行動裝置數位匯流應用實務

相機 – 行車記錄器

王昱景 Brian Wang
brian.wang.frontline@gmail.com

- 目前 Android 行動裝置大部分都配備相當畫素的相機功能
- Android 作業系統已經內建相機程式
- 可以使用 Intent 物件啟動，但是對於應用程式需要使用相機功能時

- Android SDK 提供 android.hardware.Camera 類別 (不是 android.graphics.Camera 類別)
- 它是硬體相機的介面類別，相機服務的客戶端類別
- 可以照相、擷取圖片、預覽圖片和更改相關設定

- 程式提供兩種照相功能
- 一是使用 Intent 物件啟動內建相機程式，然後將照相結果顯示在 ImageView 元件
- 另一是使用 SurfaceView 元件預覽畫面，和 Camera 類別照相和儲存至 SD 卡

- Android 應用程式可以使用 android.media.MediaRecorder 類別進行錄影

方法	說明
setVideoSource()	指定視訊來源是 Camera 相機或 Default 預設來源
setOutputFormat()	指定錄影輸出的檔案格式，在 setVideoSource() 之後；prepare() 之前呼叫
setVideoEncoder()	指定錄影的編碼方式，在 setOutputFormat() 之後；prepare() 之前呼叫
setOutputFile()	指定輸出的檔案名稱， 在 setOutputFormat() 之後；prepare() 之前呼叫
setPreviewDisplay()	指定預覽顯示的 SurfaceView 元件，在 prepare() 之前呼叫
prepare()	準備好裝置，可以開始錄影，它需要在設定好視訊相關參數後，start() 之前呼叫
start()	開始錄影至 setOutputFile() 指定的檔案，它是在 prepare() 之後呼叫
stop()	停止錄影
release()	釋放 MediaRecorder 物件佔用的資源

DrivingRecordDemo

- 開啟和執行 Android 專案
- 建立主活動使用介面的版面配置
- 建立主活動類別
- 建立 VideoRecorder 錄影活動類別
- 建立預覽錄影的 VideoPreview 類別
- 建立播放錄影的活動類別與版面配置
- 在 AndroidManifest.xml 註冊活動和新增權限

1.開啟和執行Android專案

- 請啟動 Eclipse IDE
- 建立 Android 專案
  - Project Name: DrivingRecordDemo
  - Build Target: Android API 4.2.2
  - Package Name

2. 建立主活動使用介面的版面配置

- 使用 LinearLayout 編排 3 個 Button 和 1 個 TextView 元件

```
<Button
    android:id="@+id/button1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="10px"
    android:onClick="button1_Click"
    android:text="@string/button1" />

<Button
    android:id="@+id/button2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="10px"
    android:onClick="button2_Click"
    android:text="@string/button2" />

<Button
    android:id="@+id/button3"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="10px"
    android:onClick="button3_Click"
    android:text="@string/button3" />

<TextView
    android:id="@+id/file"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

3. 建立主活動類別

- 在活動類別提供按鈕來啟動內建相機程式來錄影和直接進行錄影
- 在類別開頭定義常數，和宣告成員的 File 物件變數

```
public class MainActivity extends Activity {

    private static final int REQUEST_VIDEO_CAPTURE = 101;
    private File file;
    private String fileName = "myVideo.3gp";

}
```

onCreate() 方法

- 在覆寫 onCreate() 方法載入版面配置後，建立儲存檔案路徑的 File 物件

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

    // 建立儲存檔案路徑的File物件
    file = new File(Environment.getExternalStorageDirectory(), fileName);
}
```

onActivityResult() 方法

- 覆寫 onActivityResult() 方法可以取得相機程式回傳儲存的檔案路徑

- 然後在 TextView 元件顯示此路徑

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == REQUEST_VIDEO_CAPTURE && resultCode == Activity.RESULT_OK) {
        // 取得傳回媒體儲存的檔案路徑
        String path = data.getData().toString();
        TextView output = (TextView) findViewById(R.id.file);
        output.setText(path); // 顯示路徑
    }
}
```

button1~3_Click() 事件處理方法

- Button 元件的事件處理，都是建立 Intent 物件來啟動活動

```
// Button元件的事件處理
public void button1_Click(View view) {
    // 使用Intent物件啟動VideoRecorder活動
    Intent intent = new Intent(this, VideoRecorder.class);
    startActivity(intent);
}

public void button2_Click(View view) {
    // 建立使用內建程式錄影的Intent物件
    Intent intent = new Intent(MediaStore.ACTION_VIDEO_CAPTURE);
    // 新增附件為儲存的媒體檔案
    intent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(file));
    // 指定錄影品質
    intent.putExtra(MediaStore.EXTRA_VIDEO_QUALITY, 0);
    startActivityForResult(intent, REQUEST_VIDEO_CAPTURE);
}

public void button3_Click(View view) {
    // 使用Intent物件啟動VideoPlayer活動
    Intent intent = new Intent(this, VideoPlayer.class);
    startActivity(intent);
}
```

4. 建立 VideoRecorder 錄影活動類別

- 在 VideoRecorder 錄影活動類別開頭宣告成員的 MediaRecorder 和 VideoPreview 物件變數
- 旗標變數 isRecording 判斷是否在錄影中

