Chapter

9

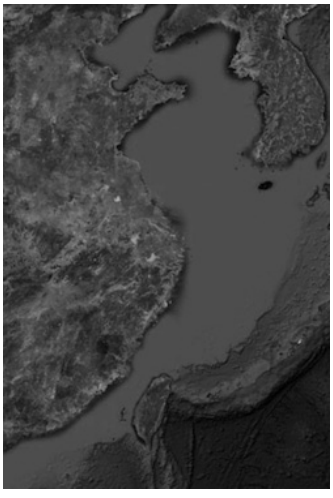
Google 地圖

9-1 Google 地圖功能簡介與建立方式

欲建立帶有 Google 地圖功能的 Android 應用程式，必須使用 Google Maps Android API（以下簡稱為 Maps API），目前版本為第 2 版（全名為 Google Maps Android API v2）。透過 Maps API 可以存取 Google 地圖伺服器上的資料用以呈現在 Android 裝置上；除此之外，Maps API 還可以協助開發者在 Google 地圖上加標記（markers）、繪製線條與多邊形（polygons），以及加上圖層（overlays）以達到更豐富的圖資呈現；甚至使用者可以改變地圖呈現方式（例如：地圖傾斜度、切換交通圖與衛星圖等），如圖 9-1，並可與地圖做多樣的互動。

加標記

多邊形



衛星圖

圖 9-1

想要在 Android 應用程式加上 Google 地圖功能，可以按照下列步驟；而各步驟的詳細說明則可繼續參看之後接續的各節。

1. 產生數位憑證指紋
2. 申請 Google Maps API Key（以下簡稱 API 金鑰）
3. 安裝 Google Play services SDK
4. 建立基本 Google 地圖

9-2 產生數位憑證指紋

Android 應用程式如果需要呼叫 Maps API，就必須使用 API 金鑰[1]。欲取得 API 金鑰的必須提供數位憑證指紋（application's digital certificate fingerprint）與 Android 應用程式套件名稱。以下先說明產生除錯憑證[2]指紋的步驟：

STEP 1 透過 Eclipse 主選單「Windows」>「Preferences」>「Android」>「Build」，右邊窗格「Default debug keystore」可以找到預設 debug.keystore 檔案[3]位置，如圖 9-2，而 debug.keystore 檔案內儲存的就是除錯憑證。

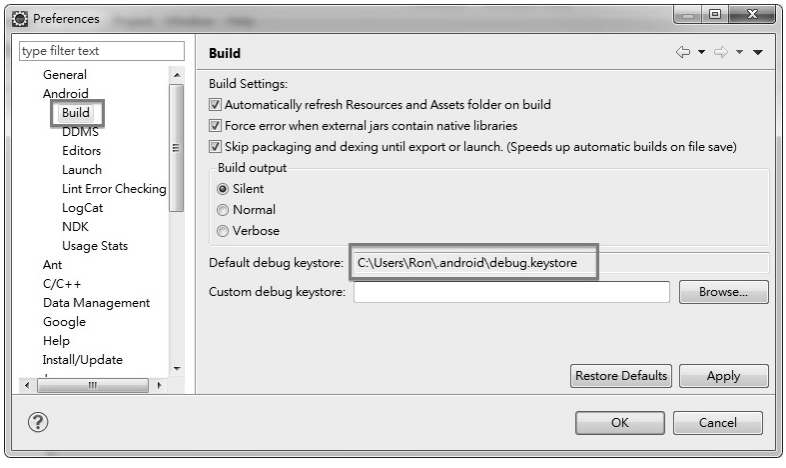


圖 9-2

---

1 Maps API 第 2 版使用新系統來管理 API 金鑰，第 1 版的 API Key（在 MapView 上設定）則無法使用第 2 版 API 的功能。目前您的應用程式使用第 2 版的金鑰來存取圖資，沒有使用者人數上限；換句話說，無論多少使用者使用您的應用程式並透過第 2 版的金鑰來存取 Google 地圖伺服器上的資源，都沒有人數上限制，這點與第 1 版相同。

2 應用程式的數位憑證有 2 種，開發階段使用除錯憑證（debug certificate）即可；如果要將應用程式發佈到 Play 商店，就必須使用發佈憑證（release certificate）。因為目前還在開發階段，所以本章說明以除錯憑證為主；發佈憑證則在第 14 章說明。

