

第十八章

GCM

運作原理

GCM

- 行動裝置可能會在各種不同的網路下
- 若要做一個App的推播系統或app聊天軟體，要自行處理網路連線是一件非常困難的事情
- **Google Clouding Messaging (GCM)**相對於自己建立系統來說簡單許多，就可以完成推播的系統
- iOS上有APNS，也是類似的概念

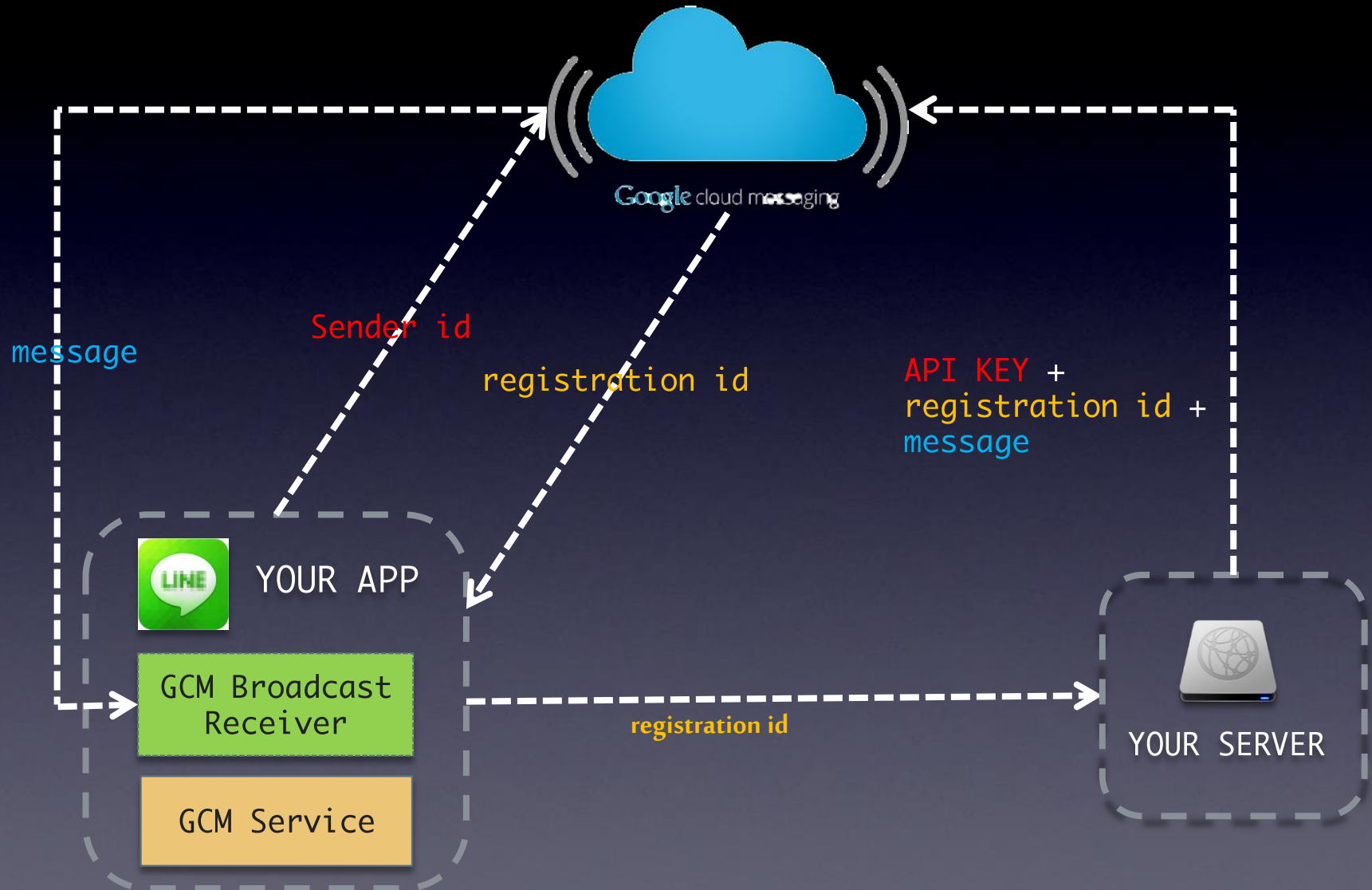
運作原理

- 首先，你需要準備自己的Server
- App需要照GCM的規定建立Broadcast Receiver和IntentService
- App第一次開啟時，藉由從Google申請GCM時給的SENDER ID，向GCM的Server進行註冊
- GCM的Server會回傳App一個註冊的id，稱為registration id

