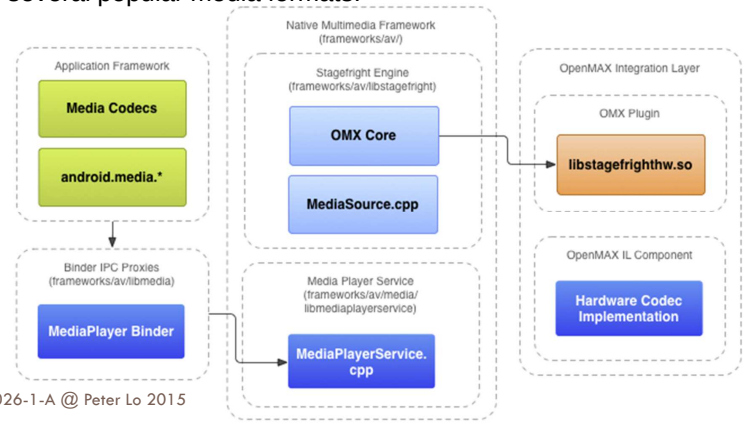
ANDROID APPS DEVELOPMENT FOR MOBILE GAME

Lecture 6: Multimedia

Peter Lo

Media Playback Engine

- Android provides a media playback engine at the native level called Stagefright that comes built-in with software-based codecs for several popular media formats.



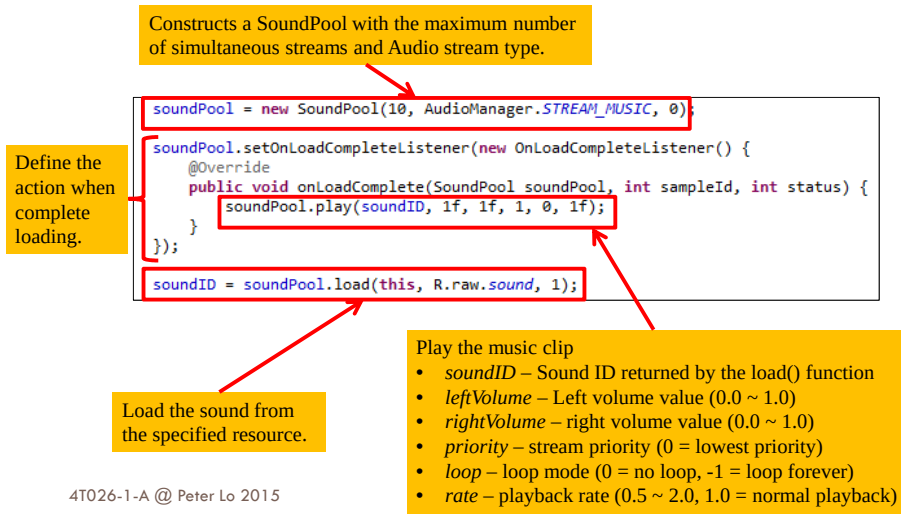
Playing Sound

- Android provides two main API's for playing sounds. The first one via the *SoundPool* class and the other one via the *MediaPlayer* class.
  - **SoundPool** can be used for small audio clips (<1MB). It can repeat sounds and play several sounds simultaneously.
  - **MediaPlayer** is better suited for longer music and movies.

SoundPool

- A *SoundPool* is a collection of samples that can be loaded into memory from a resource inside the APK or from a file in the file system.
- The *SoundPool* library uses the MediaPlayer service to decode the audio into a raw 16-bit PCM mono or stereo stream.
- This allows applications to ship with compressed streams without having to suffer the CPU load and latency of decompressing during playback.

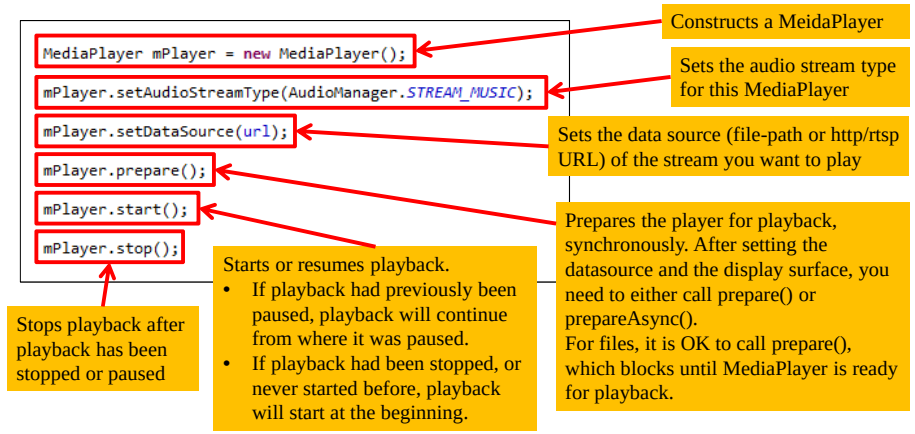
Using SoundPool



MediaPlayer

- One of the most important components of the media framework is the *MediaPlayer* class.
- An object of this class can fetch, decode, and play both audio and video with minimal setup.
- It supports several different media sources such as:
  - Local resources (**raw** resources)
  - Internal URIs, such as one you might obtain from a Content Resolver
  - External URLs (streaming)

