

Android Tutorial using Kotlin 第三堂

（2）儲存與讀取應用程式資訊

[Android Tutorial using Kotlin 第三堂（1）為ListView元件建立自定畫面](#) << 前情

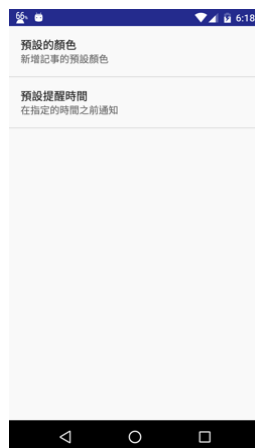
應用程式在運作的時候，可能需要儲存與讀取一些簡單的資料，另外也需要提供一個畫面讓使用者設定一些應用程式需要的資訊。例如一個遊戲應用程式，需要在使用者完成一個關卡後，儲存分數或花費的時間。還有提供遊戲效果的設定畫面，讓使用者設定是否需要背景音樂、音效和震動的效果。應用程式可以讀取這些設定的資料，用來控制遊戲進行的時候，是否需要執行這些效果。

Android 系統提供一種「Preference」的架構，它可以在應用程式中儲存一些「名稱＝值」這類簡單的資料，這些資料可以用應用程式的狀態，或是儲存使用者執行的設定。這些資料在應用程式中執行儲存與讀取的工作都非常容易，如果有這類的需求，用它來處理是最方便的。

這一章介紹Android提供的設計元件「PreferenceActivity」與「PreferenceFragment」，使用它提供的設計方式，可以簡化設定元件。完成這一章的工作以後，使用者可以在應用程式的主畫面選擇設定功能項目：



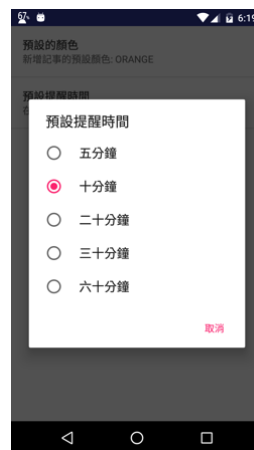
啟動設定元件以後，使用者可以選擇設定預設顏色或提醒時間：



使用者選擇設定預設顏色的項目，應用程式啟動之前已經設計好的選擇顏色元件：



使用者選擇設定預設提醒時間的項目，應用程式開啟選擇時間的對話框：



10-1 系統Preference設計架構介紹

一般的應用程式，通常需要提供使用者執行設定的功能，這樣可以讓應用程式比較方便一些，不會是固定的畫面或操作的行為。
Android 為應用程式提供一個專門用來設計應用程式設定功能的元件，它是一個比較特別的Activity 元件，規劃與設計的作法一樣。要使用這種比較方便而且容易的設定元件，需要設計這些設定檔與元件：

- 設定畫面配置檔 – 這種設定元件有它特有的畫面配置檔，也是一個XML 格式的檔案，放在專案的「res/xml」目錄下，使用特別的標籤設計畫面。
- PreferenceActivity元件 – 設定元件專用的Activity元件，讓你的元件類別繼承自這個類別。

設定元件專用的畫面配置檔放在專案的「res/xml」目錄下，這個設定檔的最外層使用「PreferenceScreen」標籤，根據應用程式要的設定資料，在這裡標籤中加入這些需要的設定元件標籤：

- EditTextPreference – 使用對話框讓使用者輸入文字資料。
- CheckBoxPreference – 勾選元件，儲存boolean 資料。
- SwitchPreference – 在Android 4.0（API level 14）加入，提供開關式的元件，儲存boolean 資料。
- ListPreference – 使用對話框讓使用者在列表中選擇一個項目，儲存字串資料。
- MultiSelectListPreference – 在Android 3.0（API level 11）加入，使用對話框讓使用者在列表中選擇多個項目，儲存Set<String> 資料。
- RingtonePreference – 開啟系統內建選擇來電鈴聲的對話框讓使用者選擇，儲存文字資料。

