Hello city操作說明

**目錄**

[**目錄** i](#_Toc34908127)

[圖目錄 i](#_Toc34908128)

[表目錄 ii](#_Toc34908129)

[1. Android Studio專案加入Zenbo SDK模組 1](#_Toc34908130)

[(1) 點選左上角 [ File ] -> [ New ] -> [ Import Module ] 1](#_Toc34908131)

[(2) 選擇想要加入的 [ Module ] 2](#_Toc34908132)

[(3) build.gradle 加入 dependencies 設定 3](#_Toc34908133)

[2. Zenbo App Management 3](#_Toc34908134)

[(1) 網址： 3](#_Toc34908135)

[(2) 新增App資訊： 3](#_Toc34908136)

[3. DDE Editor實作 6](#_Toc34908137)

[(1) 網址： 6](#_Toc34908138)

[(2) 建立專案 7](#_Toc34908139)

[(3) 必要設定 8](#_Toc34908140)

[a. 註冊「DDE Domain ID」資訊 8](#_Toc34908141)

[b. 完成DDE專案 9](#_Toc34908142)

[(4) AndroidManifest.xml加入zenbo相關資訊 9](#_Toc34908143)

[a. Zenbo dde資訊。 9](#_Toc34908144)

[b. Zenbo launch資訊。 10](#_Toc34908145)

[(5) MainActivity.java必定要加入 10](#_Toc34908146)

[a. MainActivity::onResume() 10](#_Toc34908147)

[b. RobotCallback.Listen::onResult() 11](#_Toc34908148)

[4. Concept、Intent、Plan功能介紹 11](#_Toc34908149)

[(1) Plan 12](#_Toc34908150)

[參考資料 13](#_Toc34908151)

# 圖目錄

[圖 1‑1、Import Module 1](#_Toc34908152)

[圖 1‑2、import ZenboSDK module 2](#_Toc34908153)

[圖 1‑3、Import RobotActivityLibrary module 2](#_Toc34908154)

[圖 1‑4、build.gradle加入dependencies設定。 3](#_Toc34908155)

[圖 2‑1、新增App資訊。 3](#_Toc34908156)

[圖 2‑2、新增App Console步驟。 4](#_Toc34908157)

[圖 2‑3、完成新增App Console (1) 5](#_Toc34908158)

[圖 2‑4、完成新增App Console (2) 6](#_Toc34908159)

[圖 3‑1、建立專案。 7](#_Toc34908160)

[圖 3‑2、建立App名稱 7](#_Toc34908161)

[圖 3‑3、註冊「DDE Domain ID」資訊。 8](#_Toc34908162)

[圖 3‑4、註冊「Domain Name」資訊。 8](#_Toc34908163)

[圖 3‑5、設定Domain ID與App Management資訊。 8](#_Toc34908164)

[圖 3‑6、publish與deploy專案。 9](#_Toc34908165)

[圖 3‑7、zenbo相關資訊。 9](#_Toc34908166)

[圖 3‑8、跳至要開啟的plan內。 10](#_Toc34908167)

[圖 3‑9、回傳值資訊。 11](#_Toc34908168)

[圖 4‑1、Plan說明。 12](#_Toc34908169)

# 表目錄

[表 3‑1、Plan項目說明 12](#_Toc34908170)

