

# App Inventor2 連接 Firebase資料庫 與聊天室製作



國立臺中教育大學 大學部人工智慧應用  
數位系三年級

---

吳智鴻

教學網站：[HTTP://120.108.221.55/PROFCHWU/DCTEC](http://120.108.221.55/PROFCHWU/DCTEC)

FB社團： 106 數位系人工智慧

APP INVENTOR PROJECT: FIREBASE\_CHATROOM

A solid green horizontal bar at the bottom of the slide.

# Firestore資料庫簡介

---

## 成立

- 2011/9月，2014年被google買下。

## 免費方案

同時100個連線

1GB容量

10GB流量限制

可以與很多程式連結使用

ex. App Inventor2, Webduino, python, …..

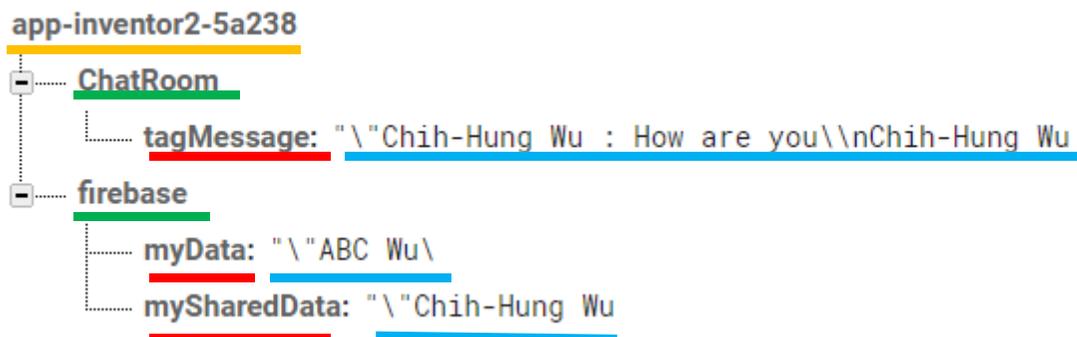
# Firestore的儲存資料方式

FIREBASE的資料長這樣

他是以TAG方式來儲存資料。

屬於NOSQL資料庫。

非傳統關連式資料庫。



TAG就像是關連式資料庫的主鍵key。利用TAG名稱來辨識出每一筆資料。

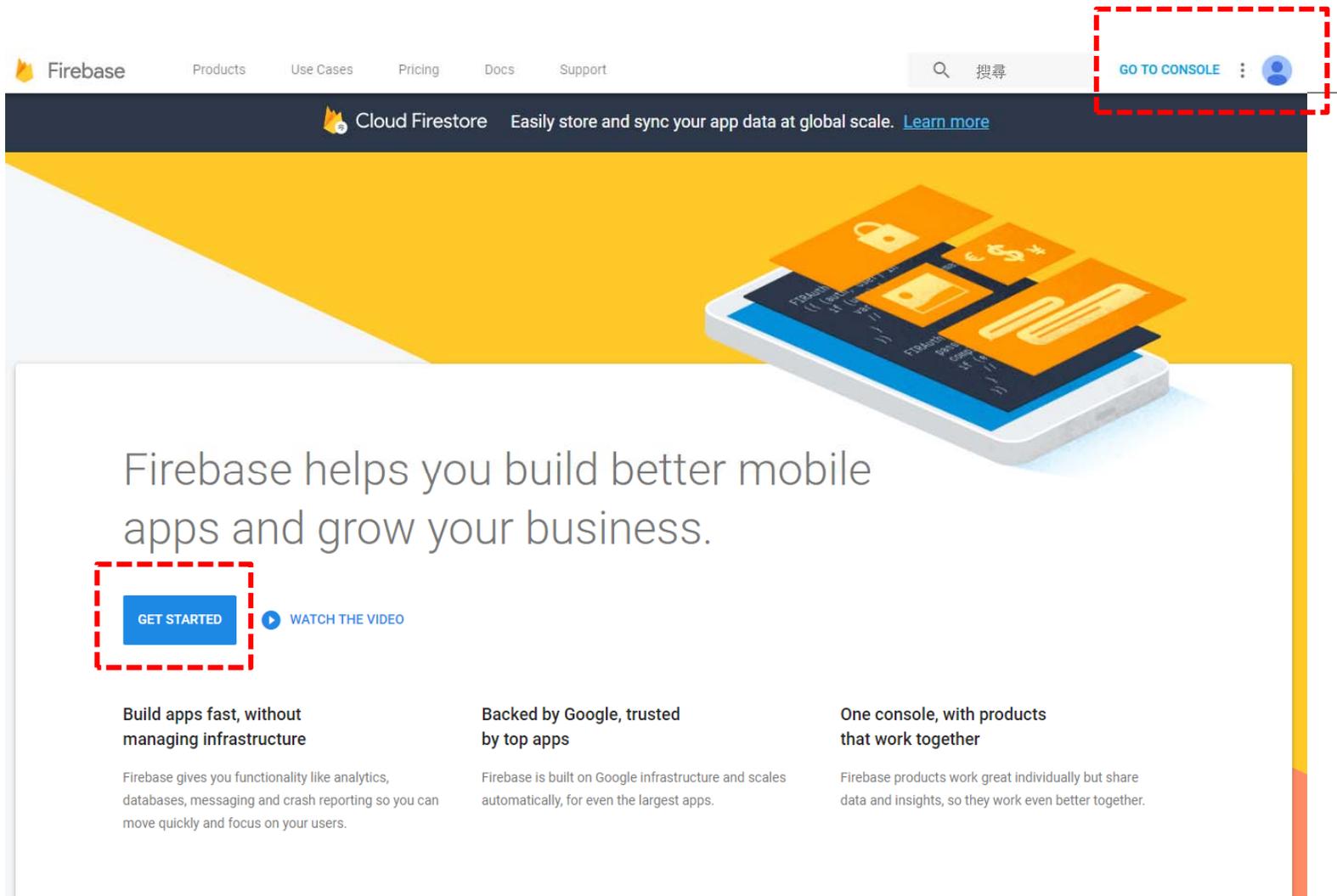
「紅色線條」部分就是TAG名稱

「藍色線條」部分是資料內容

「綠色線條」部分是ProjectBucket的名稱。「橘色」的是專案名稱。

# Step 1. Login

## https://firebase.google.com/



The screenshot shows the Firebase website homepage. At the top, there is a navigation bar with the Firebase logo, links for Products, Use Cases, Pricing, Docs, and Support, a search bar with the Chinese characters '搜尋', and a 'GO TO CONSOLE' button next to a user profile icon. Below the navigation bar is a dark blue banner for Cloud Firestore with the text 'Easily store and sync your app data at global scale. Learn more'. The main content area features a large yellow and white graphic of a smartphone with various app icons. Below the graphic, the text reads 'Firebase helps you build better mobile apps and grow your business.' Underneath this text are two buttons: 'GET STARTED' and 'WATCH THE VIDEO'. At the bottom, there are three columns of text describing Firebase's benefits: 'Build apps fast, without managing infrastructure', 'Backed by Google, trusted by top apps', and 'One console, with products that work together'.

Firebase

Products Use Cases Pricing Docs Support

GO TO CONSOLE