- PreferenceCategory – 用來執行設定資料的分組。
- Preference – 啟動其它元件執行設定的工作。

10-2 設計設定元件使用的畫面配置檔

你的應用程式可以依照需要保存的資訊，設計好設定畫面讓使用者使用，設定畫面使用一組特定的元件標籤。這個記事應用除了說明一個一般的元件外，也會說明啟動元件的作法，其它的設定元件就會比較簡單一些，你可以依照自己的需求加入與刪除設定元件。

設定元件需要使用一些文字資源，開啟「res/values/strings.xml」，加入下列的文字資源：

```
<string name="default_color">預設的顏色</string>
<string name="default_color_summary">新增記事的預設顏色</string>

<string name="default_notify">預設提醒時間</string>
<string name="default_notify_summary">在指定的時間之前通知</string>

<string-array name="notify_minutes_array">
    <item>五分鐘</item>
    <item>十分鐘</item>
    <item>二十分鐘</item>
    <item>三十分鐘</item>
    <item>六十分鐘</item>
</string-array>

<string-array name="notify_minutes_value_array">
    <item>5</item>
    <item>10</item>
    <item>20</item>
    <item>30</item>
    <item>60</item>
</string-array>
```

設定畫面配置檔必須放在專案的「res/xml」目錄下，如果專案中還沒有這個目錄，在「res」目錄按滑鼠右鍵，選擇「New -> Android resource file」，在File name輸入「mypreference」，Resource type選擇「XML」，最後選擇「OK」。把「mypreference.xml」修改為下列的內容：

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android" >
    <!-- 預設顏色 -->
    <!-- android:key 設定資料名稱 -->
    <!-- android:title 設定畫面上顯示的標題 -->
    <!-- android:summary 設定畫面上顯示的說明 -->
    <Preference
        android:key="DEFAULT_COLOR"
        android:title="@string/default_color"
```

```

        android:summary="@string/default_color_summary">
    </Preference>

    <!-- 預設提醒時間 -->
    <!-- android:entries 設定畫面顯示選項內容的陣列資源 -->
    <!-- android:entryValues 設定儲存選項資料的陣列資源 -->
    <!-- android:defaultValue 設定選項預設項目編號 -->
    <ListPreference
        android:key="NOTIFY_MINUTES"
        android:title="@string/default_notify"
        android:summary="@string/default_notify_summary"
        android:entries="@array/notify_minutes_array"
        android:entryValues="@array/notify_minutes_value_array"
        android:defaultValue="5" />
</PreferenceScreen>

```

這個檔案在設定畫面中提供兩個設定用的項目，一個用來設定新增記事的預設顏色，還有設定提醒的預設時間，這一章會先介紹顏色的設定值，提醒的預設時間在後面才會用到。

10-3 設計設定元件

設定元件是一個比較特殊的Activity元件，它是繼承自「PreferenceActivity」的子類別，必須搭配PreferenceFragment元件使用。要先建立需要的PreferenceFragment元件，在「net.macdidi.atk」目錄按滑鼠右鍵，選擇「New -> Fragment -> Fragment(Blank)」，在Fragment Name輸入「PrefFragment」，取消勾選「Create Layout XML」、「Include fragment fact methods?」與「Include interface callbacks?」選項，最後選擇「OK」。把PrefFragment.kt改為下列的內容：

```

package net.macdidi.atk

import android.content.SharedPreferences
import android.os.Bundle
import android.preference.Preference
import android.preference.PreferenceFragment
import android.preference.PreferenceManager

class PrefFragment : PreferenceFragment(),
    SharedPreferences.OnSharedPreferenceChangeListener {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        // 指定使用的設定畫面配置資源
        addPreferencesFromResource(R.xml.mypreference)
    }

