



The University of
Nottingham

UNITED KINGDOM • CHINA • MALAYSIA

Lecture 06B

Maintainable GUI Development (2/2)

JavaFX advanced topics: Multi-threading and animation

Horia A. Maior and Marjahan Begum



COMP2013

Developing Maintainable Software

Lecture 06B

More Maintainable GUI Development (2/2)

JavaFX advanced topics: Multi-threading and animation

Horia A. Maior and Marjahan Begum

Topics for this Week



- Lecture 05A
 - Build Tools
 - JavaFX(ML) advanced topics: controls and styling
- Lecture 05B
 - CW release
- Lecture 06A
 - Setting up Git for your project
- Lecture 06B
 - JavaFX advanced topics: Multi-threading and animation

Request Access to UoN Git before Friday (Tomorrow)



<https://forms.office.com/e/ieDqqj1JUu>



JavaFX advanced topics

Multi-threading: Dealing with unresponsive GUIs

Stopping GUIs becoming unresponsive



- The facts:
 - JavaFX launches the UI on a JavaFX Application thread
 - This thread should be left handling the UI interaction
 - Heavy computation should be done elsewhere to prevent freezing!
- JavaFX provides a solution:
 - Package "javafx.concurrent"
 - A way for JavaFX to create and communicate with other threads

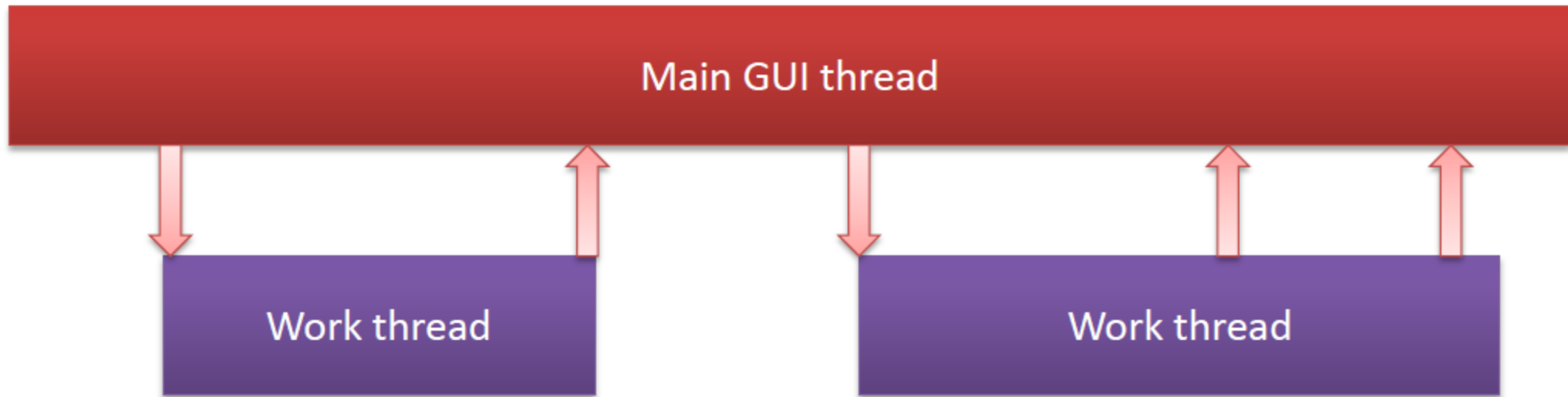
JavaFX Application Thread



```
© JavaFXMLControlsApplication.java × pom.xml (JavaFXMLControlsMultiThreads) © JavaFXMLControlsController.java © Pro...

1 package com.comp2013.javafxmlcontrolsmultithreads;
2
3 > import ...
4
5
6
7
8
9
10 ▶ public class JavaFXMLControlsApplication extends Application {
11     @Override
12     @ public void start(Stage stage) throws IOException {
13         FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource("name: \"javafxmlcontrols-view.fxml\"));
14         Scene scene = new Scene(fxmlLoader.load());
15         stage.setTitle("Hello Java FXML Controls!");
16         stage.setScene(scene);
17         stage.show();
18     }
19
20 ▶ public static void main(String[] args) {
21     launch();
22 }
23 }
```

