CS371m - Mobile Computing

Audio and Haptic Feedback

Clicker

- On the whole, do you like apps to provide audio and / or haptic feedback or not?

A. No

B. Yes

Audio on Device

- Devices have multiple audio streams:
  - music, alarms, notifications, incoming call ringer, in call volume, system sounds, DTMF tones (dual tone multi frequency, the "traditional" phone sounds and tones)
- Most of the streams are restricted to system events, so we almost always use STREAM_MUSIC via a MediaPlayer or SoundPool

MEDIA PLAYER

Android Audio

- Using the Android MediaPlayer class
- Common Audio Formats supported:
  - MP3, MIDI (.mid and others), Vorbis (.ogg), WAVE (.wav) and others
- Sources of audio
  - local resources (part of app)
  - internal URIs (Content Provider for other audio available)
  - External URLs (streaming)

Audio Resources

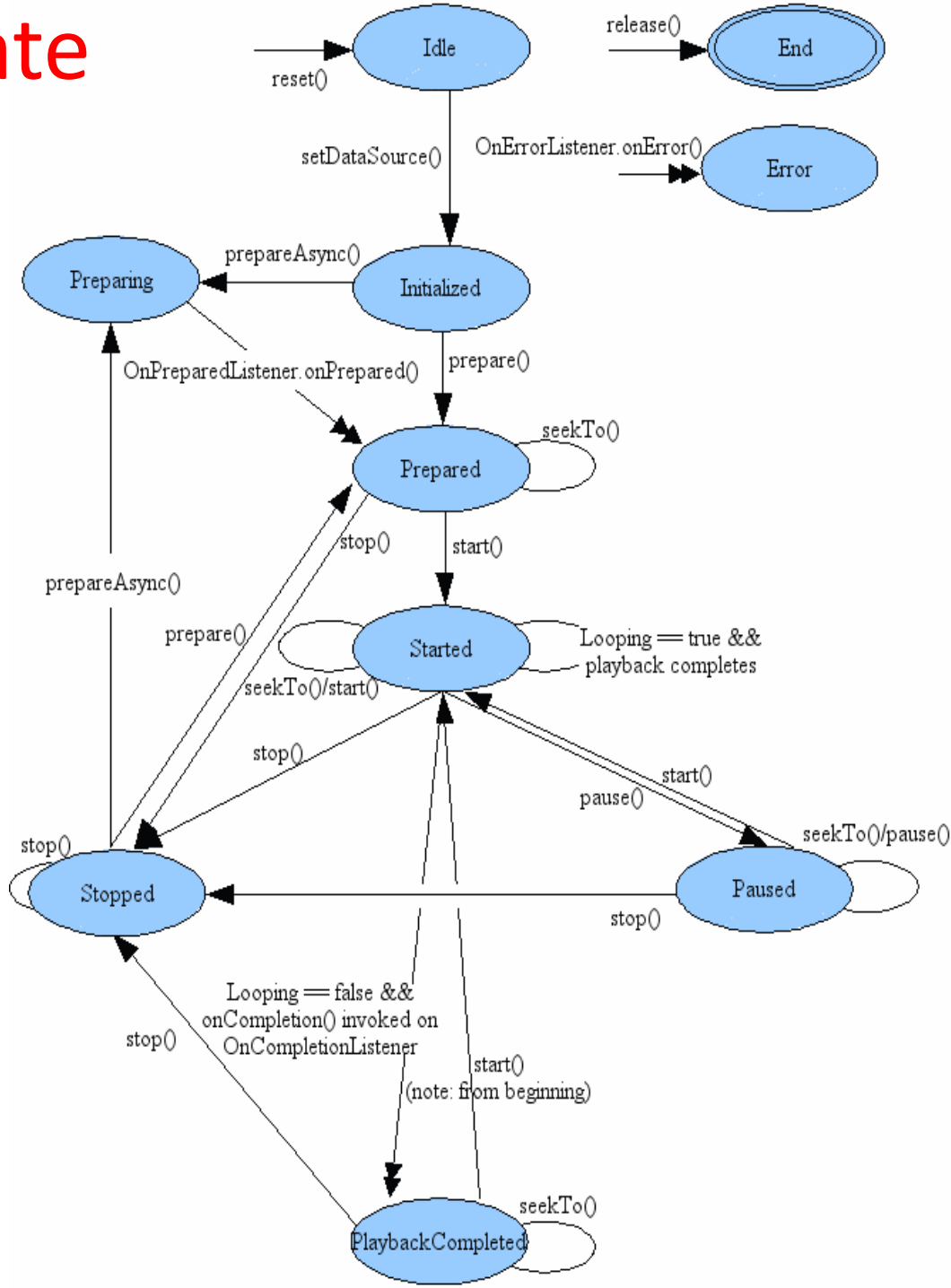
- res/raw directory
- assets/ directory
  - reference as file://andorid_asset in a URI
  - can share with other apps via URI
- Store media in application local directory, pull from network / web and store locally
- Store and use media on SD card
- Stream via the Internet

MediaPlayer

- Playback control of MediaPlayer managed as a state machine
- Idle
- Initialized
- Preparing
- Prepared
- Started
- Paused
- Playback Complete
- Stopped
- End
- Invalid state transitions result in errors

