Outline

1. Development Platforms for Multimedia Programming
   1.1. Classification of Development Platforms
   1.2. A Quick Tour of Various Development Platforms
2. Multimedia Programming with Python and Pygame
   2.1. Introduction to Python
   2.2. Pygame: A Multimedia/Game Framework for Python
3. Multimedia Programming with Java FX
4. Multimedia Programming with JavaScript and CreateJS
5. History of Multimedia Programming
6. Programming with Images
7. Programming with Vector Graphics and Animations
8. Programming with Sound
9. Programming with Video
10. Software Engineering Techniques for Multimedia Programs
    10.1. Design Patterns
    10.2. Multimedia Modeling Languages
3 Multimedia Programming with JavaFX

3.1 Basic Concepts of JavaFX
3.2 Observable Properties and Binding
3.3 Timeline and Animation
3.4 Interactive JavaFX Applications

Literature:
   docs.oracle.com/javafx
JavaFX - Idea and History

• Chris Oliver, 2006 (?): “Form follows function” (F3)
  – Working for company “SeeBeyond”, but personal project
• Acquisition of SeeBeyond by Sun, 2005
  – F3 is not in the center of interest, apparently
  – First announcement of JavaFX (ex F3) May 2007 (JavaOne conference)
• Builds on Java runtime environment:
  – Common programming model for multimedia applications across many
    platforms, including mobile devices
• In Versions 1.X, JavaFX was built on JavaScript language (not Java!)
• JavaFX 2.0 (October 2011): JavaFX as native Java library
  (and introduction of declarative FXML language)
• Since Java SE7 update 6 and JavaFX 2.2:
  JavaFX contained in Java SE standard distribution
• Current version (renumbered): JavaFX 8 (March 2014)
JavaFX Application

• JavaFX program always extends the class
  `javafx.application.Application`

• JavaFX runtime carries out the following steps:
  – Construct an instance of the application class
  – Calls `init()` method (override if needed)
  – Calls `start(javafx.stage.Stage)` abstract method
    » Needs to be overridden
  – Waits for the application to finish
  – Calls `stop()` method (override if needed)

• Standard template for a JavaFX application:

  ```java
  public class MyApp extends Application {
      public void start(Stage stage) {
          ...
      }
  }
  ```
JavaFX Stage

• Stage (*Bühne*):
  – Frequently used metaphor for the space for animated program behavior

• `javafx.stage.Stage` class:
  – Top-level container for JavaFX application
  – Elements are visible/audible only if assigned to stage
  – Primary stage is constructed by platform (parameter of `start()`)
  – There may be several stages

• Stage has a title
  – `setTitle()`

• Stage contents are organized in the contained scene:
  – `setScene(javafx.scene.Scene)`

• Stage contents have to be made visible explicitly:
  – `show()`
JavaFX Scene

• JavaFX elements are always organized in a *scene graph*
  – *Frequently used concept also in other platforms*

• Scene graph is a *directed acyclic graph (DAG)*
  – Leaf nodes are media object representations: *Rectangle, Text, ImageView, MediaView, ...*
  – Superclass of all contained objects: *Node*
    » *Composite* design pattern, see later
  – Branching class: *Group (extends Node)*
Skeleton Code for Slideshow in JavaFX

```java
public class JFXSlideshowBasic extends Application {

    @Override
    public void start(Stage primaryStage) {

        Color background = Color.rgb(255, 228, 95);

        final ImageView picture = new ImageView();
        picture.setX(50);
        picture.setY(50);
        picture.setImage(…);

        Group root = new Group(picture);
        Scene scene = new Scene(root, 356, 356, background);

        primaryStage.setTitle("Simple Slide Show with JavaFX");
        primaryStage.setScene(scene);
        primaryStage.sizeToScene();
        primaryStage.show();
    }
}
```
QUIZ

• Give a graphical picture of the scene graph for the code skeleton just shown!
3 Multimedia Programming with JavaFX

3.1 Basic Concepts of JavaFX
3.2 Observable Properties and Binding
3.3 Timeline and Animation
3.4 Interactive JavaFX Applications

Literature:
docs.oracle.com/javafx
Observable Properties

• Program variable $x$ depending on the value of another variable $y$
  $x = f(y)$
  – Example: Bill amount depending on tax rate
  – Example: Image displayed depending on current index in show

• What happens if value of $y$ changes?
  – If $y$ is a standard variable?
  – If $y$ is a writable and observable value?
  – Idea: Re-evaluate $x$ as soon as $y$ changes

• Java implementation:
  – Writable and observable values: Analogous to Java Bean properties
  – Re-evaluation through Observer pattern: Registering a change listener

• More advanced concept: Binding within expressions
Example: Observable Property “slideindex”

public class JFXSlideshowBasic extends Application {

    @Override
    public void start(Stage primaryStage) {
      ...

      final ImageView picture = new ImageView();
      picture.setImage(imagearray[0]);

      final IntegerProperty slideindex =
          new SimpleIntegerProperty();
      slideindex.addListener(
          new ChangeListener(){
              @Override
              public void changed
                  (ObservableValue o, Object oldVal, Object newVal){
                  picture.setImage(imagearray[slideindex.getValue()]);
              }
          });

