

Android Tutorial using Kotlin 第四堂

（1）使用照相機與麥克風

[Android Tutorial using Kotlin 第三堂（3）使用Android內建的SQLite資料庫 << 前情](#)

現在行動裝置的硬體設備技術已經越來越好了，螢幕的尺寸不斷的增加，提供使用者清楚又美觀的畫面。觸控螢幕也幾乎是行動裝置的標準設備，使用觸控的方式操作應用程式快速又方便。Android系統內建的音樂播放應用程式，也可以讓行動裝置成的音樂播放設備。還有畫素也越來越高的照像功能，一台行動裝置幾乎可以應付所有的需求。

行動裝置提供高畫質的攝影鏡頭，讓使用者隨時可以拍攝照片與錄影，也幾乎已經是行動裝置基本的設備與功能了。使用Android系統內建的API與元件，可以在應用程式需要的時候，讓使用者拍攝照片與錄影，並且把照片或影片檔案儲存在指定的位置。例事本應用程式中，可以加入照片與錄影備忘的功能。

應用程式需要錄音的時候，可以使用內建的API執行錄音的工作，並且把錄音完成的檔案儲存在指定的位置，例如在記事本應中，可以加入錄製語音備忘的功能，讓使用者可以隨時查詢與播放這些錄音資訊。

這一章為記事資料加入照相與錄音的功能，讓這個應用程式的功能可以更完整，使用者可以在新增或修改記事資料的時候，用相機拍照，還有使用麥克風錄製語音備忘。

12-1 使用相機拍攝照片

不論是行動電話或平板電腦，幾乎都有高畫質的攝錄鏡頭設備，讓使用者可以隨時拍攝與錄影。加入拍攝照片的功能可以讓原式的功能更完整，例如在記事本應用程式加入拍照的功能，記錄影像會比文字更清楚與方便。

應用程式需要執行拍照的功能，可以啟動系統相機元件執行拍照的工作，它的系統Action名稱變數是「MediaStore.ACTION_IMAGE_CAPTURE」，使用這個Action名稱建立好的Intent物件，可以呼叫putExtra函式加入照片檔案儲存的設定資料，資料的名稱是「MediaStore.EXTRA_OUTPUT」，如果沒有指定的話，會使用系統預設的名稱儲存在預設的位置

應用程式要執行拍照的功能，裝置必須有攝錄鏡頭的設備才可以正確的執行，所以需要在應用程式設定檔中加入硬體設備需求。如果需要儲存照片檔案到外部儲存設備，例如記憶卡，需要在應用程式設定檔中加入授權設定：

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

    <!-- 需要攝錄鏡頭設備 -->
    <uses-feature
        android:name="android.hardware.camera"
        android:required="true" />

    <!-- 寫入外部儲存設備 -->
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

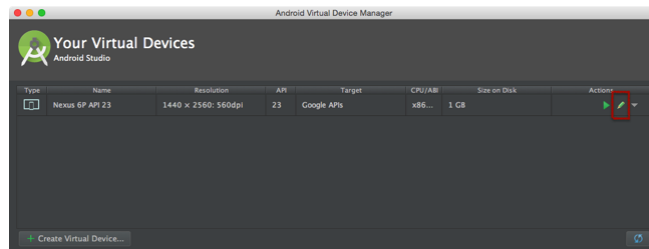
```
<application ...>

...

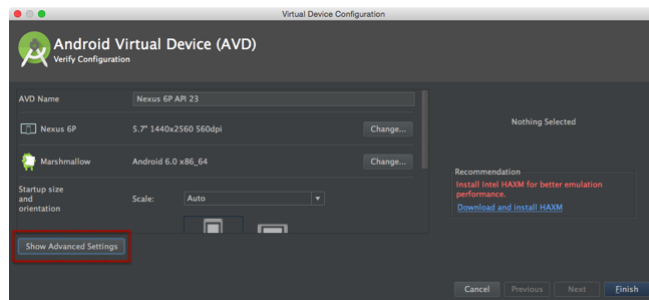
</application>

</manifest>
```

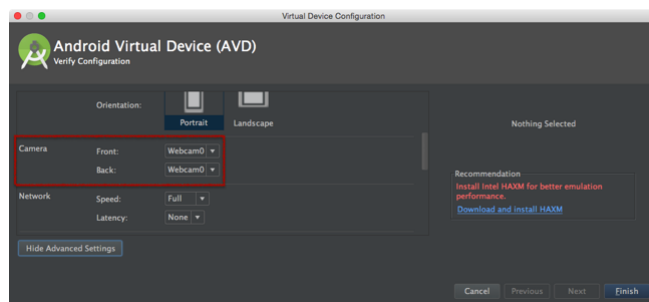
Android模擬裝置也可以測試相機的功能，不過要先確認模擬裝置的設定，關閉已經啟動的模擬裝置，在Android Studio選擇「Tools -> Android -> AVD Manager」，選擇模擬裝置的編輯圖示：



