時間服務及多媒體

AnalogClock 元件是圖形化時鐘，TextClock 元件是數字型態的時鐘，Timer 類別不但可設定執行程式的間隔時間，也能指定多少時間後才開始執行。

MediaPlayer 元件可以播放音訊及視訊，並且進行控制。VideoView 元件可以播放視訊。MediaRecorder 元件可以進行媒體採樣，要製作手機錄音功能軟體就不是件困難的任務了！

學習重點

- Timer 類別
- MediaPlayer 元件
- VideoView 視訊播放器
- SurfaceView 元件語法
- MediaPlayer 與 SurfaceView 結合
- 錄製音訊

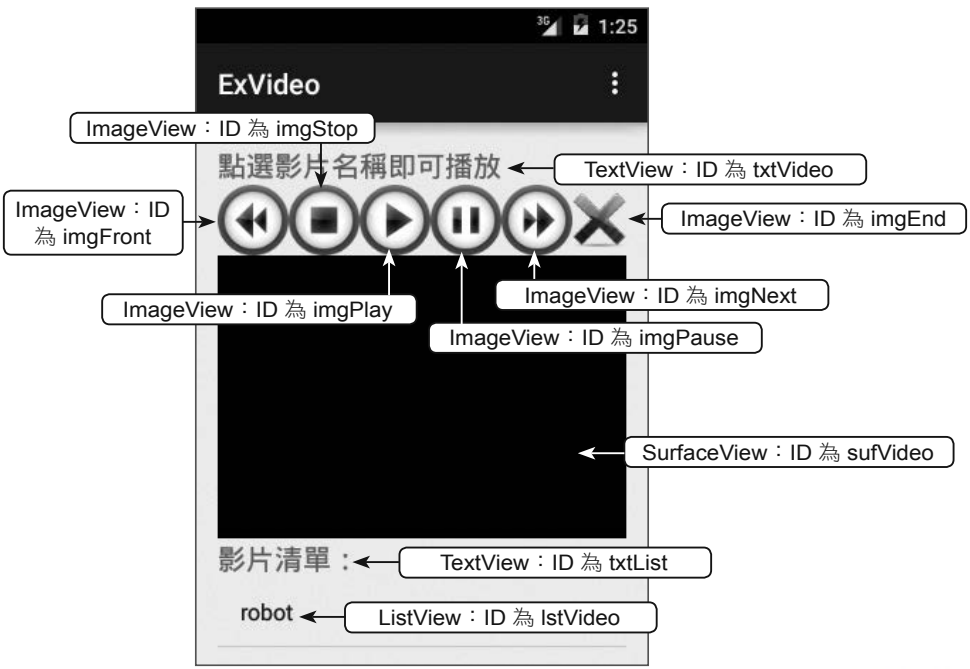
14

14.3.5 範例：自訂格式視訊播放器

本範例視訊播放器功能、介面與操作都與 14.2.4 節的播放音訊範例相同。

新增專案並完成版面配置

新增 <ExVideo> 專案，<activity_main.xml> 版面配置檔完成如下：



加入執行的程式碼

整體變數宣告及 onCreate() 啟動程式碼：

<ExVideo/app/java/com.ehappy.exvideo/MainActivity.java>

```
…略
25 private SurfaceView sufVideo;
…略
33 private MediaPlayer mediaplayer;
34 private SurfaceHolder sufHolder;
35 @Override
36 protected void onCreate(Bundle savedInstanceState) {
…略
57    mediaplayer=new MediaPlayer();
58    //建立 Surface 相關物件
59    sufHolder=sufVideo.getHolder();
60 }
```

- 第 34 列，宣告 SurfaceHolder 物件整體變數，於第 59 列做相關設定後，此 SurfaceHolder 物件即可做為視訊顯示用。

播放視訊的 playVideo 方法程式碼：

續：<ExVideo/app/java/com.ehappy.exvideo/MainActivity.java>

```
106 private void playVideo(String path) {
107   mediaplayer.reset();
108   mediaplayer.setAudioStreamType(AudioManager.STREAM_MUSIC);
109   mediaplayer.setDisplay(sufHolder);
110   try
111   {
112      mediaplayer.setDataSource(path); //播放影片路徑
113      mediaplayer.prepare();
114      mediaplayer.start(); //開始播放
115      txtVideo.setText("影片名稱：" + videoname[cListItem]); //更新名稱
116      mediaplayer.setOnCompletionListener(new OnCompletionListener() {
117         public void onCompletion(MediaPlayer arg0) {
118            nextVideo(); //播放完後播下一片
119         }
120      });
121   } catch (IOException e) {}
122   falgPause=false;
123 }
```

