LabW08 – Media Access

Objectives:

1. Understand the multimedia components for Android SDK

Tasks:

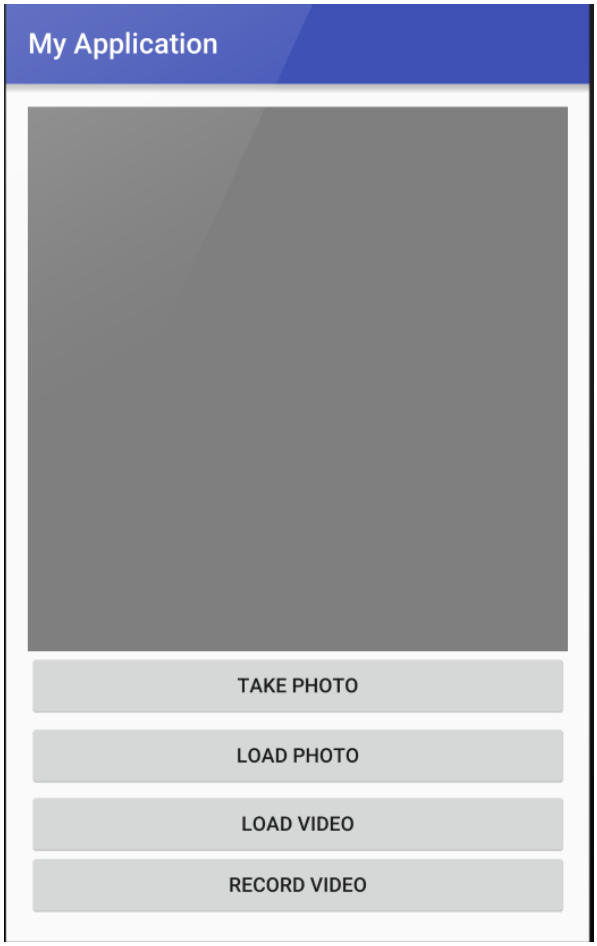
1. Use a basic layout
2. Take a photo using the phone's built-in Camera app
3. Select a photo from the gallery
4. Select a video from the gallery
5. Record a video
6. Handle the media files
7. Audio recording
8. Implement your own camera interface [Optional]

The Android multimedia framework includes support for playing variety of common media types, so that you can easily integrate audio, video and images into your applications. You can play audio or video from media files stored in your application's resources (raw resources), from standalone files in the filesystem, or from a data stream arriving over a network connection.

This tutorial shows you how to write a media-playing application that interacts with the user and the system in order to obtain good performance and a pleasant user experience.

Task 1: Use a basic layout

This layout has four buttons at the bottom, and have a VideoView and ImageView (overlapping each other) on the top.



1. The root layout is a RelativeLayout
2. Add a button for "**Record Video**" at the bottom using android:layout_alignParentBottom="true".
   Also assign an id to it: @+id/recordvideo.

3. Add a button for "**Load Video**" above the "**Record Video**" button using android:layout_above="@+id/recordvideo"
4. Do the same for "**Load Photo**" and "**Take Photo**"
5. Add an VideoView above the "take photo" button. Also add an ImageView above the "**Take Photo**" button.
6. If you are stuck, have a look at the sample layout at **activity_week08.xml** (in lab files)

Task 2: Take a photo using the phone's built-in Camera app

1. In a new Activity, use the layout above. And then define a few global request codes. The values can be any integer, but each one must be different from others.

```
public final String APP_TAG = "MobileComputingTutorial";
public final static int CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE = 1034;
public final static int CAPTURE_VIDEO_ACTIVITY_REQUEST_CODE = 1035;
public final static int PICK_PHOTO_CODE = 1046;
public final static int PICK_VIDEO_CODE = 1047;
public String photoFileName = "photo.jpg";
public String videoFileName = "video.mp4";
```