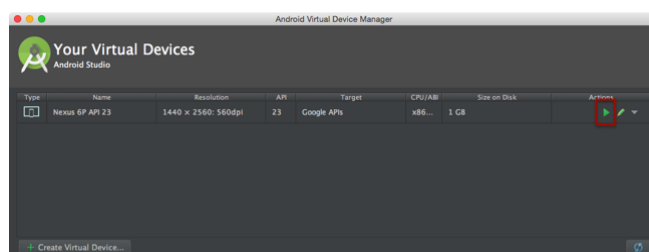
在模擬裝置編輯視窗選擇「Show Advanced Settings」：



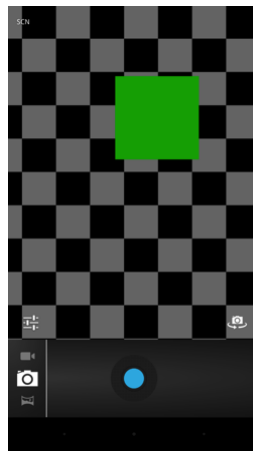
如果你的電腦沒有連接WebCam，在「Front」與「Back」選擇「Emulated」。如果電已經連接WebCam，就可以選擇「Webcam0」。完成設定後選擇「Finish」：



回到AVD Manager視窗後，選擇模擬裝置的啟動圖示：



模擬裝置啟動以後，如果相機設定為Emulated，開啟「照相」應用程式，就可以看到模擬照相機的畫面：



因為記事元件的畫面加入照片以後，在螢幕比較小的裝置運作時，畫面會超過螢幕的範圍，所以需要調整畫面的設計。另外加入顯示照片用的**ImageView**元件。開啟「**res/layout/activity_item.xml**」，參考下列的內容修改這個畫面配置檔：

```
<?xml version="1.0" encoding="utf-8"?>

<!-- 使用ScrollView為最外層的元件 -->
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <!-- 刪除xmlns:android的設定 -->
    <TableLayout
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:stretchColumns="1"
        tools:context="net.macdidi.myandroidtutorial.ItemActivity">

        <TableRow>
            ...
        </TableRow>

        <TableRow>
            ...
        </TableRow>

        <!-- 顯示圖片 -->
        <ImageView
            android:id="@+id/picture"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@drawable/rectangle_drawable"
            android:padding="6sp"
            android:layout_margin="2sp"
            android:visibility="invisible" />

        <TableLayout ...>
            <TableRow>
```

```

        ...
    </TableRow>
</TableLayout>

<TableLayout ...>
    <TableRow>
        ...
    </TableRow>
</TableLayout>
</TableLayout>

<!-- ScrollView的結束標籤 -->
</ScrollView>

```

因為需要儲存照片與錄音檔案，所以撰寫一個檔案公用類別。在「net.macdidi.atk」套件按滑鼠右鍵，選擇「New -> Kotlin File/Class」，在Name輸入「FileUtil」後選擇「OK」。參考下列的內容完成這個程式碼：

```

package net.macdidi.atk

import android.graphics.BitmapFactory
import android.util.Log
import android.widget.ImageView
import java.io.File
import java.text.SimpleDateFormat
import java.util.*

// 讀取指定的照片檔案名稱設定給ImageView元件
fun fileToImageView(fileName: String, imageView: ImageView) {
    if (File(fileName).exists()) {
        val bitmap = BitmapFactory.decodeFile(fileName)
        imageView.setImageBitmap(bitmap)
    } else {
        Log.e("fileToImageView", fileName + " not found.")
    }
}

// 產生唯一的檔案名稱
fun getUniqueFileName(): String {
    // 使用年月日_時分秒格式為檔案名稱
    val sdf = SimpleDateFormat("yyyyMMdd_HH:mm:ss")
    return sdf.format(Date())
}

```

開啟在「net.macdidi.atk」套件下的「ItemActivity」類別，加入照相功能需要的欄位變數：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    // 照片檔案名稱
    private var pictureFileName: String? = null
    // 照片元件
    private val picture: ImageView by bind(R.id.picture)
    // 寫入外部儲存設備授權請求代碼
    private val REQUEST_WRITE_EXTERNAL_STORAGE_PERMISSION = 100

    ...

