



行動裝置程式設計

Unit 11 非同步任務 Asynchronous Task (AsyncTask)

蘇維宗 (Wei-Tsung Su)
suwt@au.edu.tw
564D



雲端運算與開放資料/API的興起

許多行動裝置的App都會透過Web API從網際網路存取資料

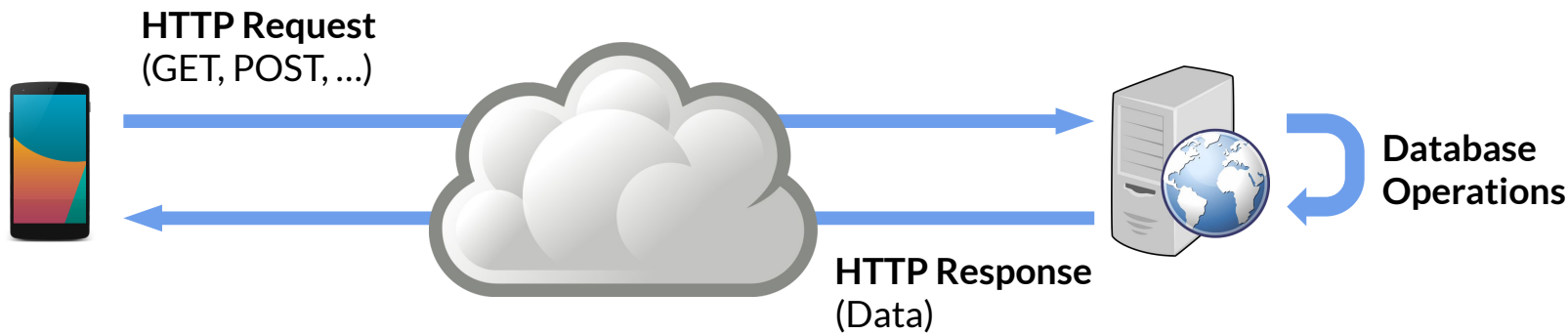




使用政府開放資料(Open Data)

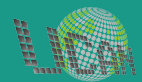
- **問題:** 想知道在新北市走哪條路比較安全?
- **想法:** 可以利用甚麼開放資料?
 - 新北市政府資料開放平台 (新北市路口監視器)
 - <http://data.ntpc.gov.tw/od/detail?oid=65C80080-8D55-407A-BDAE-A0CC1233E50F>
 - **欄位說明:**
 - Location: 地點;
 - Build_Case_Unit : 建置案單位
 - station: 單位名稱
 - bureau: 機關名稱
 - **資料格式:** JSON, XML, CSV
 - **更新頻率:** 每年

HTTP通訊協定



- HTTP Request GET方法(參數透過URL傳遞到伺服器)
 - <https://au-csie-echoservice.appspot.com/echo?msg=Hello>
- HTTP POST方法(參數依據指定格式嵌入在 HTTP封包中傳遞到伺服器)

HTTP Request / Response in Java





如何發出HTTP要求?

利用URLConnection發出HTTP要求

1. **// 準備URL與URLConnection物件**
2. URL url = new URL(" <http://xxx.xxx.xxx.xxx> ");
3. HttpURLConnection conn = (HttpURLConnection)url.openConnection();
4. **// 設定URLConnection的方法為GET**
5. conn.setRequestMethod("GET");
6. **// 發出Http Request**
7. conn.connect();





如何處理HTTP回應?

利用`URLConnection`取得HTTP回應

```
1. // 取得回應結果並判斷是否為HTTP_OK(即200)
2. int HttpStatus = conn.getResponseCode();
3. if(HttpStatus == HttpURLConnection.HTTP_OK) {
4.     InputStream input = conn.getInputStream();
5.     BufferedReader br = new BufferedReader(new InputStreamReader(input, "utf-8"));
6.     StringBuilder data = new StringBuilder();
7.     String line;
8.     while((line = br.readLine()) != null) {
9.         data.append(line + "\n");
10.    }
11.    br.close();
12.    //do something on data
13. }
```

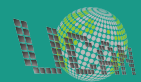


採用同步任務傳輸會造成問題

新版SDK會在執行時發生例外 (`android.os.NetworkOnMainThreadException`)



非同步任務(Asynchronous Task)





AsyncTask

AsyncTask可以在背景中執行非同步任務，並在執行過程與執行結束時將非同步任務的狀態回傳給**主執行緒(UI執行緒)**來更新介面並與使用者進行必要的互動。

例如，當App在背景下載多個檔案時，可以讓使用者在App的介面上知道目前檔案下載的進度。



AsyncTask (續)

