CMSC436: Programming Handheld Systems

Fall 2017
Multimedia
Today’s Topics

Multimedia Support Classes
Playing Audio
Watching Video
Recording Audio
Multimedia

Android provides support for encoding and decoding a variety of common media formats. Allows you to play & record audio, still images & video.
Some Multimedia Classes

AudioManager & SoundPool
RingtoneManager & Ringtone
MediaPlayer
MediaRecorder
Camera
AudioManager

Manages volume, system sound effects, and ringer mode control

Acquire AudioManager instance via

   Context.getSystemService(Context.AUDIO_SERVICE)
AudioManager

Load & play sound effects
Manage volume
Manage peripherals
SoundPool

Represents a collection of audio samples (streams)
Can mix and play multiple simultaneously
AudioVideoAudioManager

Presents two buttons that adjust the volume up or down

Presents a play button that, when pressed, plays a bubble popping sound at the current volume level
AudioVideoAudioManager

AudioVideoAudioManager

6

Down
Up
Play

10

Down
Up
Play
Ringtone and RingtoneManager

RingtoneManager provides access to

audio clips used for incoming phone calls, notifications, alarms, etc.

Allows applications to get and set ringtones and to play and stop playing them
AudioVideoRingtoneManager

Application presents three buttons labeled ringtone, notification and alarm

Pressing one of these buttons causes the associated default ringtone to play
AudioVideo
RingtoneManager
MediaPlayer

Controls playback of audio and video streams and files

Allows applications to control playback

Operates according to a complex state machine

See: http://developer.android.com/reference/android/media/MediaPlayer.html
Some MediaPlayer Methods

setDataSource()
prepare()
start()
pause()
seekTo()
stop()
release()
VideoView

SurfaceView for displaying video files
Can load video from multiple sources
Provides various display options & convenience functions
AudioVideoVideoPlay

Application plays a movie in a VideoView
public void onCreate(Bundle savedInstanceState) {
...

    // Get a reference to the VideoView
    mVideoView = findViewById(R.id.videoViewer);

    // Add a Media controller to allow forward/reverse/pause/resume
    final MediaController mMediaController =
        new MediaController(AudioVideoVideoPlayActivity.this, true);
    mMediaController.setEnabled(false);
    mVideoView.setMediaController(mMediaController);

    mVideoView.setVideoURI(Uri.parse("android.resource://" +
        getPackageName() + "/raw/moon"));
...

// Add an OnPreparedListener to enable the MediaController once the video is ready
mVideoView.setOnPreparedListener(new OnPreparedListener() {
    public void onPrepared(MediaPlayer mp) {
        mMediaController.setEnabled(true);
    }
});

// Clean up and release resources
protected void onPause() {
    if (mVideoView != null && mVideoView.isPlaying()) {
        mVideoView.stopPlayback();
        mVideoView = null;
    }
    super.onPause();
}
MediaRecorder

Used to record audio and video

Operates in accordance to a state machine

See:
Some MediaRecorder Methods

setAudioSource()
setVideoSource()
setOutputFormat()
prepare()
start()
stop()
release()
AudioVideoAudioRecording

Can record audio from the user
Can play back recorded audio
AudioVideo
AudioRecording
private void startRecoding() {

    mRecorder = new MediaRecorder();
    mRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);
    mRecorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
    mRecorder.setOutputFile(mFileName);
    mRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);

    try {
        mRecorder.prepare();
    } catch (IOException e) {
        Log.e(TAG, "Couldn't prepare and start MediaRecorder");
    }

    mRecorder.start();
}
// Playback audio using MediaPlayer
private void startPlaying() {
    MediaPlayer mPlayer = new MediaPlayer();
    mPlayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {
        public void onCompletion(MediaPlayer mp) {
            mPlayButton.performClick();
            mPlayButton.setChecked(false);
        }
    });
    try {
        mPlayer.setDataSource(mFileName);
        mPlayer.prepare();
        mPlayer.start();
    } catch (IOException e) {
        Log.e(TAG, "Couldn't prepare and start MediaPlayer");
    }
}
// Release recording and playback resources, if necessary
public void onPause() {
    super.onPause();

    if (null != mRecorder) {
        mRecorder.release();
        mRecorder = null;
    }

    if (null != mPlayer) {
        mPlayer.release();
        mPlayer = null;
    }
}
Next Time

Sensors