

# CS 475/575

## Audio

Content adapted from CS378 - Mobile Computing  
<http://www.cs.utexas.edu/~scottm/cs378/schedule.htm>  
Used with permission

# Sound

- SoundPool – easy to use, great for short sounds, use for games (background sounds looping etc.). Hard limit on size of soundfile (~1Mb but this # will likely change)
- MediaPlayer – bit more complex, better suited for longer sounds streaming music and movies

# SoundPool

- Android class for playing simple sounds

```
public SoundPool (int maxStreams, int streamType, int srcQuality)
```

Since: API

Constructor. Constructs a SoundPool object with the following characteristics:

## Parameters

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

- (For API 21 and on use Soundpool.builder instead)

# 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

|                    |   |
|--------------------|---|
| <i>soundID</i>     | a soundID returned by the load() function               |
| <i>leftVolume</i>  | left volume value (range = 0.0 to 1.0)                  |
| <i>rightVolume</i> | right volume value (range = 0.0 to 1.0)                 |
| <i>priority</i>    | stream priority (0 = lowest priority)                   |
| <i>loop</i>        | loop mode (0 = no loop, -1 = loop forever)              |
| <i>rate</i>        | 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 Example

```
SoundPool sp = null;

int napStream = UNINITIALIZED;

//get soundpool object
sp = new SoundPool(MAX_STREAMS, AudioManager.STREAM_MUSIC, 0);

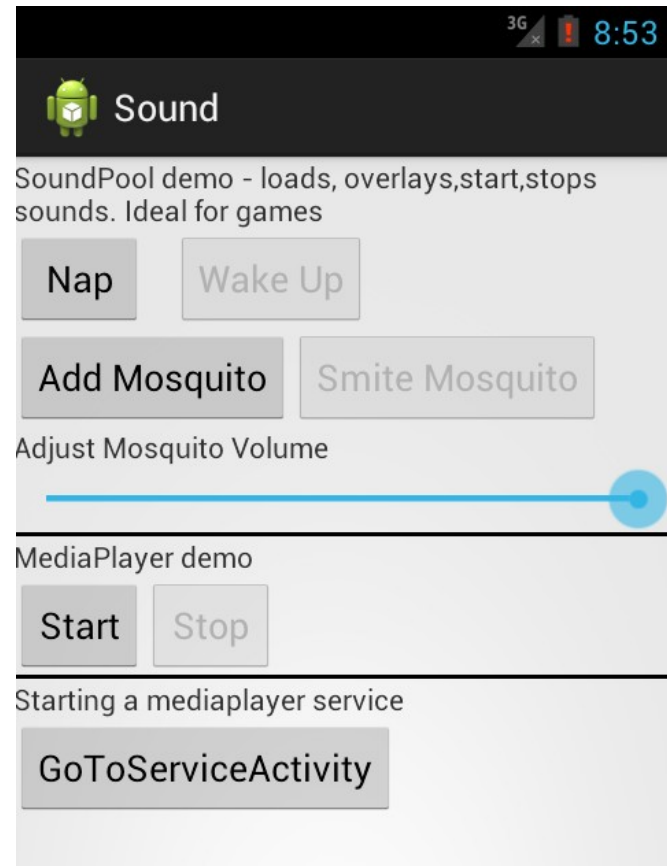
//load our sounds
trackNap = sp.load(this, R.raw.snore,0);
trackMosquito = sp.load(this, R.raw.mosquito,0);
trackSmite = sp.load(this, R.raw.flyswat,0);

private void doNap() {
    //create stream, the int returned is the value you use to clobber it
    if ( napStream == UNINITIALIZED){
        napStream = sp.play(trackNap, LEFTVOLUME, RIGHTVOLUME, PRIORITY,LOOPFOREVER , RATE);
        Log.d(TAG,"starting nap stream " + Integer.toString(napStream) );
    }
    setNapState();
}

public void doWakeup(View v){
    sp.stop(napStream);
    napStream = UNINITIALIZED;
    setNapState();
}
```

# Simple Sound Demo App

- audio files local to app **placed in res/raw**
- CAUTION
  - large sound files difficult to install on emulator:
  - better success with dev phones / actual devices