運作原理

- 接下來App將registration id傳入自己建立的Server的資料庫中
- 當Server要推播訊息時，將registration id，連同Google申請GCM時拿到的API KEY以及要傳送的訊息內容，一起傳到GCM Server
- GCM Server收到後，使用API KEY檢查是否合法若合法則使用收到的registration id找到裝置
- 找到裝置後將Server要送的訊息傳遞給裝置

運作原理

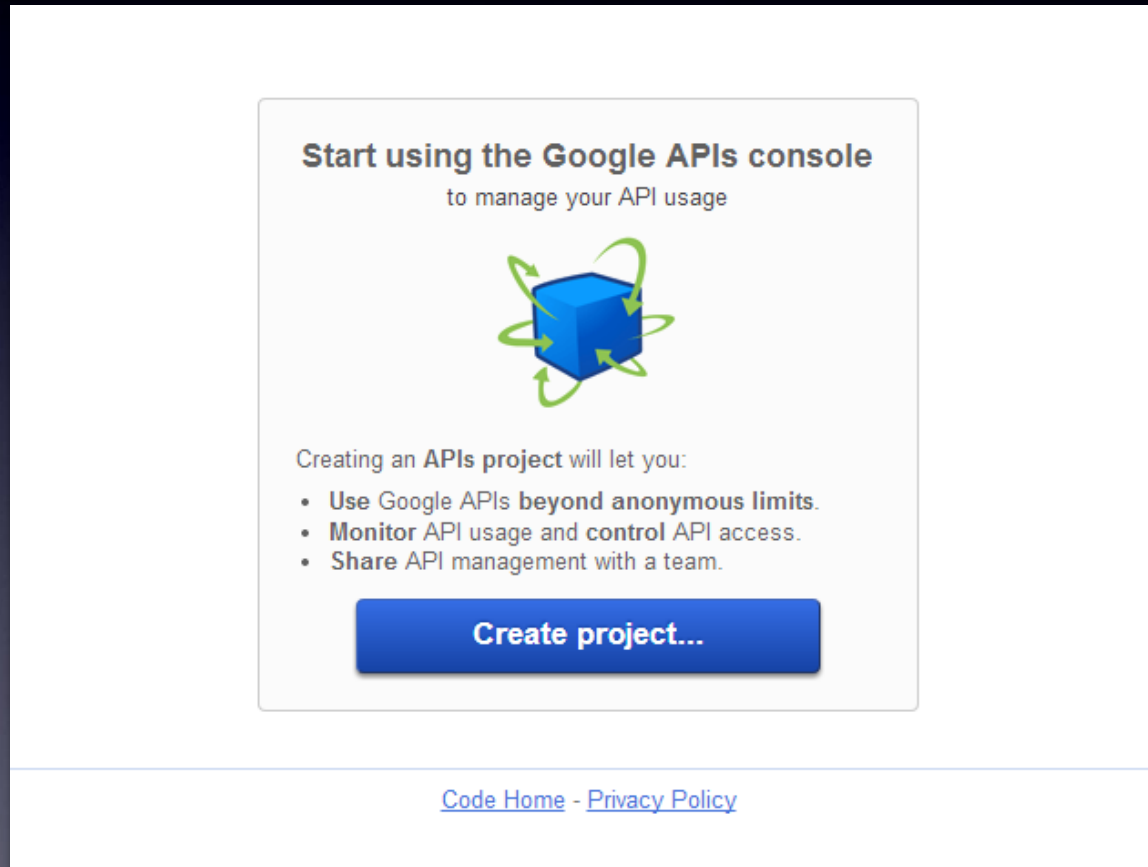


申請方法





















1. 登入Google API Console開新專案
2. 啟動Google Cloud Messaging for Android
3. 建立Server Key
4. 記住Sender ID

