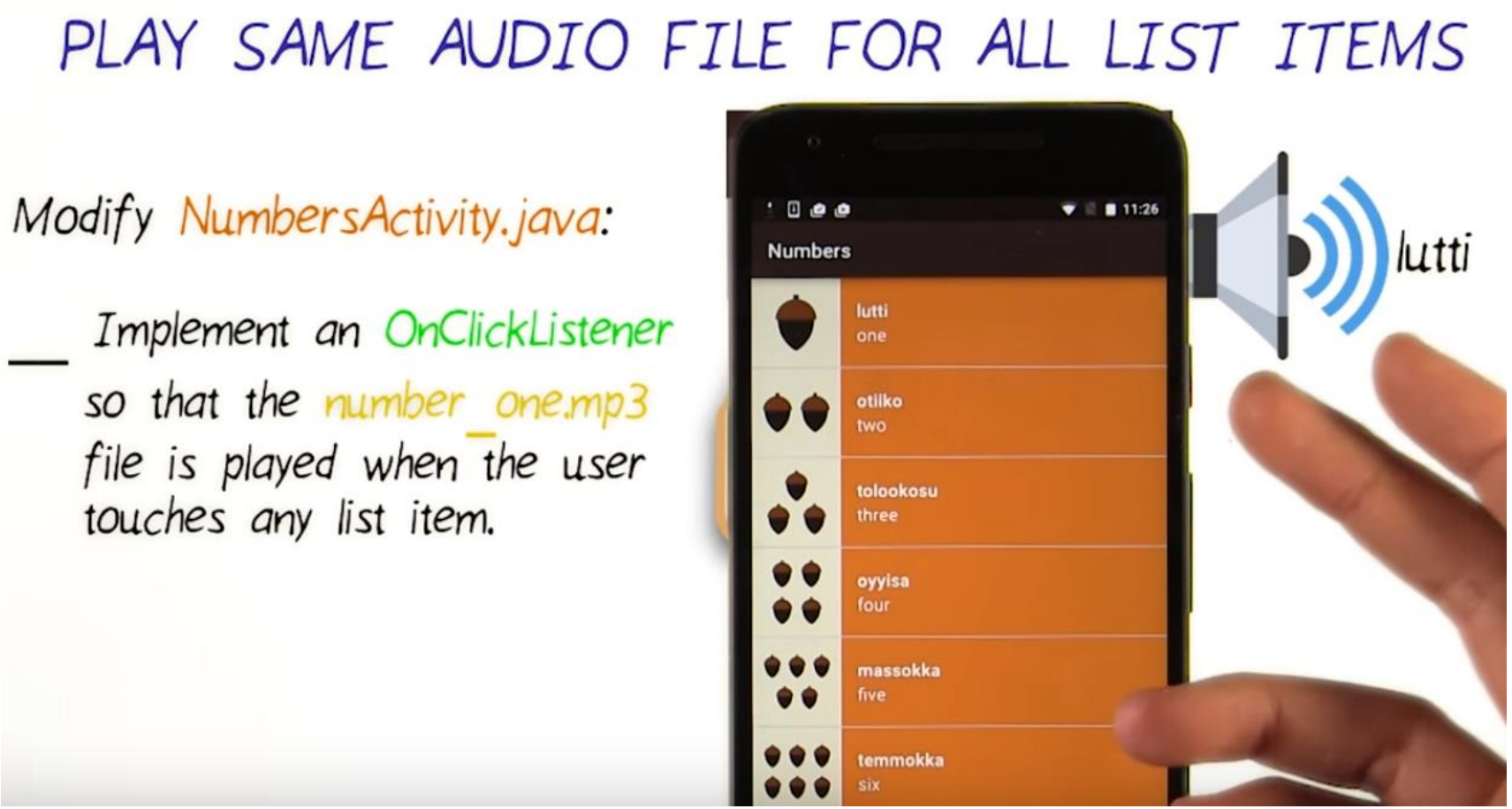
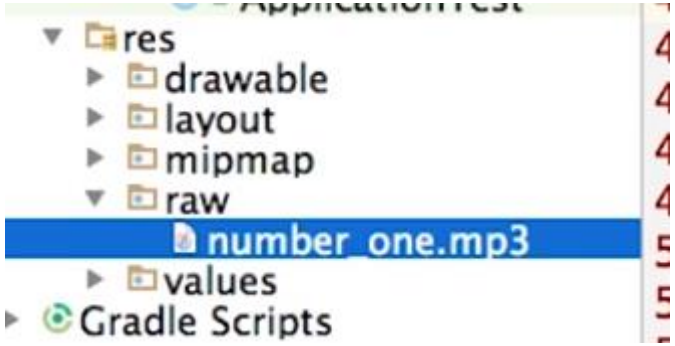
MIWOK part 4