3 **注意**：必須對任一 Android 應用程式（該應用程式不須具備 Google 地圖）執行一次，Android SDK 工具才會自動產生 debug.keystore 檔案並放在預設位置。debug.keystore 是金鑰庫，其內儲存著金鑰。

STEP 2 將 JDK 的 keytool 工具（keytool.exe）之路徑加入 path 環境變數[4]，如圖 9-3。

圖 9-3

STEP 3 請依照 debug.keystore 檔案（假設在 C:\Users\RON\.android 目錄）所在路徑下指令：

```
C:\Users\RON\.android>keytool -list -keystore debug.keystore
```

STEP 4 接著會要求輸入 keystore 密碼，預設為「android」，也可以不輸入直接按 Enter 按鈕，最後會產生 SHA1 憑證指紋如下所示，下個階段就會利用 SHA1 憑證指紋去申請 API 金鑰。

[4] Path 環境變數設定說明如下：

- Windows 7 系統：控制台 > 系統 > 進階系統設定 > 進階 > 視窗下半部系統變數找到 Path 後按下「編輯」按鈕 > 在「變數值」欄位的最後輸入「;」（前後不可有任何空白字元），接著將「keytool.exe」所在路徑－例如「C:\Program Files\Java\jdk1.7.0_05\bin」輸入。
- Windows XP 系統：與 Windows 7 系統設定幾乎完全相同，差別在於 Windows XP 少了「進階系統設定」選項。

```
C:\Users\Ron\.android>keytool -list -keystore debug.keystore
輸入金鑰儲存庫密碼: （預設為 android）
金鑰儲存庫類型: JKS
金鑰儲存庫提供者: SUN
您的金鑰儲存庫包含 1 項目
androiddebugkey, 2012/6/19, PrivateKeyEntry,
憑證指紋 (SHA1): 93:99:DF:BA:6A:13:71:ED:03:75:AD:96:E9:46:5A:D0:31:C9:43:E4
```

9-3 申請 API 金鑰

要申請 API 金鑰，必須透過 Google APIs Console 管理介面，申請步驟如下：

STEP 1 開啟瀏覽器（建議使用 Google Chrome 瀏覽器），網址列輸入「https://code.google.com/apis/console/」，接下來輸入 Gmail 帳號密碼，如圖 9-4。

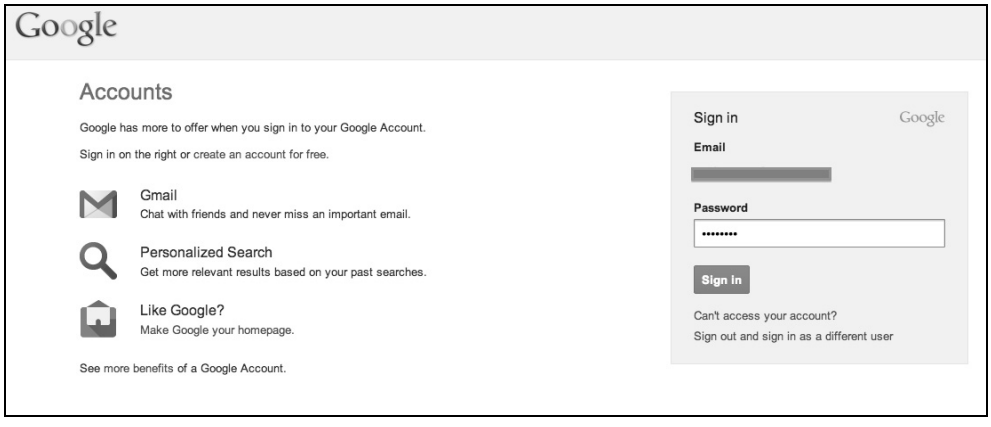


圖 9-4

STEP 2 接著會跳出如圖 9-5 畫面，按下「Create project」按鈕繼續。

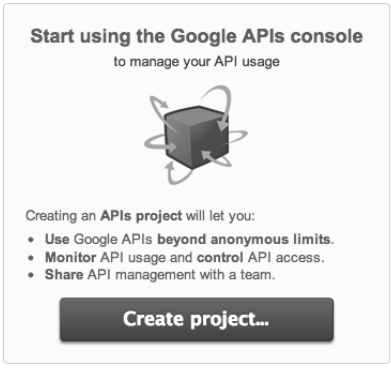


圖 9-5

STEP 3 點擊左邊導覽列的「Services」，在右邊搜尋到「Google Maps Android API v2」，如圖 9-6；點擊箭頭所指之處會跳出服務條款，接受條款後即可啟動該服務。最後點擊「API Access」，準備申請 API 金鑰。

