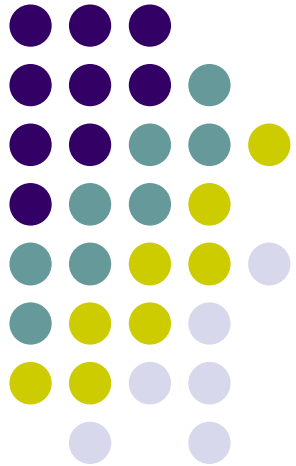
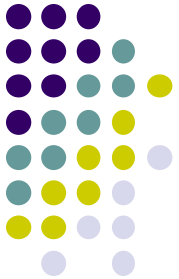
CS 4518 Mobile and Ubiquitous Computing

Lecture 9: Playing Sound and Video

Emmanuel Agu

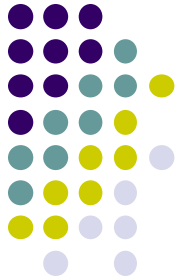




Playing Audio and Video

MediaPlayer

http://developer.android.com/guide/topics/media/mediaplayer.html

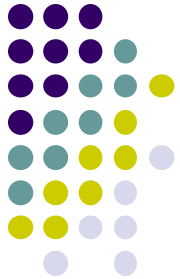


- Classes used to play sound and video in Android
  - **MediaPlayer:** Plays sound and video
  - **AudioManager:** plays audio

- MediaPlayer:
  - Can playback audio/video files & streams
  - Audio/video files stored in app's resource folders (e.g. **res/raw/)**
  - Using MediaPlayer APIs, app can easily integrate video/audio playback functionality

MediaPlayer

http://developer.android.com/guide/topics/media/mediaplayer.html



- A MediaPlayer object can fetch, decode and play audio and video from:
  - Local resources (e.g. app's res/raw folder)
  - External URLs (over the Internet)

- MediaPlayer supports:
  - **Streaming network protocols:** RTSP, HTTP streaming
  - **Media Formats:**
    - Audio (AAC, MP3, MIDI, etc),
    - Image (JPEG, GIF, PNG, BMP, etc)
    - Video (H.263, H.264, H.265 AVC, MPEG-4, etc)

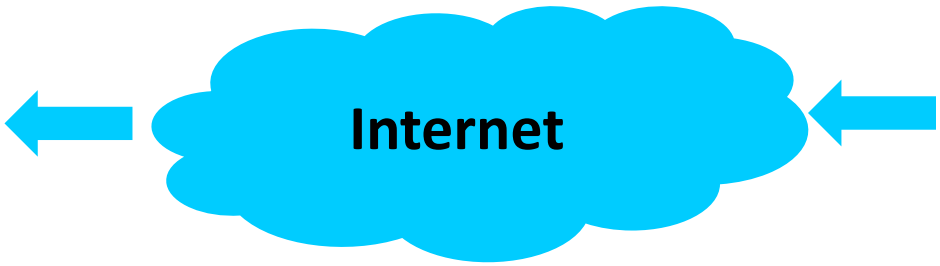
Using Media Player:

http://developer.android.com/guide/topics/media/mediaplayer.html

Step 1: Request Permission in AndroidManifest or Place video/audio files in res/raw

- If streaming video/audio over Internet (network-based content), request network access permission in AndroidManifest.xml:

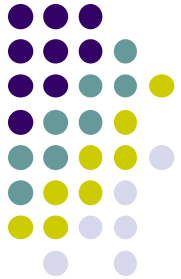
```
<uses-permission android:name="android.permission.INTERNET" />
```



- If playing back local file stored on user's smartphone, put video/audio files in **res/raw** folder

Using MediaPlayer

Step 2: Create MediaPlayer Object, Start Player



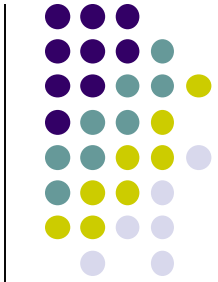
- To play audio file saved in app's **res/raw/** directory

```
MediaPlayer mediaPlayer = MediaPlayer.create(context, R.raw.sound_file_1);
mediaPlayer.start(); // no need to call prepare(); create() does that for you
```