Using MediaPlayer



Volume Control

- Android supports different audio streams for different purposes.
- The phone volume button can be configured to control a specific audio stream,
  - E.g. during a call the volume button allow increase / decrease the caller volume.
- To set the button to control the sound media stream set the audio type in your application.

```
context.setVolumeControlStream(AudioManager.STREAM_MUSIC);
```

Manifest Declarations

- Before starting development using MediaPlayer, manifest has the appropriate declarations to allow use of related features.
  - **Internet Permission** – For using MediaPlayer to stream network-based content, the application must request network access.

```
<uses-permission android:name="android.permission.INTERNET" />
```

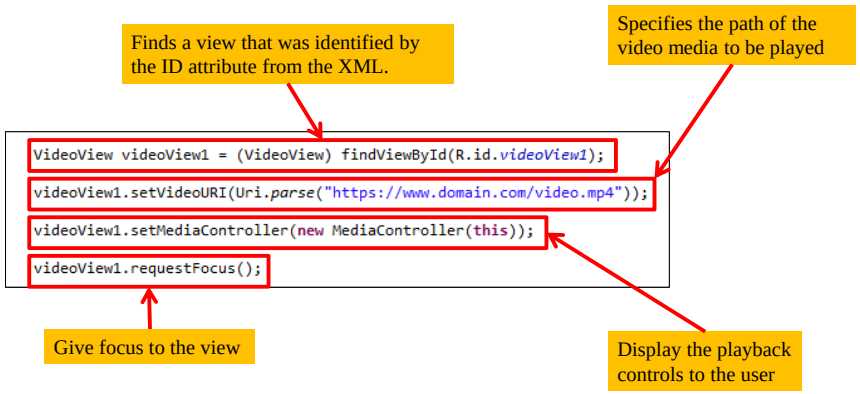
  - **Wake Lock Permission** – If player application needs to keep the screen from dimming or the processor from sleeping, must request this permission.

```
<uses-permission android:name="android.permission.WAKE_LOCK" />
```


Video Playback

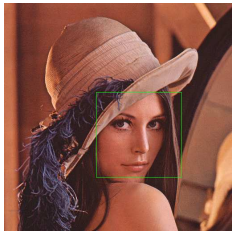
- The VideoView class can load video from various sources, takes care of computing its measurement from the video so that it can be used in any layout manager, and provides various display options such as scaling and tinting.
- VideoView does not retain its full state when going into the background. In particular, it does not restore the current play state, play position, selected tracks.

Using VideoView



Face Detection

- Face detection is a computer technology that determines the locations and sizes in arbitrary images. *Do not confuse it with face recognition*.
- Face detection extracts people's faces in images but face recognition tries to find out who they are.



Face Detection in Android

- Through two main APIs, Android provides a simple way for you to identify the faces of people in a bitmap image, with each face containing all the basic location information.
  - *android.media.FaceDetector:* Identifies the faces of people in a Bitmap graphic object.
  - *android.media.FaceDetector.Face:* Contains all the information identifying the location of a face in a bitmap.

Face Detection API

- The class *android.media.FaceDetector* is used to detect faces on the image.
  - To detect faces call *findFaces* method of *FaceDetector* class.
  - *findFaces* method returns a number of detected faces and fills the *FaceDetector.Faces[]* array.
- Each instance of the *FaceDetector.Face* class contains the following information:
  - Confidence that it's actually a face – a float value between 0 and 1.
  - Distance between the eyes – in pixels.
  - Position (x, y) of the mid-point between the eyes.
  - Pose rotations (X, Y, Z).

Controlling Vibration

- Vibration is an excellent way to provide haptic user feedback.
- Applications needs the VIBRATE permission in application manifest:

```
<uses-permission android:name="android.permission.VIBRATE"/>
```

- Example:

```
String vibratorService = Context.VIBRATOR_SERVICE;
Vibrator vibrator = (Vibrator)getSystemService(vibratorService);

long[] pattern = {1000, 2000, 4000, 8000, 16000 };
vibrator.vibrate(pattern, 0); // Execute vibration pattern.
vibrator.vibrate(1000); // Vibrate for 1 second.
```