



Java Foundations

9-3

Gráficos, audio y MouseEvents

ORACLE
Academy



Copyright © 2022, Oracle y/o sus filiales. Oracle, Java y MySQL son marcas comerciales registradas de Oracle y/o sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios.

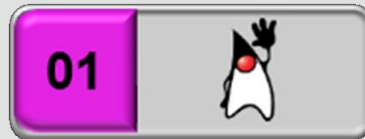
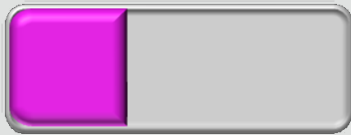
Objetivos

- En esta lección se abordan los siguientes objetivos:
 - Crear y usar una imagen JavaFX y un objeto ImageView
 - Crear y usar audio JavaFX
 - Crear y usar MouseEvents
 - Comprender las expresiones Lambda en aplicaciones de GUI



Uso de sus propios gráficos

- JavaFX puede proporcionar elementos de la UI, formas y texto
 - Pero, si tiene talento para el arte, puede utilizar sus propios gráficos en lugar de los que proporciona JavaFX
- Por ejemplo:



- El arte para el botón de selección de nivel no lo creó JavaFX
- Pero utilizamos JavaFX para agregar por procedimiento números de nivel, texto y el gráfico de Duke

Una imagen JavaFX y un objeto ImageView

- Una imagen es un objeto que describe la ubicación de un archivo de gráficos (.png, .jpg, .gif...)

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

- Un objeto ImageView es el nodo real
 - La llamada a su constructor necesita un argumento de imagen

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

- Un objeto ImageView también contiene las mismas propiedades que cualquier otro nodo: posición x, posición y, anchura, altura...

También deberá importar `javafx.scene.image.Image` y `javafx.scene.image.ImageView`.

¿Por qué tener una imagen y un objeto ImageView?

- Una gran ventaja es la animación
 - Las imágenes se pueden cambiar en y fuera del objeto ImageView
- El ventilador en Java Puzzle Ball lo aprovecha
 - El ventilador pasa por 2 imágenes cuando está en funcionamiento



- Los botones personalizados también se benefician
 - Puede utilizar diferentes imágenes para los botones según su estado:
 - ¿El mouse está encima del botón?
 - ¿El usuario está haciendo clic en el botón?

Indicaciones de un objeto ImageView

- Cómo crear imágenes:

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

- Cómo crear un objeto ImageView:

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

- Cómo intercambiar una imagen en un objeto ImageView:

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

- Un objeto Imageview mantiene sus propiedades como posicionamiento

Recuerde importar
`javafx.scene.image.Image;` y
`javafx.scene.image.ImageView;`

Creación de objetos con las propiedades del nodo

- Hasta ahora, hemos escrito todos los códigos JavaFX en el método `start()`
 - Es similar al principio del curso, en que la mayor parte del código se escribió en el método `main()`
- El código orientado al objeto no se debe escribir de esta forma
 - En su lugar, los objetos deben tener campos de nodo
- Los métodos `start()` y `main()` están destinados a ser controladores

Ejemplo: La clase Goal

- Fields

- `private Image dukeImage;`
 - `private ImageView dukeImageView;`

- Constructor

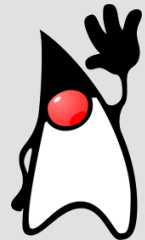
- Toma argumentos para las posiciones x e y
 - Asigna la imagen a su respectivo objeto ImageView
 - Coloca dukeImageView según los argumentos x e y



El objetivo de los niveles de Java Puzzle Ball básicos y heredados es que la bola llegue a la meta. La meta se representa visualmente como Duke. Hay muchos más campos y métodos para la clase Goal y el constructor hace algunas cosas más de las que se describen aquí.

Ejercicio 1

- Cree un nuevo proyecto Java y asígnele el nombre `GoalTest`
- Haga clic con el botón derecho en el proyecto y seleccione `New-> Package` - Asígnele al paquete el nombre `goaltest`
- Agregue los archivos java `Goal.java` y `GoalTest.java` proporcionados al paquete
- Vuelva a hacer clic con el botón derecho en el proyecto y cree otro paquete nuevo y asígnele el nombre `goaltest.Images`
- Haga clic con el botón derecho una vez más en el proyecto y cree otro paquete nuevo y asígnele el nombre `goaltest.Audio`
- De esta forma se crea una estructura de carpetas que se puede utilizar para hacer referencia fácilmente a archivos de imagen y audio