}

```

同樣在「ItemActivity」類別，新增執行拍攝照片與檔案名稱的函式：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    // 拍攝照片
    private fun takePicture() {
        // 取得照片檔案物件
        val file = getFileName("P", ".jpg")
        val uri : Uri

        // 如果是LOLLIPOP_MR1或更新的版本
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP_MR1) {
            // 使用FileProvider建立Uri物件
            uri = FileProvider.getUriForFile(this,
                BuildConfig.APPLICATION_ID + ".provider",
                file)
        }
        else {
            uri = Uri.fromFile(file)
        }

        // 啟動相機元件用的Intent物件
        val intentCamera = Intent(MediaStore.ACTION_IMAGE_CAPTURE)
    }
}

```

```

        // 設定檔案名稱
        intentCamera.putExtra(MediaStore.EXTRA_OUTPUT, uri)
        // 啟動相機元件
        startActivityForResult(intentCamera, ItemAction.CAMERA.ordinal)
    }

    // 取得照片檔案名稱物件
    private fun getFileName(prefix: String, extension: String): File {
        // 如果記事資料已經有照片檔案名稱
        if (!item.fileName.isNullOrEmpty()) {
            pictureFileName = item.fileName
        }
        // 產生檔案名稱
        else {
            pictureFileName = getUniqueFileName()
        }

        // 儲存照片的目錄
        val photoPath = File(Environment.getExternalStorageDirectory(), "photo")

        if (!photoPath.exists()) {
            // 建立儲存照片的目錄
            photoPath.mkdir()
        }

        // 傳回照片檔案物件
        return File(photoPath, "$prefix$pictureFileName$extension")
    }

    ...
}

```

開啟「res/values/strings.xml」，加入需要的文字資源：

```
<string name="write_external_storage_denied">沒有寫入外部儲存設備授權</string>
```

開啟「ItemActivity」類別，為了處理Android 6的授權架構，新增讀取與處理寫入外部儲存設備授權請求的函式，還有覆寫使用授權選擇以後執行的函式：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

```

...

// 讀取與處理寫入外部儲存設備授權請求

```
private fun requestStoragePermission() {
    // 如果裝置版本是6.0（包含）以上
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
        // 取得授權狀態，參數是請求授權的名稱
        val hasPermission = ContextCompat.checkSelfPermission(
            this, Manifest.permission.WRITE_EXTERNAL_STORAGE)

        // 如果未授權
        if (hasPermission != PackageManager.PERMISSION_GRANTED) {
            // 請求授權
            // 第一個參數是請求授權的名稱
            // 第二個參數是請求代碼
            requestPermissions(
                arrayOf<String>(Manifest.permission.WRITE_EXTERNAL_STORAGE),
                REQUEST_WRITE_EXTERNAL_STORAGE_PERMISSION)

            return
        }
    }

    // 如果裝置版本是6.0以下，
    // 或是裝置版本是6.0（包含）以上，使用者已經授權，
    // 拍攝照片
    takePicture()
}
```

```
override fun onRequestPermissionsResult(requestCode : Int,
                                         permissions : Array<String>,
                                         grantResults : IntArray) {

    // 如果是寫入外部儲存設備授權請求
    if (requestCode == REQUEST_WRITE_EXTERNAL_STORAGE_PERMISSION) {
        // 如果在授權請求選擇「允許」
        if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {
            // 拍攝照片
            takePicture()
        }
        // 如果在授權請求選擇「拒絕」
        else {
            Toast.makeText(this, R.string.write_external_storage_denied,
                Toast.LENGTH_SHORT).show()
        }
    }
    else {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults)
    }
}
```

...

```
}
```

同樣在「ItemActivity」類別，找到「clickFunction」函式，參考下列的程式碼，加入啟動相機元件的程式碼：

```
package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    fun clickFunction(view: View) {
        when (view.id) {
            R.id.take_picture -> {
                // 讀取與處理寫入外部儲存設備授權請求
                requestStoragePermission()
            }
            R.id.record_sound -> {
            }
            R.id.set_location -> {
            }
            R.id.set_alarm -> {
            }
            //選擇設定顏色功能
            R.id.select_color -> {
                // 啟動設定顏色的Activity元件
                startActivityForResult(Intent(this, ColorActivity::class.java),
                    ItemAction.COLOR.ordinal)
            }
        }
    }

    ...

}
```

同樣在「ItemActivity」類別，找到「onActivityResult」函式，參考下列的程式碼，處理完成照相工作後的程式碼：

```
package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {
```



```

...