# 資訊服務內容：

## 常見問題

## 樓層介紹

## 服務導覽

## 館藏查詢

## 資料庫查詢

## 期刊查詢

## 設定

# Android Studio專案加入Zenbo SDK模組

## 點選左上角 [ File ] -> [ New ] -> [ Import Module ]

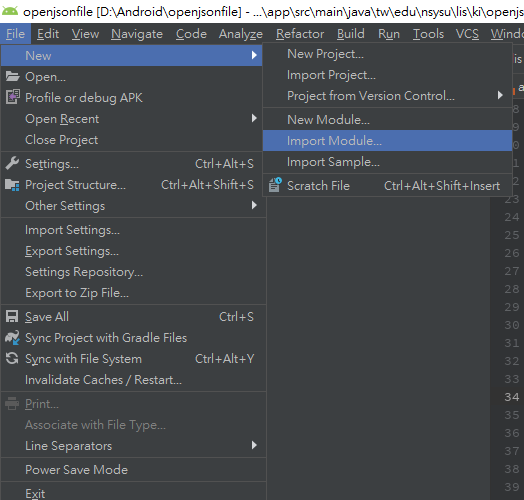


圖 1‑1、Import Module

## 選擇想要加入的 [ Module ]

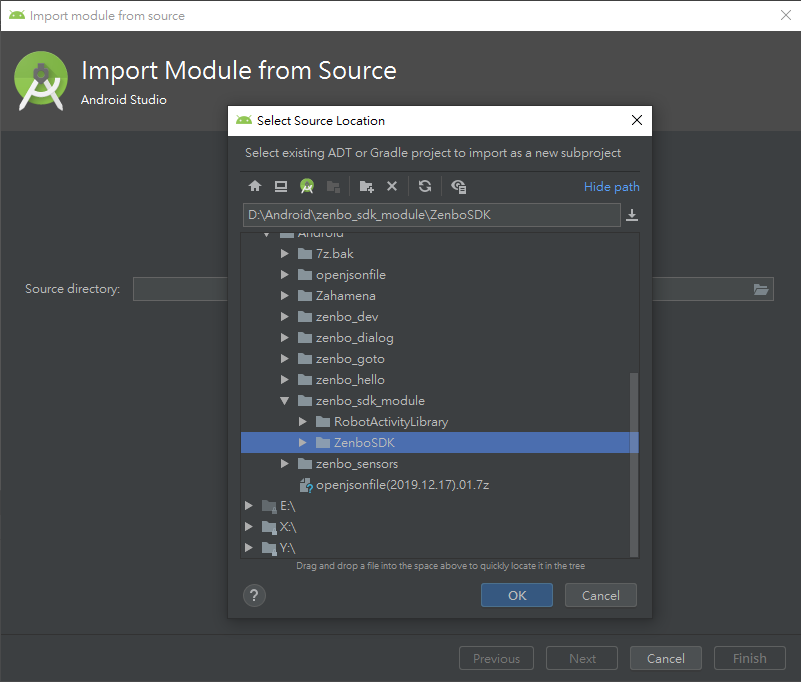


圖 1‑2、import ZenboSDK module

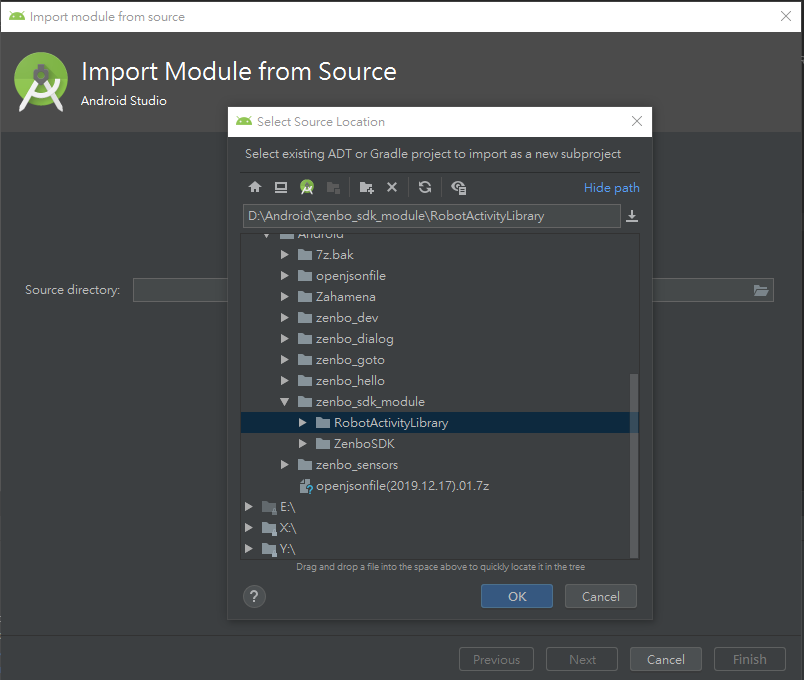


圖 1‑3、Import RobotActivityLibrary module

## build.gradle 加入 dependencies 設定

在app下的build.gradle加入dependencies 設定：

dependencies {

implementation project(':RobotActivityLibrary')

api files('../ZenboSDK/ZenboSDK.jar')