Stopping GUIs becoming unresponsive

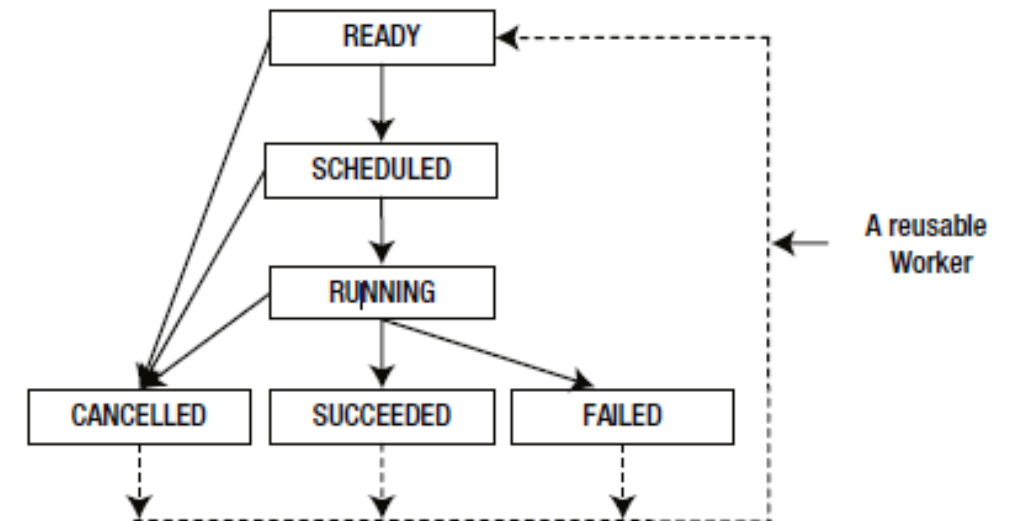
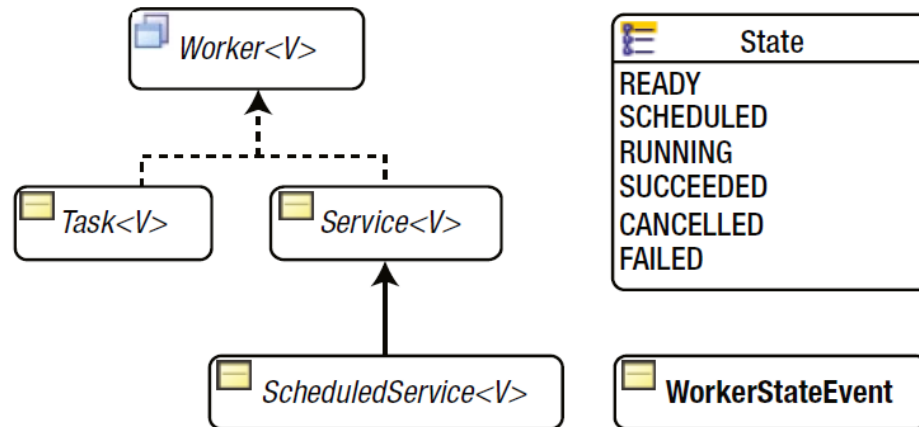


- JavaFX provides some helper classes

Concurrency Framework in Java



- Concurrency
 - The ability of different parts of a program to be executed out-of-order or in partial order, without affecting the final outcome
 - java.util.concurrent package
- Classes in the JavaFX Concurrency Framework



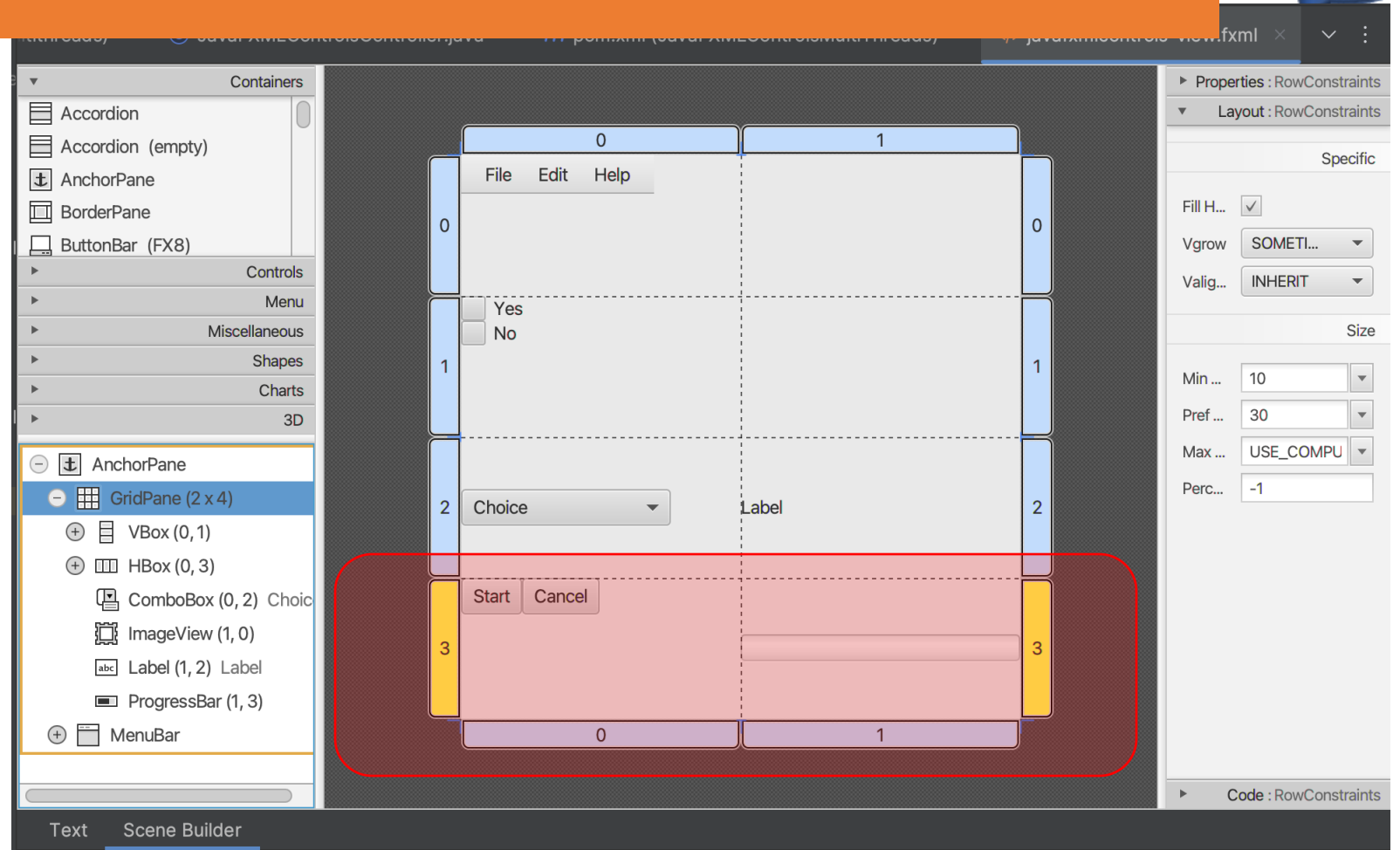
Source: Learn JavaFX 8 - Building User Experience and Interfaces with Java 8 (Apress.2015)

