SNS COLLEGE OF TECHNOLOGY

(An Autonomous Institution)

Re-accredited by NAAC with A+ grade, Accredited by NBA(CSE, IT, ECE, EEE & Mechanical)
Approvedy by AICTE, New Delhi, Recognized by UGC, Affiliated to Anna University, Chennai

Department of MCA

MULTIMEDIA

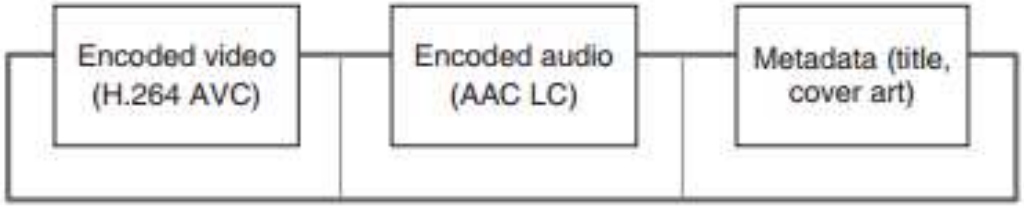
Course: **Mobile Application Development**

Unit : IV –Sprucing Up Mobile Apps

Class / Semester: II MCA / III Semester

Understanding Multimedia files

- MP4 / any other video file is just like a zip file that can package (container format) video elements, audio elements, and metadata that provide information about the title or cover art of the video
- Keeping audio and video elements together is usually referred to as multiplexing
- Audio/video, in its raw form, is typically very large that makes it unsuitable for storage or transport; needs to be compressed with algorithms, referred as codecs (compressor–decompressor)
- Android platform supports several media codecs and container formats1 to provide capabilities to play and record a wide variety of audio, video, and images



MPGE-4 container

Understanding Multimedia files

- Once media is available in a container, it is ready to play
- To play it, first audio and video elements get separated (de-multiplexed)
- Individual elements are then decompressed back to the raw format using respective codec
- finally processed by the smart device in order to play
- Decompressing the video elements results in displaying a series of images on the screen, and
- Decompressing the audio elements results in a sound stream on the speakers of the device
- Both these elements are synchronized with the help of metadata

Introduction

- Android provides multimedia capabilities to developer via APIs
- With API, Build apps with features that require interacting, recording, playing, and storing audio, video, and image elements
- Media file may be stored on the device itself as an app resource, or streamed over a network
- A developer may simply use the built-in app of Android to playback the media, but to get more flexibility and control on playback, in-app mechanism comes handy.
- In-app mechanism is implemented using MediaPlayer API, which is pivotal to audio and media playback

In App Mechanism

- Built-in app of Android is available to playback the media, but to get more flexibility and control on playback, in-app mechanism comes handy
- in-app mechanism is implemented using MediaPlayer API
- use an implicit Intent to start the intended music player app
- The setDataAndType() method is used to set the data source and the type of audio

```
Intent i = new Intent(Intent.ACTION_VIEW);
Uri uri = Uri.parse("http://www.mobmusicstore.com/music.mp3");
 i.setDataAndType(uri,"audio/*");
 startActivity(i);
```

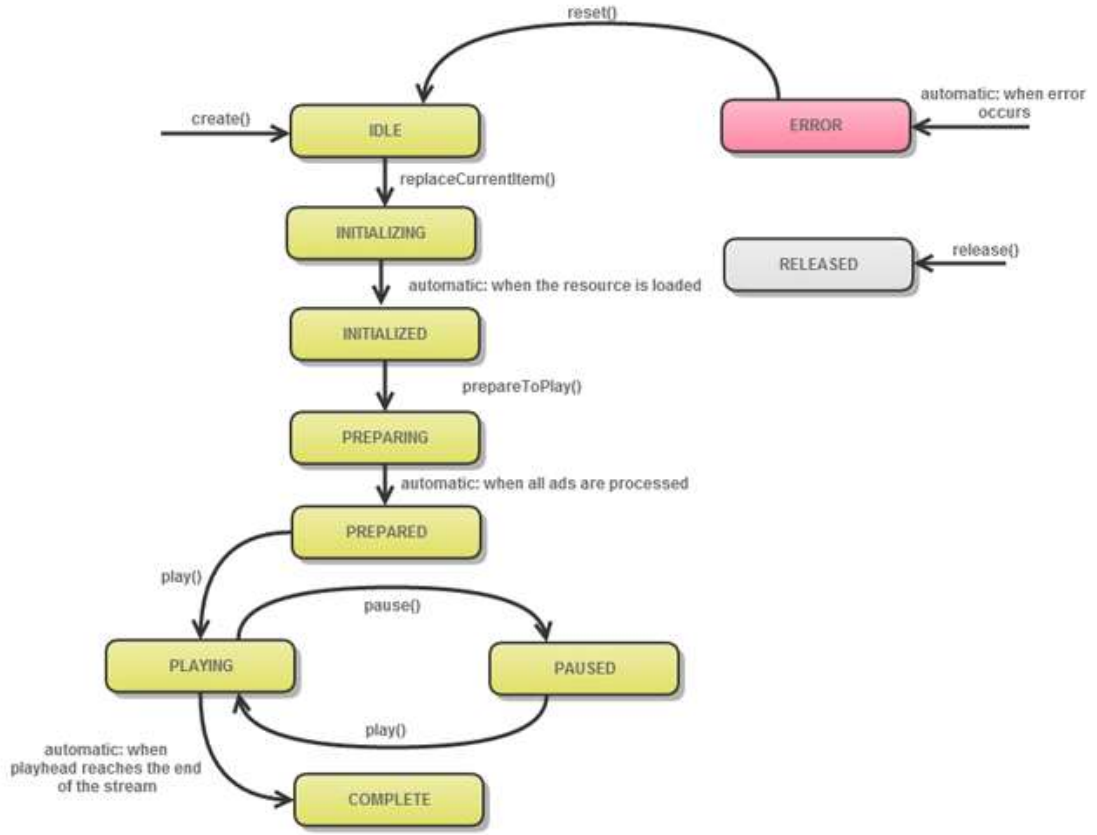
Playing audio file

Playing video file

```
Intent i = new Intent(Intent.ACTION_VIEW);
Uri uri = Uri.parse ("http://www.mobmusicstore.com/video.mp4");
i.setDataAndType(uri,"video/*");
startActivity(i);
```