}

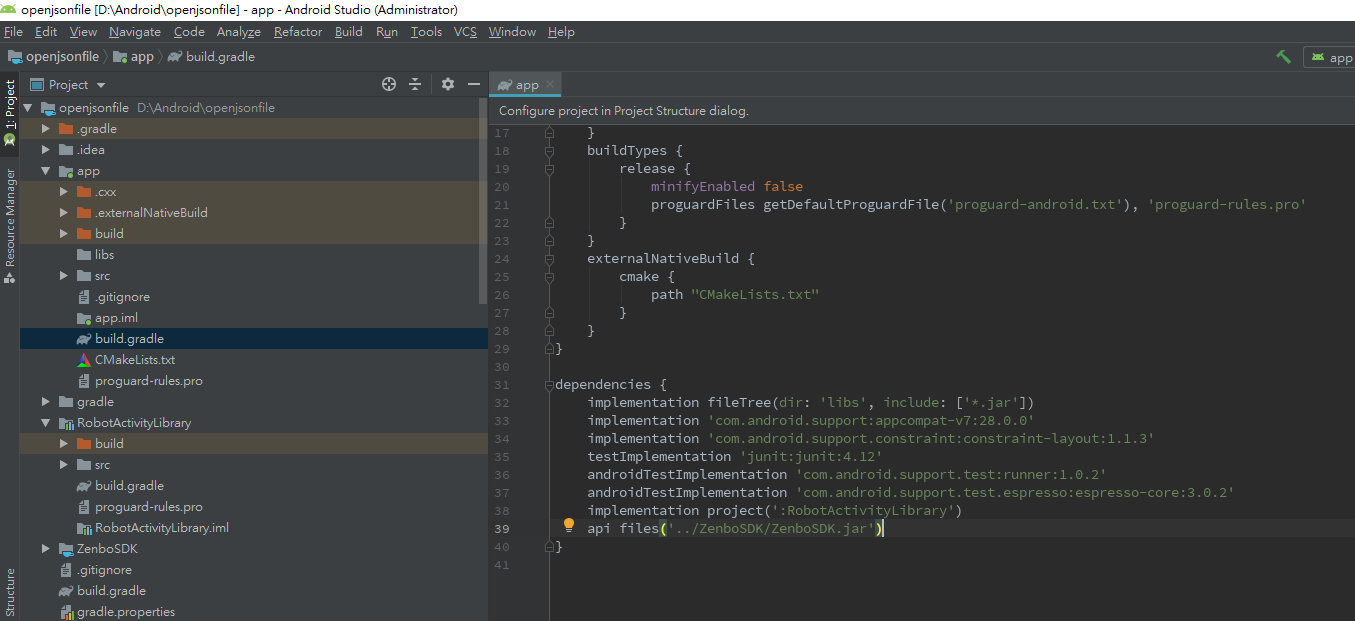


圖 1‑4、build.gradle加入dependencies設定。

# Zenbo App Management

App Managemenet相關設定都在網頁上匾及與設定。

## 網址：

<https://zenbo.asus.com/developer/console/>