AsyncTask類別有三個主要的方法

- `Result doInBackground(Params... param) {...}`
 - 背景執行緒。執行需要長時間的非同步任務。
- `void onProgressUpdate(Progress... values) {...}`
 - 主(UI)執行緒。可以在非同步任務執行過程中更新UI。
- `void onPostExecute(Result result) {...}`
 - 主(UI)執行緒。可以在非同步任務完成後更新UI。





doInBackground()

```
1. public class MyAsyncTask extends AsyncTask< Params, Progress, Result> {
2.     @Override
3.     protected Result doInBackground(Params... param) {
4.         // TODO Auto-generated method stub
5.         // 這裡的程式會在背景執行，所以不會影響主執行緒上 UI與使用者的互動
6.         // 可以這裡執行需要長時間執行的任務（如，發出HTTP要求）
7.     }
8.     ...
9. }
```





onPostExecute ()

```
1. public class MyAsyncTask extends AsyncTask< Params, Progress, Result> {
2.     ...
3.     @Override
4.     protected void onPostExecute(Result result) {
5.         // TODO Auto-generated method stub
6.         // 這裡的程式會在主執行緒執行 (所以可以用來更新UI上的資訊)
7.         // 當doInBackground() 的工作結束後才會被呼叫
8.     }
9. }
```



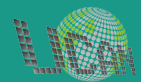


onProgressUpdate ()

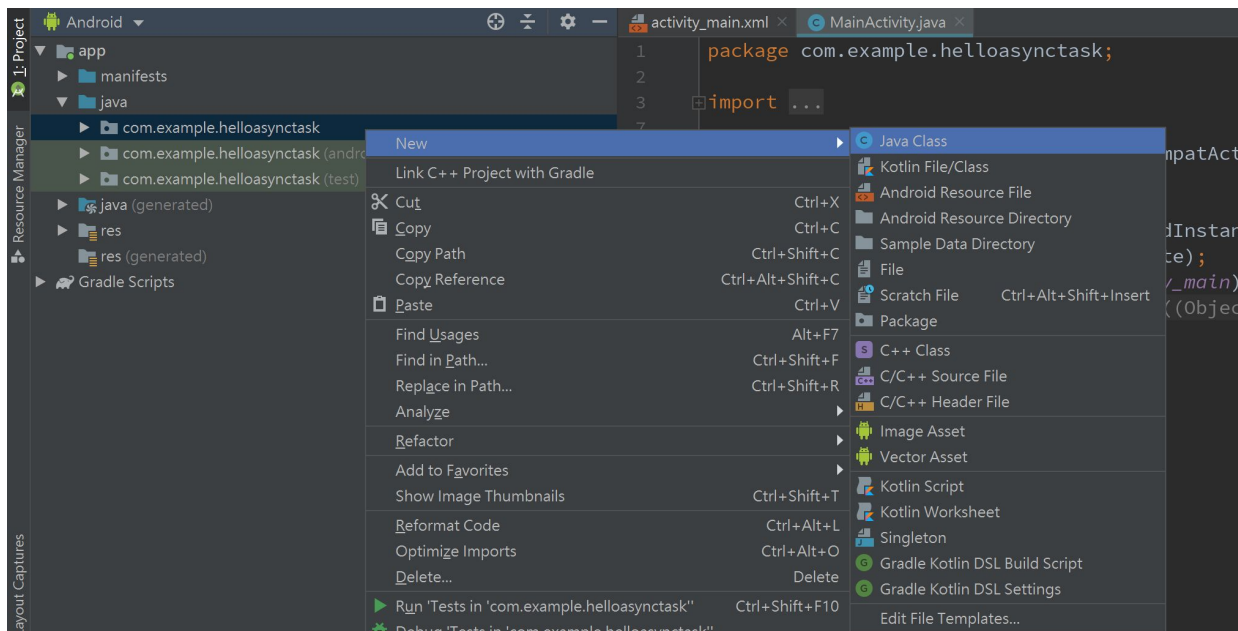
```
1.  public class MyAsyncTask extends AsyncTask< Params, Progress, Result> {
2.      ...
3.      @Override
4.      protected void onProgressUpdate(Progress... values) {
5.          // TODO Auto-generated method stub
6.          // 這裡的程式會在主執行緒執行 (所以可以用來更新UI上的資訊)
7.          // 當呼叫publishProgress() 時就會被呼叫
8.      }
9.      ...
10. }
```