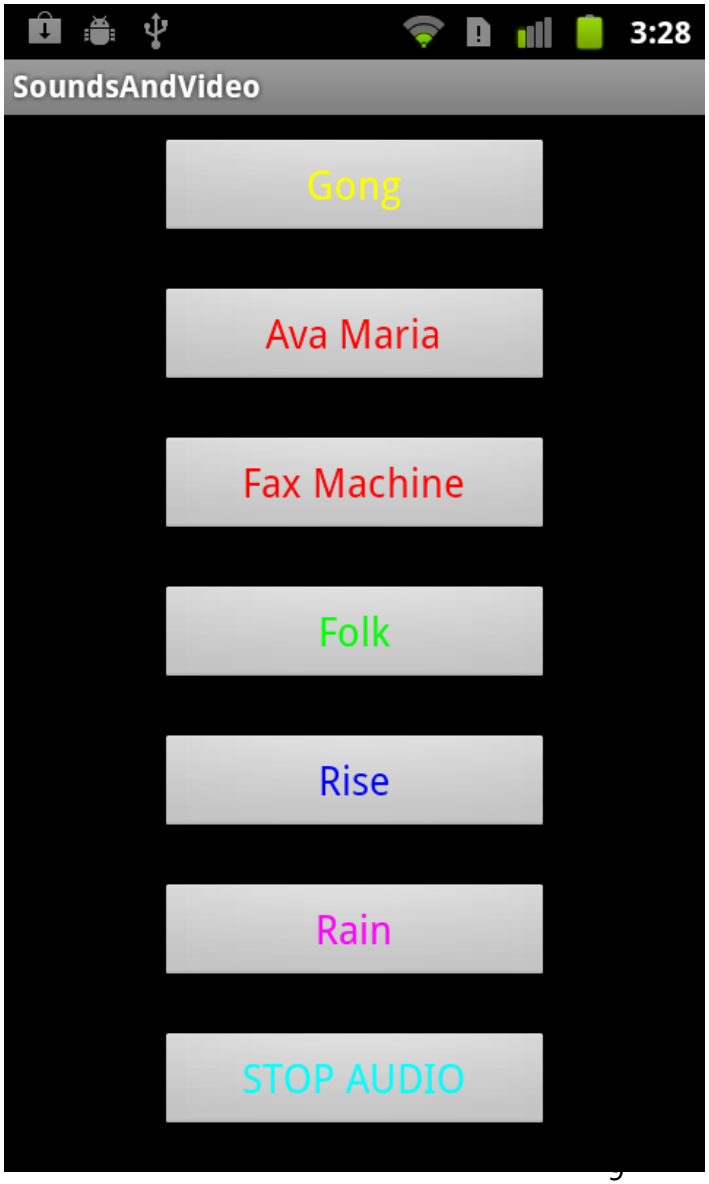
MediaPlayer State Diagram

- Single arrows are synchronous transitions
- Double arrows are asynchronous transitions



Simple Sound Demo App

- audio files local to app placed in res/raw
- CAUTION
  - large sound files difficult to install on emulator:
  - http://tinyurl.com/3pwljfj
  - better success with dev phones / actual devices



Using MediaPlayer

- MediaPlayer.create() method used for raw resources
- MediaPlayer.create() method for URIs
- various MediaPlayer.setDataSource() methods if not a raw resource and not a URI

Playing Local Audio

- To play audio local to the app
- use the MediaPlayer.create convenience method
  - when complete MediaPlayer in the **prepared** state
- start MediaPlayer
- approach:
  - build listeners for each button to call the playSound method with appropriate song id when clicked

Simple Approach

button ids

ids for sound files

```
private void buildListeners() {
    int[] ids = {R.id.gong, R.id.ava, R.id.fax,
            R.id.folk, R.id.rise, R.id.rain};
    int[] songs = {R.raw.gong, R.raw.ava_maria,
            R.raw.fax, R.raw.music,
            R.raw.rise, R.raw.rain};

    for(int i = 0; i < ids.length; i++) {
        final Button button = (Button) findViewById(ids[i]);
        final int SONG_ID = songs[i];
        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                playSound(SONG_ID);
            }
        });
    }
}
```

playSound method initial version

```
private void playSound(int songID) {
    MediaPlayer mediaPlayer = MediaPlayer.create(this, songID);
    mediaPlayer.start();
    // no need to call prepare(); create() does that for you
}
```

- useful for *short* sounds
- downsides:
  - plays to completion
  - multiple sounds play at same time (desirable in some cases)
  - audio continues to play when app paused

Changing Behavior

- Add instance variable for MediaPlayer
- If playing stop and release before creating new Player

```
private void playSound(int songID) {
    if(player == null || !player.isPlaying()) {
        Log.d(TAG, "player null or not playing " +
                "- creating new player");
        player = MediaPlayer.create(this, songID);
    }
    if(player.isPlaying()) {
        Log.d(TAG, "player playing - " +
                "stopping and releasing");
        player.stop();
        player.release();
        player = MediaPlayer.create(this, songID);
    }

    player.start();
}
```

Cleaning Up

- Initial version did not end well
- Audio continued to play if back button pressed and even if home button pressed!
- Activity Life Cycle
- in onPause for app we should stop MediaPlayer and release

stopPlayer method

- Connect app stop button to stopPlayer
  - could use XML onClick and add View parameter or set up listener ourselves