## 新增App資訊：

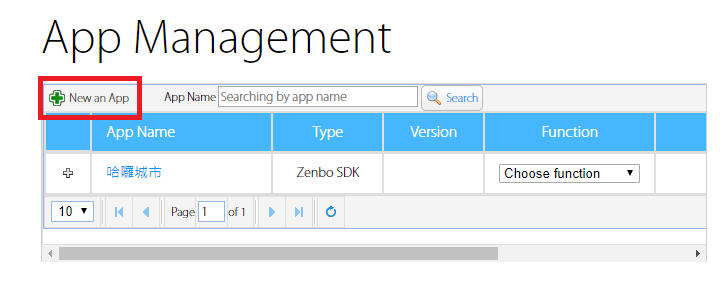


圖 2‑1、新增App資訊。

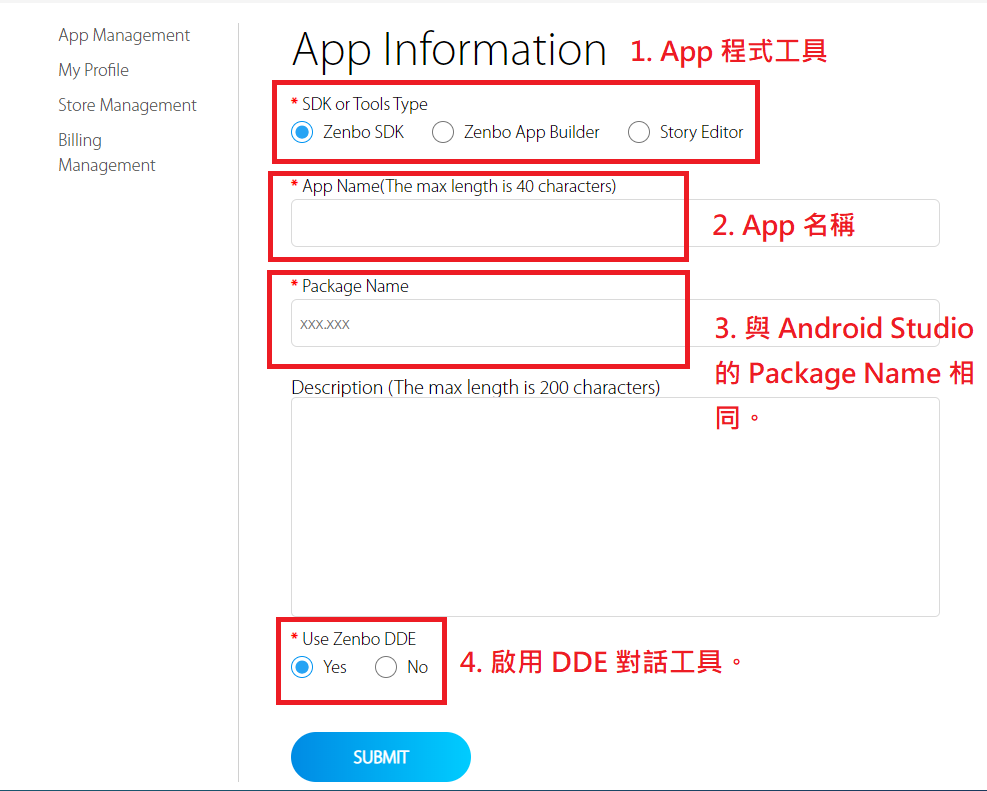


圖 2‑2、新增App Console步驟。

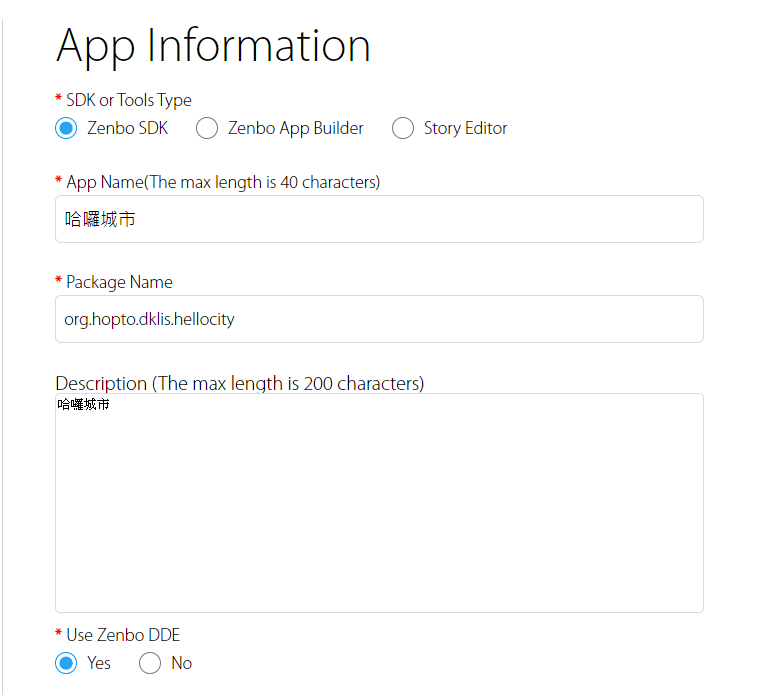


圖 2‑3、完成新增App Console (1)