Cloud Firestore Easily store and sync your app data at global scale. [Learn more](#)

Firebase helps you build better mobile apps and grow your business.

GET STARTED WATCH THE VIDEO

**Build apps fast, without managing infrastructure**

Firestore gives you functionality like analytics, databases, messaging and crash reporting so you can move quickly and focus on your users.

**Backed by Google, trusted by top apps**

Firestore is built on Google infrastructure and scales automatically, for even the largest apps.

**One console, with products that work together**

Firestore products work great individually but share data and insights, so they work even better together.

# Step #2

## 新增專案



取得說明文件



C

### 歡迎使用 Firebase !

Google 提供多項工具，協助您開發優質應用程式、與使用者交流互動，以及透過行動廣告賺取更多收益。

[瞭解詳情](#) [說明文件](#) [支援](#)

近期專案

匯入 GOOGLE 專案



新增專案



探索示範專案

app inventor2

app-inventor2-5a238

您目前在 Firebase.com 上的專案

在 Firebase.com 中找不到您的專案嗎？點選[這裡](#)即可連結帳戶。

# Step # 3

## 規則，設定為開放。將read, write都改為true

The screenshot shows the Firebase console interface for a project named 'app inventor2'. The left sidebar contains various service categories: DEVELOP (Authentication, Database, Storage, Hosting, Functions), STABILITY (Crash Reporting, Performance, Test ...), ANALYTICS (Dashboard, Events, Audiences, Attrib...), and GROW (Predictions, Notifications, Remote C...). The 'Database' service is selected, and the 'Realtime Database' is chosen. The '規則' (Rules) tab is active, showing a warning message: '您的安全性規則定義成公開狀態，因此任何人都能讀取或寫入您的資料庫' (Your security rules are defined as public, so anyone can read or write to your database). Below the warning, the rules are displayed in a code editor:

```
1 {  
2   "rules": {  
3     ".read": true,  
4     ".write": true  
5   }  
6 }
```