Concurrency Framework in JavaFX



- Framework consist of one interface + four classes + one enum
 - **Worker** interface specifies the methods available to background (work) threads
 - **Task** class instance represents a one-shot task
 - **Service** class instance represents a reusable task
 - **ScheduledService** class instance represents a reusable task that runs repeatedly following a specified interval
 - **WorkerStateEvent** class instance represents an event that occurs as state of Worker changes
 - You can add event handlers to all three types of tasks to listen to the changes in their states
 - **State** enum constants represents different states of a worker

"Task



Let's build a progress bar to show some work done in another thread

"Task" Example



- Worker interface
 - Specifies the methods available to **background threads** when working with JavaFX
 - Various useful methods such as `isRunning()`, `getProgress()`, `cancel()`...
 - <https://docs.oracle.com/javase/8/javafx/api/javafx/concurrent/Worker.html>
- Task class
 - Abstract class which implements the Worker interface
 - We program the actual work that needs to be done on a separate thread
 1. Extend the Task class
 2. Implement `call()` to do the work
 - Don't directly touch UI components from here
 - Can update the UI with `updateProgress()`, `updateMessage()`, `updateTitle()` methods
 3. Start a new thread, passing the relevant "Worker" as a parameter
 - <https://docs.oracle.com/javase/8/javafx/api/javafx/concurrent/Task.html>



Properties : Button


Layout : Button


Code : Button

Identity

fx:id


Main

On Action 


doWork 

DragDrop


On Drag Detected




On Drag Done




On Drag Dropped




On Drag Entered



On Drag Exited




On Drag Over




File Edit Help

☐ Yes

☐ No

Choice 

Label

Start  Cancel



The screenshot displays a software development environment with a central canvas and a right-hand properties panel.

Central Canvas:

- A menu bar at the top contains "File", "Edit", and "Help".
- Below the menu bar are two radio buttons labeled "Yes" and "No".
- Further down is a "Choice" dropdown menu and a "Label" text field.
- At the bottom, there is a "Start" button and a "Cancel" button. The "Cancel" button is highlighted with a red rectangular selection box.

Properties Panel (Right):

- The panel has three tabs: "Properties : Button", "Layout : Button", and "Code : Button". The "Code : Button" tab is currently selected.
- Under the "Code : Button" tab, there are sections for "Identity", "Main", and "DragDrop".
- In the "Main" section, the "On Action" property is set to "# onCancel", which is highlighted with a red rectangular selection box.
- The "DragDrop" section contains several event handlers, each with a dropdown menu for selecting an action: "On Drag Detected", "On Drag Done", "On Drag Dropped", "On Drag Entered", "On Drag Exited", and "On Drag Over".