1. 登入GOOGLE API CONSOLE開新專案

- <https://code.google.com/apis/console>



2. 啟動GCM FOR ANDROID

	Google Apps Marketplace API		<input type="checkbox"/> OFF	Courtesy limit: 10,
	Google Apps Marketplace SDK		<input type="checkbox"/> OFF	
	Google Apps Reseller API		<input type="checkbox"/> OFF	Courtesy limit: 10,
	Google Civic Information API		<input type="checkbox"/> OFF	Courtesy limit: 25,
	Google Cloud Datastore API		<input type="checkbox"/> OFF	Courtesy limit: 10,
	Google Cloud Messaging for Android		<input checked="" type="checkbox"/> ON	
	Google Cloud Messaging for Chrome		<input type="checkbox"/> OFF	Courtesy limit: 10,
	Google Cloud SQL		<input type="checkbox"/> OFF	Pricing
	Google Cloud SQL API		<input type="checkbox"/> OFF	
	Google Cloud Storage		<input type="checkbox"/> OFF	Pricing

3. 建立SERVER KEY

Simple API Access

Use API keys to identify your project when you do not need to access user data. [Le...](#)

Key for server apps (with IP locking)

API key: AIzaSyCvEDnixmPqav3VFBIWKk2JnzBYOs41O2U
IPs: Any IP allowed
Activated on: Sep 24, 2013 8:16 PM
Activated by: bowman.liu@gmail.com – you

Key for browser apps (with referers)

API key: AIzaSyC8vp_ovlBBPRfbaHh9BZL-ePQK6-M2gkw
Referers: Any referer allowed
Activated on: Sep 24, 2013 8:16 PM
Activated by: bowman.liu@gmail.com – you

[Create new Server key...](#) [Create new Browser key...](#) [Create new Android key...](#)

