

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 two horizontal dark gray bars, one at the top and one at the bottom.

ORACLE

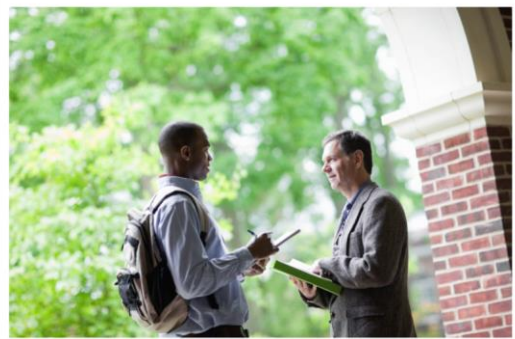
Academy

Java Foundations

9-3

Elementos Gráficos, Áudio e Eventos do Mouse

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.

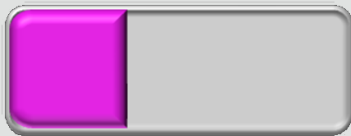
Objetivos

- Esta lição abrange os seguintes objetivos:
 - Criar e usar uma imagem e a ImageView do JavaFX
 - Criar e usar o áudio do JavaFX
 - Criar e usar Eventos do Mouse
 - Entender as expressões Lambda em aplicativos da interface gráfica do usuário (GUI)



Usando Seus Próprios Elementos Gráficos

- O JavaFX pode fornecer elementos de interface de usuário, formas e texto
 - Mas, se você tiver um talento para arte, poderá usar seus próprios elementos gráficos, em vez dos que o JavaFX fornece
- Por exemplo:



- A arte no botão de seleção de nível não foi criada pelo JavaFX
- Mas usamos o JavaFX para adicionar números de nível, texto e elemento gráfico do Duke de forma procedural

Image e ImageView do JavaFX

- Image é um objeto que descreve o local de um arquivo gráfico (.png, .jpg, .gif...)

```
Image image;  
String imagePath = "Images/Fan1.png";  
image = new Image(getClass().getResource(imagePath).toString);
```

- ImageView é o Nó propriamente dito
 - Chamar seu construtor requer um argumento Imagem

```
ImageView imageView = new ImageView(image);
```

- Uma ImageView também contém as mesmas propriedades que qualquer outro nó: posição de x, posição de y, largura, altura...

Você também precisará importar `javafx.scene.image.Image` e `javafx.scene.image.ImageView`.

Por que É Preciso Ter uma Imagem e uma ImageView?

- Uma grande vantagem é uma animação
 - As imagens podem ser alternadas na mesma ImageView
- O Ventilador do Java Puzzle Ball tira vantagem dessa possibilidade
 - O ventilador alterna duas imagens quando está circulando



- Os botões personalizados também beneficiam-se
 - Você poderia usar diferentes imagens para os botões dependendo do estado de cada botão:
 - O mouse está sendo arrastado sobre o botão?
 - O usuário está clicando no botão?

Dicas de ImageView

- Como criar Imagens:

```
Image image1 = new  
Image(getClass().getResource("Images/fan1.png").toString());  
Image image2 = new  
Image(getClass().getResource("Images/fan2.png").toString());
```

- Como criar uma ImageView:

```
ImageView imageView = new ImageView(image1);
```

- Como alternar uma Imagem em uma ImageView:

```
imageView.setImage(image2);
```

– imageView retém suas propriedades, como posicionamento

Lembre-se de importar
`javafx.scene.image.Image;` e
`javafx.scene.image.ImageView;`

Criando Objetos com Propriedades de Nó

- Até o momento, escrevemos todo o código do JavaFX no método `start()`
 - Isso é semelhante ao início do curso, quando a maioria do código foi escrita no método `main()`
- O código orientado a objetos não deve ser escrito dessa maneira
 - Em vez disso, os objetos devem ter campos `Nó`
- Os métodos `start()` e `main()` têm como objetivo serem drivers

Exemplo: a Classe Goal

- Campos

- private Image dukeImage;
 - private ImageView dukeImageView;

