الفصل السابع:

القوائم، تحويل النص إلى كلام والكلام إلى نص.

الكلمات المفتاحية:

عرض القائمة ListView، النص إلى كلام Text to Speech، الكلام إلى نص Speech to Text.

ملخص:

نستعرض في هذا الفصل آليات استخدام القوائم، إضافة إلى طرق التحويل من نص إلى كلام ومن كلام إلى نص.

أهداف تعليمية:

يتعرف الطالب في هذا الفصل على:

- استخدام القوائم.
- التحويل من نص إلى كلام.
- التحويل من كلام إلى نص

المخطط:

القوائم، تحويل النص إلى كلام والكلام إلى نص

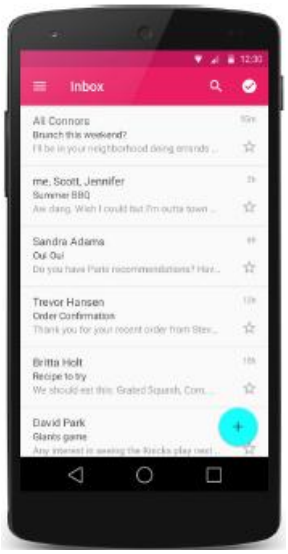
- 3 وحدات (Learning Objects)

1. القوائم Lists

الأهداف التعليمية:

- استخدام القوائم

عرض القائمة ListView



- عرض القائمة هو عبارة عن عنصر تحكم يحوي مجموعة مرتبة من الخيارات القابلة للتحديد.

- من أهم الخصائص:

إلغاء تفعيل القائمة بإسناد القيمة false	set to false to disable the list	android:clickable="*bool*"
المُعرّف الفريد للاستخدام في الكود	unique ID for use in Java code	android:id="@+id/*theID*"
مجموعة عناصر القائمة (تُقابل مصفوفة في الملف strings.xml)	set of options to appear in the list (must match an array in strings.xml)	android:entries="@array/*array* "

القوائم الساكنة Static Lists

- يكون محتواها ثابت ومعروف قبل التنفيذ.
- يتم تعريف عناصر القائمة في الملف strings.xml. مثلاً:

```
<!-- res/values/strings.xml -->

<resources>

<string-array name="oses">

<item>Android</item>

<item>iPhone</item>

        ...

<item>Max OS X</item>

</string-array>

</resources>
```

ويتم استخدامها في النشاط:

```
<!-- res/layout/activity_main.xml -->

<ListView ... android:id="@+id/mylist"

        android:entries="@array/oses" />
```

القوائم الديناميكية Dynamic Lists

- يتم توليد محتواها خلال تنفيذ البرنامج كقراءته من ملف أو من الوب.

```
// res/raw/oses.txt

Android

iPhone
...

Max OS X
```

موائم القوائم List Adapter

- يساعد الموائم في تحويل عناصر مصفوفة إلى عناصر عرض قائمة List View.
- يكون الشكل العام لإنشاء موائم:

```
ArrayAdapter<String> name =

        new ArrayAdapter<String>(activity, layout, array);
```

- وحيث:
  - تكون activity عادةً this.
  - النسق الافتراضي للقائمة هو android.R.layout.simple_list_item_1.
  - للحصول على المصفوفة array، يُمكن القراءة من ملف أو من مصدر بيانات. (يُمكن للمصفوفة أن تكون String[] أو ArrayList<String>).
- بعد إعداد الموائم، يُمكن استدعاء الطريقة setAdapter للغرض ListView. كما يُبين المثال التالي:

```
ArrayList<String> myArray = ...; // load data from file

ArrayAdapter<String> adapter =

        new ArrayAdapter<String>(

        this,

        android.R.layout.simple_list_item_1,

        myArray);

ListView list = (ListView) findViewById(R.id.mylist);

list.setAdapter(myAdapter);
```

الربط مع حدث النقر على عنصر من القائمة

- يتم ذلك باستخدام الشكل العام التالي:

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

list.setOnItemClickListener(

        new AdapterView.OnItemClickListener() {

 @Override

 public void onItemClick(AdapterView<?> list,

        View row,

        int index,

        long rowID) {

   // code to run when user clicks that item

        ...

