Video playing

Android Video Player

- **MediaController and VideoView**
- **MediaController class**
- The `android.widget.MediaController` is a view that contains media controls like play/pause, previous, next, fast-forward, rewind etc.
- **VideoView class**
- The `android.widget.VideoView` class provides methods to play and control the video player.

Methods of VideoView class

- **public void setMediaController(MediaController controller)** : sets the media controller to the video view.
- **public void setVideoURI (Uri uri) :** sets the URI of the video file.
- **public void start()** : starts the video view.
- **public void stopPlayback()** : stops the playback.
- **public void pause():** pauses the playback.
- **public void suspend():** suspends the playback.
- **public void resume()** : resumes the playback.
- **public void seekTo(int millis)** : seeks to specified time in miliseconds.

setVideoUri(Uri uri):

•initiate a video view

```
•VideoView simpleVideoView = (VideoView) findViewById(R.id.simpleVideoView);
simpleVideoView.setVideoURI(Uri.parse("android.resource://" + getPackageName() + "/"
+ R.raw.fishvideo));
```

```
Uri uri = Uri.parse("/ui/wp-content/uploads/2016/04/videoviewtestingvideo.mp4");
VideoView simpleVideoView = (VideoView) findViewById(R.id.simpleVideoView); //
initiate a video view
simpleVideoView.setVideoURI(uri);
 simpleVideoView.start();
```

•Note :

•<!--Add this before application tag in AndroidManifest.xml-->

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

URI stands for Uniform Resource Identifier

It is a string of characters used to identify the resource

```
Uri uripath = Uri.parse("resource to be identified" )
```

- Resource can be location of the place, a person's name, a phone call number, email id etc.
- Commonly used uri are tel for identifying phone call number, maps for identifying location of the place, mailto for identifying email id

setMediaController(MediaController controller)

- // create an object of media controller

```
MediaController mediaController = new MediaController(this);
```

- // initiate a video view

```
VideoView simpleVideoView = (VideoView)
findViewById(R.id.simpleVideoView);
```

- // set media controller object for a video view

```
simpleVideoView.setMediaController(mediaController);
```

setOnPreparedListener(MediaPlayer.OnPreparedListener):

- Allows a callback method to be called when the video is ready to play.

```
/ initiate a video view
VideoView simpleVideoView = (VideoView) findViewById(R.id.simpleVideoView);
/ perform set on prepared listener event on video view
simpleVideoView.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {
@Override
public void onPrepared(MediaPlayer mp) {

    // do something when video is ready to play
}
});
```

setOnErrorListener(MediaPlayer.OnErrorListener)

- Allows a callback method to be called when an error occurs during the video playback. Below we show the use of setOnErrorListener event of a video view.

/ initiate a video view

```
VideoView simpleVideoView = (VideoView) findViewById(R.id.simpleVideoView);
```

/ perform set on error listener event on video view

```
simpleVideoView.setOnErrorListener(new MediaPlayer.OnErrorListener() {
@Override
public boolean onError(MediaPlayer mp, int what, int extra) {

    // do something when an error is occur during the video playback
    return false;
}
});
```

Methods Of MediaController

1.setAnchorView(View view): setAnchorView is used to designates the view to which the controller is to be anchored. This controls the location of the controls on the screen.

```
MediaController mediaController = new MediaController(this); // create an object of media controller

mediaController.setAnchorView(simpleVideoView); // set anchor view for video view
```

2. show(): Used to show the controller on the screen.

```
MediaController mediaController = new MediaController(this); // create an object of media controller

mediaController.show(); // show the controller on the screen
```

Methods Of MediaController

3. **show(int timeout):** This method is used to set the time to show the controller on the screen.

```
MediaController mediaController = new MediaController(this); // create an object of media controller

mediaController.show(500); // set the time to show the controller on the screen
```

4. hide(): This method is used to hide the controls from the screen.

```
MediaController mediaController = new MediaController(this); // create an object of media controller

mediaController.hide(); // hide the control from the screen
```

- **Write an android application to play a video with mediacontroller.**

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
        <VideoView
            android:id="@+id/video_view"
            android:layout_width="match_parent"
            android:layout_height="match_parent" />
</FrameLayout>
```

MainActivity.java

```
package com.example.videoplaying;
import androidx.appcompat.app.AppCompatActivity;
import android.net.Uri;
import android.os.Bundle;
import android.widget.MediaController;
import android.widget.VideoView;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        VideoView videoView =
findViewById(R.id.video_view);
        //Storing video resource in string variable
        String videoPath = "android.resource://" +
getPackageName() + "/" + R.raw.video;
        Uri uri = Uri.parse(videoPath);
        videoView.setVideoURI(uri);

        MediaController mediaController = new
MediaController(this);
        videoView.setMediaController(mediaController);
        mediaController.setAnchorView(videoView);
    }
}
```