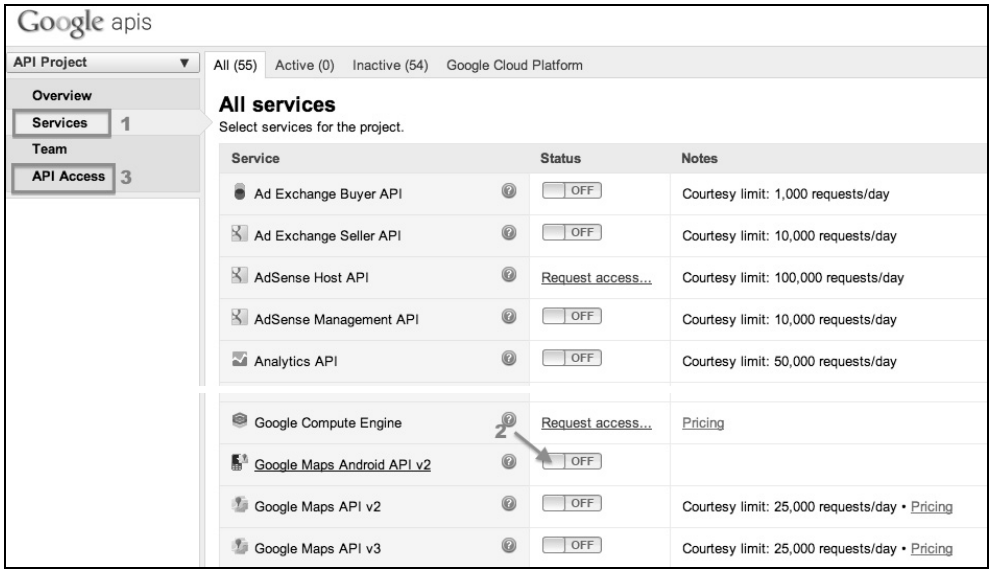


圖 9-6

STEP 4 點擊導覽列「API Access」選項，並在右邊按下「Create new Android key」按鈕，如圖 9-7。

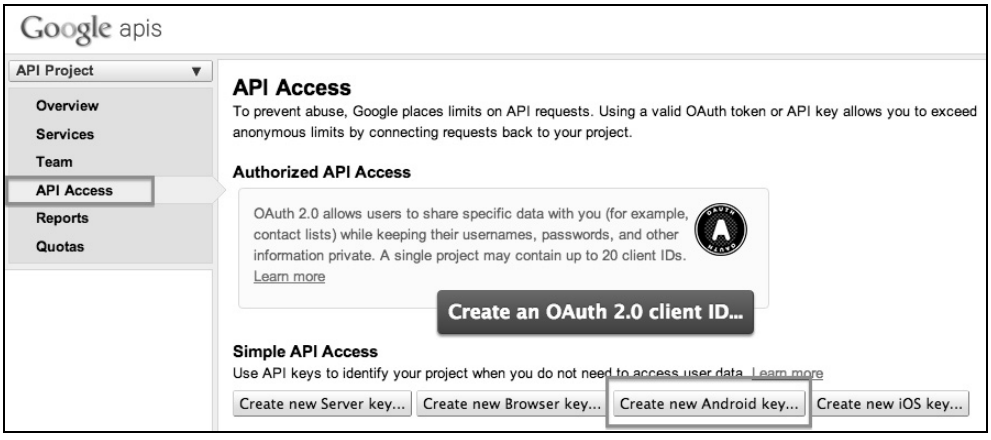


圖 9-7

STEP 5 申請 API 金鑰需要「憑證指紋」與「應用程式套件名稱」。請依照圖 9-8 所示，在文字輸入方塊中先貼上憑證指紋，加上分號，再貼上套件名稱，格式為「憑證指紋;套件名稱」(分號作為分隔)，然後按下「Create」按鈕。