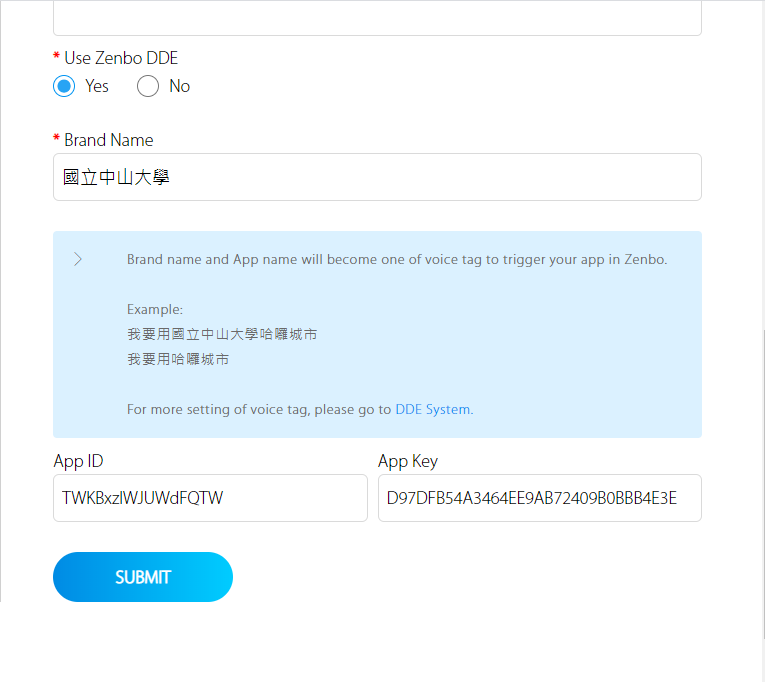


圖 2‑4、完成新增App Console (2)