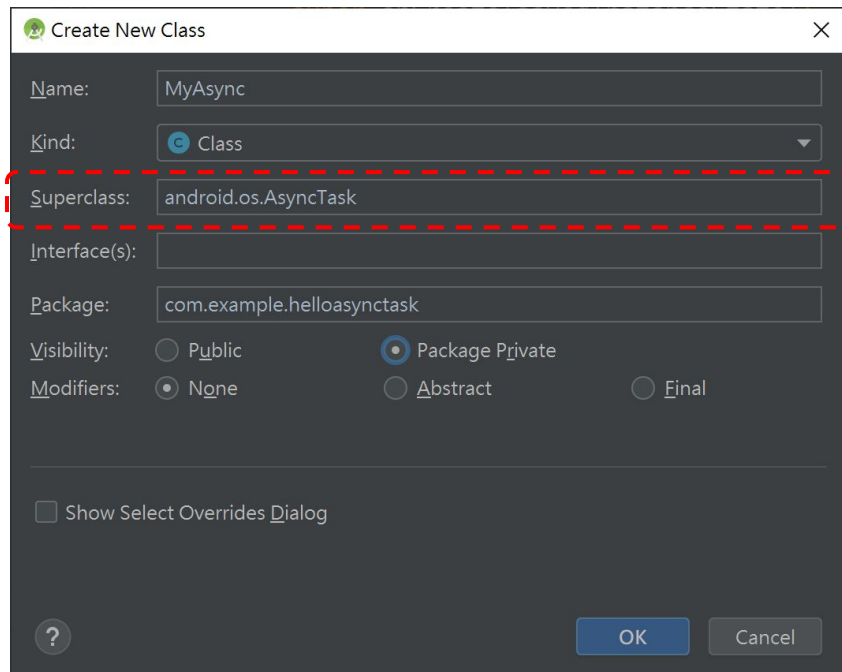
實作AsyncTask



新增繼承AsyncTask的Java類別



新增繼承AsyncTask的Java類別(續)



Create New Class

Name: MyAsync

Kind: Class

Superclass: android.os.AsyncTask

Interface(s):

Package: com.example.helloasynctask

Visibility: ☐ Public ☒ Package Private

Modifiers: ☒ None ☐ Abstract ☐ Final

☐ Show Select Overrides Dialog

OK Cancel



依據需要修改AsyncTask類別

```
1.  public class MyAsyncTask extends AsyncTask< Params, Progress, Result> {
2.      @Override
3.      protected Result doInBackground(Params... param) {
4.          // TODO Auto-generated method stub HTTP
5.      }
6.      @Override
7.      protected void onProgressUpdate(Progress... values) {
8.          // TODO Auto-generated method stub
9.      }
10.     @Override
11.     protected void onPostExecute(Result result) {
12.         // TODO Auto-generated method stub
13.     }
14. }
```



依據需要修改AsyncTask類別(續)

```
1.  public class MyAsyncTask extends AsyncTask< URL, Integer, String> {
2.      @Override
3.      protected String doInBackground(URL... urls) {
4.          // TODO Auto-generated method stub HTTP
5.      }
6.      @Override
7.      protected void onProgressUpdate( Integer... progress) {
8.          // TODO Auto-generated method stub
9.      }
10.     @Override
11.     protected void onPostExecute( String result) {
12.         // TODO Auto-generated method stub
13.     }
14. }
```





依據需要修改AsyncTask類別(續)

```
1. public class MyAsyncTask extends AsyncTask< URL, Integer, String> {  
2.     // 為了能在這裡可以更新主執行緒 (UI介面)  
3.     Context mContext;  
4.     MyAsyncTask(Context context) {  
5.         mContext = context;  
6.     }  
7.     ...  
8. }
```




在MainActivity中執行AsyncTask

```
1. public class MainActivity extends AppCompatActivity {  
2.     @Override  
3.     protected void onCreate(Bundle savedInstanceState) {  
4.         super.onCreate(savedInstanceState);  
5.         ...  
6.         URL url = new URL("http://xxx.xxx.xxx.xxx");  
7.         new MyAsyncTask(this).execute(url);  
8.     }
```



當需要將執行狀況回傳到主執行緒時...

```
1.  public class MyAsyncTask extends AsyncTask<URL, Integer, String> {
2.      @Override
3.      protected String doInBackground(URL... urls) {
4.          // TODO Auto-generated method stub
5.          ...
6.          publishProgress(progress);
7.          ...
8.          return result;
9.      }
10.     @Override
11.     protected void onProgressUpdate(Integer... progress) {
12.         // TODO Auto-generated method stub
13.     }
14. }
```





作業

<http://au-csie-echoservice.appspot.com/echo?msg=1>

<http://au-csie-echoservice.appspot.com/echo?msg=2>

<http://au-csie-echoservice.appspot.com/echo?msg=3>

...

請撰寫一個App可以利用AsyncTask發出上述HTTP要求(其中msg從1到100共100個HTTP要求)並可以在執行過程中在UI上顯示進度。

Q & A



Computer History Museum, Mt. View, CA