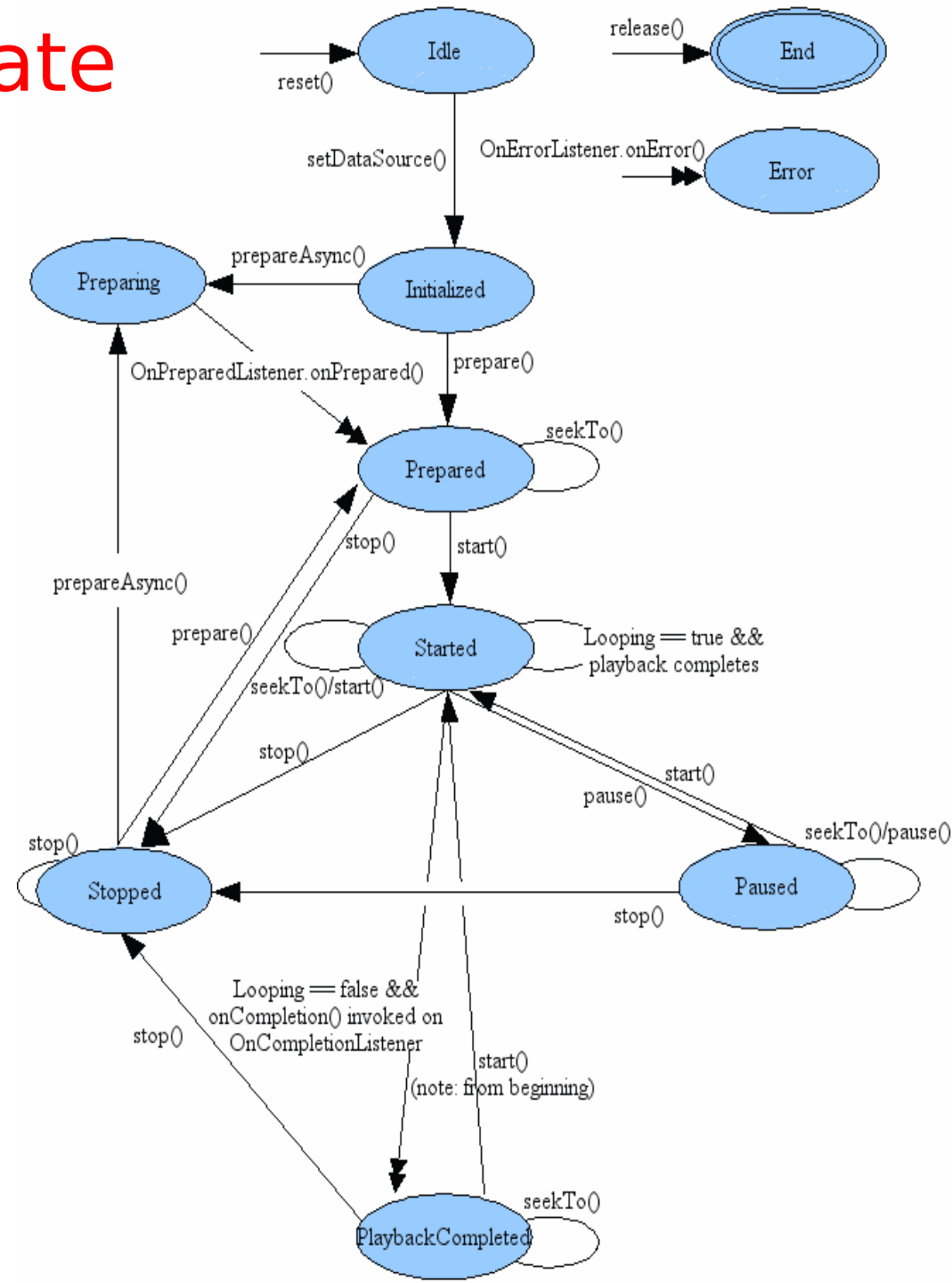


# Android Audio

- Use the 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)

# MediaPlayer State Diagram

- Single arrows are synchronous transitions
- Double arrows are asynchronous transitions
- Delicate!! See <http://developer.android.com/reference/android/media/MediaPlayer.html#StateDiagram>



# Playing Local Audio

- To play audio local to the app
- use the MediaPlayer.create convenience method
  - when complete MediaPlayer in the **prepared** state
- Then just start MediaPlayer

# playSound method

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

- okay 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

## Starting and Stopping

- Add instance variable for MediaPlayer
- Have start and stop methods

```
private void startMP() {  
    mp = MediaPlayer.create(this, R.raw.mosquito);  
    mp.start();  
    mp.setOnCompletionListener(this);  
    setMediaPlayerButtonState(false);  
}
```

```
private void stopMP(MediaPlayer mp) {  
    if (mp != null){  
        mp.stop();  
        mp.release();  
        mp = null;  
    }  
    setMediaPlayerButtonState(true);  
}
```

in ActivitySound class

# Cleaning Up

- Current version does not end well
- Audio continues to play if back button pressed and even if home button pressed!
- Activity Life Cycle - onPause we should stop MediaPlayer and release

```
@Override
protected void onPause() {
    super.onPause();
    if (mp != null)
        onStop(null);
}
```

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

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

# 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

# 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, etc)  
        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

# SoundPool or MediaPlayer

- Soundpool

- Short clips (~10s) that are loaded fast
- Loaded up front when soundpool created
- Multiple streams can play at same time
- Good for game sound effects (gun fire, explosions etc)

- MediaPlayer

- Longer audio tracks (also streamed)
- Loaded in chunks, decoding is on-the-fly.
- One audio stream per player, (can have multiple players though)
- Good for a music player or podcast app

-

# References

- [SeekBar http://webtutsdepot.com/2011/12/03/android-sdk-tutorial-seekbar-example/](http://webtutsdepot.com/2011/12/03/android-sdk-tutorial-seekbar-example/)
- <http://www.vogella.com/articles/AndroidMedia/article.html>