- Construtor

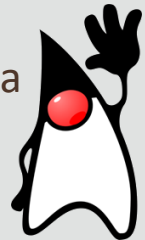
- Escolha argumentos para as posições de x e y
 - Atribui a imagem à respectiva ImageView
 - Posiciona dukeImageView de acordo com os argumentos x e y



O objetivos dos níveis Basic e Inheritance do Java Puzzle Ball é conduzir a bola até o gol. O gol é representado visualmente como Duke. Existem muito mais campos e métodos para a classe Goal, e o construtor faz algumas ações a mais além do que é descrito aqui.

Exercício 1

- Crie um novo projeto Java e atribua o nome `GoalTest` a ele
- Clique com o botão direito do mouse no projeto e selecione `New-> pacote - Atribua o nome goaltest ao pacote`
- Adicione os arquivos java `Goal.java` e `GoalTest.java` fornecidos ao pacote
- Clique com o botão direito do mouse no projeto novamente. Crie outro pacote novo e atribua a ele o nome `goaltest.Images`
- Clique novamente com o botão direito do mouse no projeto e crie outro pacote novo. Atribua a ele o nome `goaltest.Audio`
- Isso cria uma estrutura de pastas que pode ser usada para fazer referência facilmente a arquivos de imagem e de áudio



Localizações de arquivo

- Adicione os arquivos de imagem e áudio fornecidos ao local correto (arraste e solte ou copie e cole) nas pastas do pacote em seu IDE

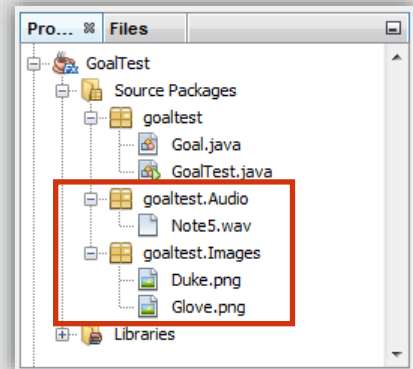
```
Image image = new Image(getClass().getResource("Images/Duke.png").toString());
```

- Images/Duke.png refere-se a uma pasta dentro da pasta GoalTest

– ... \GoalTest\src\goaltest\Images

Pasta do Origem projeto *Pacote principal* *Outro pacote*

– Ou um pacote dentro de outro



Exercício 1 - continuação

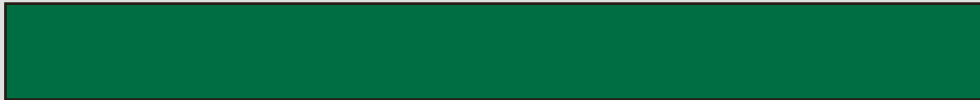
- Observe que...
 - O Root Node está publicamente disponível
 - A classe Goal é um tipo de arquivo de classe Java comum
- Escreva a classe Goal, de acordo com as especificações descritas no slide 9
 - Você também precisará adicionar a ImageView dessa classe ao Root Node
- Instancie alguns objetos Goal do método start()



Os nós têm métodos getter e setter de propriedades, como posição. Você pode obter e definir as posições de x e y de uma ImageView da mesma forma que faria com qualquer outro Nó.

Dimensionando um Nó

- É muito fácil tornar um retângulo mais largo:



- Mas se você tentar fazer o mesmo com uma ImageView...
 - Ele poderia ficar horrível!



Dimensionando um Nó da Maneira Correta

- JavaFX é muito bom para dimensionar elementos gráficos
 - É pouco provável que a qualidade da imagem seja prejudicada
- Você pode preservar a proporção de uma ImageView
 - A largura e a altura de uma ImageView são dimensionadas juntas
 - Isso evita distorção

```
imageView.setPreserveRatio(true);  
imageView.setFitWidth(25);
```

Ordenando Nós

- Algumas vezes, os testadores do Java Puzzle Ball não perceberam que a meta era passar a bola para o Duke
- Pensamos que adicionar uma luva de beisebol ajudaria a resolver o problema
- Duke e a luva são duas ImageViews separadas
 - Elas precisavam ser ordenadas corretamente para que a luva não aparecesse atrás da mão



Correto



Incorreto

ORACLE
Academy

JFo 9-3
Elementos Gráficos, Áudio e Eventos
do Mouse

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.