      ...
    }
}
Simple Example for Binding in JavaFX

```java
final IntegerProperty propx = new SimpleIntegerProperty();
final IntegerProperty propy = new SimpleIntegerProperty();

propy.bind(propx);
propx.setValue(3);
...
System.out.println("Value of propy is "+propy.getValue());
```

Result:

```
Value of propy is 3
```

![Diagram of bindings between propx and propy]
3 Multimedia Programming with JavaFX

3.1 Basic Concepts of JavaFX
3.2 Observable Properties and Binding
3.3 Timeline and Animation
3.4 Interactive JavaFX Applications

Literature:
  docs.oracle.com/javafx
Concept: Timeline and Key Frames

• From JavaFX (Ver. 1!) documentation: “Timeline provides the capability to update the property values along the progression of time.”

• Some variables take new values at certain points in time
  – Example slide index in slide show

• **Timeline**:
  – Defines a sequence of *key frames*
  – Each key frame defines a certain configuration of values
    – Writable property + concrete value for it
    – Each key frame is associated to a point in time

• Timelines are suitable to express animations
  – *Interpolation* of values (see later)

• A timeline is a sequential time container.
JavaFX Timeline

• class Timeline extends Animation
  – Property cycleCount: Repetition counter, may be INDEFINITE
  – Property KeyFrames: List of KeyFrame objects
  – Adding key frames to a timeline object:
    » By obtaining the keyframe list and adding to that

• Main methods for timelines:
  – play()
  – playFromStart()
  – stop()
JavaFX KeyFrame

- **KeyFrame** object constructor needs:
  - `Duration` object (time in ms from beginning)
  - `KeyValue` object
- **KeyValue** object constructor needs:
  - Writable property *(not* a normal variable)
    - e.g. `IntegerProperty`, *not int*
  - Suitable concrete value
- E.g.:
  ```java
  new KeyFrame(
      new Duration(8000),
      new KeyValue(slideindex, 2)
  )
  ```
Example: Timeline for Slideshow

Timeline timeline = new Timeline();
List<KeyFrame> keyframes = timeline.getKeyFrames();
for(int i=0;
i<imagearray.length; i++) {
    keyframes.add(
        new KeyFrame(
            new Duration(i*4000),
            new KeyValue(
                slideindex,
                i
            )
        )
    );
}
Slideshow with JavaFX (1)

package jfxslideshowbasic;

import java.util.List;
import javafx.animation.*;
import javafx.application.Application;
import javafx.beans.property.*;
import javafx.beans.value.*;
import javafx.scene.*;
import javafx.scene.image.*;
import javafx.scene.paint.*;
import javafx.stage.Stage;
import javafx.util.Duration;

public class JFXSlideshowBasic extends Application {

    @Override
    public void start(Stage primaryStage) {

        Color background = Color.rgb(255, 228, 95);

        final Image[] imagearray = {
            new Image("file:pics/tiger.jpg", true),
            new Image("file:pics/elephant.jpg", true),
            new Image("file:pics/jbeans.jpg", true),
            new Image("file:pics/peppers.jpg", true),
            new Image("file:pics/butterfly.jpg", true),
        };

        ...
    }
}
Slideshow with JavaFX (2)

... 

```java
final ImageView picture = new ImageView();
picture.setX(50);
picture.setY(50);
picture.setImage(imagearray[0]);

final IntegerProperty slideindex = new SimpleIntegerProperty();
slideindex.addListener(new ChangeListener(){
    @Override public void changed(ObservableValue o, Object oldVal, Object newVal){
        picture.setImage(imagearray[slideindex.getValue()]);
    }
});

Group root = new Group(picture);
Scene scene = new Scene(root, 356, 356, background);

Timeline timeline = new Timeline();
List<KeyFrame> keyframes = timeline.getKeyFrames();
for(int i=0; i<imagearray.length; i++) {
    keyframes.add(new KeyFrame(new Duration(i*4000), new KeyValue(slideindex, i)));
}

primaryStage.setTitle("Simple Slide Show with JavaFX");
primaryStage.setScene(scene);
primaryStage.sizeToScene();
primaryStage.show();
timeline.play();
```
3 Multimedia Programming with JavaFX

3.1 Basic Concepts of JavaFX
3.2 Observable Properties and Binding
3.3 Timeline and Animation
3.4 Interactive JavaFX Applications

Literature:

docs.oracle.com/javafx
Slideshow in JavaFX - Interactive Version

```java
scene.setOnKeyPressed(new EventHandler<KeyEvent>() {
    @Override
    public void handle(KeyEvent ke) {
        int currentslide = slideindex.getValue();
        if (ke.getCode() == KeyCode.RIGHT) {
            if (currentslide+1 < imagearray.length) {
                slideindex.setValue(currentslide+1);
            }
        }
        if (ke.getCode() == KeyCode.LEFT) {
            if (currentslide+1 > 0) {
                slideindex.setValue(currentslide-1);
            }
        }
    }

    Using JavaFX “Convenience Methods”
    for event handling
```
QUIZ

• What are the conceptual differences between the Pygame and JavaFX versions of the same program?