- **Note:** Audio file called by create must be encoded in one of supported media formats

Using MediaPlayer

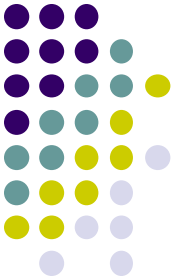
Step 2: Create MediaPlayer Object, Start Player



- To play audio from remote URL via HTTP streaming over the Internet

```
String url = "http://........"; // your URL here
MediaPlayer mediaPlayer = new MediaPlayer();
mediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);
mediaPlayer.setDataSource(url);
mediaPlayer.prepare(); // might take long! (for buffering, etc)
mediaPlayer.start();
```

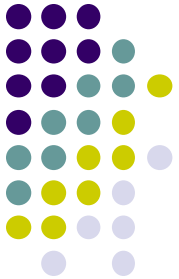
Releasing the MediaPlayer



- MediaPlayer can consume valuable system resources
- When done, call **release( )** to free up system resources
- In **onStop( )** or **onDestroy( )** methods, call

```
mediaPlayer.release();
mediaPlayer = null;
```

- **MediaPlayer in a Service:** Can play media (e.g. music) in background while app is not running
  - Start MediaPlayer as service



Playing Audio File using MediaPlayer Example from Android Nerd Ranch 1st edition

MediaPlayer Example to Playback Audio

from Android Nerd Ranch (1st edition) Ch. 13

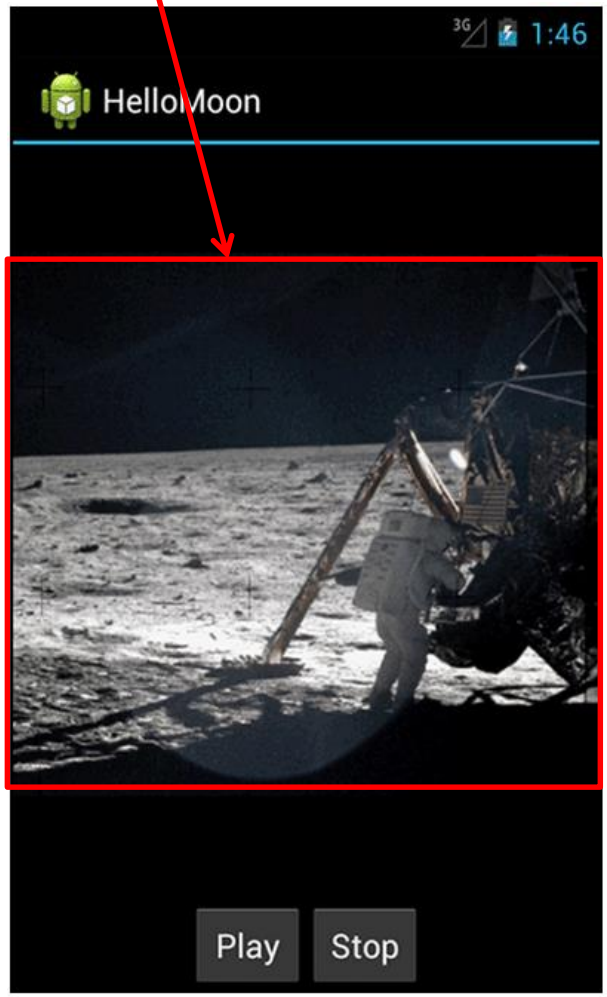
- **HelloMoon app** that uses **MediaPlayer** to play audio file

HelloMoon App

- Put image **armstrong_on_moon.jpg** in **res/drawable-mdpi/** folder
- Place audio file to be played back (**one_small_step.wav**) in **res/raw** folder
- Create **strings.xml** file for app