15

Ordenando Nós da Maneira Correta

- A ordem em que os Nós são adicionados ao Root Node determina a ordem em que são exibidos
- Os nós adicionados anteriormente ficam enterrados embaixo dos nós adicionados depois

```
root.getChildren().addAll(gloveImageView, dukeImageView);
```



- Para corrigir isso, você poderia...
 - Alterar a ordem em que os Nós são adicionados ao Root Node
 - Trazer uma ImageView para frente ou para trás

```
gloveImageView.toFront();    //Um desses  
dukeImageView.toBack();      //resolverá o problema
```



A Classe Goal

- Campos

- private Image dukeImage;
 - private ImageView dukeImageView;
 - private Image gloveImage;
 - private ImageView gloveImageView;



- Construtor

- Escolhe argumentos para as posições de x e y
 - Atribui cada imagem à respectiva ImageView
 - Posiciona dukeImageView de acordo com os argumentos x e y
 - Posiciona e dimensiona gloveImageView em relação à dukeImageView

Exercício 2

- Continue a editar o projeto `GoalTest`
- Escreva a classe `Goal` de acordo com as especificações descritas no slide anterior
 - O construtor deve receber somente dois argumentos
 - Uma luva deve aparecer na mão de Duke
- **Dica:** os nós, inclusive as `ImageViews`, têm métodos `getter` e `setter` de propriedades como `posição`



Semelhanças entre os objetos Image e Audio

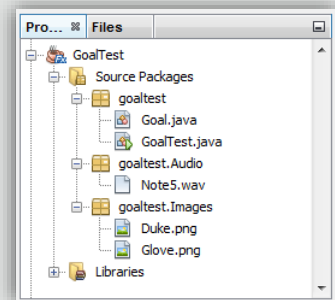
- A criação de um objeto Image do JavaFX...

```
Image image = new  
Image(getClass().getResource("Images/fan1.png").toString());
```

- é muito parecida com a criação de um objeto AudioClip do JavaFX

```
AudioClip audio = new  
Audio(getClass().getResource("Audio/Note5.wav").toString());
```

- É comum armazenar imagens e áudio em seus próprios pacotes/pastas



Diferenças entre os objetos Image e Audio

- Um objeto AudioClip descreve a localização de um arquivo de áudio (.wav, .mp3 ...)

```
AudioClip audio = new  
    AudioClip(getClass().getResource("Audio/Note5.wav").toString());
```

- E, ao contrário de um objeto Image...
 - Não existe um objeto AudioClip equivalente de uma ImageView
 - O áudio pode ser reproduzido fazendo referência ao objeto AudioClip diretamente

```
audio.play();
```

- Você pode chamar muitos outros métodos AudioClip

A Classe Goal

- Campos

```
-private Image dukeImage;  
-private ImageView dukeImageView;  
-private Image gloveImage;  
-private ImageView gloveImageView;  
-private AudioClip tone;
```



- A classe Goal contém um objeto AudioClip como um campo

- tone é reproduzido quando o mouse é pressionado no Duke
- Veremos como implementar esse recurso na próxima parte desta lição

Exercício 3

- Continue a editar o projeto `GoalTest`
- Declare um objeto `AudioClip` como um campo
- Instancie o objeto `AudioClip`
 - Use o arquivo `.wav` no diretório de projeto

Lembre-se de importar
`javafx.scene.media.AudioClip;`

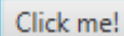


Eventos do Mouse e do Teclado

- Os nós podem detectar eventos do mouse e do teclado
 - Isso também acontece com as ImageViews!
 - Você não está limitado a botões e a outros componentes da interface gráfica do usuário

- Estes são alguns métodos úteis para fazer isso acontecer:

- `setOnMouseClicked()`
- `setOnMouseDragged()`
- `setOnMouseEntered()`
- `setOnMouseExited()`
- `setOnMouseMoved()`
- `setOnMousePressed()`
- `setOnMouseReleased()`



Click me!