Properties : ProgressBar

Layout : ProgressBar

Code : ProgressBar

Identity

fx:id

myProgressBar

DragDrop

On Drag Detected

#

On Drag Done

#

On Drag Dropped

#

On Drag Entered

#

On Drag Exited

#

On Drag Over

#

On Mouse Drag Entered

#

On Mouse Drag Exited

#

16 </>

```
@FXML private ProgressBar myProgressBar;
```

3 usages

17

```
private ProgressWorker pw=new ProgressWorker();
```



DEMO

59

```
@FXML
```

60

```
private void doWork(){
```

61

```
new Thread(pw).start();
```

62

```
myProgressBar.progressProperty().bind(pw.progressProperty());
```

63

```
}
```



16 </>

```
@FXML private ProgressBar myProgressBar;
```

3 usages

17

```
private ProgressWorker pw=new ProgressWorker();
```



DEMO

59

```
@FXML
```

60

```
private void doWork(){
```

61

```
    new Thread(pw).start();
```

62

```
    myProgressBar.progressProperty().bind(pw.progressProperty());
```

63

```
}
```

65

```
@FXML
```

66

```
private void onCancel(){
```

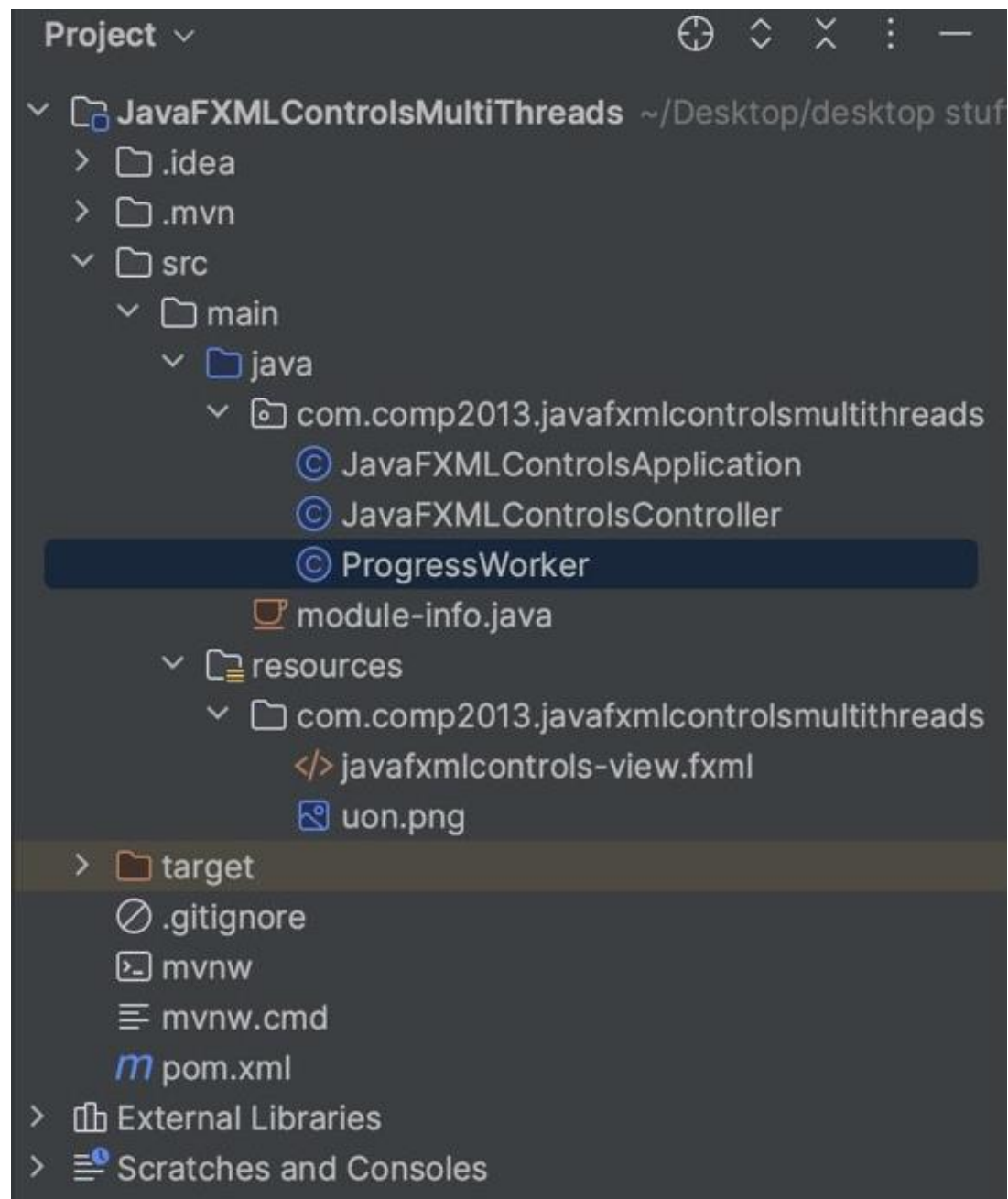
67

```
    pw.cancel();
```

68

```
}
```







DEMO


```
1 package com.comp2013.javafxcontrols multithreads;
2
3 import javafx.concurrent.Task;
4
5 public class ProgressWorker extends Task {
6     @Override
7     protected Object call() throws Exception {
8         int n=500000;
9         for(int i=0;i<n;i++){
10             System.out.println(i);
11             updateProgress(i,n);
12             if(isCancelled())break;
13         }
14         return null;
15     }
16 }
17
```

Result



Hello Java FXML Controls!


File Edit Help

 **University of Nottingham**
UK | CHINA | MALAYSIA

☒ Yes
☐ No

Choice ▾


Start Cancel



499990
499991
499992
499993
499994
499995
499996
499997
499998
499999

Hello Java FXML Controls!


File Edit Help

 **University of Nottingham**
UK | CHINA | MALAYSIA

☒ Yes
☐ No

Choice ▾

Start Cancel



299277
299278
299279
299280
299281
299282
299283
299284
299285
299286

Concurrency in Java Advice



<https://docs.oracle.com/javase/8/javafx/interoperability-tutorial/index.html>



Search Java SE Documentation

Looking for a different release? [Other releases](#)

Java Platform, Standard Edition (Java SE) 8







[Send Feedback](#) | [Print](#) | [PDF](#) | [ePub](#) | [Mobi](#)

JavaFX: Interoperability

Table of Contents

[Next Page](#)

[Expand](#) | [Collapse](#)

-  [Title and Copyright Information](#)
-  [Preface](#)
-  [Part I Concurrency in JavaFX](#)
-  [Part II JavaFX-Swing Interoperability](#)
-  [Part III Interoperability with SWT](#)
-  [Part IV Source Code for the Interoperability Tutorial](#)



JavaFX advanced topics

Animation

Types of animations in JavaFX



- Timeline Animation
- Transition Animation
- Path Animation
- the list goes on :)