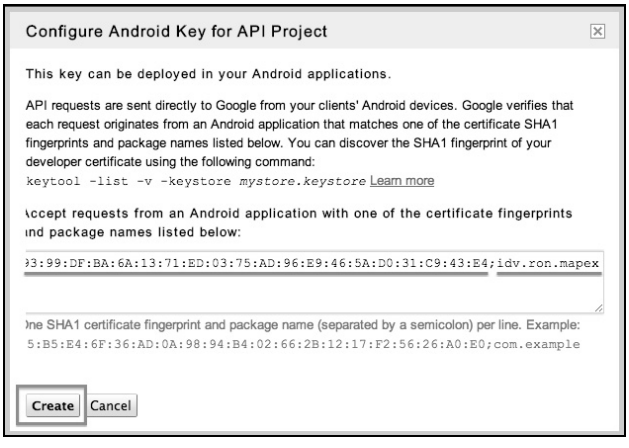


圖 9-8

STEP 6 最後會產生 API 金鑰，如圖 9-9。之後建立 Android 專案時，套件名稱要填寫申請 API 金鑰時所輸入的套件名稱；另外 manifest 檔案也必須設定 API 金鑰，在下一節將會詳細說明。

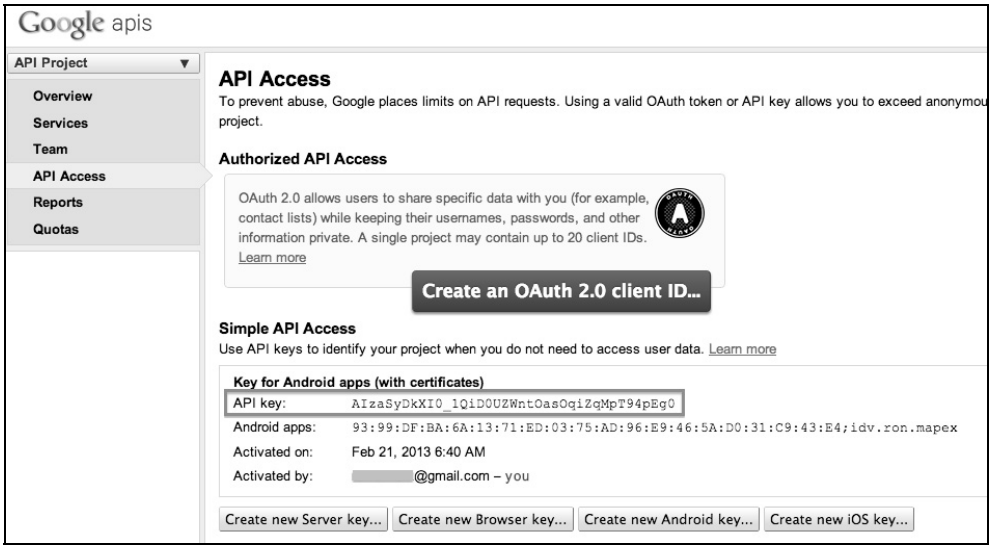


圖 9-9

9-4 安裝 Google Play services SDK

Maps API 現在已經成為 Google Play services SDK 一部分，所以要使用 Google 地圖相關功能，必須先安裝 Google Play services SDK。安裝步驟為：

