AsyncTask

陳健文

什麼是 AsyncTask？

一種可執行非同步任務的物件。

運用 Thread 執行需長時間運行的工作，且在運行過程中可以有更新畫面的機制。

有執行前、背景處理、更新處理與執行後等四個作業階段。

只有背景處理的方法是在另一個執行緒上運行，其它的方法運行在 UI 執行緒上。

使用流程

定義自定的 AsyncTask 類別

定義執行前方法(onPreExecute)、背景處理方法(doInBackground)、更新處理方法(onProgressUpdate)與執行後方法(onPostExecute)。

產生 AsyncTask 物件，傳入 Activity (Context)。

執行 AsyncTask 物件的 excute 方法

建立類別並定義建構方法與執行前方法

```
public class BobAT extends AsyncTask<Void, Integer, Void>{

    private Context atContext;

    public BobAT(Context context) { atContext = context.getApplicationContext(); }

    @Override
    protected void onPreExecute(){
        Toast.makeText(atContext, "AsyncTask is about to start.", Toast.LENGTH_SHORT).show();
    }
```

定義背景處理方法

```
protected Void doInBackground(Void... params){
    AudioAttributes at = new AudioAttributes.Builder()
            .setContentType(AudioAttributes.CONTENT_TYPE_MUSIC)
            .setUsage(AudioAttributes.USAGE_GAME)
             .build();

    SoundPool sp = new SoundPool.Builder()
            .setMaxStreams(1)
            .setAudioAttributes(at)
            .build();

    int snd = sp.load(atContext, R.raw.scifi018, 1);

    //delay for loading sound clip
    try{
        Thread.sleep(1000);
    }
    catch(InterruptedException e){
        e.printStackTrace();
    }

    for(int i=0; i<10; i++) {
        sp.play(snd, 1, 1, 0, 0, 1);
        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        publishProgress(i+1);
    }
    return null;
}
```

定義更新處理與執行後方法

```
@Override
protected void onProgressUpdate(Integer... i){
    super.onProgressUpdate(i);
    //Toast.makeText(atContext, String.valueOf(i[0]), Toast.LENGTH_SHORT).show();
    playTimes.setText(String.valueOf(i[0]));
}

@Override
protected void onPostExecute(Void i){
    super.onPostExecute(i);
    Toast.makeText(atContext, "AsyncTask is finished.", Toast.LENGTH_SHORT).show();
}
```

產生 AsyncTask 物件並將之啟動

```
BobAT bat = new BobAT(this);

bat.execute();
```

重點說明

public class BobAT extends AsyncTask<Void, Integer, Void>{}

<Void, Integer, Void> 是泛型的寫法，代表傳給 doInBackground 的參數型別為 Void，傳給 onProgressUpdate 的參數型別為 Integer，而傳給 onPostExecute 的參數型別為 Void。

protected Void doInBackground(Void... params){}

Void… params 代表傳進來的 params 參數可能是一個 Void 也可能有多個 Void，這種寫法稱為變動長度的參數列表 (variable-length argument list)。

在 doInBackground 中以 publishProgress 來傳遞參數

Void 是一個表示型別為基本型別 void 的類別，回傳值須為 null。

AsyncTask 的缺點

一個實例只能運行一次

當 AsyncTask 是內部類別時，在 Activity 或 Fragment 被銷毀後，會留下 AsyncTask 繼續執行。

當 AsyncTask 是外部類別或靜態的內部類別時，則所傳進來的 Context 參考並不可靠，使用前要檢查其是否為 null。

Demo