Key Concepts for Timeline Animation

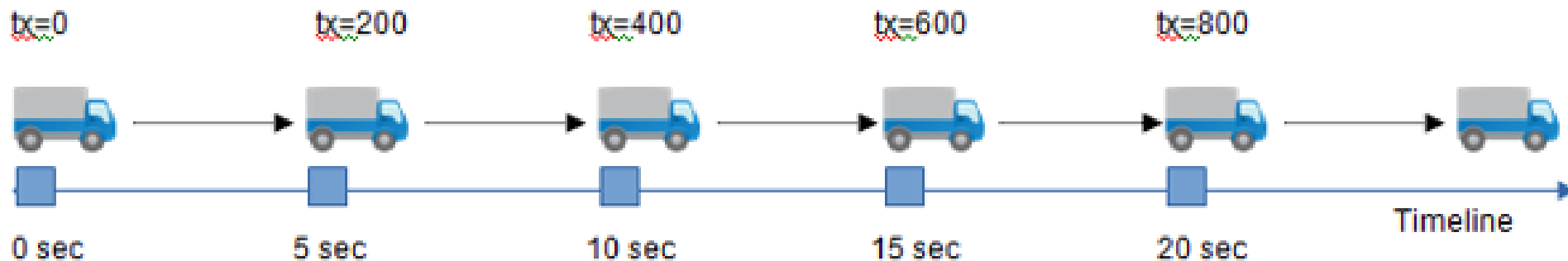


- Timeline
 - Denotes the progression of time during animation with an associated key frame at a given instance
- Key frame
 - Represents the state of the node being animated at a specific instant on the timeline
- Key value
 - Represents the value of a property of a node along with an interpolator to be used
- Interpolator
 - By default linear; changes the property being animated linearly with time

Key Concepts for Timeline Animation



- Lorries represent key frames at specific instants of the timeline
- The key values (here the value for the translateX property) associated with key frames are shown at the top
- By default linear interpolation is used between the key frames



Key Concepts for Timeline Animation



DEMO

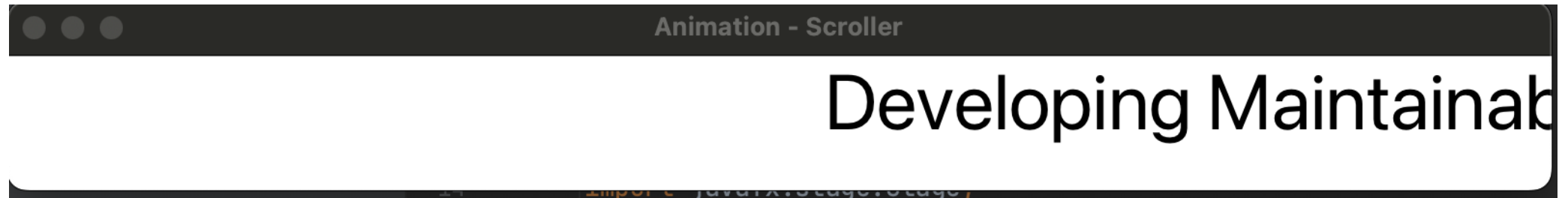
- Using timeline animation involves the following steps
 1. Construct key frames
 2. Create timeline object with key frames
 3. Set the animation properties
 4. Use the play() method to run the animation
- The timeline keeps all key frames in the `ObservableList<KeyFrame>` object
- The `getKeyFrames()` method returns the list

Key Concepts for Timeline Animation



DEMO

- Let's look at an example



- Problem:
 - Scroll text does not update its initial position when the width of the scene changes
- Solution:
 - Update the initial key frame whenever the scene width changes
 - Use a ChangeListener for this

```

1 package com.javafxanimation;
2 > import ...
15 ▶ public class HelloApplication extends Application {
16     @Override
17     @Override
18     public void start(Stage primaryStage) throws Exception{
19         Media sound = new Media(HelloApplication.class.getResource("Axe1F.mp3").toString());
20         MediaPlayer mediaPlayer = new MediaPlayer(sound);
21         mediaPlayer.play();
22
23         Text msg=new Text(s: "Developing Maintainable Software is Cool");
24         msg.setFont(Font.font(v: 40));
25         msg.setTextOrigin(VPos.TOP);
26
27         Pane root=new Pane(msg);
28         root.setPrefSize(v: 800, v1: 70);
29         primaryStage.setTitle("Animation - Scroller");
30         Scene scene=new Scene(root);
31         primaryStage.setScene(scene);
32         primaryStage.show();
33
34         double sceneWidth=primaryStage.getWidth();
35         double msgWidth=msg.getLayoutBounds().getWidth();
36         KeyValue initKeyValue=new KeyValue(msg.translateXProperty(),sceneWidth);//-msgWidth
37         KeyFrame initFrame=new KeyFrame(Duration.ZERO,initKeyValue);
38         KeyValue endKeyValue=new KeyValue(msg.translateXProperty(),-msgWidth);//0
39         KeyFrame endFrame=new KeyFrame(Duration.seconds(v: 3),initKeyValue,endKeyValue);
40         Timeline timeline =new Timeline(initFrame,endFrame);
41         timeline.setCycleCount(Timeline.INDEFINITE);
42         timeline.setAutoReverse(true);
43         timeline.setRate(2);
44         timeline.play();
45     }
46
47     public static void main(String[] args) {
48         launch(args);
49     }
50