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">HelloMoon</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="hellomoon_play">Play</string>
    <string name="hellomoon_stop">Stop</string>
    <string name="hellomoon_description">Neil Armstrong stepping
            onto the moon</string>

</resources>
```

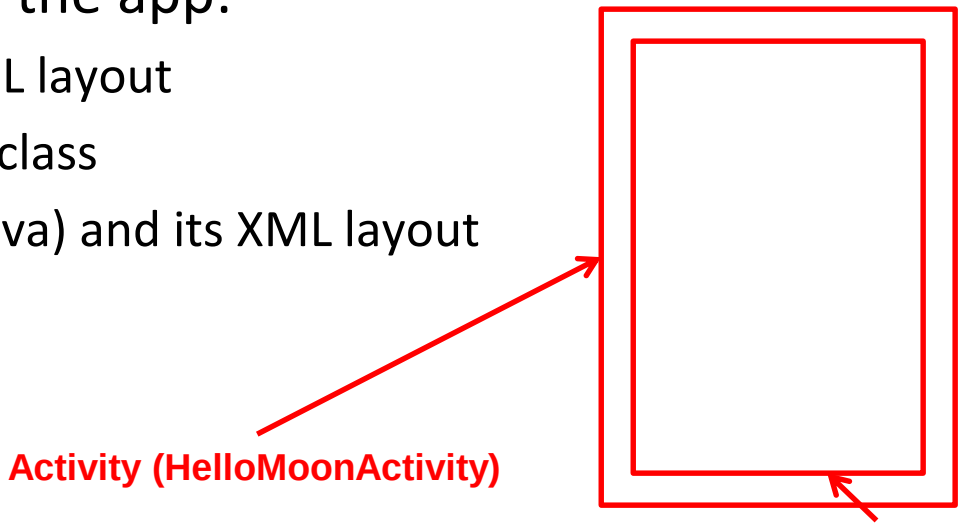
armstrong_on_moon.jpg



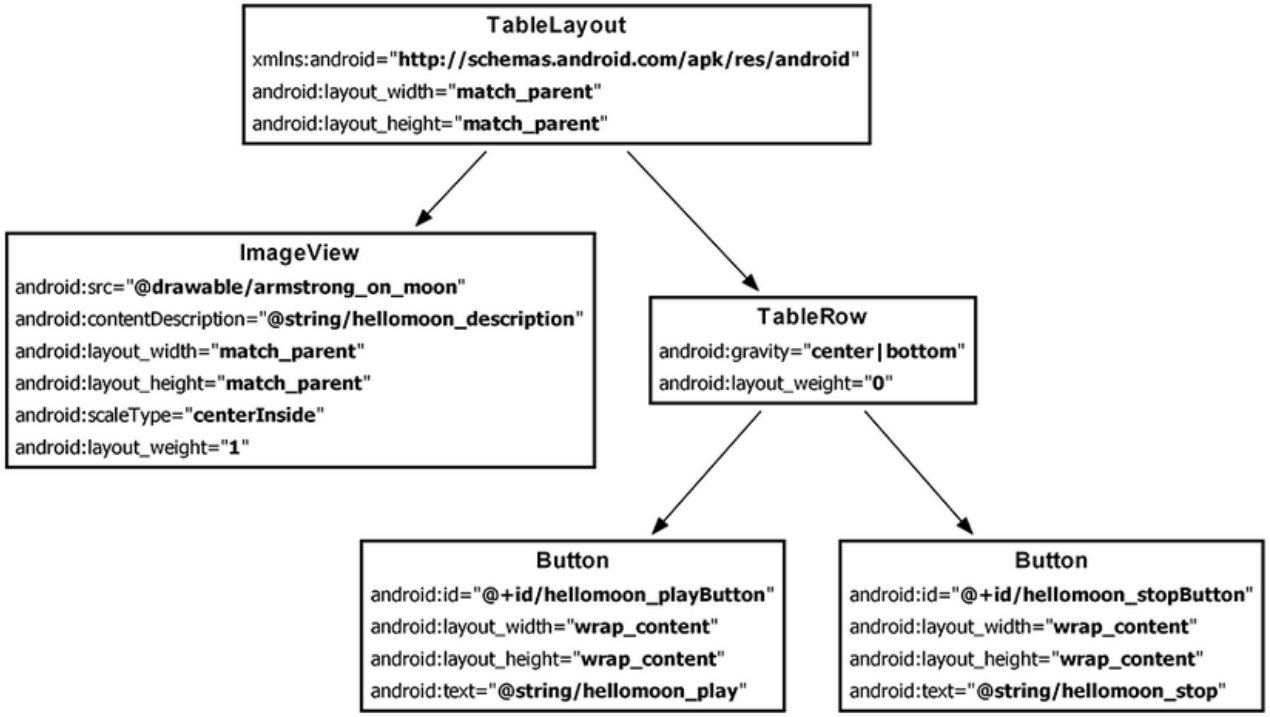
HelloMoon App

- HelloMoon app will have:
  - 1 activity (**HelloMoonActivity**) that hosts **HelloMoonFragment**
- **AudioPlayer** class will be created to encapsulate **MediaPlayer**

- First set up the rest of the app:
  1. Define fragment's XML layout
  2. Create fragment java class
  3. Modify the activity (java) and its XML layout to host the fragment



Defining the Layout for HelloMoonFragment



Define XML for HelloMoon UI (fragment_hello_moon.xml)

Creating a Layout Fragment

- Previously added Fragments to activity's java code
- **Layout fragment:** Can also add fragments to hosting Activity's XML file
- We will use a layout fragment instead
- Create activity's XML layout (**activity_hello_moon.xml**)
- **Activity's** XML layout file contains/hosts fragment