Notification Endpoints

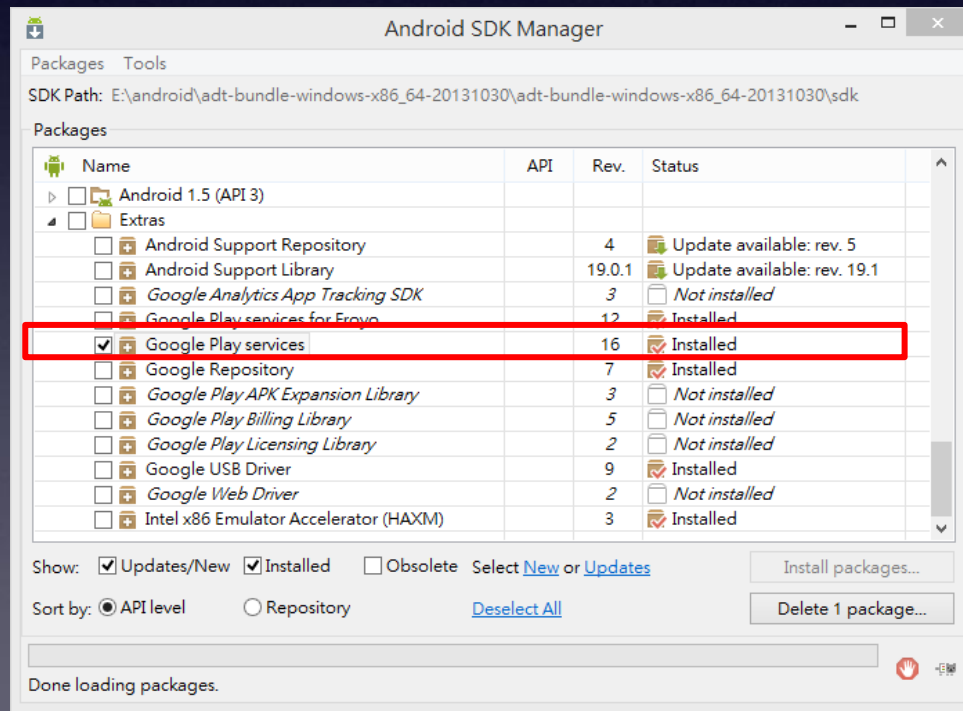
4. 記住SENDER ID

Dashboard	
Project Summary	
Name	GCM Test
Project Number	154560742341
Project ID	Register...
Google+ Page	Request connection

前置作業

前置作業

- 確定有下載Google Play Service Library
- 確定執行手機內有Google Play Store
- 確定專案有參照到Google Play Service Library



GCM/SimpleGCM

專案設定

ANDROID MANIFEST

- 根據Android官方網站的敘述，對於專案的AndroidManifest.xml的設定，直接用複製的就可以
 - <http://developer.android.com/google/gcm/client.html>

ANDROID MANIFEST

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="com.google.android.c2dm.permission.RECEIVE"/>
<permission android:name="com.example.simplegcm.permission.C2D_MESSAGE"
    android:protectionLevel="signature" />
<uses-permission android:name="com.example.simplegcm.permission.C2D_MESSAGE"/>
```

- 上方程式碼黃字的部分，改為自己的package name

ANDROID MANIFEST

```
<meta-data
    android:name="com.google.android.gms.version"
    android:value="@integer/google_play_services_version" />
<receiver android:name=".GcmBroadcastReceiver"
    android:permission="com.google.android.c2dm.permission.SEND">
    <intent-filter>
        <action
            android:name="com.google.android.c2dm.intent.RECEIVE"/>
        <category android:name="com.example.simplegcm" />
    </intent-filter>
</receiver>
<service android:name=".GcmIntentService" />
```

- 依照運作原理的介紹，專案一定要建立一個receiver和一個service

ANDROID MANIFEST

```
<meta-data
    android:name="com.google.android.gms.version"
    android:value="@integer/google_play_services_version" />
<receiver android:name=".GcmBroadcastReceiver"
    android:permission="com.google.android.c2dm.permission.SEND">
    <intent-filter>
        <action
            android:name="com.google.android.c2dm.intent.RECEIVE"/>
        <category android:name="com.example.simplegcm" />
    </intent-filter>
</receiver>
<service android:name=".GcmIntentService" />
```

- 上方黃字部分改為自己的package name

GCM/SimpleGCM

建立GCM RECEIVER

建立GCM RECEIVER

- 依照Android官方網站對GCM的描述，GCM所使用的Broadcast Receiver的寫法也都是固定的
- <http://developer.android.com/google/gcm/client.html>
見Receive a message篇

建立GCM RECEIVER

```
public class GcmBroadcastReceiver extends WakefulBroadcastReceiver{  
    public void onReceive(Context context, Intent intent) {  
        ComponentName comp = new  
            ComponentName(context.getPackageName(),  
                GcmIntentService.class.getName());  
        startWakefulService(context,  
            (intent.setComponent(comp)));  
        setResultCode(Activity.RESULT_OK);  
    }  
}
```

- class必須得繼承WakefulBroadcastReceiver
 - WakefulBroadcastReceiver是BroadcastReceiver的子類別
 - 要能持續收到訊息，必須得適當的讓裝置的CPU啟動，而這個Broadcast Receiver可以處理這個需求

建立GCM RECEIVER

```
public class GcmBroadcastReceiver extends WakefulBroadcastReceiver{  
    public void onReceive(Context context, Intent intent) {  
        ComponentName comp = new  
            ComponentName(context.getPackageName(),  
                GcmIntentService.class.getName());  
        startWakefulService(context,  
            (intent.setComponent(comp)));  
        setResultCode(Activity.RESULT_OK);  
    }  
}
```

- 使用ComponentName指定要啟動的GCM Service
- 啟動時使用startWakefulService()來啟動Service同時啟動電力

GCM/SimpleGCM

建立GCM SERVICE

建立GCM SERVICE