```



```
18
19
20 @Override
21 public void start(Stage primaryStage) throws Exception{
22     Media sound = new Media(HelloApplication.class.getResource(name: "AxelF.mp3").toString());
23     MediaPlayer mediaPlayer = new MediaPlayer(sound);
24     mediaPlayer.play();
25
26     Text msg=new Text(s: "Developing Maintainable Software is Cool");
27     msg.setFont(Font.font(v: 40));
28     msg.setTextOrigin(VPos.TOP);
29
30     Pane root=new Pane(msg);
31     root.setPrefSize(v: 800, v1: 70);
32     primaryStage.setTitle("Animation - Scroller");
33     Scene scene=new Scene(root);
34     primaryStage.setScene(scene);
35     primaryStage.show();
36
37     double sceneWidth=primaryStage.getWidth();
38     double msgWidth=msg.getLayoutBounds().getWidth();
39     KeyValue initKeyValue=new KeyValue(msg.translateXProperty(),sceneWidth);//-msgWidth
40     KeyFrame initFrame=new KeyFrame(Duration.ZERO,initKeyValue);
41     KeyValue endKeyValue=new KeyValue(msg.translateXProperty(),-msgWidth);//0
42     KeyFrame endFrame=new KeyFrame(Duration.seconds(v: 3),initKeyValue,endKeyValue);
43     Timeline timeline =new Timeline(initFrame,endFrame);
44     timeline.setCycleCount(Timeline.INDEFINITE);
45     timeline.setAutoReverse(true);
46     timeline.setRate(2);
47     timeline.play();
48
49     public static void main(String[] args) {
50         launch(args);
51     }
52
53
54
```



```
18
19 @Override
20 @ public void start(Stage primaryStage) throws Exception{
21     Media sound = new Media(HelloApplication.class.getResource(name: "AxelF.mp3").toString());
22     MediaPlayer mediaPlayer = new MediaPlayer(sound);
23     mediaPlayer.play();
24
25     Text msg=new Text(s: "Developing Maintainable Software is Cool");
26     msg.setFont(Font.font(v: 40));
27     msg.setTextOrigin(VPos.TOP);
28
29     Pane root=new Pane(msg);
30     root.setPrefSize(v: 800, v1: 70);
31     primaryStage.setTitle("Animation - Scroller");
32     Scene scene=new Scene(root);
33     primaryStage.setScene(scene);
34     primaryStage.show();
35
```

```
36     double sceneWidth=primaryStage.getWidth();
37     double msgWidth=msg.getLayoutBounds().getWidth();
38     KeyValue initKeyValue=new KeyValue(msg.translateXProperty(),sceneWidth);//-msgWidth
39     KeyFrame initFrame=new KeyFrame(Duration.ZERO,initKeyValue);
40     KeyValue endKeyValue=new KeyValue(msg.translateXProperty(),-msgWidth);//0
41     KeyFrame endFrame=new KeyFrame(Duration.seconds(v: 3),initKeyValue,endKeyValue);
42     Timeline timeline =new Timeline(initFrame,endFrame);
43     timeline.setCycleCount(Timeline.INDEFINITE);
44     timeline.setAutoReverse(true);
45     timeline.setRate(2);
46     timeline.play();
47 }
48
49 public static void main(String[] args) {
50
51     launch(args);
52 }
53
54
```

```
18
19 @Override
20 public void start(Stage primaryStage) throws Exception{
21     Media sound = new Media(HelloApplication.class.getResource(name: "AxelF.mp3").toString());
22     MediaPlayer mediaPlayer = new MediaPlayer(sound);
23     mediaPlayer.play();
24
25     Text msg=new Text(s: "Developing Maintainable Software is Cool");
26     msg.setFont(Font.font(v: 40));
27     msg.setTextOrigin(VPos.TOP);
28
29     Pane root=new Pane(msg);
30     root.setPrefSize(v: 800, v1: 70);
31     primaryStage.setTitle("Animation - Scroller");
32     Scene scene=new Scene(root);
33     primaryStage.setScene(scene);
34     primaryStage.show();
35
```

```
36 double sceneWidth=primaryStage.getWidth();
37 double msgWidth=msg.getLayoutBounds().getWidth();
38 KeyValue initKeyValue=new KeyValue(msg.translateXProperty(),sceneWidth);//-msgWidth
39 KeyFrame initFrame=new KeyFrame(Duration.ZERO,initKeyValue);
40 KeyValue endKeyValue=new KeyValue(msg.translateXProperty(),-msgWidth);//0
41 KeyFrame endFrame=new KeyFrame(Duration.seconds(v: 3),initKeyValue,endKeyValue);
42 Timeline timeline =new Timeline(initFrame,endFrame);
43 timeline.setCycleCount(Timeline.INDEFINITE);
44 timeline.setAutoReverse(true);
45 timeline.setRate(2);
46 timeline.play();
47 }
48
49 public static void main(String[] args) {
50
51     launch(args);
52 }
53 }
54
```

```
18
19
20 @Override
21 public void start(Stage primaryStage) throws Exception{
22     Media sound = new Media(HelloApplication.class.getResource(name: "AxelF.mp3").toString());
23     MediaPlayer mediaPlayer = new MediaPlayer(sound);
24     mediaPlayer.play();
25
26     Text msg=new Text(s: "Developing Maintainable Software is Cool");
27     msg.setFont(Font.font(v: 40));
28     msg.setTextOrigin(VPos.TOP);
29
30     Pane root=new Pane(msg);
31     root.setPrefSize(v: 800, v1: 70);
32     primaryStage.setTitle("Animation - Scroller");
33     Scene scene=new Scene(root);
34     primaryStage.setScene(scene);
35     primaryStage.show();
```

```
36     double sceneWidth=primaryStage.getWidth();
37     double msgWidth=msg.getLayoutBounds().getWidth();
38     KeyValue initKeyValue=new KeyValue(msg.translateXProperty(),sceneWidth); //-msgWidth
39     KeyFrame initFrame=new KeyFrame(Duration.ZERO,initKeyValue);
40     KeyValue endKeyValue=new KeyValue(msg.translateXProperty(),-msgWidth); //0
41     KeyFrame endFrame=new KeyFrame(Duration.seconds(v: 3),initKeyValue,endKeyValue);
42     Timeline timeline =new Timeline(initFrame,endFrame);
43     timeline.setCycleCount(Timeline.INDEFINITE);
44     timeline.setAutoReverse(true);
45     timeline.setRate(2);
46     timeline.play();
47 }
48
49 public static void main(String[] args) {
50     launch(args);
51 }
52
53
54
```

```

36 double sceneWidth=primaryStage.getWidth();
37 double msgWidth=msg.getLayoutBounds().getWidth();
38 KeyValue initKeyValue=new KeyValue(msg.translateXProperty(),sceneWidth);//-msgWidth
39 KeyFrame initFrame=new KeyFrame(Duration.ZERO,initKeyValue);
40 KeyValue endKeyValue=new KeyValue(msg.translateXProperty(),-msgWidth);//0
41 KeyFrame endFrame=new KeyFrame(Duration.seconds( v: 3),initKeyValue,endKeyValue);
42 Timeline timeline =new Timeline(initFrame,endFrame);
43 timeline.setCycleCount(Timeline.INDEFINITE);
44 timeline.setAutoReverse(true);
45 timeline.setRate(2);
46 timeline.play();
47 }