Ubicaciones de archivos

- Agregue los archivos de imagen y audio suministrados a la ubicación correcta (arrastre y suelte o copie y pegue) en las carpetas de paquetes del IDE

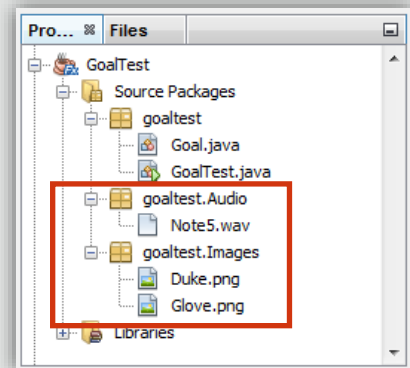
```
Image image = new Image(getClass().getResource("Images/Duke.png").toString());
```

- Images/Duke.png hace referencia a una carpeta dentro de la carpeta GoalTest

– ... \GoalTest\src\goaltest\Images

Carpeta de Origen
proyecto paquete
principal Otro
paquete

– O un paquete dentro de un paquete



Ejercicio 1 (continuación)

- Observe que...
 - El nodo raíz está disponible públicamente
 - La clase Goal es un tipo de archivo de clase Java normal
- Escriba la clase Goal de acuerdo con las especificaciones de la diapositiva 9
 - También tendrá que agregar ImageView de esta clase al nodo raíz
- Instancie algunos objetos Goal desde el método start()



Los nodos tienen métodos getter y setter para propiedades como la posición. Puede obtener y definir las posiciones x e y de una instancia de ImageView de la misma manera que lo haría con cualquier otro nodo.

Escala de un nodo

- Es muy fácil hacer un rectángulo más amplio:



- Pero si intenta lo mismo con un objeto ImageView...
 - ¡Puede parecer horrible!



Escala de un nodo de forma correcta

- JavaFX es excelente para escalar gráficos
 - La calidad de la imagen tiene menos probabilidades de deteriorarse
- Tiene la opción de mantener el ratio de aspecto de un objeto ImageView
 - La altura y la anchura de un objeto Imageview se escalan juntas
 - Evita la distorsión

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

Ordenación de nodos

- A veces, los probadores de Java Puzzle Ball no sabían que su objetivo era llevar la bola hasta Duke
- Pensamos que agregar un guante de béisbol ayudaría a solucionar el problema
- Duke y el guante son dos objetos ImageView independientes
 - Tenían que ordenarse de forma adecuada ya que el guante no se muestra detrás de la mano



Correcto



Incorrecto

ORACLE
Academy

JFo 9-3
Gráficos, audio y MouseEvents

Copyright © 2022, Oracle y/o sus filiales. Oracle, Java y MySQL son marcas comerciales registradas de Oracle y/o sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios.

15

Ordenación de nodos de forma correcta

- El orden en el que se agregan los nodos al nodo raíz determina el orden en que se muestran
- Los nodos agregados más tarde tapan a los nodos agregados anteriormente

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



- Para solucionarlo podría...
 - Cambiar el orden en el que se agregan los nodos al nodo raíz.
 - Poner un objeto ImageView delante o detrás

```
gloveImageView.toFront();    //Either one of these  
dukeImageView.toBack();      //will solve the problem
```



Clase Goal

- Campos

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



- Constructor

- Toma argumentos para las posiciones x e y
 - Asigna cada imagen a su respectivo objeto ImageView
 - Coloca dukeImageView según los argumentos x e y
 - Coloca y escala gloveImageView en relación con dukeImageView

Ejercicio 2

- Continúe editando el proyecto `GoalTest`
- Escriba la clase `Goal` de acuerdo con las especificaciones de la diapositiva anterior
 - El constructor también debe tomar solo dos argumentos
 - Debe aparecer un guante encima de la mano de Duke
- **Indicación:** Los nodos, incluidos los objetos `ImageView`, tienen los métodos `getter` y `setter` para las propiedades como posición



Similitudes de imagen y audio

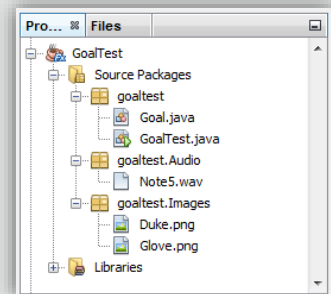
- Creación de un objeto de imagen JavaFX...