The code editor shows the following JSON rules:

```
{  
  "rules": {  
    ".read": true,  
    ".write": true  
  }  
}
```

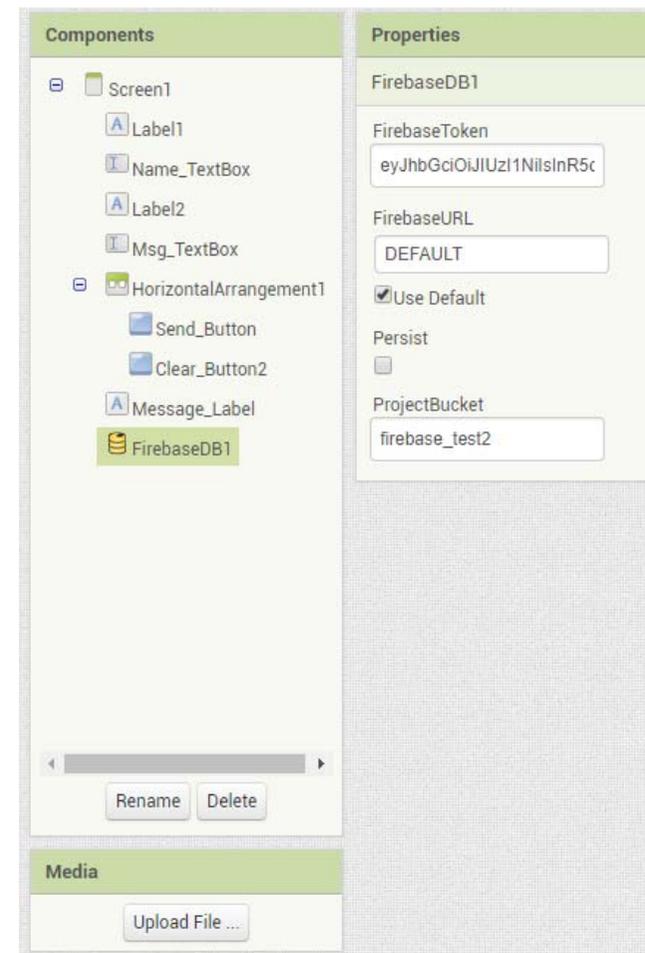
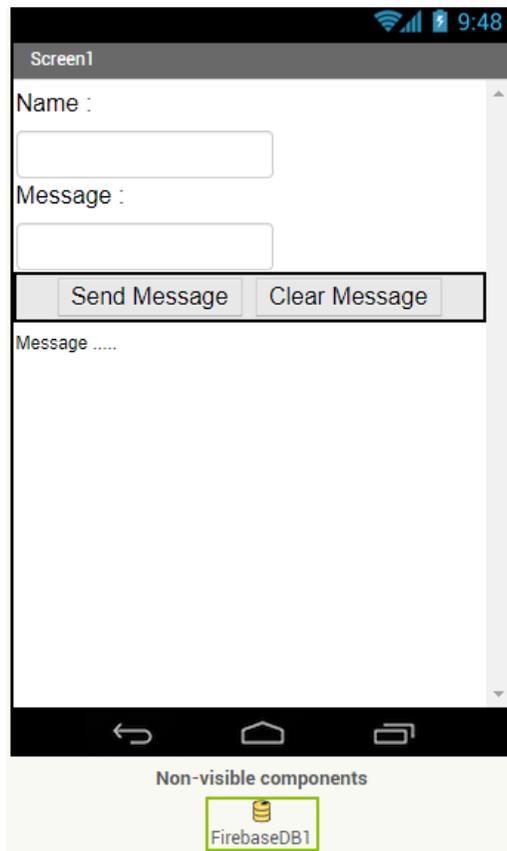
# Step #4 選擇Database

可以看到裡面的資料，以及連結此資料的網址

The screenshot displays the Firebase console interface. On the left sidebar, the 'Database' option is highlighted with a red dashed box. The main content area shows the URL 'https://app-inventor2-5a238.firebaseio.com/' and a warning message: '您的安全性規則定義成公開狀態，因此任何人都能讀取或寫入您的資料庫'. Below the warning, the project name 'app-inventor2-5a238' and the Firebase logo are also highlighted with a red dashed box.