```

```

49 ▶ public static void main(String[] args) {
50
51     launch(args);
52 }
53 }
54

```

```

pom.xml (JavaFXAnimation)  HelloApplication.java  module-info.java (JavaFXAnimation)
18
19 @Override
20 @
21 public void start(Stage primaryStage) throws Exception{
22     Media sound = new Media>HelloApplication.class.getResource( name: "AxeLF.mp3").toString());
23     MediaPlayer mediaPlayer = new MediaPlayer(sound);
24     mediaPlayer.play();
25
26     Text msg=new Text( s: "Developing Maintainable Software is Cool");
27     msg.setFont(Font.font( v: 40));
28     msg.setTextOrigin(VPos.TOP);
29
30     Pane root=new Pane(msg);
31     root.setPrefSize( v: 800, v1: 70);
32     primaryStage.setTitle("Animation - Scroller");
33     Scene scene=new Scene(root);
34     primaryStage.setScene(scene);
35     primaryStage.show();

```



```

36     double sceneWidth=primaryStage.getWidth();
37     double msgWidth=msg.getLayoutBounds().getWidth();
38     KeyValue initKeyValue=new KeyValue(msg.translateXProperty(),sceneWidth);//-msgWidth
39     KeyFrame initFrame=new KeyFrame(Duration.ZERO,initKeyValue);
40     KeyValue endKeyValue=new KeyValue(msg.translateXProperty(),-msgWidth);//0
41     KeyFrame endFrame=new KeyFrame(Duration.seconds( v: 3),initKeyValue,endKeyValue);
42     Timeline timeline =new Timeline(initFrame,endFrame);
43     timeline.setCycleCount(Timeline.INDEFINITE);
44     timeline.setAutoReverse(true);
45     timeline.setRate(2);
46     timeline.play();
47 }
48
49 ▶ public static void main(String[] args) {
50
51     launch(args);
52 }
53 }
54

```

```

pom.xml (JavaFXAnimation)  HelloApplication.java  module-info.java (JavaFXAnimation)
18
19 @Override
20 @
21 public void start(Stage primaryStage) throws Exception{
22     Media sound = new Media(HelloApplication.class.getResource( name: "AxeLF.mp3").toString());
23     MediaPlayer mediaPlayer = new MediaPlayer(sound);
24     mediaPlayer.play();
25
26     Text msg=new Text( s: "Developing Maintainable Software is Cool");
27     msg.setFont(Font.font( v: 40));
28     msg.setTextOrigin(VPos.TOP);
29
30     Pane root=new Pane(msg);
31     root.setPrefSize( v: 800, v1: 70);
32     primaryStage.setTitle("Animation - Scroller");
33     Scene scene=new Scene(root);
34     primaryStage.setScene(scene);
35     primaryStage.show();