        }

   });
```

- لاحظ أننا نستخدم أسلوب الصف الداخلي المجهول anonymous inner class لأن عرض القائمة لا يوفر طريقة للربط المباشر لحدث النقر على عنصر من القائمة.
- تذكرة: <?> تعني عام generic.

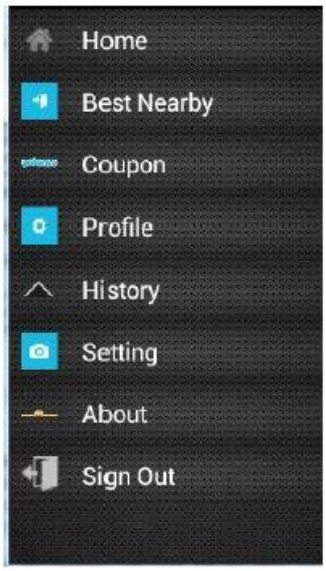
أحداث القائمة List events

نعرض فيما يلي لأهم أحداث القائمة:

- **setOnItemClickListener**(AdapterView.OnItemClickListener)
- //Listener for when an item in the list has been clicked.
- **setOnItemLongClickListener**(AdapterView.OnItemLongClickListener)
- //Listener for when an item in the list has been clicked and held.
- **setOnItemSelectedListener**(AdapterView.OnItemSelectedListener)
- //Listener for when an item in the list has been selected.
- onDrag
- onFocusChanged
- onHover
- onKey
- onScroll
- onTouch

تنسيق القائمة المخصص

- يُمكنك إجراء تنسيق مخصص للقائمة كما يلي:
  - كتابة ملف تنسيق layout XML لتنسيق عنصر القائمة.
  - كتابة صف مشتق من الصف ArrayAdapter مع كتابة الطريقة getView لتوصيف العرض الواجب إرجاعه لكل عنصر من القائمة.
- يعرض المثال التالي كيفية إنشاء قائمة مخصصة:



يكون ملف النشاط:

```
<!-- res/layout/mylistlayout.xml -->
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout ... android:orientation="horizontal">
<ImageView ... android:id="@+id/list_row_image"
    android:layout_width="100dp"
    android:layout_height="100dp"
    android:src="@drawable/smiley" />
<TextView ... android:id="@+id/list_row_text"
    android:textStyle="bold"
    android:textSize="22dp"
    android:text=""
```

```
    android:background="#336699" />

</LinearLayout>
```

ويكون ملف الكود:

```
// MyAdapter.java

public class MyAdapter extends ArrayAdapter<String> {

private int layoutResourceId;

private List<String> data;


public MyAdapter
  (Context context, int layoutId, List<String> list) {

        super(context, layoutResourceId, data);

        layoutResourceId = layoutId;

        data = list;

}

@Override

public View getView(int index, View row, ViewGroup parent) {

row = getLayoutInflater().inflate
     (layoutResourceId, parent, false);

TextView text = (TextView) row.findViewById(R.id.list_row_text);

text.setText(data.get(index));

return row;

    }

}
```

2. تحويل النص إلى كلام والكلام إلى نص

الأهداف التعليمية:

- تحويل النص إلى كلام
- تحويل الكلام إلى نص

التحويل من نص لكلام Text to Speech

- يسمح الأندرويد بتحويل سلسة نصية إلى كلام مسموع:
  - أنشئ غرض من الصف TextToSpeech
  - استدع الطريقة speak

الصف TextToSpeech

يكون للصف TextToSpeech الباني والطرق التالية:

- new TextToSpeech(***activity***, ***listener***) - constructor
- `getVoice`, `getVoices`, `setVoice` – change speaking voice
- `getLanguage`, `setLanguage` – sets language used
- `getPitch`, `setPitch` – sets vocal tone used
- `isSpeaking` – returns true if speaking
- `shutdown` – kills TTS engine
- `speak(`*text*`,` *mode*`,` *params*`)` – speaks given text aloud
- `stop` – halts any speech
- `synthesizeToFile(`***text***`,` ***params***`,` ***filename***`)` – speaks to file

التهيئة

- يُمكن لخدمة TextToSpeech أن تأخذ بعض الوقت حتى تتهيأ ولذا تكون التهيئة كما يلي:

```
TextToSpeech tts = new TextToSpeech(this,

        new TextToSpeech.OnInitListener() {

@Override

public void onInit(int status) {

// code to run when done loading

        }

});
```

- يجب إذاً انتظار انتهاء الإجرائية onInit قبل استدعاء speak وإلا فسيحدث استثناء.
- يُمكن استخدام متغير منطقي (flag) في النشاط يتم إعطائه القيمة true عند الانتهاء من التهيئة ولا تُستدعى الطريقة speak ما لم تكن قيمته true.

الطريقة speak

- تأخذ الطريقة speak ثلاثة معاملات:
  - النص (سلسلة نصية).
  - النمط: يأخذ إحدى القيمتين:

TextToSpeech.QUEUE_ADD: التكلم بعد انتهاء النصوص الأخرى.

TextToSpeech.QUEUE_FLUSH: إيقاف أي نص آخر والتكلم مباشرة.

  - متغير ثالث لن نتعرض له (نضعه null).