// 更改參數data的型態為Intent?
override fun onActivityResult(requestCode: Int,
                               resultCode: Int,
                               data: Intent?) {
    if (resultCode == Activity.RESULT_OK) {
        val actionRequest = ItemAction.values()[requestCode]

        when (actionRequest) {
            // 照像
            ItemAction.CAMERA -> {
                // 設定照片檔案名稱
                item.fileName = pictureFileName
            }
            ...
        }
    }
}

...

}

```

同樣在「ItemActivity」類別，新增覆寫「onResume」函式的程式碼，執行顯示照片的工作：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    override fun onResume() {
        super.onResume()

        // 如果有照片檔案名稱
        if (!item.fileName.isNullOrEmpty()) {
            // 照片檔案物件
            val file = getFileName("P", ".jpg")

            // 如果照片檔案存在
            if (file.exists()) {
                // 顯示照片元件
                picture.visibility = View.VISIBLE
                // 設定照片
                fileToImageView(file.absolutePath, picture)
            }
        }
    }
}

```

```

        }

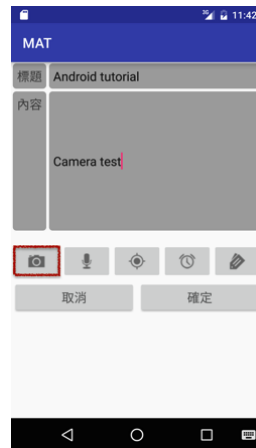
    }

    ...

}

```

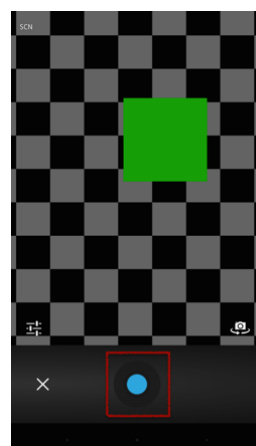
完成照相功能的工作了，執行應用程式，新增一個記事資料，選擇照相功能：



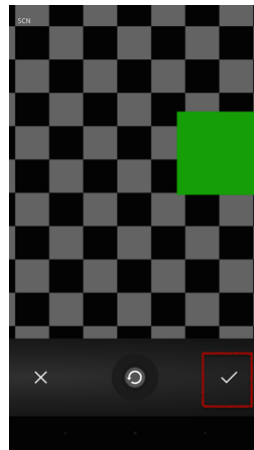
畫面出現詢問使用者是否授權應用程式儲存照片檔案的對話框，選擇「允許」：



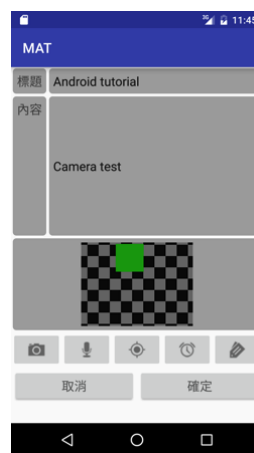
畫面出現像這樣的相機模擬畫面，選擇照像按鈕：



選擇確定按鈕：



記事資料顯示拍攝的照片，儲存記事資料後也會儲存照片：



如果在詢問使用者授權的畫面選擇「拒絕」，應用程式會顯示沒有授權的訊息：



如果使用者決定不要使用應用程式的照相功能，可以勾選「不要再詢問我」以後選擇「拒絕」，下次就不會再出現授權對話框



12-2 錄製語音備忘

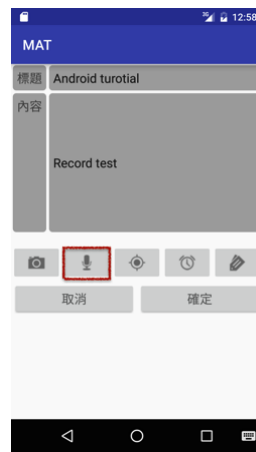
在行動裝置的應用程式使用錄音功能，可以讓很多工作變得更方便，例如語音備忘錄的功能，可以省掉很多輸入文字的時間。應用程式需要執行錄音的工作，使用宣告在「android.media」套件下的「MediaRecorder」類別，應用程式可以設定錄音的輸出格式、編碼和儲存檔案的位置。這些是執行設定與錄音的函式，要特別注意在程式碼中呼叫它們的順序：