# DDE Editor實作

同樣的DDE Editor都是在網頁上測試與編寫。

## 網址：

<https://zenbo.asus.com/developer/tools/ds-editor.jsp>

## 建立專案

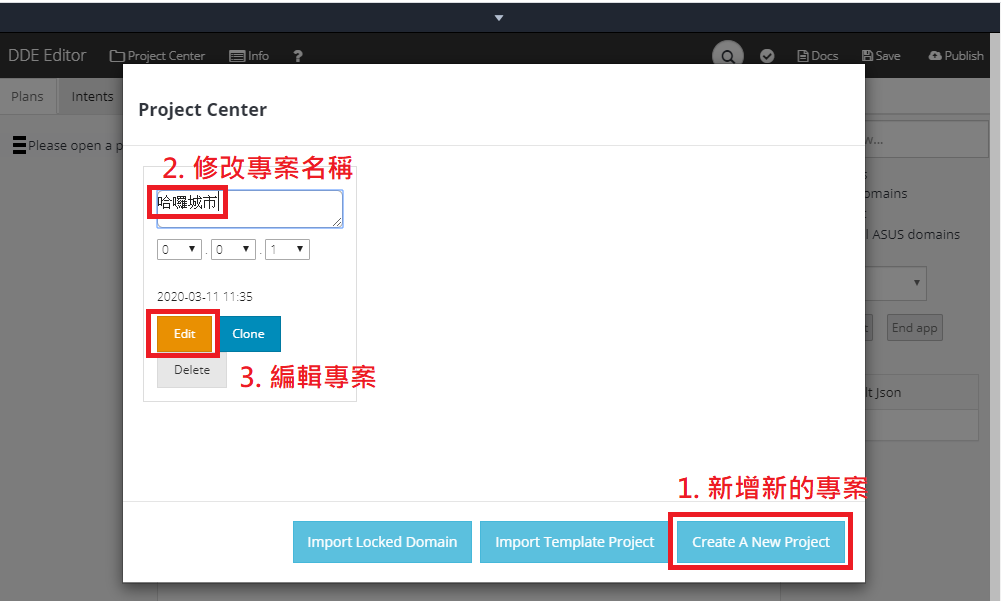


圖 3‑1、建立專案。

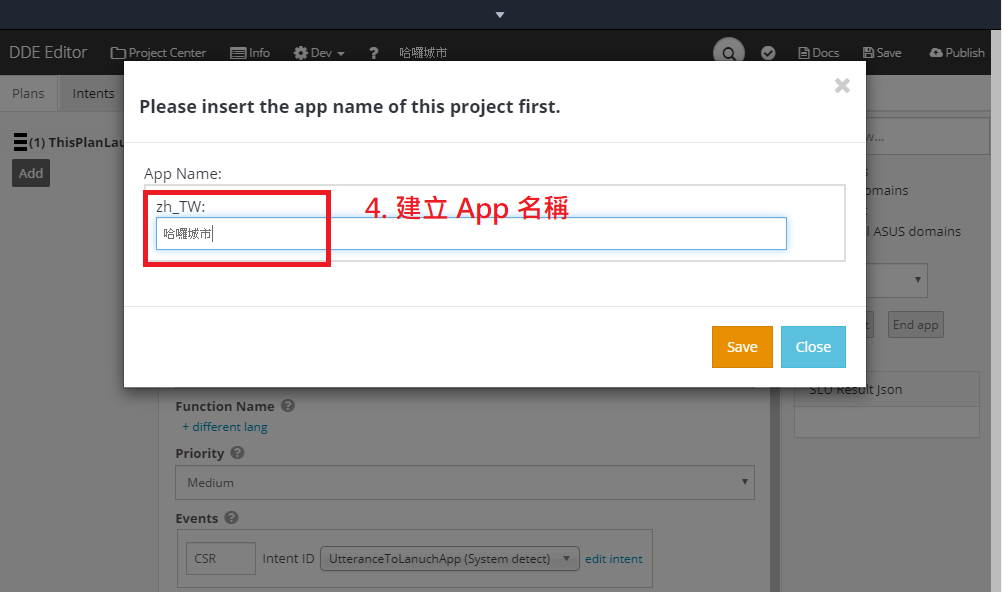


圖 3‑2、建立App名稱

## 必要設定

### 註冊「DDE Domain ID」資訊

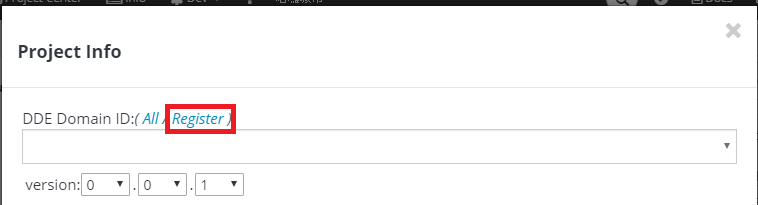


圖 3‑3、註冊「DDE Domain ID」資訊。



圖 3‑4、註冊「Domain Name」資訊。

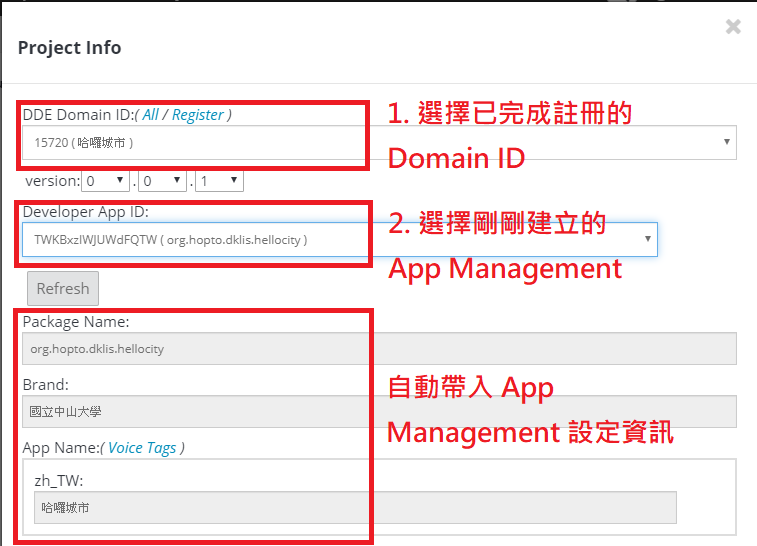


圖 3‑5、設定Domain ID與App Management資訊。

### 完成DDE專案

完成DDE專案後必須「Publish」與「Deploy to Download Server」才能與Android專案連結。

