox
 - O HBox funciona como um recipiente para botões



ORACLE
Academy

JFo 9-1
Introdução ao JavaFX

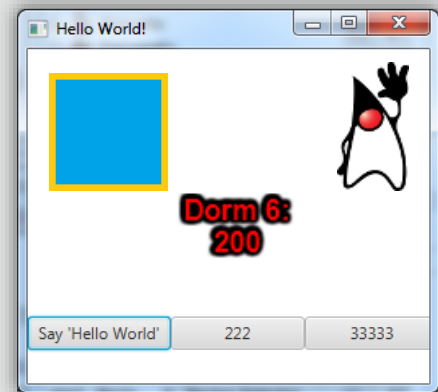
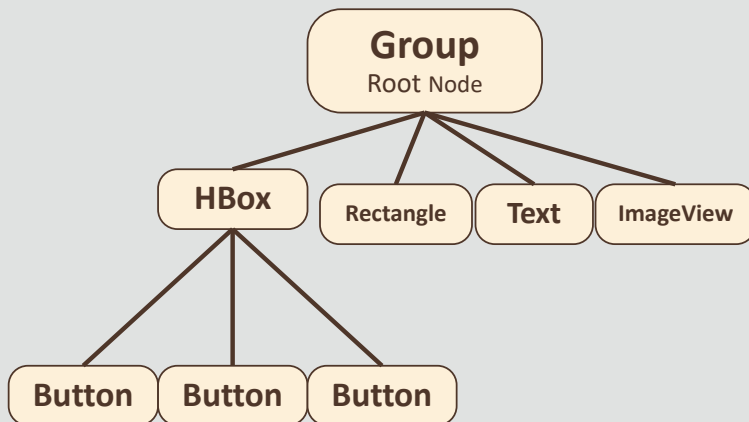
Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

27

Observação para os Instrutores: as imagens nesses dois slides devem estar na mesma posição.

O Scene Graph

- O HBox mantém a interface gráfica do usuário organizada e bem localizada
- O restante da janela poderia ser usado para outros Nós



ORACLE
Academy

JFo 9-1
Introdução ao JavaFX

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

28

Observação para os Instrutores: as imagens nesses dois slides devem estar na mesma posição.

A Scene e o Stage

- Se olharmos para o restante do programa JavaFX padrão, perceberemos duas outras coisas:
 - Uma Scene (que contém o Root Node)
 - Um Stage (que contém a Scene)

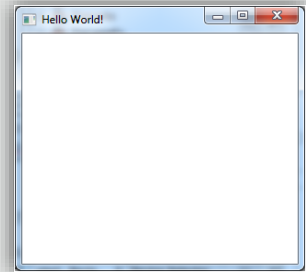
```
public void start(Stage primaryStage) {  
    ...  
    Scene scene = new Scene(root, 300, 250);  
  
    primaryStage.setTitle("Hello World!");  
    primaryStage.setScene(scene);  
    primaryStage.show();  
} //fim do método start
```

O que é a Scene?

- Existem algumas propriedades notáveis que descrevem uma Scene:
 - Scene Graph
 - A Scene é o recipiente de todo o conteúdo no Scene Graph do JavaFX
 - Tamanho
 - É possível definir a largura e a altura da Scene
 - Plano de fundo
 - O plano de fundo pode ser definido como uma Cor ou uma Imagem de plano de fundo
 - Informações do Cursor
 - A Scene pode detectar eventos do mouse e tratar propriedades do cursor
- ```
Scene scene = new Scene(Root Node root, largura 300, altura 250, plano de fundo Color.BLACK);
```

# O que é o Stage?

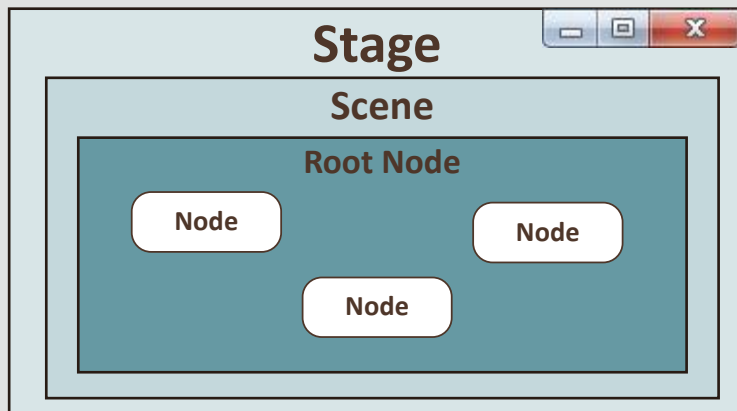
- Considere o Stage como uma janela do aplicativo
- Veja a seguir duas propriedades notáveis do Stage:
  - Título
    - É possível definir o título do Stage
  - Scene
    - O Stage contém uma Scene



```
primaryStage.setTitle("Hello World!");
primaryStage.setScene(scene);
primaryStage.show();
```