```
// speak text aloud, if my init boolean flag is set

if (myTTSisReady) {

        tts.speak("Hello, world!",

                TextToSpeech.QUEUE_FLUSH, null);

        }
```

التحويل من كلام لنص Speech to Text

- يقوم المستخدم بالتكلم فيُسجل الجهاز ومن ثم يحول الكلام إلى سلسلة نصية.
- يكون في الأندرويد نشاط جاهز لذلك.
- يُمكنك استدعاء هذا النشاط باستخدام جسر كما يُبين المثال التالي:

```
Intent intent = new Intent(
        RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE,
        Locale.getDefault());
        intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
        RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
// prompt text is shown on screen to tell user what to say
intent.putExtra(RecognizerIntent.EXTRA_PROMPT, "text");
startActivityForResult(intent, requestCode);
```

- بعد انتهاء النشاط text-to-speech، يُعيد جسره كلام المستخدم ضمن ArrayList وحيث يحوي عادةً العنصر ذو الفهرس 0 النص المطلوب.

```
@Override
protected void onActivityResult(int requestCode,
        int resultCode, Intent intent) {
super.onActivityResult(requestCode, resultCode, intent);
ArrayList<String> list = intent.getStringArrayListExtra(
        RecognizerIntent.EXTRA_RESULTS);
String spokenText = list.get(0);
// ...
}
```

- يُمكن ألا تدعم بعض الأجهزة هذه الميزة لذا من الأفضل معالجة الاستثناء الذي قد يحصل:

```
Intent intent = new Intent(
        RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
        ...
        try {
        startActivityForResult(intent, requestCode);
        } catch (ActivityNotFoundException anfe) {
    // code to handle the exception
        }
```

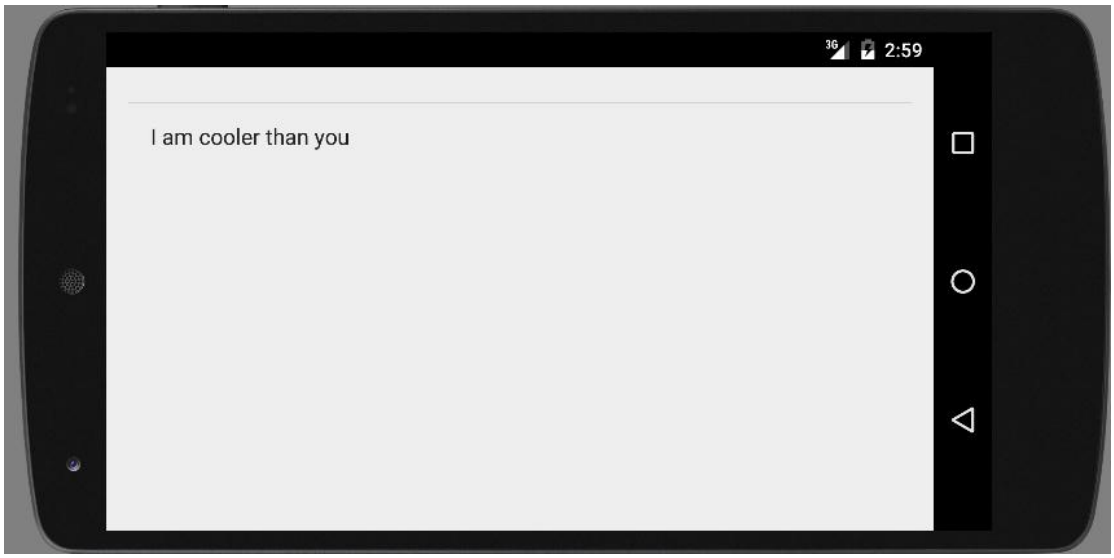
3. مثال تعليمي

الأهداف التعليمية:

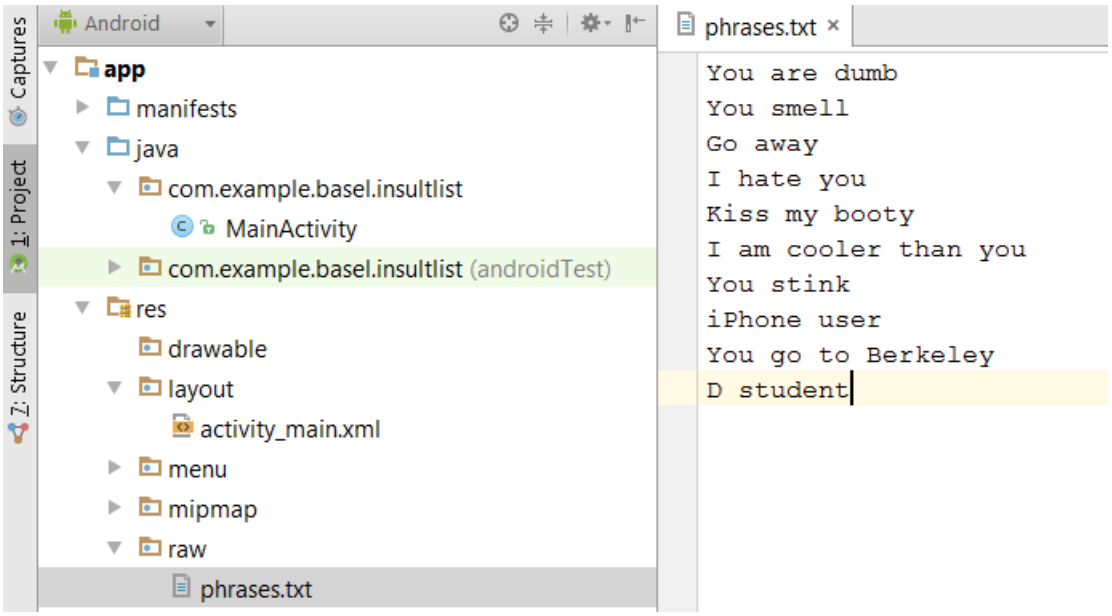
- مثال تعليمي.

مثال تعليمي

- عند تشغيل التطبيق تظهر قائمة منزلقة من عبارات التوبيخ:



- عند النقر على عبارة يتم قراءتها من قبل الجهاز.
- تكون عبارات التوبيخ في مثالنا في ملف نصي raw/phrases.txt:



- يحوي ملف النشاط activity_main.xml على عنصر عرض قائمة ListView ضمن عرض منزلق ScrollView.