24 - Play same audio file for all words in NumbersActivity



NumbersActivity.java

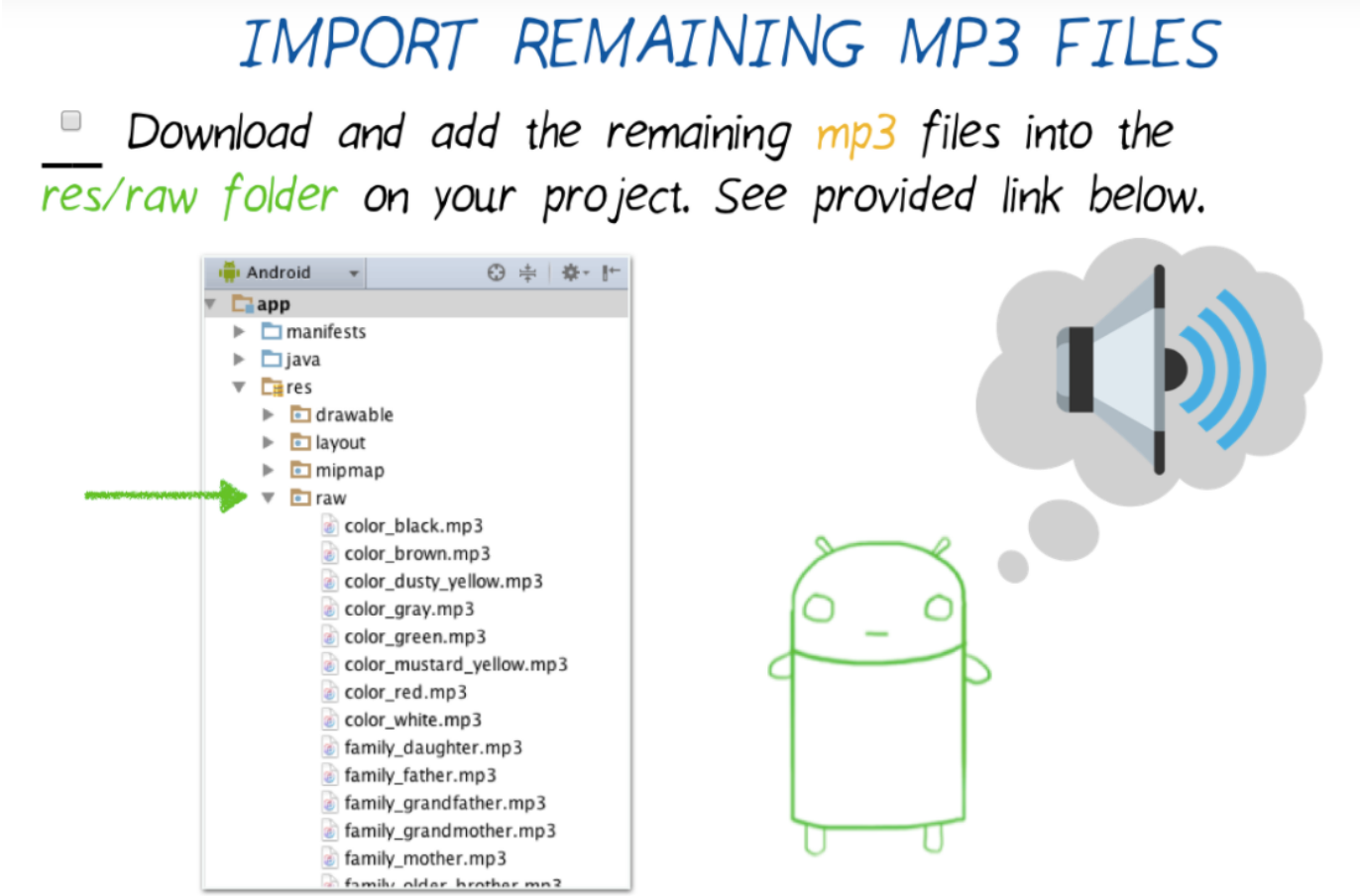



```
/** Handles playback of all the sound files */
private MediaPlayer mMediaPlayer;
```