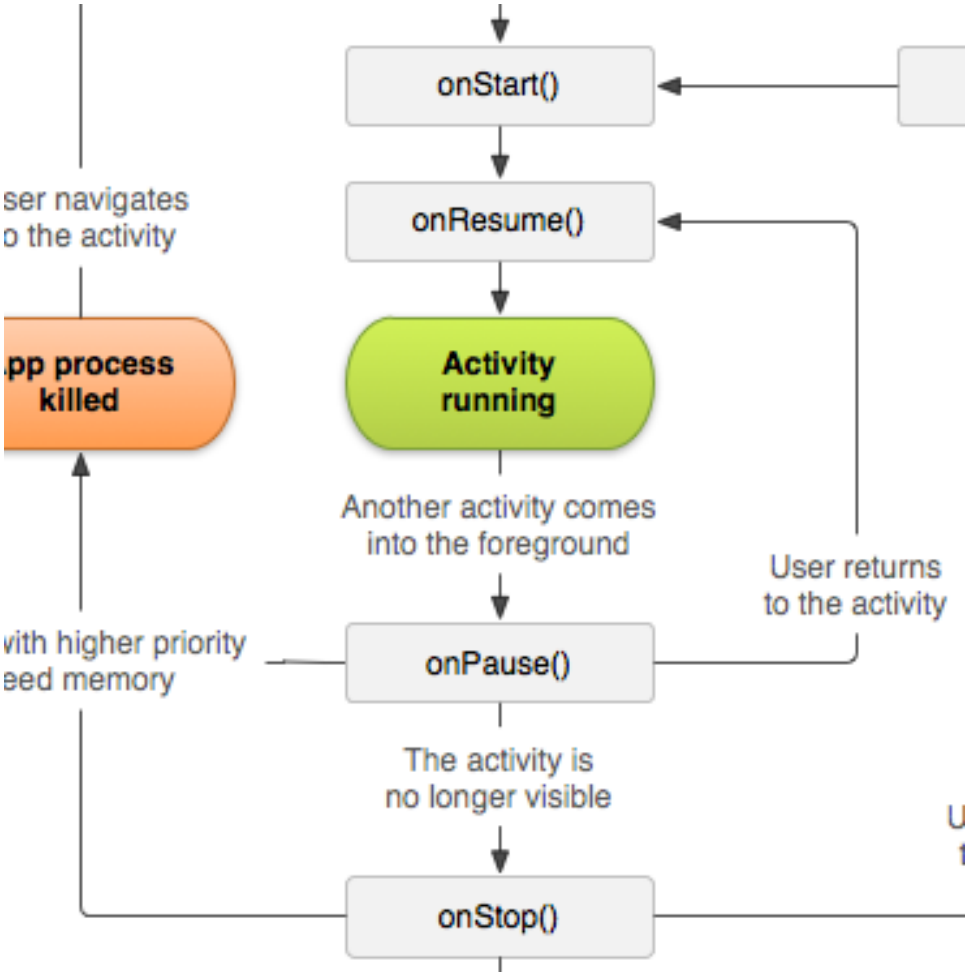
```
    // set up the stop button
    Button stop = (Button) findViewById(R.id.stop);
    stop.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            stopPlayer();
        }
    });
}

private void stopPlayer() {
    if(player != null) {
        player.stop();
        player.release();
        player = null;
    }
}
```

in buildListeners method

onPause

- onPause() should call the stopPlayer method
- what happens if activity resumed?

```
@Override
protected void onPause() {
    super.onPause();
    // stop the music!!
    stopPlayer();
}
```



Saving State

- Resume music where we left off if paused or activity destroyed due to orientation change

```
@Override
protected void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);

    stopPlayer();
}

@Override
protected void onPause() {
    super.onPause();

    stopPlayer();
}
```

Saving MediaPlayer State

- Not a lot of data so used the SharedPreferences

```
private void stopPlayer() {
    if(player != null) {
        if(player.isPlaying()) {
            SharedPreferences mPrefs
                    = getSharedPreferences("sound_demo", MODE_PRIVATE);
            SharedPreferences.Editor ed = mPrefs.edit();
            ed.putInt("songID", currentSongID);
            ed.putInt("audioLocation", player.getCurrentPosition());
            ed.commit();
        }
        player.stop();
        player.release();
        player = null;
    }
}
```

Restarting Audio

- In onResume check if audio was interrupted recreate player with same id and move to correct position
- Can write data to shared preferences or bundle (onSaveInstanceState) and pull out in onResume

AUDIO SOURCES

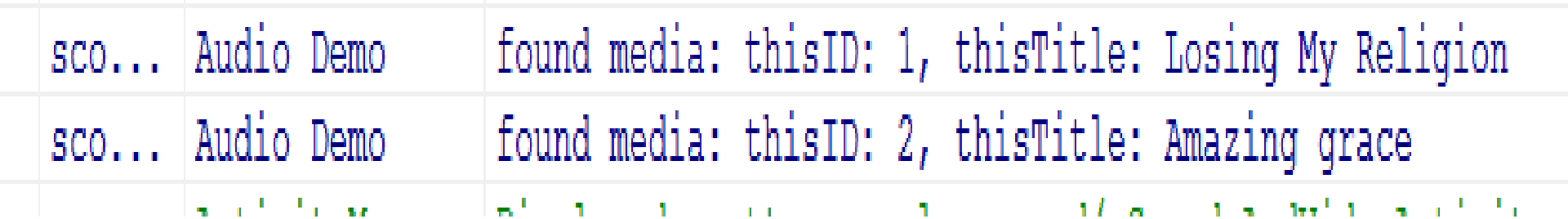
Playing Audio from Phone

- If audio is on device / system, but not local to app use a URI
- Obtain URIs of Music via a Content Resolver
- Example of simply listing URIs to the logcat