```
<RelativeLayout

    xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools"
```

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">
<ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <ListView
            android:id="@+id/list_of_insults"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:theme="@android:style/Animation">
        </ListView>
</ScrollView>
</RelativeLayout>
```

- يكون ملف الكود MainActivity.java:

```
package com.example.basel.insultlist;
/*
 * This file implements the main activity for the insult list app.
 * It shows a list of insult phrases and speaks them aloud when
 * the user clicks on each one.
 */
```

```
import android.app.Activity;

import android.os.Bundle;

import android.speech.tts.TextToSpeech;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.ListView;

import java.util.ArrayList;

import java.util.Scanner;


public class MainActivity extends Activity {

private ArrayList<String> lines;        // lines of file of insults

private TextToSpeech tts;               // TTS engine

private boolean speechReady = false;// true when TTS engine is loaded


/*

* Initializes the state of the activity.

*/

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);


// read input file

lines = readEntireFile(R.raw.phrases);
```

```
// set up the ListView to use the lines from the file

ListView myList = (ListView)
       findViewById(R.id.list_of_insults);

ArrayAdapter<String> adapter = new ArrayAdapter<String>(

         this, android.R.layout.simple_list_item_1, lines);

myList.setAdapter(adapter);



// set up event listening for clicks on the list

myList.setOnItemClickListener(new
   AdapterView.OnItemClickListener() {

            @Override

            public void onItemClick(AdapterView<?> parent,
                           View view, int index, long id) {

                handleClick(index);

            }

        });


// set up text-to-speech engine

tts = new TextToSpeech(this, new TextToSpeech.OnInitListener() {

            @Override

            public void onInit(int status) {

                speechReady = true;

            }

        });
```

```
    }


/*
* Handles a click on the list item at the given 0-based index.
* Speaks the given insult aloud using text-to-speech.
*/
private void handleClick(int index) {
        String text = lines.get(index);
        if (speechReady) {
            tts.speak(text, TextToSpeech.QUEUE_FLUSH, null);
        }
}


/*
* Reads the lines of the file with the given resource ID,
* returning them as an array list of strings.
* Assumes that the file with the given ID exists in res/raw folder.
*/
private ArrayList<String> readEntireFile(int id) {
        ArrayList<String> list = new ArrayList<String>();
        Scanner scan = new
        Scanner(getResources().openRawResource(id));
        while (scan.hasNextLine()) {
            String line = scan.nextLine();
            list.add(line);
        }
```

```
        return list;

    }

}
```