- 第 108-109 列，設定 MediaPlayer 元件的資料流格式及顯示的 SurfaceHolder 物件後就可播放視訊。
- 第 112 列，因為本範例的視訊檔位於 SD 卡，所以要使用 setDataSource 方法設定視訊來源檔案。

其餘處理按鈕程式 (listener)、選取 ListView 項目處理程式 (lstListener)、播放上一片 (frontVideo) 及播放下一片 (nextVideo) 等的程式碼皆與 MusicPlayer 專案相同，不再贅述。

儲存專案後執行專案。

14.4 錄製音訊

需要使用錄音的場合非常多，例如參加研討會時要將研習內容錄製下來供以後參考，聽到喜歡的音樂可以錄下來做為手機鈴聲等。以往錄音就要攜帶體積龐大的錄音機，後來進步到一支小小的錄音筆，但最讓使用者困擾的是：要錄音時才發現手邊沒帶錄音器材。手機上的錄音功能解決了這些問題，因為絕大多數手機使用者無時無刻都帶著手機，也就是隨時都有錄音器材在身邊。

14.4.1 MediaRecorder 元件語法

Android 提供 MediaRecorder 元件來進行媒體採樣，MediaRecorder 元件常用的方法如下表：

方法	說明
prepare	多媒體準備錄製。
release	結束 MediaRecorder 元件。
reset	重置 MediaRecorder 元件。
setAudioEncoder	設定音訊編碼格式。
setAudioSource	設定音訊來源。
setMaxDuration	設定最大錄製時間長度。
setMaxFileSize	設定最大檔案大小。
setOutputFile	設定輸出檔案。
setOutputFormat	設定輸出檔案格式。
setVideoEncoder	設定視訊編碼格式。
setVideoFrameRate	設定視訊影格頻率。
setVideoSize	設定視訊解析度。
setVideoSource	設定視訊來源。
start	開始錄製。
stop	停止錄製。

使用這些 MediaRecorder 元件的方法，製作錄音功能是輕而易舉的事！

在 Android 中執行錄製音訊的步驟如下：

1. 錄製音訊的第一步是建立 MediaRecorder 元件，例如建立名稱為 mediarecorder 的 MediaRecorder 元件：

```
MediaRecorder mediarecorder= new MediaRecorder();
```

2. 接著設定麥克風為輸入音訊來源：

```
mediarecorder.setAudioSource(MediaRecorder.AudioSource.MIC);
```

3. 設定輸出檔案格式及音訊編碼方式皆為預設值：

```
mediarecorder.setOutputFormat(MediaRecorder.OutputFormat.DEFAULT);
mediarecorder.setAudioEncoder(MediaRecorder.AudioEncoder.DEFAULT);
```

4. 在 Android 中預設的音訊輸出檔案格式為「amr」。

   最後以 setOutputFile 方法指定錄音後的儲存檔案名稱，例如將檔案存於 SD 卡根目錄，名稱為 <record.amr>：

```
mediarecorder.setOutputFile(Environment.
  getExternalStorageDirectory().getPath() + "/record.amr");
```

5. 與播放音訊相同，錄製音訊前也要先做好準備工作再開始錄製：

```
mediarecorder.prepare();
mediarecorder.start();
```

6. 因為系統要使用到麥克風輸入音訊，並將檔案寫入 SD 卡，所以要在 <AndroidManifest.xml> 檔中設定允許使用麥克風及存取 SD 卡功能的權限，語法為：

```
<uses-permission android:name="android.permission.
  RECORD_AUDIO"></uses-permission>
<uses-permission android:name="android.permission.
  WRITE_EXTERNAL_STORAGE"></uses-permission>
```

如此就完成錄製音訊的工作！

14.4.2 範例：MediaRecorder 錄音機

有了 MediaRecorder 元件，要製作手機錄音功能軟體就不是件困難的任務了！Android 模擬器可以模擬錄音功能，但是錄音的品質很差，開發者可以在模擬器中測試錄製音訊功能，等到應用程式開發完成，再於實機錄製真實的音訊檔案。

程式開啟後，會自動讀取 SD 卡中的錄音檔顯示於下方列表，點按檔名即可播放錄音檔。 按 鈕就開始錄音，按 鈕停止錄音，按 鈕可播放剛錄的音訊檔，螢幕上方會隨時顯示目前的工作狀態。錄音檔自動以錄製的日期及時間組合做為檔案名稱，以記錄錄製時間。按鈕會隨不同功能變化，灰階表示按鈕無效。