- `setAudioSource(int)` – 設定錄音來源，必須在`setOutputFormat`函式之前呼叫。設定為「`MediaRecorder.AudioSource.MIC`」錄音來源是麥克風。
- `setOutputFormat(int)` – 設定輸出格式，必須在`setAudioSource`函式之後。設定為「`MediaRecorder.OutputFormat.THREE_G`」輸出為3GP壓縮格式。
- `setAudioEncoder(int)` – 設定編碼方式，必須在`setOutputFormat`函式之後。一般設定為「`MediaRecorder.AudioEncoder.AMR_NB`」。
- `setOutputFile(String)` – 設定輸出的檔案名稱，必須在`setOutputFormat`函式之後。
- `prepare()` – 使用設定的內容準備錄音。
- `start()` – 開始錄音。
- `stop()` – 停止錄音。
- `release()` – 清除錄音資源。

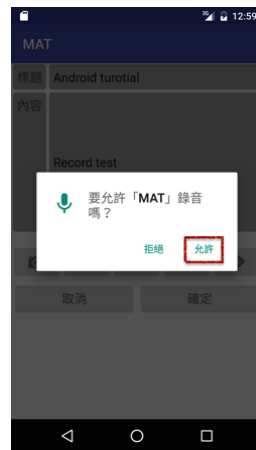
如果應用程式需要使用裝置的錄音設備，必須在應用程式設定檔「`AndroidManifest.xml`」加入授權的設定：

```
<!-- 使用錄音設備 -->
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
```

加入錄音功能的記事應用程式，可以讓使用者選擇錄音功能按鈕：



如果使用者在錄音設備授權畫面選擇「允許」：



啟動錄音的Activity元件以後，按下錄音按鈕就可以開始錄音：



錄音的時候，錄音按鈕會切換為紅色的圖示，錄音的音量變化會在右側顯示：



設計錄音元件的畫面配置檔，使用的圖形資源可以在GitHub中取得，需要「recordddardicon.png」與「recordredicon.png」兩
示檔案。開啟「res/values/strings.xml」，加入這個元件需要的文字資源：

```
<string name="title_record">語音備忘</string>
<string name="title_play">播放語音備忘</string>
<string name="record_play">播放</string>
<string name="record_new">重新錄製</string>
<string name="record_audio_denied">沒有錄音設備授權</string>
```

在「net.macdidi.atk」套件按滑鼠右鍵，選擇「New -> Activity -> Empty Activity」，在Name輸入「RecordActivity」後選擇
「OK」。參考下列的內容完成這個Activity元件的程式碼：（這裡提供的設計包含顯示錄音中的音量，你可以考慮移除這個部
式碼，這個元件的設計就會比較簡單一些）

```
package net.macdidi.atk

import android.app.Activity
import android.media.MediaRecorder
import android.os.AsyncTask
import android.os.Bundle
import android.util.Log
import android.view.View
import android.widget.ImageButton
import android.widget.ProgressBar
import java.io.IOException

// 從AppCompatActivity改為Activity
class RecordActivity : Activity() {

    private val record_button : ImageButton by bind(R.id.record_button)
    private var isRecording : Boolean = false
    private val record_volumn : ProgressBar by bind(R.id.record_volumn)

    private lateinit var fileName : String
    private var myRecorder: MyRecorder? = null
```

```

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_record)
    // 隱藏狀態列ProgressBar
    setProgressBarIndeterminateVisibility(false);

    // 讀取檔案名稱
    fileName = intent.getStringExtra("fileName")
}

fun onSubmit(view: View) {
    if (isRecording) {
        // 停止錄音
        myRecorder?.stop()
    }

    // 確定
    if (view.getId() === R.id.record_ok) {
        setResult(Activity.RESULT_OK, intent)
    }

    finish()
}

fun clickRecord(view: View) {
    // 切換
    isRecording = !isRecording

    // 開始錄音
    if (isRecording) {
        // 設定按鈕圖示為錄音中
        record_button.setImageResource(R.drawable.record_red_icon)
        // 建立錄音物件
        myRecorder = MyRecorder(fileName)
        // 開始錄音
        myRecorder?.start()
        // 建立並執行顯示麥克風音量的AsyncTask物件
        MicLevelTask().execute()
    }
    // 停止錄音
    else {
        // 設定按鈕圖示為停止錄音
        record_button.setImageResource(R.drawable.record_dark_icon)
        // 麥克風音量歸零
        record_volumn.progress = 0
        // 停止錄音
        myRecorder?.stop()
    }
}

// 在錄音過程中顯示麥克風音量
private inner class MicLevelTask : AsyncTask<Void, Void, Void>() {