Retrieving Music URIs Using a Content Resolver

```
private void showContent() {
    ContentResolver contentResolver = getContentResolver();
    Uri uri = android.provider.MediaStore.Audio.Media.EXTERNAL_CONTENT_URI;
    Cursor cursor = contentResolver.query(uri, null, null, null, null);
    if (cursor == null) {
        Log.d(TAG, "cursor == null, query falied");
    } else if (!cursor.moveToFirst()) {
        Log.d(TAG, "no media on the device");
    } else {
        int titleColumn
            = cursor.getColumnIndex(android.provider.MediaStore.Audio.Media.TITLE);
        int idColumn
            = cursor.getColumnIndex(android.provider.MediaStore.Audio.Media._ID);
        do {
           long thisId = cursor.getLong(idColumn);
           String thisTitle = cursor.getString(titleColumn);
           Log.d(TAG, "found media: thisID: "
                    + thisId + ", thisTitle: " + thisTitle);
        } while (cursor.moveToNext());
    }
}
```

MediaPlayer and System Audio



- After URI retrieved can play audio with MediaPlayer
- this approach requires calling MediaPlayer prepare yourself
- Or could use MediaPlayer create method that takes a URI

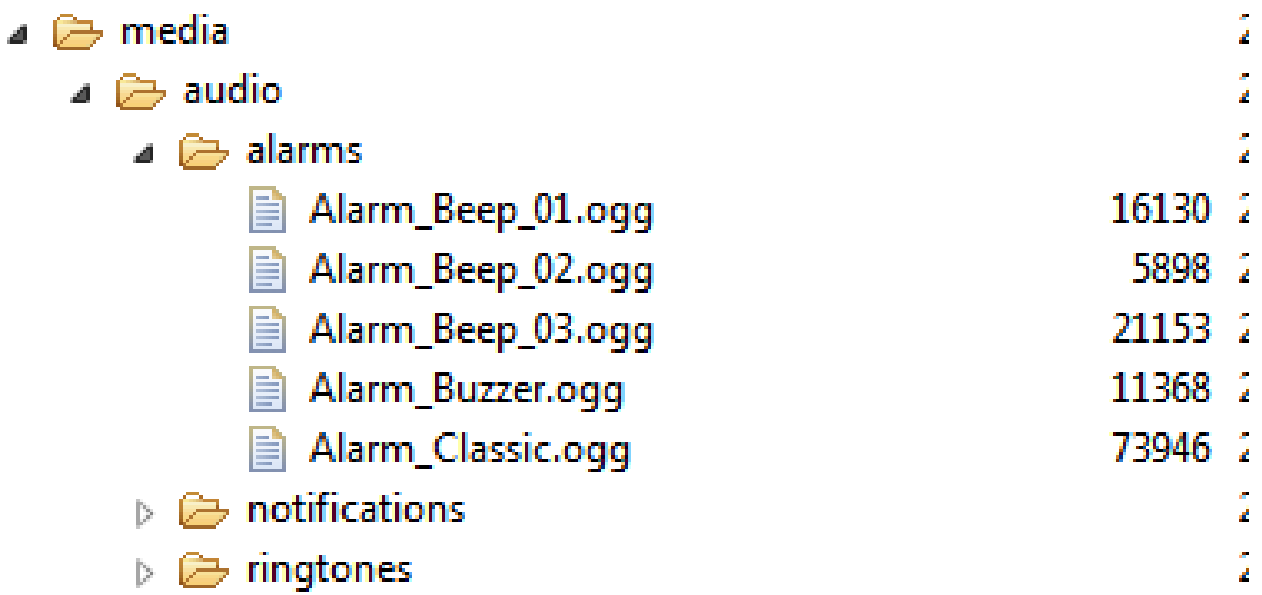
Playing Audio Via Local URI

- id obtained via approach from showContent method

```
private void playRandomSong() {
    stopPlayer();

    // get id of random song
    long id = showContent();

    Uri contentUri = ContentUris.withAppendedId(
            android.provider.MediaStore.Audio.Media.EXTERNAL_CONTENT_URI, id);

    player = new MediaPlayer();
    player.setAudioStreamType(AudioManager.STREAM_MUSIC);

    try {
        player.setDataSource(this, contentUri);
        player.prepare();
        player.start();
    }
```

Other Audio

- Other audio for ringtones, notifications, and alarms can be accessed via a RingtoneManager
- Obtain URIs and play with media player
- from DDMS:



Listing Other Audio

```
private void showRingtones() {
    RingtoneManager rm = new RingtoneManager(this);
    rm.setType(RingtoneManager.TYPE_ALL);
    Cursor cursor = rm.getCursor();
    if (cursor == null) {
        Log.d(TAG, "cursor == null, query failed");
    } else if (!cursor.moveToFirst()) {
        Log.d(TAG, "no ringtones on the device");
    } else {
        int count = cursor.getCount();
        Log.d(TAG, "count of ringtones: " + count);
        for(int i = 0; i < count; i++) {
            Ringtone r = rm.getRingtone(i);
            Log.d(TAG, "ringtone num: " + i
                    + " name: " + r.getTitle(this));
        }
    }
}
```

Playing Other Audio

- Once the URI is obtained, playing other audio is same as playing song

```
int count = cursor.getCount();
Log.d(TAG, "count of ringtones: " + count);
for(int i = 0; i < count; i++) {
    Ringtone r = rm.getRingtone(i);
    Log.d(TAG, "ringtone num: " + i
            + " name: " + r.getTitle(this));
}*/
int num = (int) (Math.random() * count);
result = rm.getRingtoneUri(num);
```

