

Java Programming

Arthur Hoskey, Ph.D.
Farmingdale State College
Computer Systems Department

- JavaFX Graphics
 - Playing Videos

Today's Lecture

Now on to playing videos...

Playing Videos

Playing Videos

- You can play videos within your JavaFX application.
- You can play, pause, stop etc...
- The instructions in these slides are for playing MP4 videos.
- NASA has free MP4 video downloads at:
<https://www.nasa.gov/multimedia/downloadable-video-page/>

Playing Videos

Setup Project to Play Videos

1. Add the following dependency to Maven (in pom.xml):

<dependency>

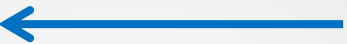
<groupId>org.openjfx</groupId>

<artifactId>javafx-media</artifactId>

<version>13</version>

</dependency>

Change the version to
match the JavaFX
version you are using



2. Make updates to **module-info.java**:

- Add a requires statement to the project's module-info.java file:

requires javafx.media;

- Update the opens statement in module-info.java (adding javafx.media):
opens com.mycompany.testlabjavafxvideo to javafx.fxml, **javafx.media;**

3. Put any mp4 videos that you wish to play in the package directory under resources. For example:

...\src\main\resources\com\mycompany\testlabjavafxvideo

This is the same directory where your .fxml files are located.

Setup Project to Play Videos

- **Media** (not a GUI control) – Used to access the video file that you want to play. Pass it the URL of the file you want to play.
- **MediaPlayer** (not a GUI control) – Manages playing the video. It should be given an instance of Media to operator on.
- **MediaView** (GUI control) – Control used to display the video in a GUI. It should be given an instance of MediaPlayer.

JavaFX Video Related Classes

- The controller will have code to manipulate the media player.

```
@FXML private MediaView mediaView;  
private MediaPlayer mediaPlayer;
```

MediaView is a control in the window.
MediaPlayer class is responsible for playing the video.

Use controller's initialize method for setup

```
public void initialize() {
```

```
    URL url = PrimaryController.class.getResource("myvideo.mp4");
```

Get the url for the video file

```
    Media media = new Media(url.toExternalForm());
```

```
    mediaPlayer = new MediaPlayer(media);
```

```
    mediaView.setMediaPlayer(mediaPlayer);
```

```
}
```

Associate the video file with the MediaPlayer and the MediaView

```
mediaPlayer.play();  
mediaPlayer.pause();  
mediaPlayer.stop();
```

You can use the play, pause, and stop methods on the MediaPlayer to control the playback.

Playing Videos - Startup

- When the video ends you cannot just call play again to restart the video.
- You must put the playback position back to the beginning of the video.
- For example:

```
mediaPlayer.seek(Duration.ZERO);
```

Go to the beginning
of the video



Note: If the video is playing and you seek to position 0 the video will just keep playing. You can call pause right after seeking if you do not want the video to restart. For example:

```
mediaPlayer.seek(Duration.ZERO);  
mediaPlayer.pause();
```

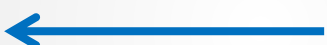
Playing Videos - Restarting

Video Finished Event Handler

- You can add event handlers for when the MediaPlayer enters different states. For example, when a video ends.
- The event handler will be called automatically.
- The code below sets up an event handler for when the video ends.

```
mediaPlayer.setOnEndOfMedia(  
    new Runnable() {  
        public void run() {  
            System.out.println("Video ended");  
        }  
    }  
);
```

Create an anonymous instance of Runnable and pass it to the MediaPlayer instance



Video Event Handlers

Current Time Changed Event Handler

- You can get updates for when the value of the current time changes.
- The invalidated method gets called on the InvalidationListener interface when the current time value is updated.

**Listening to the currentTimeProperty
of the mediaPlayer instance**

```
mediaPlayer.currentTimeProperty().addListener(new InvalidationListener() {  
    public void invalidated(Observable ov)  
    {  
        double minutes, seconds;  
        minutes = mediaPlayer.getCurrentTime().toMinutes();  
        seconds = mediaPlayer.getCurrentTime().toSeconds();  
        // Code to update the time on screen goes here...  
    }  
});
```

**Use the following import for Observable:
import javafx.beans.Observable;**

**Create an anonymous instance of InvalidationListener
and pass it to the currentTimeProperty of the
MediaPlayer instance**

Video Event Handlers

End of Slides