}

```

接下來設計PreferenceActivity元件，在「net.macdidi.atk」目錄按滑鼠右鍵，選擇「New -> Activity -> Empty Activity」，在Name輸入「PrefActivity」，取消勾選「Generate Layout File」選項，最後選擇「OK」。把PrefActivity.kt改為下列的內容：

```
package net.macdidi.atk

import android.os.Bundle
import android.preference.PreferenceActivity

// 繼承自PreferenceActivity類別
class PrefActivity : PreferenceActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        // 載入PrefFragment元件
        fragmentManager.beginTransaction().replace(
            android.R.id.content, PrefFragment()).commit()
    }

    // Android 4.4、API level 19加入的函式
    // API level 19以後的版本必須覆寫這個函式，
    // 檢查使用的Fragment是否有效
    override fun isValidFragment(fragmentName: String): Boolean {
        return PrefFragment::class.java.name == fragmentName
    }
}
```

開啟「res/menu/main_menu.xml」檔案，為功能表新增啟動設定元件的項目：

```
<!-- 設定 -->
<item
    android:title="Setting"
    app:showAsAction="always"
    android:icon="@android:drawable/ic_menu_preferences"
    android:onClick="clickPreferences" />
```

開啟「net.macdidi.atk」套件下的「MainActivity」類別，加入啟動設定元件的函式宣告：

```
package net.macdidi.atk

...

class MainActivity : AppCompatActivity() {
```

```

...

// 設定
fun clickPreferences(item: MenuItem) {
    // 啟動設定元件
    startActivity(Intent(this, PrefActivity::class.java))
}

}

```

完成這個階段的工作以後，可以先執行應用程式，選擇主功能表上的設定圖示，看看可不可以正確的啟動設定元件。

10-4 在設計設定元件中啟動其它元件

選擇記事分類顏色的元件在之前已經設計好了，所以需要讓使用者選擇預設顏色最好的作法，應該是使用原來設計好的元件微修改就可以了。選擇記事分類顏色元件在這裡需讓設定元件使用，所以在「AndroidManifest.xml」檔案修改它的設定：

```

<!-- 選擇顏色 -->
<activity
    android:name=".ColorActivity"
    android:theme="@android:style/Theme.Dialog">
    <!-- 加入設定元件啟動用的Action名稱 -->
    <intent-filter>
        <action android:name="net.macdidi.atk.CHOOSE_COLOR"/>
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>

```

開啟「res/xml/mypreference.xml」，加入啟動選擇顏色元件的設定：

```

<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android" >
    <Preference
        android:key="DEFAULT_COLOR"
        android:title="@string/default_color"
        android:summary="@string/default_color_summary">
        <!-- 啟動選擇顏色元件 -->
        <intent
            android:action="net.macdidi.atk.CHOOSE_COLOR"
            android:targetPackage="net.macdidi.atk"
            android:targetClass="net.macdidi.atk.ColorActivity"/>
        </Preference>
    ...
</PreferenceScreen>

```

開啟在「net.macdidi.atk」套件下的「ColorActivity」類別，找到「ColorListener」監聽類別，依照下列的說明修改原來的程

```
package net.macdidi.atk

...

class ColorActivity : Activity() {

    ...

    private inner class ColorListener : View.OnClickListener {

        override fun onClick(view: View) {
            val action = this@ColorActivity.intent.action

            // 經由設定元件啟動
            if (action != null && action == "net.macdidi.atk.CHOOSE_COLOR") {
                // 建立SharedPreferences物件
                val editor = PreferenceManager.getDefaultSharedPreferences(
                    this@ColorActivity).edit()
                // 儲存預設顏色
                editor.putInt("DEFAULT_COLOR", view.id)
                // 寫入設定值
                editor.commit()
            }
            // 經由新增或修改記事的元件啟動
            else {
                val result = intent
                result.putExtra("colorId", view.id)
                setResult(Activity.RESULT_OK, result)
            }

            finish()
        }
    }
}
```

為了接下來設計讀取顏色設定的功能，開啟在「net.macdidi.atk」套件下的「ItemActivity」類別，找到「getColors」函式，依照下列的說明修改這個函式的宣告：

```
package net.macdidi.atk

...
```

```

class ItemActivity : AppCompatActivity() {

    ...