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

- Es muy similar a la creación de un objeto de audio JavaFX

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

- Es muy común almacenar imágenes y audio en sus propios paquetes/carpetas



Diferencias de imagen y audio

- Un objeto de audio describe la ubicación de un archivo de audio (.wav, .mp3...)

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

- A diferencia de una imagen...
 - No hay ningún equivalente de audio de un objeto ImageView
 - Se puede reproducir audio haciendo referencia directamente al objeto de audio

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

- Hay varios métodos de audio a los que puede llamar.

Clase Goal

- Campos

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



- La clase Goal contiene un objeto de audio como campo
 - Se reproduce un tono cuando hace clic en Duke
 - Observaremos cómo implantar esta función en la siguiente parte de esta lección

Ejercicio 3

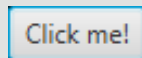
- Continúe editando el proyecto `GoalTest`
- Declare un objeto de audio como campo
- Instancie el objeto de audio
 - Utilice el archivo `.wav` suministrado, `Note5.wav`

Recuerde importar
`javafx.scene.media.AudioClip;`



Eventos del mouse y del teclado

- Los nodos pueden detectar los eventos del mouse y del teclado
 - Se aplica también a los objetos ImageView
 - No está limitado a los botones y a otros componentes de GUI
- Los métodos útiles para que esto suceda incluyen:
 - `setOnMouseClicked()`
 - `setOnMouseDragged()`
 - `setOnMouseEntered()`
 - `setOnMouseExited()`
 - `setOnMouseMoved()`
 - `setOnMousePressed()`
 - `setOnMouseReleased()`



Recuerde importar
`javafx.scene.input.MouseEvent`

Expresiones lambda

- Estos métodos utilizan un argumento especial denominado expresión Lambda:

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

- Las expresiones Lambda utilizan una sintaxis especial:

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

No hay ningún
punto y coma

- Las llaves permiten que las expresiones Lambda incluyan varias sentencias:

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

Puntos y
comas

Las lambdas se introdujeron en Java SE 8. Proporcionan una sintaxis más eficaz y ordenada para trabajar con aplicaciones de GUI y listas de ordenación.

Expresiones lambda como argumentos

- Cuando se combinan, obtenemos lo siguiente:

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

- Qué hace este código:
 - Permite un objeto imageView detectar una pulsación del mouse en cualquier momento
 - Si ocurre, las dos sentencias de impresión se ejecutan
 - De lo contrario, este código se ignora

MouseEvent

- Un objeto MouseEvent solo existe en el ámbito de la expresión lambda
- Contiene muchas propiedades y métodos útiles:

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

- En este ejemplo:
 - me es el objeto MouseEvent
 - A me se accede para imprimir las posiciones x e y del cursor del mouse al pulsar imageView

Métodos MouseEvent

- `getSceneX()`
- `getSceneY()`
 - Devuelve un valor `double`
 - Devuelve la posición del cursor en la escena JavaFX
 - La esquina superior izquierda de la escena es la posición (0,0)
- `getScreenX()`
- `getScreenY()`
 - Devuelve un valor `double`
 - Devuelve la posición del cursor en la pantalla de la computadora
 - La esquina superior izquierda de la pantalla de la computadora es (0,0)

Hay muchos más métodos que los que aparecen aquí.

Recepción de eventos

- Cuando escribe un código para MouseEvents
 - Está indicando la recepción de un nodo para un evento concreto
 - Pero los eventos, en realidad, no tienen que producirse
- Mientras se está recibiendo el nodo...
 - Puede detectar cualquier evento, en cualquier momento
- Un nodo puede recibir muchos eventos

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

Ejercicio 4

- Continúe editando el proyecto `GoalTest`
- Complete el método `interactions()` para que...
 - Duke reciba una pulsación del mouse y un arrastre del mouse
 - Se reproduzca un sonido al pulsar el mouse
 - Se impriman las posiciones x e y del evento arrastrado del mouse.
 - Esto resultará útil para el juego de problemas
- ¿Qué pasa si nunca se llama a `interactions()`?
 - Comente esta llamada de método en el constructor



Resumen

- En esta lección, debe haber aprendido lo siguiente:
 - Crear y usar una imagen JavaFX y un objeto ImageView
 - Crear y usar audio JavaFX
 - Crear y usar MouseEvents
 - Comprender las expresiones Lambda en aplicaciones de GUI



ORACLE
Academy

JFo 9-3
Gráficos, audio y MouseEvents

Copyright © 2022, Oracle y/o sus filiales. Oracle, Java y MySQL son marcas comerciales registradas de Oracle y/o sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios.

30