如果先前有存過資料成功的話，資料會顯示在這邊。這裡顯示的預先自己打好的資料。

# Step #5. App Inventor 2 螢幕設計



# Firestore的資料長這樣 ：先輸入這樣的資料

app inventor2

取得說明文件

## Database

Realtime Database

資料 規則 備份 用量

https://app-inventor2-5a238.firebaseio.com/

您的安全性規則定義成公開狀態，因此任何人都能讀取或寫入您的資料庫

瞭解詳情 關閉

```
app-inventor2-5a238
├── firebase
│   ├── myData: "\\\"ABC Wu\\"
│   └── mySharedData: "\\\"Chih-Hung Wu\""
```

Spark  
免費 每月 \$0 美元 升級

# Step #6. 設定那邊可以找到必要的資訊，並輸入App Inventor2

The image shows a screenshot of the Firebase console interface. On the left is a dark sidebar with navigation options like 'Project Overview', 'Authentication', 'Database', 'Storage', 'Hosting', 'Functions', 'STABILITY', 'ANALYTICS', 'GROW', and 'Spark'. The main content area is titled 'app inventor2' and '設定' (Settings). A red dashed box highlights the '設定' tab and a table of project information:

專案名稱	app inventor2
公開名稱	project-840125511149
專案 ID	app-inventor2-5a238
網路 API 金鑰	[Redacted]

A green callout bubble points to the '網路 API 金鑰' field, stating 'App Inventor 2 裡面的設定'. Below this is a 'Properties' panel for 'FirebaseDB1'. A red dashed box highlights the 'FirebaseToken' field, which contains the value 'AlzaSyA\_h1tHDntnrskucbl'. A green dashed arrow points from the '網路 API 金鑰' field in the settings table to this 'FirebaseToken' field. Other fields in the Properties panel include 'FirebaseURL' (https://app-inventor2-5a238), 'Persist' (unchecked), and 'ProjectBucket' (ChatRoom). At the bottom of the main content area, there are three buttons for adding Firebase to iOS, Android, and web applications.

# Step #6. 設定那邊可以找到必要的資訊，並輸入App Inventor2

App Inventor 2 裡面的設定

Database

Realtime Database

資料 規則 備份 用量

https://app-inventor2-5a238.firebaseio.com/

您的安全性規則定義成公開狀態，因此任何人都能讀取或寫入您的資料庫

瞭解詳情 關閉

app-inventor2-5a238

```
firebase
├── myData: "\"ABC Wu\"
└── mySharedData: "\"Chih-Hung Wu"
```

Properties

FirebaseDB1

FirebaseToken

AlzaSyA\_h1tHDntnrskucbl

FirebaseURL

https://app-inventor2-5a238

Use Default

Persist

ProjectBucket

ChatRoom

# Firestore的資料長這樣 ：先輸入這樣的資料

app inventor2

取得說明文件

## Database

Realtime Database

資料 規則 備份 用量

https://app-inventor2-5a238.firebaseio.com/

您的安全性規則定義成公開狀態，因此任何人都能讀取或寫入您的資料庫

瞭解詳情 關閉

```
app-inventor2-5a238
└── firebase
    ├── myData: "\"ABC Wu\"
    └── mySharedData: "\"Chih-Hung Wu"
```

Spark  
免費 每月 \$0 美元 升級

# App Inventor 2的螢幕設計與 Firebase元件的設定

The screenshot displays the App Inventor 2 interface, divided into three main sections: Viewer, Components, and Properties.

- Viewer:** Shows a mobile app preview. The screen contains:
  - A status bar at the top with signal, Wi-Fi, and battery icons, and the time 9:48.
  - A header labeled "Screen1".
  - A "Name:" label with a text box containing "Chih-Hung Wu".
  - A "Message:" label with a text box containing "Hi!".
  - Two buttons: "Send Message" and "Clear Message".
  - A "Message ....." label with a large empty text area below it.
  - A bottom navigation bar with back, home, and recent apps icons.
- Components:** Lists the components used in the app:
  - Screen1
    - Label1
    - Name\_TextBox
    - Label2
    - Msg\_TextBox
    - HorizontalArrangement1
      - Send\_Button
      - Clear\_Button2
    - Message\_Label
    - FirestoreDB1 (highlighted in green)

- Properties:** Shows the configuration for the selected FirestoreDB1 component:
- FirestoreDB1
  - FirestoreToken: AlzaSyA\_h1tHDntnrskucbl
  - FirestoreURL: https://app-inventor2-5a238
  - Use Default:
  - Persist:
  - ProjectBucket: ChatRoom

At the bottom of the Components panel, there are "Rename" and "Delete" buttons. Below the Components panel is a "Media" section with an "Upload File ..." button. At the bottom of the Viewer panel, there is a "Non-visible components" section with a FirestoreDB1 component icon.