```

```

36     double sceneWidth=primaryStage.getWidth();
37     double msgWidth=msg.getLayoutBounds().getWidth();
38     KeyValue initKeyValue=new KeyValue(msg.translateXProperty(),sceneWidth); //-msgWidth
39     KeyFrame initFrame=new KeyFrame(Duration.ZERO,initKeyValue);
40     KeyValue endKeyValue=new KeyValue(msg.translateXProperty(),-msgWidth); //0
41     KeyFrame endFrame=new KeyFrame(Duration.seconds( v: 3),initKeyValue,endKeyValue);
42     Timeline timeline =new Timeline(initFrame,endFrame);
43     timeline.setCycleCount(Timeline.INDEFINITE);
44     timeline.setAutoReverse(true);
45     timeline.setRate(2);
46     timeline.play();
47 }
48
49 ▶ public static void main(String[] args) {
50
51     launch(args);
52 }
53 }
54

```

```

pom.xml (JavaFXAnimation)  HelloApplication.java  module-info.java (JavaFXAnimation)
18
19 @Override
20 @
21 public void start(Stage primaryStage) throws Exception{
22     Media sound = new Media>HelloApplication.class.getResource( name: "AxeLF.mp3").toString());
23     MediaPlayer mediaPlayer = new MediaPlayer(sound);
24     mediaPlayer.play();
25
26     Text msg=new Text( s: "Developing Maintainable Software is Cool");
27     msg.setFont(Font.font( v: 40));
28     msg.setTextOrigin(VPos.TOP);
29
30     Pane root=new Pane(msg);
31     root.setPrefSize( v: 800, v1: 70);
32     primaryStage.setTitle("Animation - Scroller");
33     Scene scene=new Scene(root);
34     primaryStage.setScene(scene);
35     primaryStage.show();

```

```

36     double sceneWidth=primaryStage.getWidth();
37     double msgWidth=msg.getLayoutBounds().getWidth();
38     KeyValue initKeyValue=new KeyValue(msg.translateXProperty(),sceneWidth);//-msgWidth
39     KeyFrame initFrame=new KeyFrame(Duration.ZERO,initKeyValue);
40     KeyValue endKeyValue=new KeyValue(msg.translateXProperty(),-msgWidth);//0
41     KeyFrame endFrame=new KeyFrame(Duration.seconds( v: 3),initKeyValue,endKeyValue);
42     Timeline timeline =new Timeline(initFrame,endFrame);
43     timeline.setCycleCount(Timeline.INDEFINITE);
44     timeline.setAutoReverse(true);
45     timeline.setRate(2);
46     timeline.play();
47 }

```

```

49 ▶ public static void main(String[] args) {
50
51     launch(args);
52 }
53 }
54

```

```

pom.xml (JavaFXAnimation)  HelloApplication.java  module-info.java (JavaFXAnimation)
18
19
20 @Override
21 public void start(Stage primaryStage) throws Exception{
22     Media sound = new Media>HelloApplication.class.getResource( name: "AxeLF.mp3").toString());
23     MediaPlayer mediaPlayer = new MediaPlayer(sound);
24     mediaPlayer.play();
25
26     Text msg=new Text( s: "Developing Maintainable Software is Cool");
27     msg.setFont(Font.font( v: 40));
28     msg.setTextOrigin(VPos.TOP);
29
30     Pane root=new Pane(msg);
31     root.setPrefSize( v: 800, v1: 70);
32     primaryStage.setTitle("Animation - Scroller");
33     Scene scene=new Scene(root);
34     primaryStage.setScene(scene);
35     primaryStage.show();

```

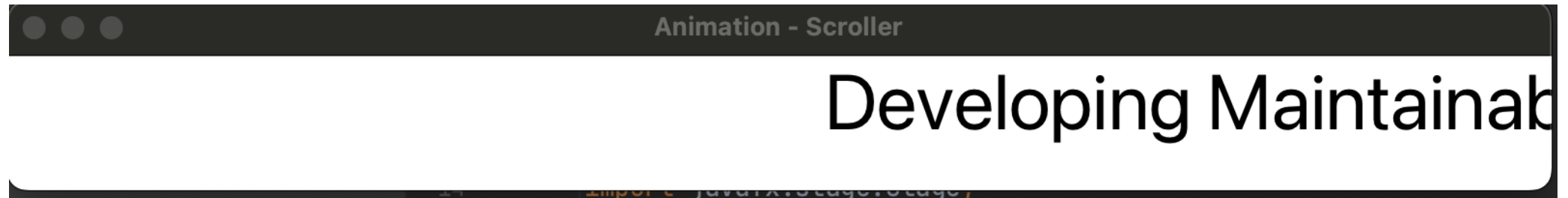


Key Concepts for Timeline Animation



DEMO

- Let's look at an example



- Problem:
 - Scroll text does not update its initial position when the width of the scene changes
- Solution:
 - Update the initial key frame whenever the scene width changes
 - Use a ChangeListener for this

Lab 05 Extension



- Challenge: Multi-threading
 - Modify the "makeCall()" method to run in another thread, emulating a 10 second call. We do not want the phone to "lock up" while we are making a call.
- Challenge: Animation
 - When the user clicks call, add a suitable animation on the front of the interface that represents a call being connected
 - Use the Timeline and KeyFrame classes



- General JavaFX
 - Comprehensive introduction to JavaFX
 - https://www3.ntu.edu.sg/home/ehchua/programming/java/Javafx1_intro.html
 - Multithreading & Concurrent Programming in Java
 - https://www3.ntu.edu.sg/home/ehchua/programming/java/J5e_multithreading.html
- Animation of Text in JavaFX
 - <https://examples.javacodegeeks.com/desktop-java/javafx/javafx-animation-example-2/>



some final remarks ...