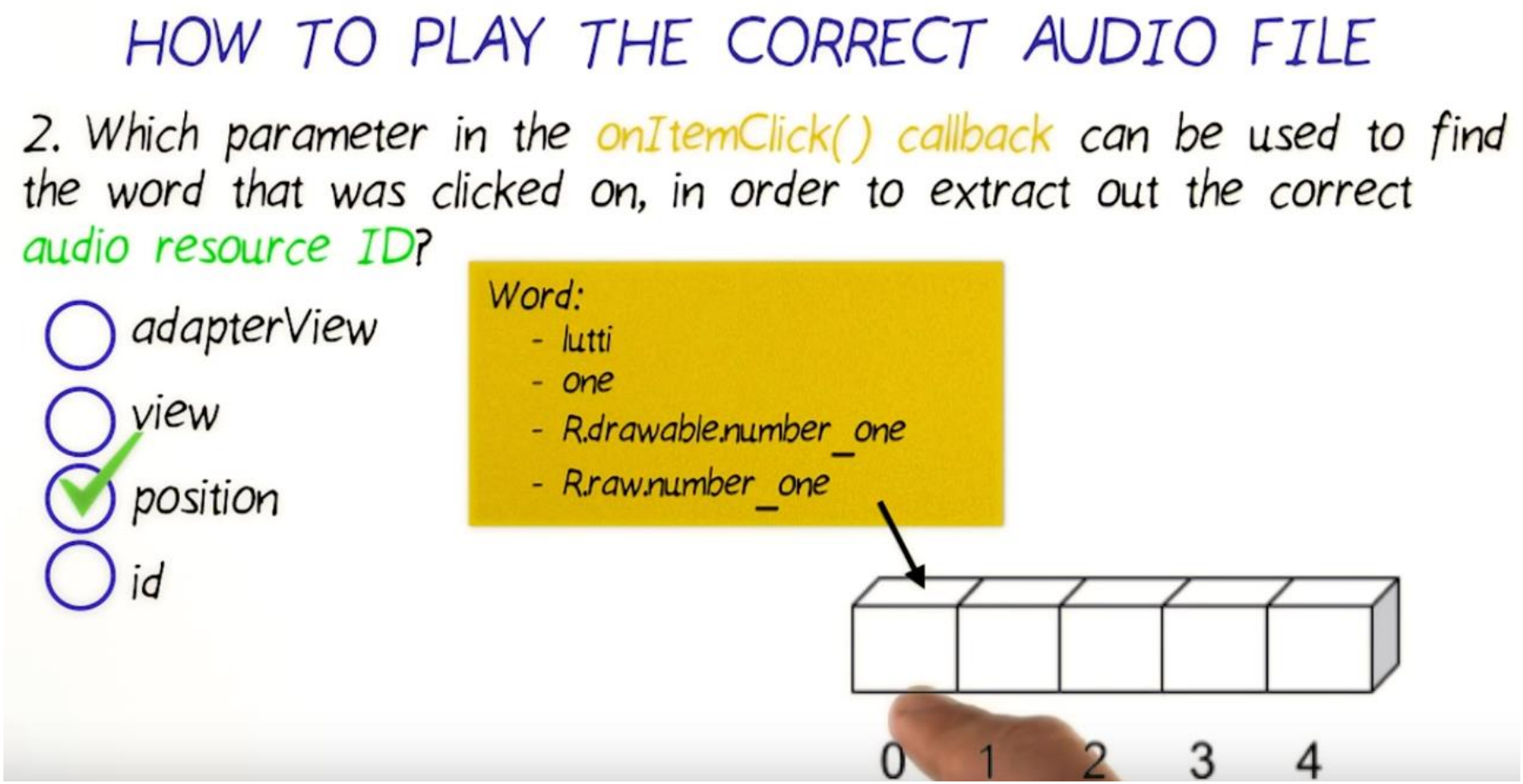
```
ListView listView = (ListView) findViewById(R.id.list);

// Make the {@link ListView} use the {@link WordAdapter} we created above, so that the
// {@link ListView} will display list items for each {@link Word} in the list.
listView.setAdapter(adapter);

listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        mMediaPlayer = MediaPlayer.create(NumbersActivity.this, R.raw.number_one);
        mMediaPlayer.start(); // no need to call prepare(); create() does that for you
    }
});
```