新增專案並完成版面配置

新增 <ExRecord> 專案，<activity_main.xml> 版面配置檔完成如下：

加入權限設定

加入允許麥克風輸入及寫入 SD 卡權限的 <AndroidManifest.xml> 檔：

<ExRecord/ExRecordManifest.xml>

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.ehappy.exrecord" >

    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-permission android:name="android.
        permission.WRITE_EXTERNAL_STORAGE" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
…略
```

加入執行的程式碼

1. 整體變數宣告及 onCreate() 啟動程式碼：

<ExRecord/app/java/com.ehappy.exrecord/MainActivity.java>

```
…略
27 private MediaPlayer mediaplayer;
28 private MediaRecorder mediarecorder;
29 private String temFile; // 以日期時間做暫存檔名
30 private File recFile, recPATH;
31 private List<String> lstArray=new ArrayList<String>(); // 檔名陣列
32 private int cListItem=0; // 目前播放錄音
33 @Override
34 protected void onCreate(Bundle savedInstanceState) {
…略
47    lstRec.setOnItemClickListener(lstListener);
48    recPATH=Environment.getExternalStorageDirectory(); //SD 卡路徑
40    mediaplayer=new MediaPlayer();
50    imgDisable(imgStop);
51    recList(); // 錄音列表
52 }
…略 (處理按鈕程式碼後續說明)
```

❶ ❷ ❸ ❹

```
134 //取得錄音檔案列表
135 public void recList() {
136   lstArray.clear(); //清除陣列
137   for(File file:recPATH.listFiles()) {
138      if(file.getName().toLowerCase().endsWith(".amr")) {
139         lstArray.add(file.getName());
140      }
141   }
142   if(lstArray.size()>0) {                                    ←❺
143      ArrayAdapter<String> adaRec=new ArrayAdapter<String>(this,
         android.R.layout.simple_list_item_1, lstArray);
144      lstRec.setAdapter(adaRec);
145   }
146 }
147
148 private void imgEnable(ImageView image) { //使按鈕有效
149   image.setEnabled(true);
150   image.setAlpha(255);                                       ←❻
151 }
152
153 private void imgDisable(ImageView image) { //使按鈕失能
154   image.setEnabled(false);
155   image.setAlpha(50);                                        ←❼
156 }
```

❶ 第 27 列宣告 MediaPlayer 物件做為播放音訊用，第 28 列宣告 MediaRecorder 物件做為錄製音訊用。第 29 列宣告 temFile 暫時存放錄音檔名，程式以日期結合時間做為錄音檔名，此檔名暫時存於 temFile 中，需結合 SD 卡路徑才是錄音檔的實體路徑。錄音檔實體路徑存於第 30 列的 recFile 中，recPATH 存放 SD 卡路徑，SD 卡路徑則是在第 48 列以「Environment.getExternalStorageDirectory()」方法取得。

❷ 本範例 ListView 顯示的資料為 SD 卡中的錄音檔案列表，其內容及長度是變動的，所以無法以簡單陣列型式儲存，必須儲存於 ArrayList 中，因為 ArrayList 才可以使用 add、remove 等方法動態加入或移除陣列元素。此處宣告 ArrayList 變數 lstArray 儲存錄音檔案列表。

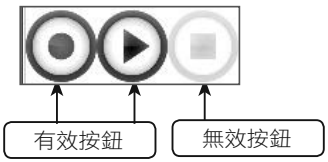
❸ 因為程式啟動時，並沒有錄音及播放音訊動作，所以在此處設定 **停止** 鈕失效。此列程式位於 onCreate 方法中，效果是程式啟動後無法使用 **停止** 鈕。

❹ 在啟動程式時執行 recList 方法取得 SD 卡中已存在的錄音檔案列表，以 ListView 元件顯示。

❺ recList 方法顯示 SD 卡中已存在的錄音檔案列表。首先於第 136 列以 clear 方法將原先陣列內容清除乾淨，第 137-141 列逐一檢查 SD 卡根目錄中每一個檔案，第 138 列判斷檔案的附加檔名是否為「amr」錄音檔，如果是錄音檔就在第 139 列加入檔案列表中。當所有檔案都檢查完畢後，第 142 列檢查是否有錄音檔存在，若有錄音檔才將陣列做為 ListView 元件的內容於第 143-144 列顯示出來，如果沒有錄音檔則不顯示 ListView 元件。

❻ 由於 ImageView 不會因圖形是否可按而顯示不同狀態，使用者將無法判斷按鈕是否有效，因此使用透明度自行訂定兩個方法來顯示不同圖形：imgEnable 方法設定圖形按鈕有效，同時將透明度設定為 255，即一般正常狀態。

❼ imgDisable 方法設定圖形按鈕無效，同時將透明度設定為 50，即較暗淡狀態。如此使用者就可明確判斷按鈕是否有效。



圖形透明度設定

第 150 及 155 列設定圖形透明度的程式碼被畫上刪除線，但不影響其功能：