STEP 1 按下 Eclipse 工具列的「Android SDK Manager」按鈕，如圖 9-10。

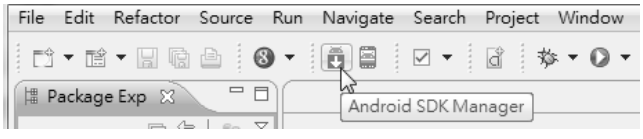


圖 9-10

STEP 2 勾選「Google Play services」，如圖 9-11，按下右下角「Install packages」即可開始安裝。

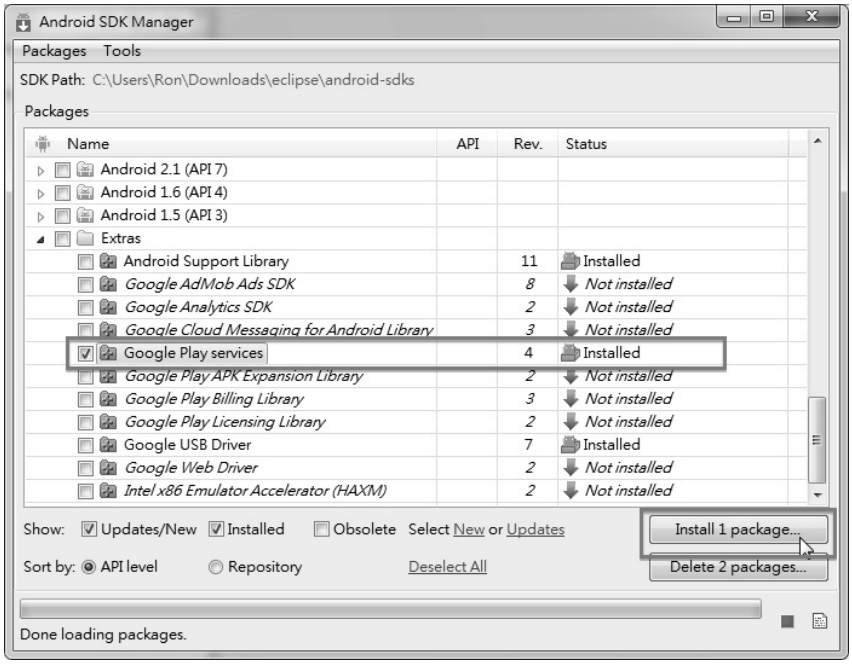


圖 9-11

Google 地圖應用程式需要使用到 Google Play services 函式庫，所以先匯入 google-play-services_lib 專案，當作其他地圖應用程式的函式庫（library project）。匯入步驟為：

STEP 1 按下 Eclipse 主選單「File」>「Import」>「Android」>「Existing Android Code Into Workspace」然後按下「Next」按鈕，如圖 9-12。

圖 9-12

STEP 2 按下「Browse」按鈕然後找尋 google-play-services_lib 目錄，如圖 9-12，路徑在「<android-sdk- 目錄 >/extras/google/google_play_services/libproject/google-play-services_lib」（注意：必須先安裝好前述 Google Play services SDK）。

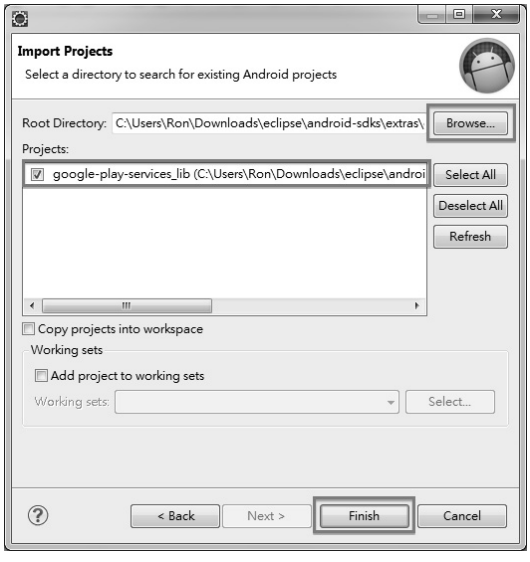


圖 9-13

STEP 3 匯入成功後可以在 Package Explorer 看到 google-play-services_lib 專案，如圖 9-14。

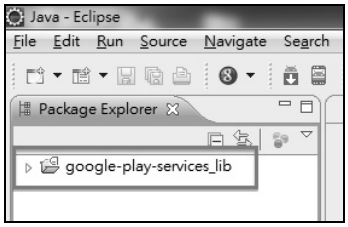


圖 9-14

9-5 建立基本 Google 地圖

想要建立具備基本 Google 地圖的 Android 應用程式，必須先建立 Android 專案，並且在 manifest 檔案內作相關設定，最後建立 layout 檔案與 Activity，可參看下面範例說明。要注意 Google 地圖功能必須實機測試，模擬器測試會失敗，因為模擬器無法更新 Google Play services。