```

```

        override fun doInBackground(vararg args: Void): Void? {
            while (isRecording) {
                publishProgress()

                try {
                    Thread.sleep(200)
                } catch (e: InterruptedException) {
                    Log.d("RecordActivity", e.toString())
                }
            }

            return null
        }

        override fun onProgressUpdate(vararg values: Void) {
            record_volumn.progress = myRecorder?.amplitudeEMA?.toInt() ?: 0
        }
    }

    // 執行錄音並且可以取得麥克風音量的錄音物件
    private inner class MyRecorder internal constructor(private val output: String) {

        private var recorder: MediaRecorder? = null
        private var mEMA = 0.0
        private val EMA_FILTER = 0.6

        val amplitude: Double
        get() = if (recorder != null)
            recorder!!.maxAmplitude / 2700.0
        else
            0.0

        // 取得麥克風音量
        val amplitudeEMA: Double
        get() {
            val amp = amplitude
            mEMA = EMA_FILTER * amp + (1.0 - EMA_FILTER) * mEMA
            return mEMA
        }

        // 開始錄音
        fun start() {
            if (recorder == null) {
                // 建立錄音用的MediaRecorder物件
                recorder = MediaRecorder()
                // 設定錄音來源為麥克風，必須在setOutputFormat函式之前呼叫
                recorder!!.setAudioSource(MediaRecorder.AudioSource.MIC)
                // 設定輸出格式為3GP壓縮格式，必須在setAudioSource函式之後，
                // 在prepare函式之前呼叫
                recorder!!.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP)
            }
        }
    }

```



```

        // 設定錄音的編碼方式，必須在setOutputFormat函式之後，
        // 在prepare函式之前呼叫
        recorder!!.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB)
        // 設定輸出的檔案名稱，必須在setOutputFormat函式之後，
        // 在prepare函式之前呼叫
        recorder!!.setOutputFile(output)

        try {
            // 準備執行錄音工作，必須在所有設定之後呼叫
            recorder!!.prepare()
        } catch (e: IOException) {
            Log.d("RecordActivity", e.toString())
        }

        // 開始錄音
        recorder!!.start()
        mEMA = 0.0
    }
}

// 停止錄音
fun stop() {
    if (recorder != null) {
        // 停止錄音
        recorder!!.stop()
        // 清除錄音資源
        recorder!!.release()
        recorder = null
    }
}
}
}

```

開啟「res/layout/activity_record.xml」，參考下列的內容完成這個畫面配置檔：

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@drawable/rectangle_drawable"
        android:layout_margin="6sp"
        android:padding="6sp">

```

```

        <ImageButton
            android:id="@+id/record_button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/record_sound_icon"
            android:onClick="clickRecord" />

        <ProgressBar
            android:id="@+id/record_volumn"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_vertical"
            android:layout_marginLeft="6dp"
            android:layout_marginRight="6dp"
            android:max="15"
            style="@android:style/Widget.ProgressBar.Horizontal" />
    </LinearLayout>

    <TableLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:stretchColumns="*">

        <TableRow>

            <Button
                android:text="@android:string/cancel"
                android:onClick="onSubmit" />

            <Button
                android:id="@+id/record_ok"
                android:text="@android:string/ok"
                android:onClick="onSubmit" />

        </TableRow>
    </TableLayout>

</LinearLayout>

```

開啟應用程式設定檔「AndroidManifest.xml」，修改錄音元件的設定：

```

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

    ...

    <application ...>
        ...
    </application>
</manifest>

```

```

        <!-- 錄音元件 -->
        <activity
            android:name=".RecordActivity"
            android:label="@string/title_record"
            android:theme="@android:style/Theme.Dialog" />

    </application>

</manifest>

```

完成錄音元件的設計後，開啟在「net.macdidi.atk」套件下的「ItemActivity」類別，加入下列的欄位變數：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    // 錄音設備授權請求代碼
    private val REQUEST_RECORD_AUDIO_PERMISSION = 101
    // 錄音檔案名稱
    private var recFileName: String? = null

    ...

}

```

同樣在「ItemActivity」類別，新增執行錄音、播放功能與檔案名稱的函式：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    // 錄音與播放
    fun processRecord() {
        // 錄音檔案名稱
        val recordFile = getRecFileName("R", ".mp3")

        // 如果已經有錄音檔，詢問播放或重新錄製
    }
}

```

```

        if (recordFile.exists()) {
            // 詢問播放還是重新錄製的對話框
            val d = AlertDialog.Builder(this)

            d.setTitle(R.string.title_record)
                .setCancelable(false)
            d.setPositiveButton(R.string.record_play,
                DialogInterface.OnClickListener { dialog, which ->
                    // 播放
                    // 在後面的說明才會處理
                })

            d.setNeutralButton(R.string.record_new,
                DialogInterface.OnClickListener { dialog, which ->
                    // 重新錄音
                    val recordIntent = Intent(this@ItemActivity, RecordActivity::class.java)
                    recordIntent.putExtra("fileName", recordFile.absolutePath)
                    startActivityForResult(recordIntent, ItemAction.RECORD.ordinal)
                })

            d.setNegativeButton(android.R.string.cancel, null)