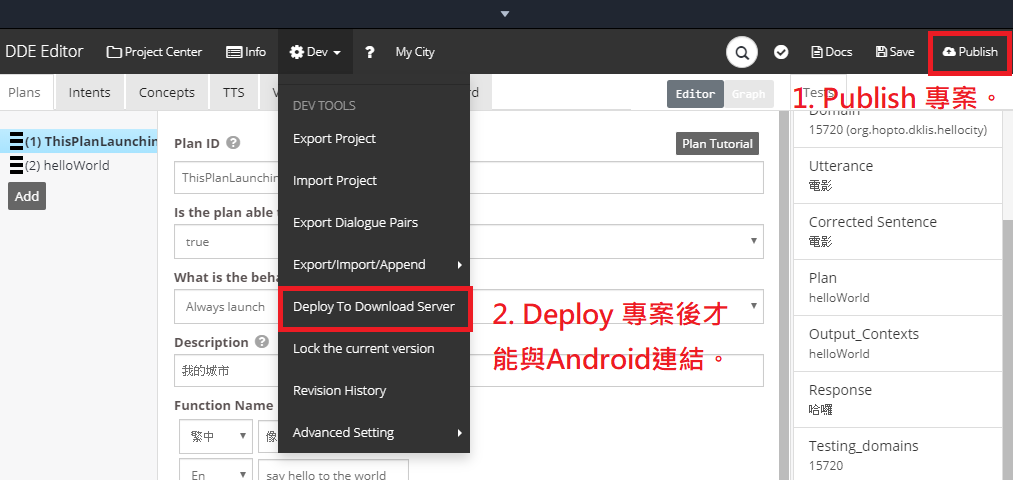


圖 3‑6、publish與deploy專案。

## AndroidManifest.xml加入zenbo相關資訊

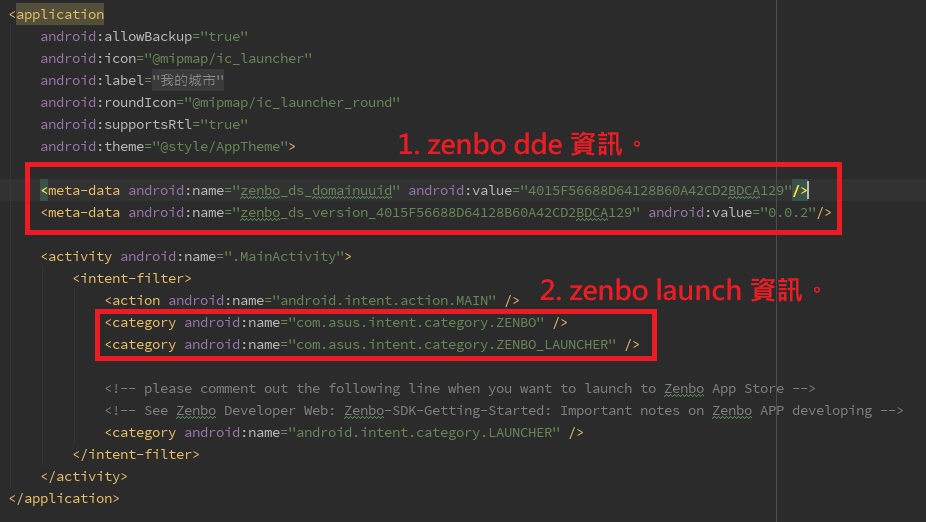


圖 3‑7、zenbo相關資訊。

### Zenbo dde資訊。

<meta-data android:name="zenbo\_ds\_domainuuid" android:value="4015F56688D64128B60A42CD2BDCA129"/>

<meta-data android:name="zenbo\_ds\_version\_4015F56688D64128B60A42CD2BDCA129" android:value="0.0.2"/>

### Zenbo launch資訊。

<category android:name="com.asus.intent.category.ZENBO" />  
<category android:name="com.asus.intent.category.ZENBO\_LAUNCHER" />