25 - Add remaining MP3 files

https://github.com/udacity/ud839_Miwok/tree/audio_assets



26 - Play correct audio file per word Word.Java



PLAY CORRECT AUDIO FILE PER WORD

Within the NumbersActivity, play the correct sound file when each word is clicked.

Repeat the same process for the other category activities.

1. Update Word class to store audio information for each word
2. Update the creation of our list of words using an updated Word Constructor
3. Update the OnItemClickListener to play the correct sound per word
4. Repeat This For Process For All Remaining Activities

Word.Java

- Add variable

```
/** Audio resource ID for the word */
private int mAudioResourceId;
```

- Modify both constructor

```
public Word(String defaultTranslation, String miwokTranslation, int audioResourceId) {
    mDefaultTranslation = defaultTranslation;
    mMiwokTranslation = miwokTranslation;
    mAudioResourceId = audioResourceId;
}
```

- Add method

```
    /**
     * Return the audio resource ID of the word.
     * @return
     */
    public int getmAudioResourceId(){ return mAudioResourceId; }
}
```

Numbers_Activity.java

```java
/** Handles playback of all the sound files */
private MediaPlayer mMediaPlayer;
```

```java
// Create a list of words
final ArrayList<Word> words = new ArrayList<~>();
words.add(new Word("one", "lutti", R.drawable.number_one, R.raw.number_one));
words.add(new Word("two", "otiiko", R.drawable.number_two, R.raw.number_two));
words.add(new Word("three", "tolookosu", R.drawable.number_three, R.raw.number_three));
words.add(new Word("four", "oyyisa", R.drawable.number_four, R.raw.number_four));
words.add(new Word("five", "massokka", R.drawable.number_five, R.raw.number_five));
words.add(new Word("six", "temmokka", R.drawable.number_six,R.raw.number_six));
words.add(new Word("seven", "kenekaku", R.drawable.number_seven, R.raw.number_seven));
words.add(new Word("eight", "kawinta", R.drawable.number_eight, R.raw.number_eight));
words.add(new Word("nine", "wo'e", R.drawable.number_nine, R.raw.number_nine));
words.add(new Word("ten", "na'aacha", R.drawable.number_ten, R.raw.number_ten));
```

```java
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Word word = words.get(position);
        mMediaPlayer = MediaPlayer.create(NumbersActivity.this, word.getAudioResourceId());
        mMediaPlayer.start(); // no need to call prepare(); create() does that for you
    }
});
```