```
public class GcmIntentService extends IntentService {
    public GcmIntentService() {
        super("GcmIntentService");
    }
    protected void onHandleIntent(Intent intent) {
        .....
    }
    private void showNotification(String message) {
        .....
    }
}
```

- Service要繼承IntentService
- 要建立預設的建構子GcmIntentService()
- 若有訊息從GCM來，會從onHandleIntent()收到
- onHandleIntent()收到訊息後，呼叫showNotification()顯示通知

建立GCM SERVICE

```
protected void onHandleIntent(Intent intent) {  
    GoogleCloudMessaging gcm = GoogleCloudMessaging.getInstance(this);  
    String type = gcm.getMessageType(intent);  
    if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_SEND_ERROR)) {  
        showNotification("receive error");  
    } else if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_DELETED)) {  
        showNotification("message deleted");  
    } else if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_MESSAGE)) {  
        Bundle extras = intent.getExtras();  
        String message = extras.getString("message");  
        showNotification(message);  
    }  
}
```

- GoogleCloudMessaging是Google提供的工具class，專門協助處理GCM的各類工作

建立GCM SERVICE

```
protected void onHandleIntent(Intent intent) {  
    GoogleCloudMessaging gcm = GoogleCloudMessaging.getInstance(this);  
    String type = gcm.getMessageType(intent);  
    if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_SEND_ERROR)) {  
        showNotification("receive error");  
    } else if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_DELETED)) {  
        showNotification("message deleted");  
    } else if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_MESSAGE)) {  
        Bundle extras = intent.getExtras();  
        String message = extras.getString("message");  
        showNotification(message);  
    }  
}
```

- `GoogleCloudMessaging.getMessageType(intent)` 可以取得收到的訊息類型
 - `MESSAGE_TYPE_SEND_ERROR` 傳送失敗時
 - `MESSAGE_TYPE_DELETED` 訊息被刪除
 - `MESSAGE_TYPE_MESSAGE` 收到訊息時

建立GCM SERVICE

```
protected void onHandleIntent(Intent intent) {
    GoogleCloudMessaging gcm = GoogleCloudMessaging.getInstance(this);
    String type = gcm.getMessageType(intent);
    if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_SEND_ERROR)) {
        showNotification("receive error");
    } else if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_DELETED)) {
        showNotification("message deleted");
    } else if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_MESSAGE)) {
        Bundle extras = intent.getExtras();
        String message = extras.getString("message");
        showNotification(message);
    }
}
```

- 由GCM來的訊息可以從onHandleIntent()的參數Intent中的extras得到

建立GCM SERVICE

```
protected void onHandleIntent(Intent intent) {  
    GoogleCloudMessaging gcm = GoogleCloudMessaging.getInstance(this);  
    String type = gcm.getMessageType(intent);  
    if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_SEND_ERROR)) {  
        showNotification("receive error");  
    } else if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_DELETED)) {  
        showNotification("message deleted");  
    } else if (type.equals(GoogleCloudMessaging.MESSAGE_TYPE_MESSAGE)) {  
        Bundle extras = intent.getExtras();  
        String message = extras.getString("message");  
        showNotification(message);  
    }  
}
```

- extras.getString()中的索引值，要看跟Server之互相的約定的值

GCM/SimpleGCM

註冊GCM

註冊GCM