## MainActivity.java必定要加入

### MainActivity::onResume()



圖 3‑8、跳至要開啟的plan內。

### RobotCallback.Listen::onResult()



圖 3‑9、回傳值資訊。

# Concept、Intent、Plan功能介紹

在語音對話過程是由多個句子組合而成，而句子的意圖以Plan來控制邏輯關係,，意圖的集合句型置於Intent之中，而Concept則是各個句型中所包含的詞彙集合。

所以設計「語音對話」建議先從「小」到「大」來設計對話方式，先從「Concept」🡪「Intent」🡪「Plan」方式設計。

## Plan

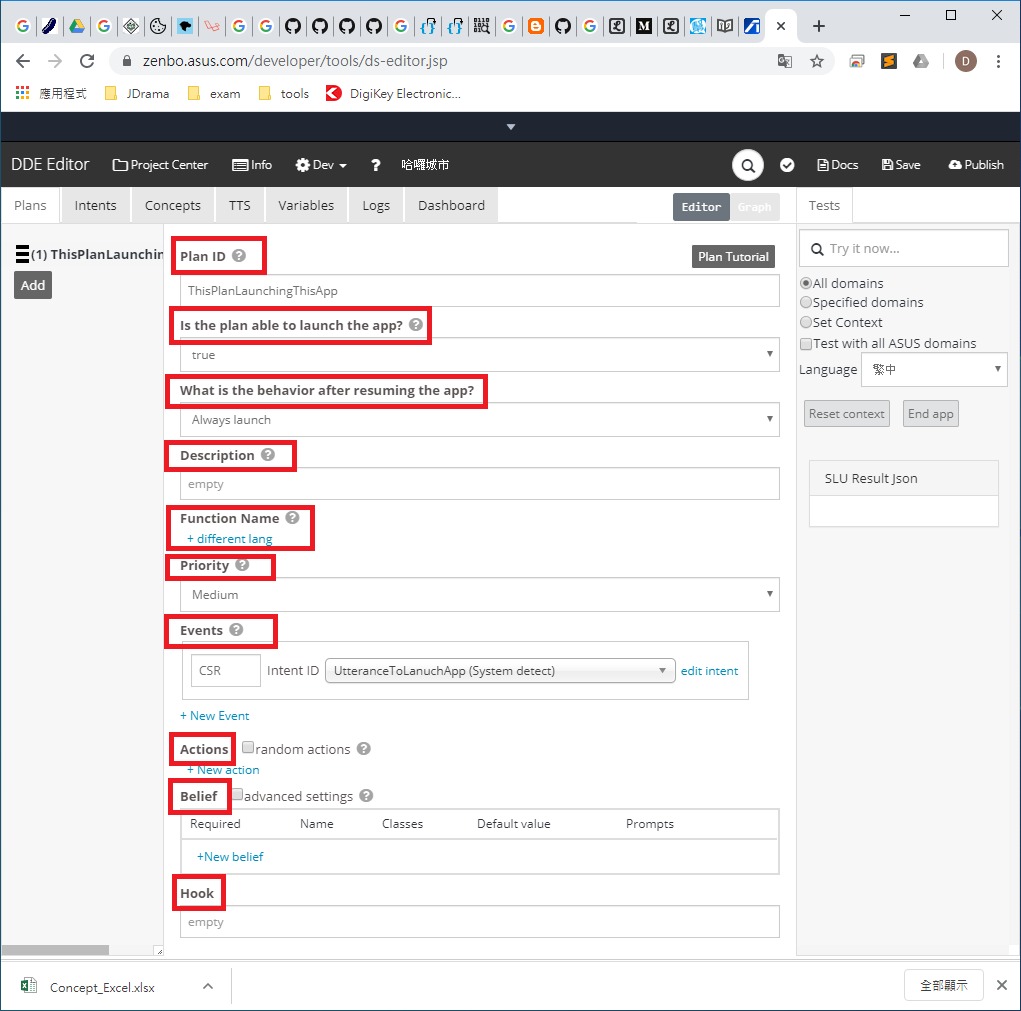


圖 4‑1、Plan說明。

表 4‑1、Plan項目說明

|  |  |
| --- | --- |
| 項目 | 說明 |
| Plans | Plan ID is the name of a plan. Every plan requires a unique plan ID. |
| Is the plan able to launch the app? | Set the plan as cross plan to launch the app or in-app plan.  True🡪 plan to launch the app.  False🡪 in-app plan. |
| What is the behavior after resuming the app? | To resume background apps, there are three ways, which are “Always launch”, “Always resume”, and “Always Resume but Set Plan Context”. If choosing the last one to resume the app, the Context’s settings will contain Plan’s Output Context. |
| Input Context | For example, set a plan’s Input Context as "A". It will be linked to other plan, which its Output Context is also set as "A". |
| Description | Description About the plan that you edit. |

# 參考資料

|  |  |
| --- | --- |
| [1] | github, “github,” github, 2020. [線上]. Available: https://github.com/. |
| [2] | git, “git,” git, 2020. [線上]. Available: https://git-scm.com/. |
| [3] | 黃保翕, “30 天精通 Git 版本控管 (24)：使用 GitHub 遠端儲存庫 - 入門篇,” 電週文化事業, 24 10 2013. [線上]. Available: https://ithelp.ithome.com.tw/articles/10139849. |