Repeat for all activities!!! By copying

- Run the app

27 - Clean up MediaPlayer resources

RELEASE MEDIAPLAYER RESOURCES

Modify the Miwok app so that we release the MediaPlayer resources at the appropriate times:

- [ ] Use provided code snippet to release the resources after the sound file has finished playing

- [ ] Also release the MediaPlayer resources before the MediaPlayer is initialized to play a different song

** Be sure to make these changes across all category activities: NumbersActivity, PhrasesActivity, ColorsActivity, and FamilyActivity **

NumbersActivity.java (and other activities)

```
/**
 * Clean up the media player by releasing its resources.
 */
private void releaseMediaPlayer() {
    // If the media player is not null, then it may be currently playing a sound.
    if (mMediaPlayer != null) {
        // Regardless of the current state of the media player, release its resources
        // because we no longer need it.
        mMediaPlayer.release();

        // Set the media player back to null. For our code, we've decided that
        // setting the media player to null is an easy way to tell that the media player
        // is not configured to play an audio file at the moment.
        mMediaPlayer = null;
    }
}
```

Call releaseMediaPlayer

Put this at the top of NumbersActivity.java

```
private MediaPlayer.OnCompletionListener mCompletionListener = new MediaPlayer.OnCompletionListener() {
    @Override
    public void onCompletion(MediaPlayer mediaPlayer) {
        // Now that the sound file has finished playing, release the media player resources.
        releaseMediaPlayer();
    }
};
```

```
    listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> adapterView, View view, int position, long l)
            // Get the {@link Word} object at the given position the user clicked on
            Word word = words.get(position);

            releaseMediaPlayer();

            // Create and setup the {@link MediaPlayer} for the audio resource associated

            mMediaPlayer = MediaPlayer.create(NumbersActivity.this, word.getAudioResourceId(

            // Start the audio file
            mMediaPlayer.start();


            // Setup a listener on the media player, so that we can stop and release the
            // media
            mMediaPlayer.setOnCompletionListener(mCompletionListener);
        }
    });
}
```

DO for all activities and Run

The result will be playing sound and interrupting it

28 - Clean up MediaPlayer resources according to activity lifecycle

NumbersActivity.java

```
        });
    }

    @Override
    protected void onStop() {
        super.onStop();
        // When the activity is stopped, release the media player resources because we won't
        // be playing any more sounds.
        releaseMediaPlayer();
    }

    /**
     * Clean up the media player by releasing its resources.
     */
    private void releaseMediaPlayer() {
        // If the media player is not null, then it may be currently playing a sound.
        if (mMediaPlayer != null) {
            // Regardless of the current state of the media player, release its resources
            // because we no longer need it.
            mMediaPlayer.release();
```

Replicate in ll activities and Run the app. And the sound will be interrupted when leaf the activity

29 - Manage audio focus properly

- Request Audio Focus (NumbersActivity.java dulu) Add the variable:

```
/**Handles audio focus when playing a
private AudioManager mAudioManager;

/**
```

- Setup audio manager to request audio focus

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

    // Create and setup the {@link AudioManager} to request audio focus
    mAudioManager = (AudioManager) getSystemService(Context.AUDIO_SERVICE);
```

Modify onItemClick

```
public void onItemClick(AdapterView<?> adapterView, View view, int position, long l) {
    // Release the media player if it currently exists because we are about to
    // play a different sound file
    releaseMediaPlayer();

    // Get the {@link Word} object at the given position the user clicked on
    Word word = words.get(position);

    // Request audio focus so in order to play the audio file. The app needs to play a
    // short audio file, so we will request audio focus with a short amount of time
    // with AUDIOFOCUS_GAIN_TRANSIENT.
    int result = mAudioManager.requestAudioFocus(mOnAudioFocusChangeListener,
            AudioManager.STREAM_MUSIC, AudioManager.AUDIOFOCUS_GAIN_TRANSIENT);

    if (result == AudioManager.AUDIOFOCUS_REQUEST_GRANTED) {
        // We have audio focus now.