2. Add a helper method. Given a filename, this method returns the Uri for this file, which will be generated when a photo is taken, or a video is recorded.

```
    // Returns the Uri for a photo/media stored on disk given the fileName
    public Uri getFileUri(String fileName) {
        // Get safe storage directory for photos
        File mediaStorageDir = new File(
Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES),
                APP_TAG);

        // Create the storage directory if it does not exist
        if (!mediaStorageDir.exists() && !mediaStorageDir.mkdirs()) {
            Log.d(APP_TAG, "failed to create directory");
        }

        // Return the file target for the photo based on filename
        return Uri.fromFile(new File(mediaStorageDir.getPath() + File.separator
                + fileName));
    }
```

3. Add a method when the “**Take Photo**” button is clicked

```
public void onTakePhotoClick(View v) {
            // create Intent to take a picture and return control to the calling
            // application
            Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
            // set file name
            intent.putExtra(MediaStore.EXTRA_OUTPUT, getFileUri(photoFileName));

            // Start the image capture intent to take photo
            startActivityForResult(intent, CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE);
      }
```

4. Add attribute android:onClick=“onTakePhotoClick" to the “**Take Photo**” button.

5. Run it, preferably on a real phone. If run it on the emulator, the emulated camera will be used. But nothing happens when the photo is captured, because the onActivityResult() is not implemented yet.

Task 3: Select a photo from the gallery

1. Add a method when the “**Load Photo**” button is clicked

```
      public void onLoadPhotoClick(View view) {
            // Create intent for picking a photo from the gallery
            Intent intent = new Intent(Intent.ACTION_PICK,
                        MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
            // Bring up gallery to select a photo
            startActivityForResult(intent, PICK_PHOTO_CODE);
      }
```

2. Add attribute android:onClick=“onLoadPhotoClick" to the “**Load Photo**” button.

Task 4: Select a video from the gallery

1. Add a method when the "**Load Video**" button is clicked

```
public void onLoadVideoClick(View view) {
        // Create intent for picking a video from the gallery
        Intent intent = new Intent(Intent.ACTION_PICK,
                        MediaStore.Video.Media.EXTERNAL_CONTENT_URI);
        // Bring up gallery to select a video
        startActivityForResult(intent, PICK_VIDEO_CODE);
}
```

2. Add attribute android:onClick="onLoadVideoClick" to the "**Load Video**" button.

Task 5: Record a video

1. Add a method when the "**Record Video**" button is clicked

```
public void onRecordVideoClick(View v) {
        if (getPackageManager().hasSystemFeature(PackageManager.FEATURE_CAMERA_FRONT)) {
        Intent intent = new Intent(MediaStore.ACTION_VIDEO_CAPTURE);
        File mediaFile = new File(
           Environment.getExternalStorageDirectory().getAbsolutePath() + "/myvideo.mp4");
        intent.putExtra(MediaStore.EXTRA_OUTPUT, getFileUri(videoFileName));
        startActivityForResult(intent, CAPTURE_VIDEO_ACTIVITY_REQUEST_CODE);
    } else {
        Toast.makeText(this, "No camera on device", Toast.LENGTH_LONG).show();
    }
}
```