    // 改為可以使用類別名稱呼叫這個函式
    companion object {
        // 轉換顏色值為Colors型態
        public fun getColors(color: Int): Colors {

            ...

        }
    }

}

```

使用者設定預設的顏色以後，通常會希望在設定元件的畫面看到設定的結果，所以回到「PrefFragment」類別，依照下列的輸入需要的程式碼：

```

package net.macdidi.atk

import android.content.SharedPreferences
import android.os.Bundle
import android.preference.Preference
import android.preference.PreferenceFragment
import android.preference.PreferenceManager

class PrefFragment : PreferenceFragment(),
    SharedPreferences.OnSharedPreferenceChangeListener {

    private val defaultColor: Preference
        by lazy { findPreference("DEFAULT_COLOR") }
    private val sharedPreferences: SharedPreferences
        by lazy { PreferenceManager.getDefaultSharedPreferences(this.activity) }

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        // 指定使用的設定畫面配置資源
        addPreferencesFromResource(R.xml.mypreference)
    }

    override fun onSharedPreferenceChanged(sharedPreference: SharedPreferences?,
        key: String?) {

        if (key == "DEFAULT_COLOR") {
            setColorSummary()
        }
    }

    override fun onResume() {
        super.onResume()
    }

```



```

        setColorSummary()
    }

    private fun setColorSummary() {
        // 讀取設定的預設顏色
        val color = sharedPreferences.getInt("DEFAULT_COLOR", -1)

        if (color != -1) {
            // 設定顏色說明
            defaultColor.summary = getString(R.string.default_color_summary) +
                ": " + ItemActivity.getColors(color)
        }
    }
}

```

完成這個階段的工作以後，執行應用程式，選擇設定預設顏色的項目，看看可不可以正確的啟動選擇顏色元件。選擇顏色並設定元件以後，預設顏色設定項目的說明也會顯示設定顏色的名稱。

10-5 使用儲存的設定值

完成設定元件與相關的設計後，使用者在新增的記事資料的時候，如果沒有為它設定顏色，就應該採用已經在設定元件設定好的顏色。開啟在「net.macdidi.atk」套件下的「ItemActivity」類別，找到「onSubmit」函式，依照下列的說明執行需要的修

```

package net.macdidi.atk

...

class ItemActivity : AppCompatActivity() {

    ...

    fun onSubmit(view: View) {
        if (view.id == R.id.ok_item) {
            val titleText = title_text.text.toString()
            val contentText = content_text.text.toString()

            item.title = titleText
            item.content = contentText

            if (intent.action == "net.macdidi.atk.EDIT_ITEM") {
                item.lastModify = Date().time
            }
            else {
                item.datetime = Date().time

                // 建立SharedPreferences物件
                val sharedPreferences = PreferenceManager.getDefaultSharedPreferences(th

```

```

        // 讀取設定的預設顏色
        val color = sharedPreferences.getInt("DEFAULT_COLOR", -1)
        item.color = getColors(color)
    }

    intent.putExtra("net.macdidi.atk.Item", item)
    setResult(Activity.RESULT_OK, intent)
} else {
    setResult(Activity.RESULT_CANCELED, intent)
}

// 結束
finish()
}

...
}

```

完成這一章所有的工作了，執行應用程式，新增一個記事資料，看看會不會設定為預設的顏色。



相關的檔案都可以在GitHub瀏覽與下載：

GitHub


<https://github.com/macdidi5/Android-Tutorial-Kotlin>

後續 >> Android Tutorial using Kotlin 第三堂（3）使用Android內建的SQLite資料庫

Does Clearly work fine?



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

 Give us feedback

Build upon ❤️ with Clearly