```
<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/helloMoonFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:name="com.bignerdranch.android.hellomoon.HelloMoonFragment">

</fragment>
```

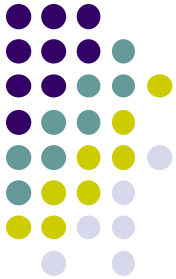
Set up HelloMoonFragment.java

```
public class HelloMoonFragment extends Fragment {

    private Button mPlayButton;
    private Button mStopButton;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup parent,
            Bundle savedInstanceState) {
        View v = inflater.inflate(R.layout.fragment_hello_moon, parent, false);

        mPlayButton = (Button)v.findViewById(R.id.hellomoon_playButton);
        mStopButton = (Button)v.findViewById(R.id.hellomoon_stopButton);

        return v;
    }
}
```

Inflate view in onCreateView( )

Get handle to Start, Stop buttons

Create AudioPlayer Class encapsulates MediaPlayer



```
public class AudioPlayer {

    private MediaPlayer mPlayer;

    public void stop() {
        if (mPlayer != null) {
            mPlayer.release();
            mPlayer = null;
        }
    }

    public void play(Context c) {
        mPlayer = MediaPlayer.create(c, R.raw.one_small_step);
        mPlayer.start();
    }
}
```

Hook up Play and Stop Buttons

```
public class HelloMoonFragment extends Fragment {
    private AudioPlayer mPlayer = new AudioPlayer();
    private Button mPlayButton;
    private Button mStopButton;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup parent,
            Bundle savedInstanceState) {
        View v = inflater.inflate(R.layout.fragment_hello_moon, parent, false);

        mPlayButton = (Button)v.findViewById(R.id.hellomoon_playButton);
        mPlayButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                mPlayer.play(getActivity());
            }
        });

        mStopButton = (Button)v.findViewById(R.id.hellomoon_stopButton);
        mStopButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                mPlayer.stop();
            }
        });
        return v;
    }
```

References

- Head First Android
- Android Nerd Ranch, 2nd edition
- Busy Coder's guide to Android version 6.3
- CS 65/165 slides, Dartmouth College, Spring 2014
- CS 371M slides, U of Texas Austin, Spring 2014