```
private GoogleCloudMessaging mGcm;  
private static final String SENDER_ID = "YOUR-SENDER-ID";  
private static final String PREF_REGISTRATION =  
    "registration_prefs";  
private static final String PREF_KEY_REGISTRATION_ID =  
    "registration_id";  
private ProgressDialog mProgress;  
private static final String REGISTER_URL = "YOUR-REGISTER-URL";
```

- SENDER_ID填入在Google API Console記下的Send id
- PREF_REGISTRATION與
PREF_KEY_REGISTRATION_ID
 - 取得GCM的registration id後，儲存在PREF_REGISTRATION中，以索引值PREF_KEY_REGISTRATION_ID儲存
- REGISTER_URL儲存自己建立Server中，可以上傳registration id的API

註冊GCM

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    int resultCode = GooglePlayServicesUtil.isGooglePlayServicesAvailable(this);  
    if (resultCode != ConnectionResult.SUCCESS) {  
        Toast.makeText(this, "你的裝置不支援GCM", Toast.LENGTH_LONG).show();  
        return;  
    }  
    mGcm = GoogleCloudMessaging.getInstance(this);  
    String registrationId = getRegistrationId();  
    if (TextUtils.isEmpty(registrationId)) {  
        registrationInBackground();  
    }  
}
```

- 使用GooglePlayServiceUtil.isGooglePlayServiceAvailable() 來檢查手機是否有支援GooglePlay Service

註冊GCM

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    int resultCode = GooglePlayServicesUtil.isGooglePlayServicesAvailable(this);  
    if (resultCode != ConnectionResult.SUCCESS) {  
        Toast.makeText(this, "你的裝置不支援GCM", Toast.LENGTH_LONG).show();  
        return;  
    }  
    mGcm = GoogleCloudMessaging.getInstance(this);  
    String registrationId = getRegistrationId();  
    if (TextUtils.isEmpty(registrationId)) {  
        registrationInBackground();  
    }  
}
```

- `getRegistrationId()`和`registrationInBackground()`都是自己建立的方法

註冊GCM

```
private void registrationInBackground() {
    AsyncTask<Void, Void, Void> task = new AsyncTask<Void, Void, Void>() {
        protected Void doInBackground(Void... params) {
            try {
                String registrationId = mGcm.register(SENDER_ID);
                sendRegistrationToBackend(registrationId);
            } catch (IOException e) { }
            return null;
        }
    };
    task.execute();
}
```

- **GoogleCloudMessaging.register(SENDER_ID)**
 - 向GCM Server註冊，這一段是需要網路連線，所以寫在AsyncTask中
- **sendRegistrationToBackend()**是自己寫的方法，負責將registration Id傳到自己的Server中儲存

GCM/SimpleGCM

自行建立的SERVER端

自行建立的SERVER端

- 本範例是使用Apache為Server
- php建立Server端的資料處理方式
- 資料庫使用mySQL
- 可以搭配簡便的解決方案 Apache + php + mySQL
WAMP

自行建立的SERVER端

- 環境建立好後，需要建立DB
- 其中需要包含一個資料表，名稱為registration
- 資料表有兩個欄位
 - _id，主鍵，型態為INT，自動遞增型
 - registration_id，型態為TEXT(256)，不得為空

自行建立的SERVER端


- 打開config.php，填入連入Server的使用者名稱、密碼以及剛剛建立的資料庫名稱

自行建立的SERVER端

- registration.php
 - 從網址帶入裝置由GCM取得的registration id
 - `http://<url location>/registration.php?id=<registration id>`
 - 帶入的registration id會存在剛剛建立的資料庫內的registration資料表的registration_id欄位中

自行建立的SERVER端

- gcm.html
 - Google API Key: 填入Google API Console申請到的Key
 - Notification Message: 要傳送給已註冊裝置的訊息



Google API Key (with IP locking) :

Get your Google API Key : [Google API](#)

Notification Message :

自行建立的SERVER端

- gcm_engine.php
 - 在gcm.html按下send notification按鈕後，會把訊息及API Key傳到gcm_engine.php中
 - gcm_engine.php會去剛剛建立的資料庫中的registration資料表的紀錄，組合要傳送給GCM Server的內容並傳遞

更多資料

更多資料

- 官方資料
 - <http://developer.android.com/google/gcm/client.html>
 - <http://developer.android.com/google/gcm/http.html>
- 很好的教學
 - <http://www.androidbegin.com/tutorial/android-google-cloud-messaging-gcm-tutorial/>