        // Create and setup the {@link MediaPlayer} for the audio resource associated
        // with the current word
        mMediaPlayer = MediaPlayer.create(NumbersActivity.this, word.getAudioResourceId());
```

Masih onItemClick

```
                // Start the audio file
                mMediaPlayer.start();

                // Setup a listener on the media player, so that we can stop and release the
                // media player once the sound has finished playing.
                mMediaPlayer.setOnCompletionListener(mCompletionListener);
            }
        }
    });
}
```

Manage focus change

```
41      private AudioManager.OnAudioFocusChangeListener mOnAudioFocusChangeListener = new AudioManager.OnAudioFocusChangeListener() {
42          @Override
43          public void onAudioFocusChange(int focusChange) {
44              if (focusChange == AudioManager.AUDIOFOCUS_LOSS_TRANSIENT ||
45                      focusChange == AudioManager.AUDIOFOCUS_LOSS_TRANSIENT_CAN_DUCK) {
46                  // The AUDIOFOCUS_LOSS_TRANSIENT case means that we've lost audio focus for a
47                  // short amount of time. The AUDIOFOCUS_LOSS_TRANSIENT_CAN_DUCK case means that
48                  // our app is allowed to continue playing sound but at a lower volume. We'll treat
49                  // both cases the same way because our app is playing short sound files.
50
51                  // Pause playback and reset player to the start of the file. That way, we can
52                  // play the word from the beginning when we resume playback.
53                  mMediaPlayer.pause();
54                  mMediaPlayer.seekTo(0);
```

```
55              } else if (focusChange == AudioManager.AUDIOFOCUS_GAIN) {
56                  // The AUDIOFOCUS_GAIN case means we have regained focus and can resume playback.
57                  mMediaPlayer.start();
58              } else if (focusChange == AudioManager.AUDIOFOCUS_LOSS) {
59                  // The AUDIOFOCUS_LOSS case means we've lost audio focus and
60                  // Stop playback and clean up resources
61                  releaseMediaPlayer();
62              }
63          }
64      };
```

Set abandon focus on releaseMediaPlayer()

```
private void releaseMediaPlayer() {
    // If the media player is not null, then it may be currently playing a sound.
    if (mMediaPlayer != null) {
        // Regardless of the current state of the media player, release its resources
        // because we no longer need it.
        mMediaPlayer.release();

        // Set the media player back to null. For our code, we've decided that
        // setting the media player to null is an easy way to tell that the media player
        // is not configured to play an audio file at the moment.
        mMediaPlayer = null;

        // Regardless of whether or not we were granted audio focus, abandon it. This also
        // unregisters the AudioFocusChangeListener so we don't get anymore callbacks.
        mAudioManager.abandonAudioFocus(mOnAudioFocusChangeListener);
    }
}
```

RUN the app

30 - Add audio icon to list item layout

Download the play arrow icon from the Material Design icons site

- (use the white 24dp version). Remember to include versions of the icon for all densities into your app (from mdpi → xxxhdpi).

List_item.xml

```
17  <RelativeLayout
18      xmlns:android="http://schemas.android.com/apk/res/android"
19      xmlns:tools="http://schemas.android.com/tools"
20      android:layout_width="match_parent"
21      android:layout_height="@dimen/list_item_height"
22      android:background="@color/tan_background"
23      android:minHeight="@dimen/list_item_height">
24
25      <ImageView
26          android:id="@+id/image"
27          android:layout_width="@dimen/list_item_height"
28          android:layout_height="@dimen/list_item_height" />
29
```