In-app Mechanism

- MediaPlayer API is the underlying engine to retrieve decode, and playback the media elements
- MediaPlayer object behaves as a state machine and has it own life cycle and states



In-app Mechanism

- **idle** state is the first state , player object gets into, as soon as it is created
- **initialized** state is to set the data source – a file path or an HTTP URL of the media stream. setDataSource() method is used to achieve this
- media stream needs to be fetched and decoded so that it can be prepared for playback by using prepareAsync() during **prepared** state
- Once it is prepared, can be started using start(), followed by play() to playback the media stream
- The media stream can now be paused or stopped by calling pause() or stop()
- call release(), once the MediaPlayer object is not required anymore, which takes it to the **end** state
- An invalid method call from any state may result in errors and exceptions

In-app Mechanism

- key life-cycle methods and states of the MediaPlayer object are listed

Methods	States
`setDataSource()`	Initialized
`prepareAsync()`	Prepared
`start()`	Started
`pause()`	Paused
`stop()`	Stopped
`release()`	End

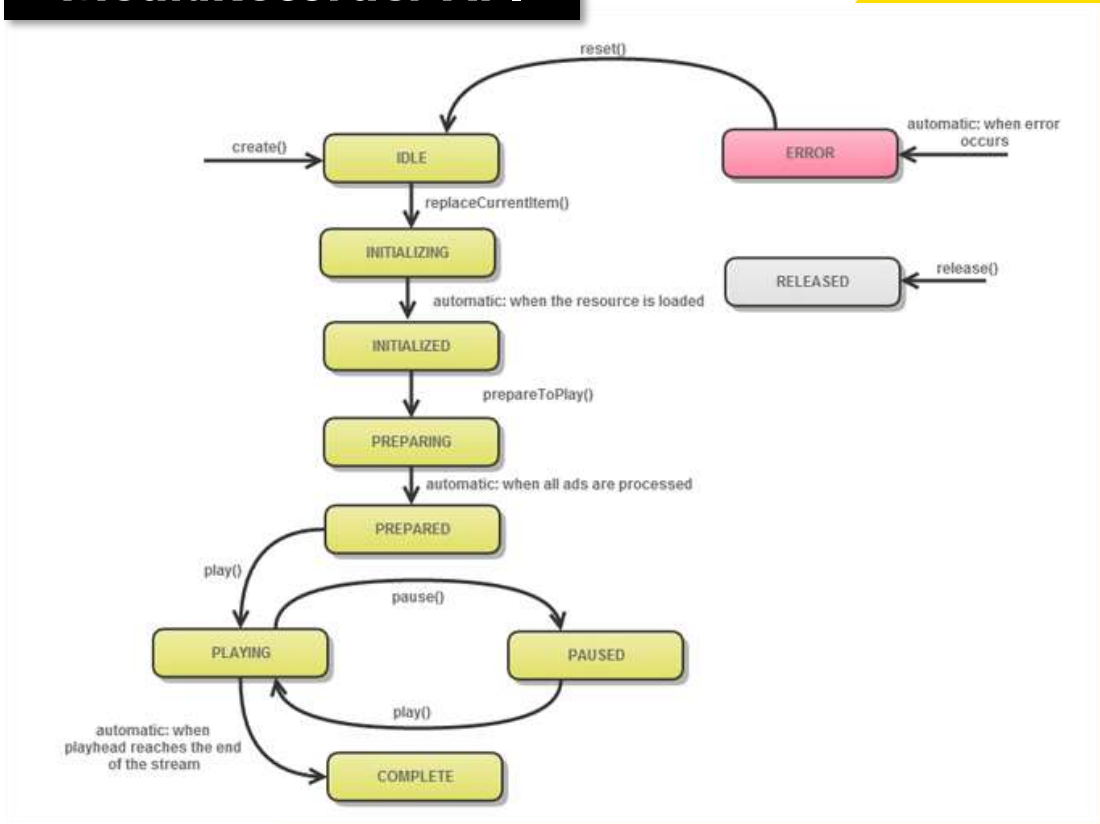
```
MediaPlayer mediaPlayer = MediaPlayer.create(this,Uri.parse (“http://www.mobmusicstore.com/music.mp3”));
if(mediaPlayer==null)
{
 Toast.makeText(this, “Unable to create Media player”, 4000).show();}
 else
Toast.makeText(this, “Playing song”, Toast.LENGTH_LONG).show();
mediaPlayer.start();
```

app requires an INTERNET permission,

```
<uses-permission android:name=“android.permission.INTERNET”/>
 <uses-permission android:name=“android.permission.WAKE_LOCK”/>
```

MediaRecorder API

MediaRecorder API



REFERENCES

- Anubhav Pradhan, Anil V Deshpande, “Composing Mobile Apps using Android”, Wiley Edition, 2014
- http://www.dre.vanderbilt.edu/~schmidt/android/android-4.0/out/target/ common/docs/doc-comment-check/guide/topics/graphics/2d-graphics.html
- https://developer.android.com/guide/topics/graphics/drawable-animation