Playing Audio from Remote URL

- Straightforward given the URL

```
private void playFromURL() {
    String url = "http://www.pacdv.com/sounds/" +
            "machine_sound_effects/chain-saw-2.mp3";
    stopPlayer();
    if(player == null)
        player = new MediaPlayer();
    player.setAudioStreamType(AudioManager.STREAM_MUSIC);
    try {
        player.setDataSource(url);
        player.prepare(); // might take long! (for buffering,
        player.start();
    }
    catch (IOException e){
```

Completion of Audio

- If action required when audio done playing implement the MediaPlayer.onCompletionListener interface

```
MediaPlayer.OnCompletionListener done
        = new MediaPlayer.OnCompletionListener() {

    @Override
    public void onCompletion(MediaPlayer mp) {
        // take necessary action on completion
    }
};
```

- could make activity the listener

Looping

- to loop sound (play over and over) simply set the isLooping method of the MediaPlayer to true

SOUND POOL

SoundPool

- Another Android class
  - Used in tutorials

public **SoundPool** (int maxStreams, int streamType, int srcQuality) Since: API L

Constructor. Constructs a SoundPool object with the following characteristics:

Parameters

maxStreams	the maximum number of simultaneous streams for this SoundPool object
streamType	the audio stream type as described in AudioManager For example, game applications will normally use `STREAM_MUSIC`.
srcQuality	the sample-rate converter quality. Currently has no effect. Use 0 for the default.

Using SoundPool

- Great for applications with a number of short sound samples
- maxStreams parameter sets maximum number of sounds that can be played at once via this SoundPool
- If max is exceeded stream with lowest priority stopped
  - and then by age (oldest) with lowest priority

SoundPool play

public final int **play** (int soundID, float leftVolume, float rightVolume, int priority, int loop, float rate)

Parameters

soundID	a soundID returned by the load() function
leftVolume	left volume value (range = 0.0 to 1.0)
rightVolume	right volume value (range = 0.0 to 1.0)
priority	stream priority (0 = lowest priority)
loop	loop mode (0 = no loop, -1 = loop forever)
rate	playback rate (1.0 = normal playback, range 0.5 to 2.0)

Using SoundPool

- Looping of sounds:
  - 0 no looping
  - -1 loop forever
  - >0, play that many times
- frequency (speed) can be changed
  - range from 0.5 to 2.0
  - 0.5 twice as long to play
  - 2.0 half as long to play

SoundPool.Builder

- Added in API level 21, Android 5.0 Lollipop
- static methods to
  - set audio attributes
  - set max number of streams
  - build and return a SoundPool object given the current audio attributes and number of streams
- Audio Attributes include why, what and how for sound
  - USAGE_GAME, CONTENT_TYPE_SONIFICATION, various flags

Other Sources of Sound

- AudioTrack
- http://developer.android.com/reference/android/media/AudioTrack.html
- low level API for handling audio streams in formats MediaPlayer cannot handle
- ToneGenerator
- http://developer.android.com/reference/android/media/ToneGenerator.html
- Traditional phone tones
- *Dual-Tone Multi-Frequency* (DTMF)
- Just to make sounds, not send tones over phone network

ToneGenerator

Public Constructors	
	ToneGenerator (int streamType, int volume) ToneGenerator class contructor specifying output stream type and volume.

Public Methods	
final int	getAudioSessionId () Returns the audio session ID.
void	release () Releases resources associated with this ToneGenerator object.
boolean	startTone (int toneType, int durationMs) This method starts the playback of a tone of the specified type for the specified duration.
boolean	startTone (int toneType) This method starts the playback of a tone of the specified type.
void	stopTone () This method stops the tone currently playing playback.

HAPTIC FEEDBACK

Haptic Feedback

- haptic: "of or relating to the sense of touch"
- Easy to add this kind of feedback to an app
- can enable haptic feedback for any view

Haptic Feedback on a View

- Methods from View class

public void setHapticFeedbackEnabled (boolean hapticFeedbackEnabled)

Set whether this view should have haptic feedback for events such as long presses.

You may wish to disable haptic feedback if your view already controls its own haptic feedback.

Related XML Attributes:

android:hapticFeedbackEnabled

public boolean performHapticFeedback (int feedbackConstant)

Added in API level 3

BZZZTT!!1!

Provide haptic feedback to the user for this view.

The framework will provide haptic feedback for some built in actions, such as long presses, but you may wish to provide feedback for your own widget.

The feedback will only be performed if `isHapticFeedbackEnabled()` is true.

HapticFeedbackConstants Class

Constants		
int	CLOCK_TICK	The user has pressed either an hour or minute tick of a Clock.
int	CONTEXT_CLICK	The user has performed a context click on an object.
int	FLAG_IGNORE_GLOBAL_SETTING	Flag for `View.performHapticFeedback(int, int)`: Ignore the global setting for whether to perform haptic feedback, do it always.
int	FLAG_IGNORE_VIEW_SETTING	Flag for `View.performHapticFeedback(int, int)`: Ignore the setting in the view for whether to perform haptic feedback, do it always.
int	KEYBOARD_TAP	The user has pressed a soft keyboard key.
int	LONG_PRESS	The user has performed a long press on an object that is resulting in an action being performed.
int	VIRTUAL_KEY	The user has pressed on a virtual on-screen key.