範例 MapEx/BasicMapActivity

圖 9-15

範例說明：

- 建立 Android 專案：套件名稱必須與前述申請 API 金鑰所用套件名稱相同，如圖 9-16，否則無法顯示地圖。

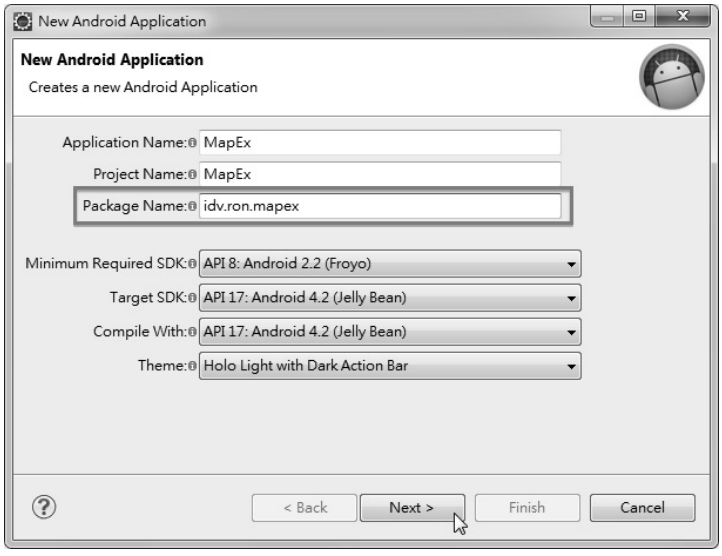


圖 9-16

引用 google-play-services_lib 專案：請先依照前述匯入 google-play-services_lib 專案，然後對著 Android 專案按滑鼠右鍵選「Properties」> 點擊「Android」選項 > 按下「Add」按鈕 > 選擇「google-play-services_lib」專案[5] > 按下「OK」按鈕，如圖 9-17。但請特別注意，依照筆者經驗，建立的 Android 專案與被引用的 google-play-services_lib 專案必須放在同一個硬碟分割區（例如都是在 C 分割區），否則會發生尋找不到 google-play-services_lib 專案而導致執行錯誤！

[5] 如果無法顯示 google-play-services_lib 專案，就必須將 google-play-services_lib 專案（注意：不是您建立的 Google 地圖專案）設定成「Is Library」，這代表要將 google-play-services_lib 專案設定成函式庫供您的 Google 地圖專案使用。設定方式為：對著 google-play-services_lib 專案按滑鼠右鍵選「Properties」> 點擊「Android」選項 > 勾選「Is Library」。

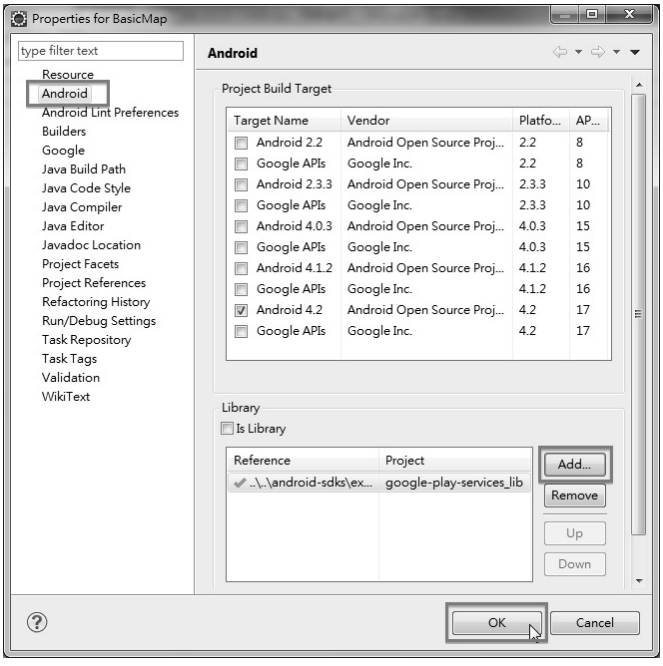


圖 9-17

- 設定 manifest 檔案：各個設定請參看下列註解部分。

```
<!-- 允許應用程式可以存取 Google Maps 伺服器，其中 「idv.ron.mapex」是套件名稱-->
<permission
    android:name="idv.ron.mapex.permission.MAPS_RECEIVE"
    android:protectionLevel="signature" />
<uses-permission android:name="idv.ron.mapex.permission.MAPS_RECEIVE" />

<!-- 允許應用程式透過 internet 下載地圖資訊 -->
<uses-permission android:name="android.permission.INTERNET" />

<!-- 允許應用程式存取 Google 所提供 web 型式的服務 -->
<uses-permission
android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />

<!-- 允許應用程式將地圖資訊暫存到 Android 裝置的外部儲存體 -->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

<!-- 允許應用程式透過 WiFi 或行動網路來定位 -->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

<!-- 允許應用程式透過 GPS 來定位 -->
```

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

<!-- 因為 Maps API 第 2 版要使用到 OpenGL ES 第 2 版功能，所以必須增加下列設定，以確保使用
者的裝置支援 -->
<uses-feature
    android:glEsVersion="0x00020000"
    android:required="true" />