            // 顯示對話框
            d.show()
        }
        // 如果沒有錄音檔，啟動錄音元件
    else {
        // 錄音
        val recordIntent = Intent(this, RecordActivity::class.java)
        recordIntent.putExtra("fileName", recordFile.absolutePath)
        startActivityForResult(recordIntent, ItemAction.RECORD.ordinal)
    }
}

private fun getRecFileName(prefix: String, extension: String): File {
    // 如果記事資料已經有錄音檔案名稱
    if (!item.recFileName.isNullOrEmpty()) {
        recFileName = item.recFileName
    } else {
        // 產生檔案名稱
        recFileName = getUniqueFileName()
    }
}

// 儲存錄音的目錄
val recordPath = File(Environment.getExternalStorageDirectory(), "record")

if (!recordPath.exists()) {
    // 建立儲存錄音的目錄
    recordPath.mkdir()
}

// 傳回錄音檔案物件

```

```

        return File(recordPath, "$prefix$recFileName$extension")
    }

    ...
}

```

同樣在「ItemActivity」類別，為了處理Android 6的授權架構，新增錄音設備授權請求的函式：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    // 讀取與處理錄音設備授權請求
    private fun requestRecordPermission() {
        // 如果裝置版本是6.0（包含）以上
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            // 取得授權狀態，參數是請求授權的名稱
            val hasPermission = ContextCompat.checkSelfPermission(
                this, Manifest.permission.RECORD_AUDIO)

            // 如果未授權
            if (hasPermission != PackageManager.PERMISSION_GRANTED) {
                // 請求授權
                // 第一個參數是請求授權的名稱
                // 第二個參數是請求代碼
                requestPermissions(
                    arrayOf(Manifest.permission.RECORD_AUDIO),
                    REQUEST_RECORD_AUDIO_PERMISSION)
                return
            }
        }

        // 如果裝置版本是6.0以下，
        // 或是裝置版本是6.0（包含）以上，使用者已經授權，
        // 錄音或播放
        processRecord()
    }

}

```

同樣在「ItemActivity」類別，找到「onRequestPermissionsResult」函式，加入錄音設備授權請求的程式碼：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    override fun onRequestPermissionsResult(requestCode : Int,
                                           permissions : Array<String>,
                                           grantResults : IntArray) {
        if (requestCode == REQUEST_WRITE_EXTERNAL_STORAGE_PERMISSION) {
            ...
        }
        // 如果是使用錄音設備授權請求
        else if (requestCode == REQUEST_RECORD_AUDIO_PERMISSION) {
            // 如果在授權請求選擇「允許」
            if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                // 錄音或播放
                processRecord();
            }
            // 如果在授權請求選擇「拒絕」
            else {
                // 顯示沒有授權的訊息
                Toast.makeText(this, R.string.record_audio_denied,
                               Toast.LENGTH_SHORT).show();
            }
        }

        else {
            super.onRequestPermissionsResult(requestCode, permissions, grantResults)
        }
    }

    ...
}

```

同樣在「ItemActivity」類別，找到「clickFunction」函式，加入讀取與處理錄音設備授權請求的程式碼：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    fun clickFunction(view: View) {

```

```

        when (view.id) {
            R.id.take_picture -> {
                ...
            }
            R.id.record_sound -> {
                // 讀取與處理錄音設備授權請求
                requestRecordPermission()
            }
            ...
        }
    }

    ...
}

```

同樣在「net.macdidi.atk」套件下的「ItemActivity」類別，找到「onActivityResult」函式，加入設定檔案名稱的程式碼：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    // 更改參數data的型態為Intent?
    override fun onActivityResult(requestCode: Int,
                                    resultCode: Int,
                                    data: Intent?) {
        if (resultCode == Activity.RESULT_OK) {
            val actionRequest = ItemAction.values()[requestCode]

            when (actionRequest) {
                ItemAction.CAMERA -> {
                    ...
                }
                ItemAction.RECORD -> {
                    // 設定錄音檔案名稱
                    item.recFileName = recFileName
                }
                ...
            }
        }
    }