More Complex Feedback

- Also possible to create more complex haptic feedback for apps:
- Request permission
- Get the Vibrator object from the system
- call vibrate method

Haptic Feedback

- Request Permission

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

- Get Vibrator

```
private Vibrator vib;
```

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    vib = (Vibrator) this.getSystemService(VIBRATOR_SERVICE)
```

Haptic Feedback

- Create feedback

```
private void initStopButton() {
    // set up the stop button
    Button stop = (Button) findViewById(R.id.stop);
    stop.setOnClickListener(new View.OnClickListener()
        public void onClick(View v) {
            vib.vibrate(200); // milliseconds
            stopPlayer();
        }
    });
}
```

Vibrator Methods

Public Methods	
abstract void	cancel () Turn the vibrator off.
abstract boolean	hasVibrator () Check whether the hardware has a vibrator.
abstract void	vibrate (long[] pattern, int repeat) Vibrate with a given pattern.
abstract void	vibrate (long milliseconds) Vibrate constantly for the specified period of time.

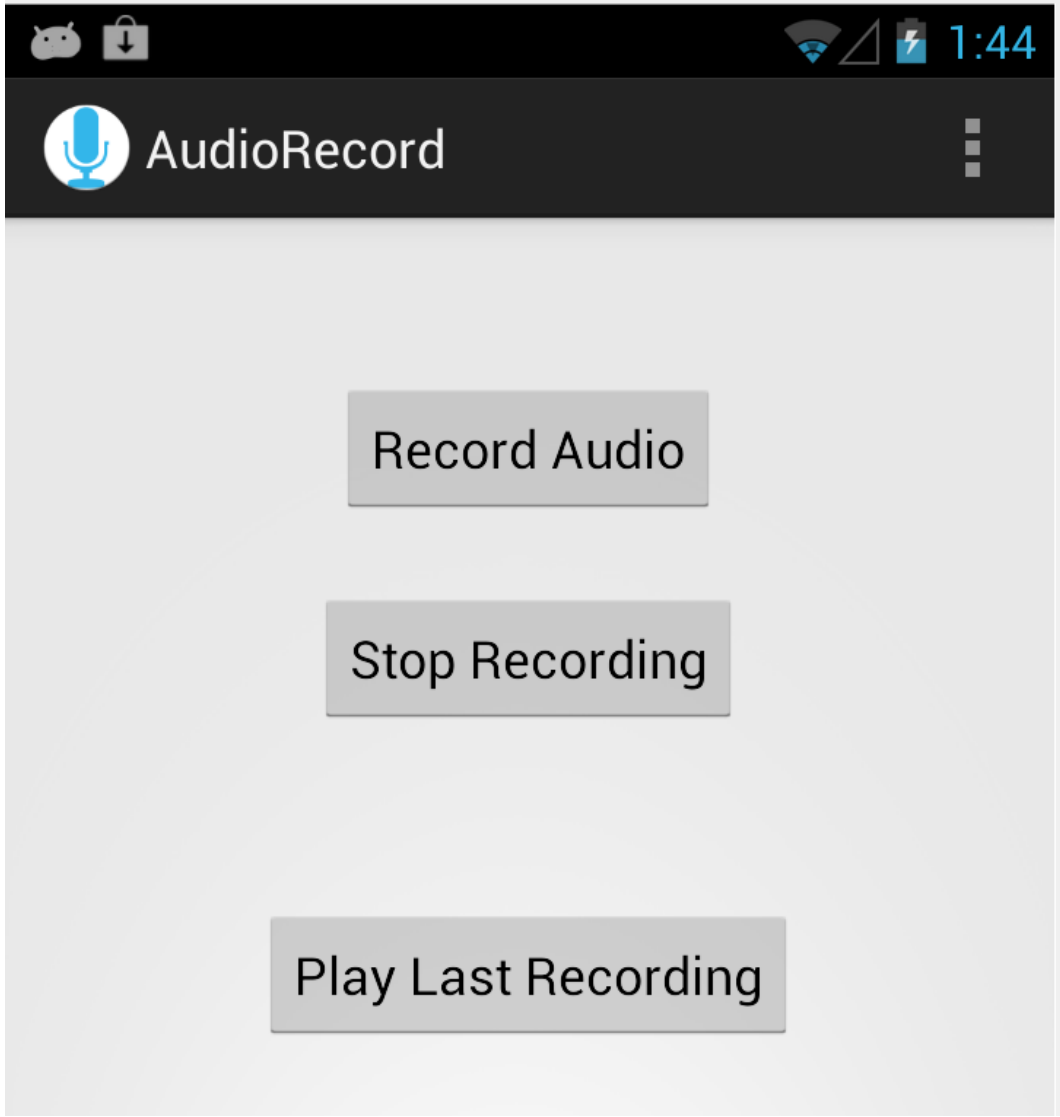
RECORDING AUDIO

Recording Audio - MediaRecorder

1. create MediaRecorder object
2. set audio source
3. set audio format
4. set file format for audio
5. set file name to record to
6. prepare MediaRecorder for recording
7. start recording
8. stop and release MediaRecorder

Record Audio

- Simple View with buttons to start and stop recording
- alternatively could change text on record audio button
  - single button or toggle switch



Record Audio

```
public void recordAudio(View v) {
    if(audioRecorder == null)
        audioRecorder = new MediaRecorder();
    String pathForAudioRecording = getFilesDir().getAbsolutePath();
    pathForAudioRecording += RECORD_FILE;

    audioRecorder.setAudioSource(
                MediaRecorder.AudioSource.MIC);
    audioRecorder.setOutputFormat(
                MediaRecorder.OutputFormat.DEFAULT);
    audioRecorder.setAudioEncoder(
                MediaRecorder.AudioEncoder.DEFAULT);

    audioRecorder.setOutputFile(pathForAudioRecording);

    try {
        audioRecorder.prepare();
        audioRecorder.start();
    }
    catch(IOException e) {
        Log.e(TAG, "unable to record. could not prepare or start Me
```

Stop Recording