2. Add attribute android:onClick="onRecordVideoClick" to the "**Load Video**" button.

Task 6: Handle the media files

Add the onActivityResult() method to handle the results from the above actions. Read the inline comments.
For a photo, the VideoView is hidden, and the photo is placed in the ImageView.
For a video, the ImageView is hidden, and the video is started in the VideoView.

```
@Override
        public void onActivityResult(int requestCode, int resultCode, Intent data) {

                final VideoView mVideoView = (VideoView) findViewById(R.id.videoview);
                ImageView ivPreview = (ImageView) findViewById(R.id.photopreview);

                mVideoView.setVisibility(View.GONE);
                ivPreview.setVisibility(View.GONE);


                if (requestCode == CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE) {
                        if (resultCode == RESULT_OK) {
                                Uri takenPhotoUri = getFileUri(photoFileName);
                                // by this point we have the camera photo on disk
                                Bitmap takenImage = BitmapFactory.decodeFile(takenPhotoUri
                                                .getPath());
                                // Load the taken image into a preview
                                ivPreview.setImageBitmap(takenImage);
                                ivPreview.setVisibility(View.VISIBLE);
                        } else { // Result was a failure
                                Toast.makeText(this, "Picture wasn't taken!",
                                                Toast.LENGTH_SHORT).show();
                        }
                } else if (requestCode == PICK_PHOTO_CODE) {
                        if (resultCode == RESULT_OK) {
                                Uri photoUri = data.getData();
                                // Do something with the photo based on Uri
                                Bitmap selectedImage;
                                try {
                                        selectedImage = MediaStore.Images.Media.getBitmap(
                                                        this.getContentResolver(), photoUri);
                                        // Load the selected image into a preview

                                        ivPreview.setImageBitmap(selectedImage);
                                        ivPreview.setVisibility(View.VISIBLE);
                                } catch (FileNotFoundException e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                                } catch (IOException e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                                }

                        }

                } else if (requestCode == PICK_VIDEO_CODE) {
                        if (resultCode == RESULT_OK) {
                                Uri videoUri = data.getData();

                                mVideoView.setVisibility(View.VISIBLE);
                                mVideoView.setVideoURI(videoUri);
                                mVideoView.requestFocus();
                                mVideoView.setOnPreparedListener(new OnPreparedListener() {
                                    // Close the progress bar and play the video
                                    public void onPrepared(MediaPlayer mp) {
                                        mVideoView.start();
                                    }
                                });

                        }

                }else if (requestCode == CAPTURE_VIDEO_ACTIVITY_REQUEST_CODE) {
                        if (resultCode == RESULT_OK) {
                                Uri takenVideoUri = getFileUri(videoFileName);
                                mVideoView.setVisibility(View.VISIBLE);
                                mVideoView.setVideoURI(takenVideoUri);
                                mVideoView.requestFocus();
                                mVideoView.setOnPreparedListener(new OnPreparedListener() {
                                    // Close the progress bar and play the video
                                    public void onPrepared(MediaPlayer mp) {
                                        mVideoView.start();
                                    }
                                });
                        }
                }
```

3. Add the permission for writing and reading external storage space (not in the private app folder). Also add the <Activity> tag inside the <Application> tag.

```
<!-- Week08 -->
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

4. Run it. The final code will be the same as that in **ActivityWeek08.java** (in lab files)

Task 7: Audio recording

This task is based on the following tutorial:
https://developer.android.com/guide/topics/media/audio-capture.html

The Android multimedia framework includes support for capturing and encoding a variety of common audio formats, so that you can easily integrate audio into your applications. You can record audio using the **MediaRecorder** APIs if supported by the device hardware.

This document shows you how to write an application that captures audio from a device microphone, save the audio and play it back.

Note: The Android Emulator does not have the ability to capture audio, but actual devices are likely to provide these capabilities

Audio capture from the device is a bit more complicated than audio and video playback, but still fairly simple:

1. Create a new instance of **android.media.MediaRecorder**.
2. Set the audio source using **MediaRecorder.setAudioSource()**. You will probably want to use **MediaRecorder.AudioSource.MIC**.
3. Set output file format using **MediaRecorder.setOutputFormat()**.
4. Set output file name using **MediaRecorder.setOutputFile()**.
5. Set the audio encoder using **MediaRecorder.setAudioEncoder()**.
6. Call **MediaRecorder.prepare()** on the **MediaRecorder** instance.
7. To start audio capture, call **MediaRecorder.start()**.

8. To stop audio capture, call **MediaRecorder.stop()**.

9. When you are done with the **MediaRecorder** instance, call **MediaRecorder**.**release**() on it. Calling **MediaRecorder**.**release**() is always recommended to free the resource immediately.

Example code please refer to **AudioRecordTest.java** in lab file.

Task 8 [Optional]: Implement your own camera interface

If you want to embed a camera view in your own app instead of using the built-in Camera support, read the explanation below (very complicated, do it only if you want to build the next Instagram):
http://developer.android.com/guide/topics/media/camera.html

Some contents are adapted from:
https://github.com/thecodepath/android_guides/wiki/Accessing-the-Camera-and-Stored-Media