```
30      <RelativeLayout
31          android:id="@+id/text_container"
32          android:layout_width="match_parent"
33          android:layout_height="@dimen/list_item_height"
34          android:layout_alignParentBottom="true"
35          android:layout_alignParentRight="true"
36          android:layout_alignParentTop="true"
37          android:layout_toRightOf="@id/image"
38          android:orientation="vertical"
39          android:paddingLeft="16dp">
40
41          <TextView
42              android:id="@+id/miwok_text_view"
43              android:layout_width="match_parent"
44              android:layout_height="44dp"
45              android:layout_weight="1"
46              android:gravity="bottom"
47              android:textAppearance="?android:textAppearanceMedium"
48              android:textColor="@android:color/white"
49              android:textStyle="bold"
50              tools:text="lutti" />
```

```
52          <TextView
53              android:id="@+id/default_text_view"
54              android:layout_width="match_parent"
55              android:layout_height="44dp"
56              android:layout_below="@id/miwok_text_view"
57              android:layout_weight="1"
58              android:gravity="top"
59              android:textAppearance="?android:textAppearanceMedium"
60              android:textColor="@android:color/white"
61              tools:text="one" />
62
63          <ImageView
64              android:layout_width="24dp"
65              android:layout_height="24dp"
66              android:layout_alignParentRight="true"
67              android:layout_centerVertical="true"
68              android:layout_marginRight="16dp"
69              android:src="@drawable/ic_play_arrow" />
70      </RelativeLayout>
71  </RelativeLayout>
```

31 - Add pressed states to category views

Activity_main.xml

```
16  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
17      xmlns:tools="http://schemas.android.com/tools"
18      android:layout_width="match_parent"
19      android:layout_height="match_parent"
20      android:background="@color/tan_background"
21      android:orientation="vertical"
22      tools:context="com.example.android.miwok.MainActivity">
23
24      <!-- Numbers category -->
25      <FrameLayout
26          android:layout_width="match_parent"
27          android:layout_height="wrap_content"
28          android:background="@color/category_numbers">
29          <TextView
30              android:id="@+id/numbers"
31              style="@style/CategoryStyle"
32              android:background="?android:attr/selectableItemBackground"
33              android:text="@string/category_numbers" />
34      </FrameLayout>
```

```
36      <!-- Family category -->
37      <FrameLayout
38          android:layout_width="match_parent"
39          android:layout_height="wrap_content"
40          android:background="@color/category_family">
41          <TextView
42              android:id="@+id/family"
43              style="@style/CategoryStyle"
44              android:background="?android:attr/selectableItemBackground"
45              android:text="@string/category_family" />
46      </FrameLayout>
47
48      <!-- Colors category -->
49      <FrameLayout
50          android:layout_width="match_parent"
51          android:layout_height="wrap_content"
52          android:background="@color/category_colors">
53          <TextView
54              android:id="@+id/colors"
55              style="@style/CategoryStyle"
56              android:background="?android:attr/selectableItemBackground"
57              android:text="@string/category_colors" />
58      </FrameLayout>
```

```
60        <!-- Phrases category -->
61        <FrameLayout
62            android:layout_width="match_parent"
63            android:layout_height="wrap_content"
64            android:background="@color/category_phrases">
65            <TextView
66                android:id="@+id/phrases"
67                style="@style/CategoryStyle"
68                android:background="?android:attr/selectableItemBackground"
69                android:text="@string/category_phrases" />
70        </FrameLayout>
71    </LinearLayout>
```

32 - Add pressed states to list item views

Bisa di list_item.xml

```
57      <View
58          android:layout_width="match_parent"
59          android:layout_height="match_parent"
60          android:background="?android:attr/selectableItemBackground"/>
61
62  </RelativeLayout>
```

Atau di word_list.xml

```
<ListView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/list"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:drawSelectorOnTop="true"/>
```