```
public void stopRecording(View v) {
    if(audioRecorder != null) {
        audioRecorder.stop();
        audioRecorder.release();
        audioRecorder = null;
    }
}
```

- a few seconds of audio results in a file size of ~ 10 kb with default settings
- PERMISSIONS! -- must request RECORD_AUDIO and WRITE_EXTERNAL_STORAGE permissions in manifest file

SPEECH RECOGNITION

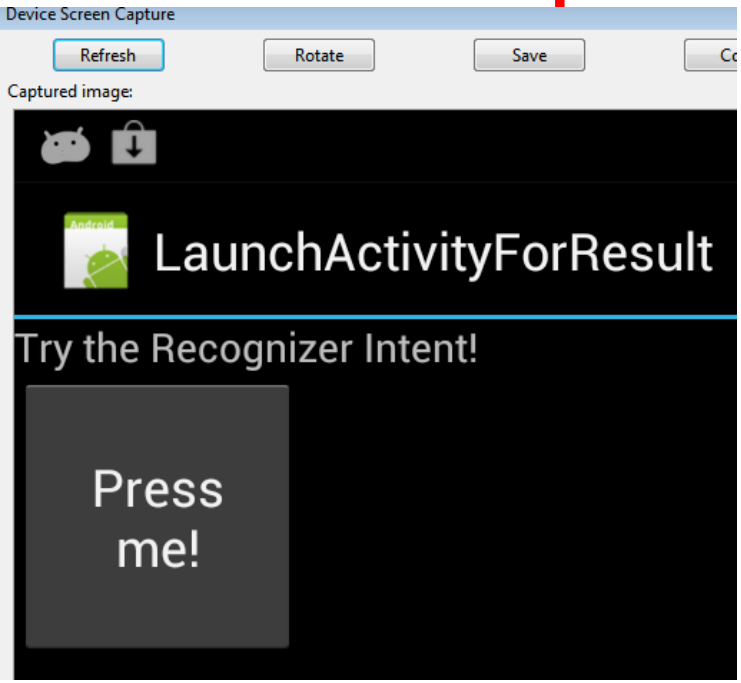
Speech Recognition

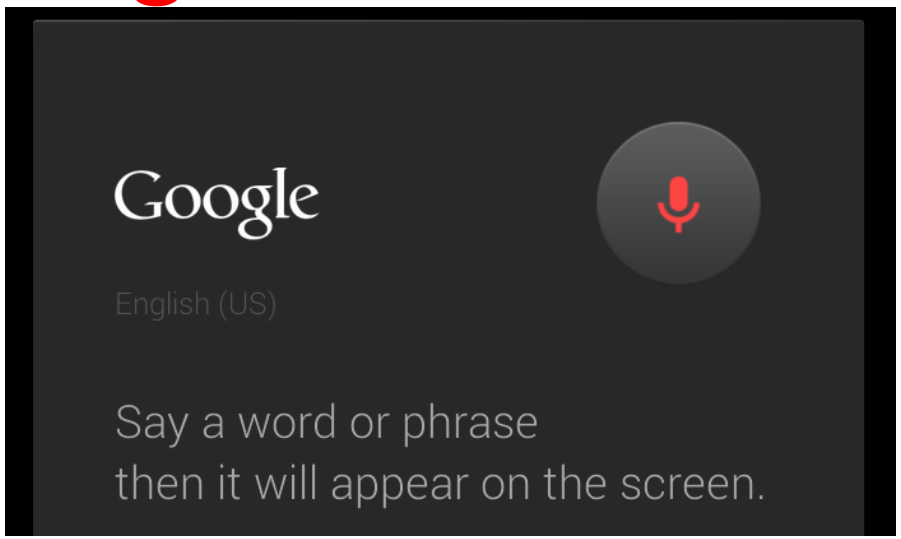
- android.speech package
- Simplest example - start an Intent for a result

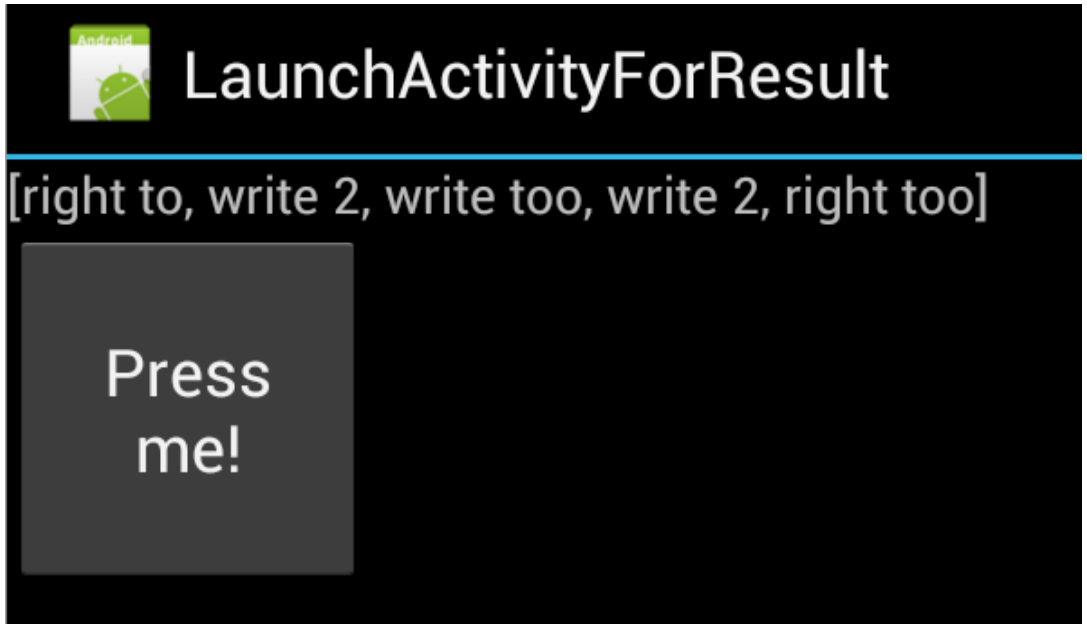
  RecognizerIntent.ACTION_RECOGNIZE_SPEECH

- uses network
  - true on the dev phones
  - doesn't work on emulator

Speech Recognition







Starting Intent

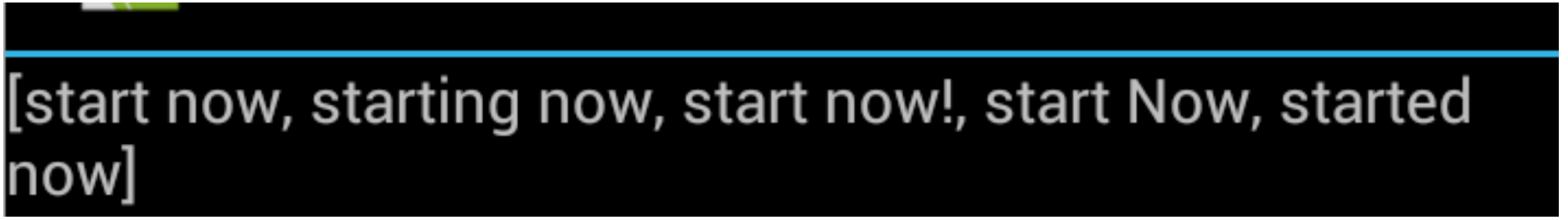
```
//setup button listener
Button startButton = (Button) findViewById(R.id.trigger);
startButton.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
        // RecognizerIntent prompts for speech and returns text
        Intent intent =
            new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);

        intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
                RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
        intent.putExtra(RecognizerIntent.EXTRA_PROMPT,
        "Say a word or phrase\nthen it will appear on the screen.");
        startActivityForResult(intent, RECOGNIZER_EXAMPLE);
    }
});
```

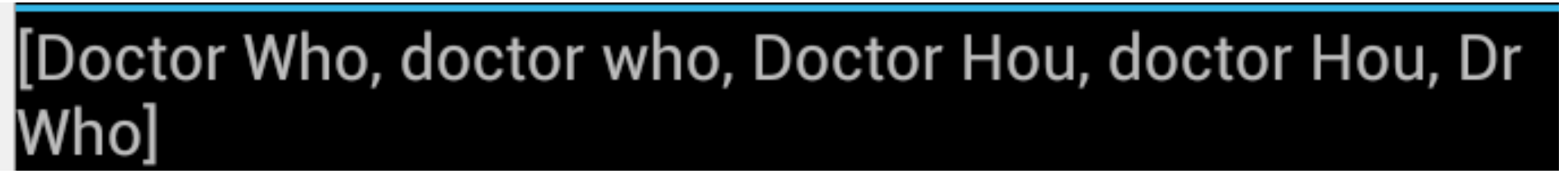
Responding to Result

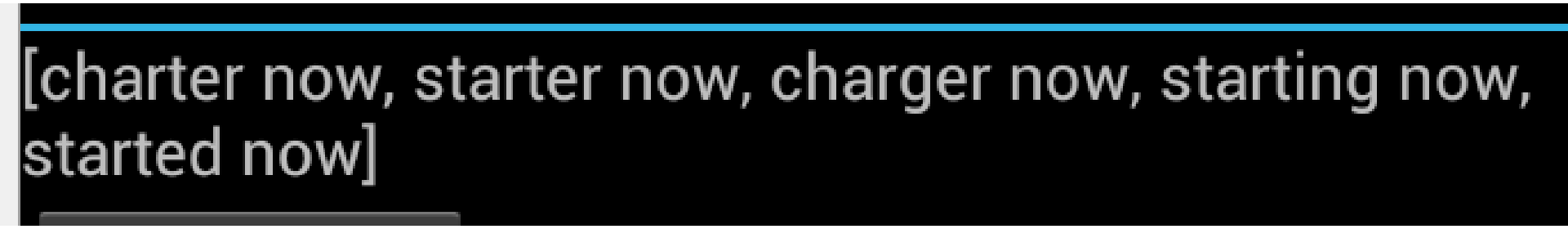
- Note: list of results, ordered by confidence

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

    if (requestCode == RECOGNIZER_EXAMPLE
                && resultCode == RESULT_OK) {
        // returned data is a list of matches to the speech
        ArrayList<String> result =
            data.getStringArrayListExtra
            (RecognizerIntent.EXTRA_RESULTS);

        //display on screen
        tv.setText(result.toString());
    }

    super.onActivityResult(requestCode, resultCode, data);
}
```

Results


[start now, starting now, start now!, start Now, started now]


[Doctor Who, doctor who, Doctor Hou, doctor Hou, Dr Who]


[charter now, starter now, charger now, starting now, started now]