*Lembre-se de importar
`javafx.scene.input.MouseEvent`*

Expressões Lambda

- Esses métodos usam um argumento especial, denominado expressão Lambda:

```
imageView.setOnMousePressed( /*Expressão Lambda*/ );
```

- As expressões Lambda usam uma sintaxe especial:

```
(MouseEvent me) -> System.out.println("Pressionado")
```

Sem
ponto e vírgula

- As chaves permitem que as expressões Lambda contenham várias instruções:

```
(MouseEvent me) -> {  
    System.out.println("Instrução 1");  
    System.out.println("Instrução 2");  
} //fim MouseEvent
```

pontos e
vírgulas

As expressões Lambdas foram introduzidas no Java SE 8. Elas fornecem uma sintaxe mais eficiente e organizada para trabalhar com aplicativos de interface gráfica do usuário, bem como para classificar listas.

Expressões Lambda como Argumentos

- Quando elas são combinadas, obtemos o seguinte:

```
imageView.setOnMousePressed( (MouseEvent me) -> {  
    System.out.println("Instrução 1");  
    System.out.println("Instrução 2");  
} );
```

- O que esse código faz:
 - Permite que a imageView detecte um pressionamento do mouse a qualquer momento
 - Se isso ocorrer, as duas instruções de impressão serão executadas
 - Caso contrário, esse código será ignorado

MouseEvent

- Um objeto MouseEvent só existe dentro do escopo da expressão Lambda
- Ele contém muitas propriedades e métodos úteis:

```
imageView.setOnMousePressed( (MouseEvent me) -> {  
    System.out.println(me.getSceneX());  
    System.out.println(me.getSceneY());  
} );
```

- Neste exemplo:
 - me é o objeto MouseEvent
 - me é acessado para imprimir as posições de x e y do cursor do mouse quando imageView é pressionado

Métodos MouseEvent

- `getSceneX()`
- `getSceneY()`
 - Retorna um double
 - Retorna a posição do cursor dentro da Scene JavaFX
 - O canto superior esquerdo da Scene é a posição (0,0)
- `getScreenX()`
- `getScreenY()`
 - Retorna um double
 - Retorna a posição do cursor na tela do computador
 - O canto superior esquerdo da tela do computador é (0,0)

Existem muitos mais métodos do que os listados aqui.

Listening de Eventos

- Quando você escreve um código para Eventos do Mouse
 - Você está informando um Nó para detectar determinado evento
 - Mas isso não significa que os eventos precisam realmente ocorrer
- Desde que o Nó esteja detectando...
 - Ele consegue detectar qualquer evento, a qualquer momento
- Um Nó pode detectar muitos eventos

```
imageView.setOnMousePressed( /*Expressão Lambda*/ );  
imageView.setOnMouseDragged( /*Expressão Lambda*/ );  
imageView.setOnMouseReleased( /*Expressão Lambda*/ );
```

Exercício 4

- Continue a editar o projeto `GoalTest`
- Complete o método `interactions()` para que...
 - Duke detecte a ação de pressionar o mouse e de arrastar o mouse
 - Reproduza um som quando o mouse for pressionado
 - Imprima as posições de x e y do evento de arrastar o mouse
Isso será útil para o conjunto de problemas
- E se `interactions()` nunca for chamado?
 - Remova essa chamada de método no construtor



Resumo

- Nesta lição, você deverá ter aprendido a:
 - Criar e usar uma imagem e a ImageView do JavaFX
 - Criar e usar o áudio do JavaFX
 - Criar e usar Eventos do Mouse
 - Entender as expressões Lambda em aplicativos da interface gráfica do usuário (GUI)