```
image.setAlpha(n);
```

這是因為從 Android 4.1.2 版 (API 16) 後建議由 setImageAlpha 方法取代：

```
image.setImageAlpha(n);
```

若使用 setImageAlpha 方法設定透明度，必須修改 <AndroidManifest.xml> 檔中 android:minSdkVersion 屬性值為 16，如此一來，若應用程式安裝在 Android 4.1.2 版以下版本時將無法執行。因此本書仍使用 setAlpha 方法設定圖形透明度。

2. 處理按鈕的程式碼：

說明：<ExRecord/app/java/com.ehappy.exrecord/MainActivity.java>

```
54 private ImageView.OnClickListener listener=new ImageView.OnClickListener() {
55     @Override
56     public void onClick(View v) {
57       switch(v.getId())
58       {
59         case R.id.imgRecord:  // 錄音
60           try {
61           Time t=new Time();
62           t.setToNow(); // 取得現在日期及時間
63           temFile="R"+add0(t.year)+add0(t.month+1)+add0(t.monthDay)
                +add0(t.hour)+add0(t.minute)+add0(t.second);
64           recFile=new File(recPATH + "/" + temFile + ".amr");
65           mediarecorder= new MediaRecorder();
66           mediarecorder.setAudioSource(MediaRecorder.AudioSource.MIC);
67           mediarecorder.setOutputFormat(MediaRecorder.OutputFormat.DEFAULT);
68           mediarecorder.setAudioEncoder(MediaRecorder.AudioEncoder.DEFAULT);
69           mediarecorder.setOutputFile(recFile.getAbsolutePath());
70           mediarecorder.prepare();
71           mediarecorder.start();
72           txtRec.setText(" 正在錄音……");
73           imgDisable(imgRecord); // 處理按鈕是否可按
74           imgDisable(imgPlay);
75           imgEnable(imgStop);
76           } catch (IOException e) {
77              e.printStackTrace();
78           }
79           break;
80         case R.id.imgPlay:  // 播放
81           playSong(recPATH + "/" + lstArray.get(cListItem).toString());
82           break;
83         case R.id.imgStop:  // 停止
84           if (mediaplayer.isPlaying()) { //// 停止播放
85              mediaplayer.reset();
86           } else if(recFile!=null) { // 停止錄音
87              mediarecorder.stop();
88              mediarecorder.release();
89              mediarecorder=null;
90              txtRec.setText("停止" + recFile.getName() + "錄音！");
91              recList();
```

```
92              }
93              imgEnable(imgRecord);
94              imgEnable(imgPlay);
95              imgDisable(imgStop);
96              break;
97          case R.id.imgEnd:  // 結束
98             finish();
99            break;
100         }
101     }
102 };
…略
158 protected String add0(int n) { // 個位數前面補零
159     if(n<10) return ("0" + n);
160     else return ("" + n);
161 }
```

- 第 60-78 列，按下 **錄音** 鈕的處理程式碼。第 61-62 列取得目前系統時間，取得目前時間的年、月、日、時、分、秒結合成暫時檔案名稱。第 158-161 列，add0 程式碼是自行撰寫的方法，功能是將個位數的數值前面補一個零，如此每個數值都會是兩個字元，看起來非常美觀，例如 2014 年 2 月 8 日 7 時 12 分 5 秒就成為「20140208071205」。這樣的檔案名稱不但記錄了錄音時間，而且絕不會重複。第 64 列將 SD 卡路徑結合暫時檔案就得到實體路徑，第 65-71 列執行錄音工作，第 72 列顯示目前正在錄音訊息。第 73-75 列處理按鈕狀態：因為正在錄音，只有 **停止** 鈕有效，所以第 73-74 列讓 **錄音** 及 **播放** 鈕失效，第 75 列設定 **停止** 鈕有效。
- 第 84-95 列是按下 **停止** 鈕的處理程式碼，此按鈕同時處理停止錄音及停止播放工作。第 84-85 列檢查是否正在播放音訊，如果是就停止播放。第 86-92 列檢查是否正在錄音，如果是就停止錄音。注意第 91 列執行 recList 方法，如此就會將剛錄製的音訊檔加入下方列表，使用者點選後可立即播放錄製內容。
- 第 110-116 列處理點選 ListView 項目處理程式，及 118 到 138 列播放音訊檔案程式都與 MusicPlayer 專案相同，不再贅述。