```
public class VideoRecorder extends Activity {

    // 宣告MediaRecorder和Preview物件變數
    private MediaRecorder recorder;
    private VideoPreview preview;
    // 是否是錄影中
    private boolean isRecording = false;
```

onCreate() 方法

- 在覆寫 onCreate() 方法建立 MediaRecorder 物件和指定錄影參數

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // 建立MediaRecorder物件
    recorder = new MediaRecorder();
    // 指定錄影的參數
    recorder.setVideoSource(MediaRecorder.VideoSource.DEFAULT);
    recorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
    recorder.setVideoEncoder(MediaRecorder.VideoEncoder.MPEG_4_SP);

    // 建立錄影預覽的VideoPreview物件
    preview = new VideoPreview(this, recorder);

    // 橫向顯示
    setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE);
    setContentView(preview); // 指定活動的使用介面
}
```

onPrepareOptionsMenu() 方法

- 在覆寫 onPrepareOptionsMenu() 方法建立選項選單，它是使用程式碼來建立選項

```
// 建立選項選單
@Override
public boolean onPrepareOptionsMenu(Menu menu) {
    super.onPrepareOptionsMenu(menu);

    menu.clear();

    // 新增選項
    menu.add(0, 0, 0, "開始錄影");
    menu.add(1, 1, 0, "停止錄影");

    // 預設指定選項為不可見
    menu.setGroupVisible(0, false);
    menu.setGroupVisible(1, false);

    if (isRecording == false) {
        menu.setGroupVisible(0, true); // 開始錄影
    } else {
        menu.setGroupVisible(1, true); // 停止錄影
    }

    return true;
}
```

onOptionsItemSelected() 方法

- 在覆寫 onOptionsItemSelected() 方法處理選項選單的選項，0 是呼叫 start() 方法開始錄影；1 是呼叫 stop() 方法停止錄影

```
// 處理選項選單的選項
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case 0: // 開始錄影
            recorder.start();
            isRecording = true;
            break;
        case 1: // 停止錄影
            recorder.stop();
            // 釋放MediaRecorder物件佔用的資源
            recorder.release();
            recorder = null;
            isRecording = false;
            break;
    }

    return super.onOptionsItemSelected(item);
}
```

5. 建立預覽錄影的 VideoPreview 類別

- 預覽錄影的 VideoPreview 類別是繼承自 SurfaceView 類別且實作 Callback 介面
- 在類別開頭宣告 SurfaceHolder 和 MediaRecorder 物件變數

```
public class VideoPreview extends SurfaceView implements Callback {

    private SurfaceHolder holder;
    private MediaRecorder recorder;

}
```

VideoPreview() 方法

- 在 VideoPreview() 建構子取得參數的 MediaRecorder 物件

```
public VideoPreview(Context context, MediaRecorder recorder) {
    super(context);

    this.recorder = recorder;

    holder = getHolder();
    holder.addCallback(this);
    holder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
}
```

getSurface() 方法

- getSurface() 方法可以傳回 SurfaceView 物件

```
public Surface getSurface() {
    return holder.getSurface();
}
```

實作 SurfaceHolder.Callback() 介面方法

- 活動類別實作 SurfaceHolder.Callback 介面的 3 個方法
- 不過只使用前 2 個方法
- 在 surfaceCreated() 方法指定輸出檔案路徑，和預覽檢視是 getSurface() 方法傳回的 SurfaceView 物件

```java
// 實作SurfaceHolder.Callback介面方法
@Override
public void surfaceCreated(SurfaceHolder holder) {
    // 指定輸出檔案路徑
    recorder.setOutputFile("/sdcard/myVideo.3gp");
    // 指定預覽檢視
    recorder.setPreviewDisplay(holder.getSurface());

    try {
        recorder.prepare(); // 準備錄影
    } catch (Exception e) {
        Log.e("DrivingRecordDemo", e.getMessage());
        recorder.release();
        recorder = null;
    }
}

@Override
public void surfaceDestroyed(SurfaceHolder holder) {
    if (recorder != null) {
        // 釋放MediaRecorder物件佔用的資源
        recorder.release();
        recorder = null;
    }
}

@Override
public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
}
```

6.建立播放錄影的活動類別與版面配置

```
<VideoView
    android:id="@+id/video"
    android:layout_width="480px"
    android:layout_height="480px"
    android:layout_x="10px"
    android:layout_y="10px" />
```

```java
public class VideoPlayer extends Activity {

    private VideoView videoView;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // 沒有標題文字
        requestWindowFeature(Window.FEATURE_NO_TITLE);

        // 橫向顯示
        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE);
        setContentView(R.layout.videoplayer);

        // 取得VideoView元件
        videoView = (VideoView) this.findViewById(R.id.video);

        // 建立MediaController物件
        MediaController mc = new MediaController(this);

        videoView.setMediaController(mc); // 指定控制物件
        // 指定媒體檔案的播放路徑的URI
        videoView.setVideoURI(Uri.parse("/sdcard/myVideo.3gp"));
        // 指定元件取得焦點
        videoView.requestFocus();
    }

}
```

7. 在 AndroidManifest.xml 註冊活動和新增權限

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

```
        <activity android:name="VideoPlayer" android:label="@string/app_name">
        </activity>
        <activity android:label="@string/app_name" android:name="VideoRecorder">
        </activity>
```