<application
   …
    <!-- android:value 屬性要輸入申請的 API 金鑰 -->
    <meta-data
        android:name="com.google.android.maps.v2.API_KEY"
        android:value="AIzaSyDkXI0_1QiD0UZWntOasOqiZqMpT94pEg0" />
   …
</application>
```

- 建立帶有 Google 地圖的 layout 檔案：加上<fragment>標籤，如果 class 屬性是 SupportMapFragment，必須搭配 support library 的 FragmentActivity，可以支援舊版本的 Android 系統。如果改用 MapFragment，就必須搭配 Activity，僅支援 API 12（Android 3.1）以後的 Android 系統。

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/map"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    class="com.google.android.gms.maps.SupportMapFragment"/>
```

- 建立 Activity 檔案：如果<fragment>標籤設定為 SupportMapFragment，就必須搭配 FragmentActivity。

```
public class BasicMapActivity extends FragmentActivity {

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

}
```

9-6 地圖種類與 UI 設定

地圖種類設定

在 Google 地圖上可以設定下列 4 種地圖種類，參看圖 9-18：

- 一般圖（normal）：典型的道路地圖，重要的人造設施或天然景觀如河流、湖泊的形狀會顯示；另外重要道路與重要設施的名稱也會以文字顯示。
- 混合圖（hybrid）：就是一般圖與衛星圖的混合體。
- 衛星圖（satellite）：衛星所拍攝的空照圖，但不會以文字顯示道路或設施的名稱。
- 地形圖（terrain）：顯示地形走勢，也會以文字顯示一些道路或設施名稱。

一般圖　　衛星圖　　地形圖

圖 9-18

要設定地圖種類可以呼叫 GoogleMap.setMapType(int) 方法並搭配 GoogleMap 的地圖種類常數來達成，例如：

```
GoogleMap map;
...
// 將地圖種類設定為混合圖
map.setMapType(GoogleMap.MAP_TYPE_HYBRID);
```

地圖 UI 設定

Google 地圖可以顯示交通資訊（traffic）與自己位置圖層[6]（my location layer），只要透過呼叫 GoogleMap.setMyLocationEnabled(boolean) 與 GoogleMap.setMyLocationEnabled(boolean)方法即可達成，例如：

```
GoogleMap map;
...
// 顯示交通資訊
map.setTrafficEnabled(true);
// 顯示自己位置
map.setMyLocationEnabled(true);
```

另外還有 UiSettings 儲存著地圖 UI（User Interface，使用者介面）設定與操作手勢設定，可以呼叫 getUiSettings()取得該物件，並呼叫 setter 方法加以設定：

```
GoogleMap map;
...
// 取得地圖 UI 設定物件
UiSettings uiSettings = map.getUiSettings();

// 顯示縮放按鈕
```

6 如果只有一張地圖，而且要在地圖上標記自己現在位置，大概不會有人直接將自己位置標在地圖上，原因很簡單，因為自己所在位置會不斷更新，一旦直接標在地圖上，之後要將其清除可能會損及地圖。最好的作法就是在地圖上加一層透明紙，而將自己位置標記在該透明紙上，這樣從上方往下看，即可看到地圖與自己位置的綜合資訊，而且之後移除或是修改該標記都不致於損及地圖，這張透明紙即可稱為圖層。Google 地圖的作法也是如此，在地圖上可以新增許多不同的圖層，例如自己位置圖層、標記圖層等以方便管理；如果使用者不想看到某一圖層資訊，只要移除該圖層即可。

```
uiSettings.setZoomControlsEnabled(true)
// 顯示指北針
uiSettings.setCompassEnabled(true)
// 顯示自己位置按鈕
uiSettings.setMyLocationButtonEnabled(true)

// 開啟地圖捲動手勢
uiSettings.setScrollGesturesEnabled(true)
// 開啟地圖縮放手勢
uiSettings.setZoomGesturesEnabled(true)
// 開啟地圖傾斜手勢
uiSettings.setTiltGesturesEnabled(true)
// 開啟地圖旋轉手勢
uiSettings.setRotateGesturesEnabled(true)
```

範例 MapEx/MapTypeUiSettingsActivity

圖 9-19

範例說明：

- 點擊右上角自己位置按鈕可以將地圖畫面移到自己現行位置。
- 點擊右下角縮放按鈕可以縮放地圖。
- 點擊左下角下拉選單可以改變地圖種類；勾選下方選項可以開啟/關閉個別 UI 顯示或操作功能。