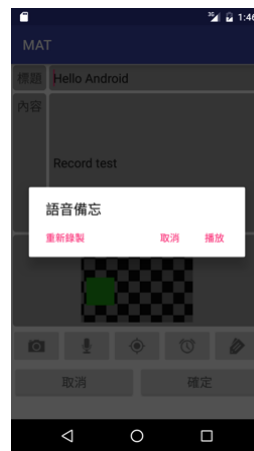
    ...
}

```

為記事資料完成錄音的功能了，在完成後面的播放功能後，再一起執行測試的工作。

12-3 播放語音備忘

在前面已經完成的功能，如果使用者選擇的記事資料已經錄製過語音備忘，應用程式可以選擇播放或是重新錄製：



使用者選擇播放功能，應用程式啟動播放語音備忘元件，這個元件提供播放、暫停與停止三個功能按鈕：



現在設計錄音元件的畫面配置檔，使用的圖形資源可以在GitHub取得，需要「playicon.png」、「pauseicon」與「stop_icon」三個圖示檔案。

在「net.macdidi.atk」套件按滑鼠右鍵，選擇「New -> Activity -> Empty Activity」，在Name輸入「PlayActivity」後選擇「C」參考下列的內容完成這個Activity元件的程式碼：

```
package net.macdidi.atk

import android.app.Activity
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.view.View

// 從AppCompatActivity改為Activity
class PlayActivity : Activity() {

    // 播放元件
```



```

private lateinit var mediaPlayer : MediaPlayer

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_play)

    // 讀取與設定錄音檔案名稱
    val fileName : String = intent.getStringExtra("fileName")
    val uri : Uri = Uri.parse(fileName)
    mediaPlayer = MediaPlayer.create(this, uri)
}

override fun onStop() {
    if (mediaPlayer.isPlaying) {
        // 停止播放
        mediaPlayer.stop()
    }

    // 清除MediaPlayer物件
    mediaPlayer.release()
    super.onStop()
}

fun onSubmit(view: View) {
    // 結束Activity元件
    finish()
}

fun clickPlay(view: View) {
    // 開始播放
    mediaPlayer.start()
}

fun clickPause(view: View) {
    // 暫停播放
    mediaPlayer.pause()
}

fun clickStop(view: View) {
    // 停止播放
    if (mediaPlayer.isPlaying) {
        mediaPlayer.stop()
    }

    // 回到開始的位置
    mediaPlayer.seekTo(0)
}
}

```

開啟「res/layout/activity_play.xml」，參考下列的內容完成這個畫面配置檔：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@drawable/rectangle_drawable"
        android:layout_margin="6sp"
        android:padding="6sp" >
        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/play_icon"
            android:onClick="clickPlay" />
        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/pause_icon"
            android:onClick="clickPause" />
        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/stop_icon"
            android:onClick="clickStop" />
    </LinearLayout>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/ok_add_teim"
        android:text="@android:string/ok"
        android:onClick="onSubmit" />

</LinearLayout>
```

開啟應用程式設定檔「AndroidManifest.xml」，加入這個Activity元件的設定：

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

```

...

<application ...>
    ...

    <!-- 播放元件 -->
    <activity
        android:name=".PlayActivity"
        android:theme="@android:style/Theme.Dialog"
        android:label="@string/title_play"/>
    </application>

</manifest>

```

完成元件的設計後，開啟在「net.macdidi.atk」套件下的「ItemActivity」類別，找到「processRecord」函式，加入啟動播放程式碼：

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    // 錄音與播放
    fun processRecord() {
        val recordFile = getRecFileName("R", ".mp3")

        if (recordFile.exists()) {
            val d = AlertDialog.Builder(this)

            d.setTitle(R.string.title_record)
                .setCancelable(false)
            d.setPositiveButton(R.string.record_play,
                DialogInterface.OnClickListener { dialog, which ->
                    // 啟動播放元件
                    val playIntent : Intent = Intent(
                        this, PlayActivity::class.java)
                    playIntent.putExtra("fileName",
                        recordFile.getAbsolutePath());
                    startActivity(playIntent);
                })
        }

        ...
    }
    else {
        ...
    }
}

```

```
}  
  
...  
  
}
```

完成這一章所有的工作了，錄音與播放的功能建議在實體的裝置上測試，試試看加入的功能是不是都可以正確的運作。

相關的檔案都可以在[GitHub](#)瀏覽與下載：

GitHub


<https://github.com/macdid5/Android-Tutorial-Kotlin>

後續 >> [Android Tutorial using Kotlin 第四堂（2）設計地圖應用程式 – Google Maps Android API](#)

Does Clearly work fine?



Shortcuts: **SHIFT+CTRL+C** to Toggle, **ESC** to Close.

 Give us feedback

Build upon ❤️ with Clearly