# Animação da Hierarquia

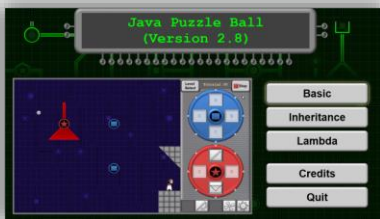
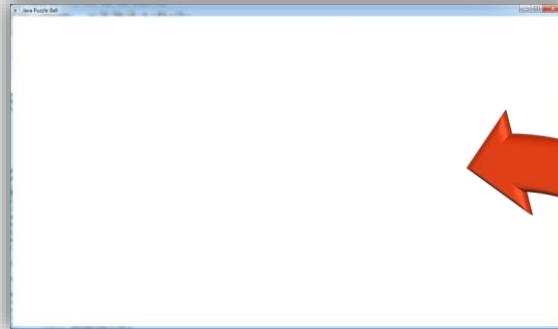
- Um Stage é um recipiente de nível superior
- Ele contém uma Scene
- Uma Scene contém um Root Node
- O Root Node contém outros Nós





# Muitas Scenes, Um Stage

- É possível alternar qualquer Scene em um único Stage



**ORACLE**  
Academy

JFo 9-1  
Introdução ao JavaFX

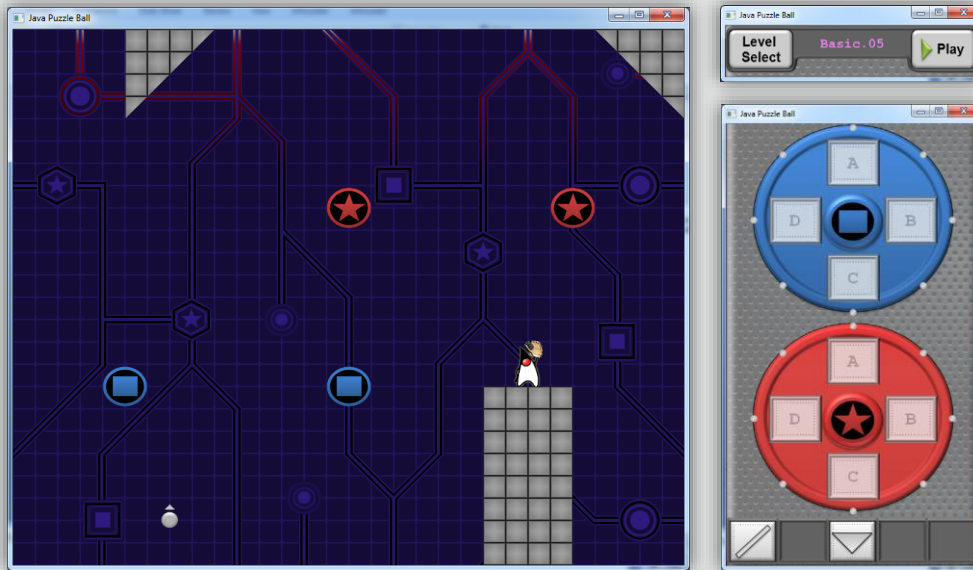
Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

33

Usamos um único Stage no Java Puzzle Ball. Se usássemos muitos Stages, pareceria desorganizado ver janelas abrindo e fechando quando você navegasse pelos menus.

# Muitas Scenes, Muitos Stages

- Também é possível criar muitos Stages



**ORACLE**  
Academy

JFo 9-1  
Introdução ao JavaFX

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

34

Isso normalmente é feito com ferramentas. Mas nem tanto com jogos.

# Resumo

- Esta lição abrange os seguintes objetivos:
  - Criar um projeto JavaFX
  - Explicar os componentes do projeto JavaFX padrão
  - Descrever os diferentes tipos de Nós e Painéis
  - Explicar os conceitos de Scene Graph, Root Node, Scenes e Stages