MapEx/MapTypeUiSettingsActivity.java

```
public class MapTypeUiSettingsActivity extends FragmentActivity {
  private GoogleMap map; // 儲存著地圖資訊
  private UiSettings uiSettings; // 儲存著地圖 UI 設定

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.map_type_ui_settings_activity);
    initMap();
    setMyMapType();
  }

  // 初始化地圖
  private void initMap() {
    // 檢查 GoogleMap 物件是否存在
    if (map == null) {
      // 從 SupportMapFragment 取得 GoogleMap 物件
      map = ((SupportMapFragment) getSupportFragmentManager()
          .findFragmentById(R.id.fmMap)).getMap();
      if (map != null) {
        setUpMap();
      }
    }
  }

  // 完成地圖相關設定
  private void setUpMap() {
    // 顯示交通資訊
    map.setTrafficEnabled(true);
    // 顯示自己位置
    map.setMyLocationEnabled(true);

    // 取得地圖地圖 UI 設定物件
    uiSettings = map.getUiSettings();
  }

  // 設定地圖種類
  private void setMyMapType() {
    // 建立地圖種類下拉選單，讓使用者可以選取欲顯示的地圖種類
    Spinner spinner = (Spinner) findViewById(R.id.sp_mapType);
    ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(
        this, R.array.mapTypes, android.R.layout.simple_spinner_item);
    adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
```

```
    spinner.setAdapter(adapter);

    spinner.setOnItemSelectedListener(new OnItemSelectedListener() {

      @Override
      public void onItemSelected(AdapterView<?> parent, View view,
          int position, long id) {
        if (!isMapReady()) {
          return;
        }

        // 將地圖設定成使用者選定的種類
        String mapType = parent.getItemAtPosition(position).toString();
        if (mapType.equals(getString(R.string.normal))) {
          map.setMapType(GoogleMap.MAP_TYPE_NORMAL);
        } else if (mapType.equals(getString(R.string.hybrid))) {
          map.setMapType(GoogleMap.MAP_TYPE_HYBRID);
        } else if (mapType.equals(getString(R.string.satellite))) {
          map.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
        } else if (mapType.equals(getString(R.string.terrain))) {
          map.setMapType(GoogleMap.MAP_TYPE_TERRAIN);
        } else {
          Log.i("MapTypeError", mapType + "設定錯誤！");
        }
      }

      @Override
      public void onNothingSelected(AdapterView<?> parent) {
        // Do nothing.
      }
    });
  }

  // 執行與地圖有關的方法前應該先呼叫此方法以檢查 GoogleMap 物件是否存在
  private boolean isMapReady() {
    if (map == null) {
      Toast.makeText(this, R.string.map_not_ready, Toast.LENGTH_SHORT)
          .show();
      return false;
    }
    return true;
  }

  // 點擊「交通資訊」CheckBox
  public void onTrafficClick(View view) {
```

```
  if (!isMapReady()) {
    return;
  }
  // 顯示/隱藏交通流量
  map.setTrafficEnabled(((CheckBox) view).isChecked());
}

// 點擊「縮放按鈕」CheckBox
public void onZoomControlsClick(View view) {
  if (!isMapReady()) {
    return;
  }
  // 顯示/隱藏縮放按鈕
  uiSettings.setZoomControlsEnabled(((CheckBox) view).isChecked());
}

// 點擊「指北針」CheckBox
public void onCompassClick(View view) {
  if (!isMapReady()) {
    return;
  }
  // 顯示/隱藏指北針
  uiSettings.setCompassEnabled(((CheckBox) view).isChecked());
}

// 點擊「自己位置按鈕」CheckBox
public void onMyLocationButtonClick(View view) {
  if (!isMapReady()) {
    return;
  }
  // 顯示/隱藏自己位置按鈕
  uiSettings.setMyLocationButtonEnabled(((CheckBox) view).isChecked());
}

// 點擊「自己位置圖層」CheckBox
public void onMyLocationLayerClick(View view) {
  if (!isMapReady()) {
    return;
  }
  // 顯示/隱藏自己位置圖層，如果未開啓則自己位置按鈕也無法顯示
  map.setMyLocationEnabled(((CheckBox) view).isChecked());
}

// 點擊「滑動手勢」CheckBox
public void onScrollGesturesClick(View view) {
```