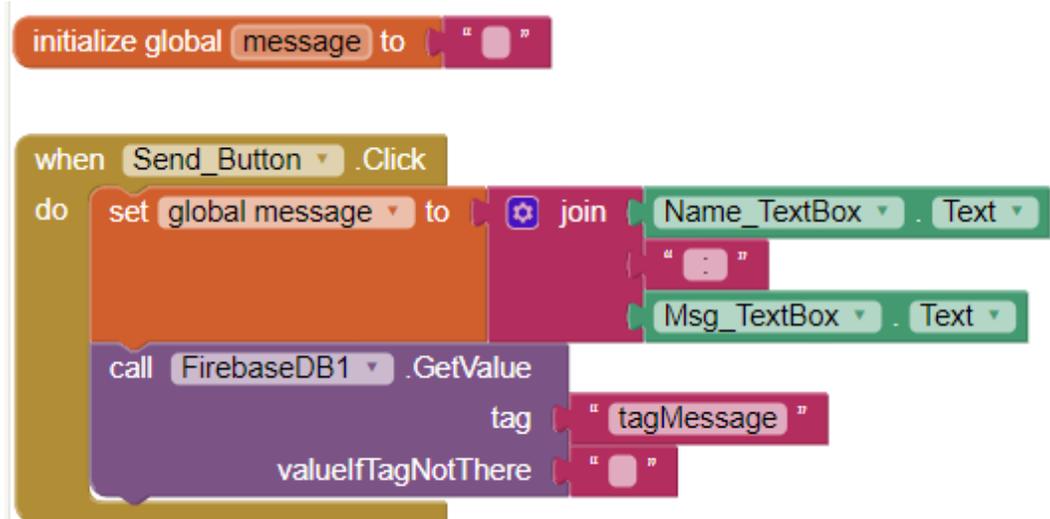
# Prg#1 送出訊息的程式

送出按鈕時

把姓名與訊息合併成一個。

Ex. 吳智鴻：Hi

呼叫firebase，抓取tag名稱  
為「tagMessage」的資料



```
initialize global message to ""

when Send_Button .Click
do
  set global message to join
  Name_TextBox .Text
  ":"
  Msg_TextBox .Text

  call FirebaseDB1 .GetValue
  tag "tagMessage"
  valueIfTagNotThere ""
```

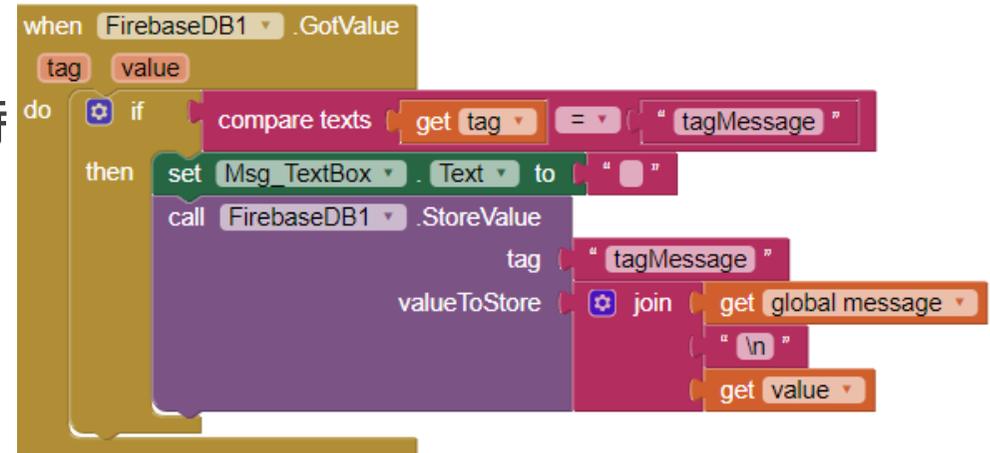
The image shows a Scratch script. It starts with an 'initialize global message to' block with an empty string. Then, a 'when Send\_Button .Click' block contains a 'do' block. Inside the 'do' block, there is a 'set global message to' block followed by a 'join' block. The 'join' block takes 'Name\_TextBox .Text', a colon character, and 'Msg\_TextBox .Text' as input. Below this, there is a 'call FirebaseDB1 .GetValue' block with a 'tag' of 'tagMessage' and a 'valueIfTagNotThere' block with an empty string.

# Prg#2 從firebase取得資料

當取得資料時

若資料tag名稱=tagMessage時

呼叫firebase，儲存資料，  
將資料存入以tag名為  
「tagMessage」的資料列中。

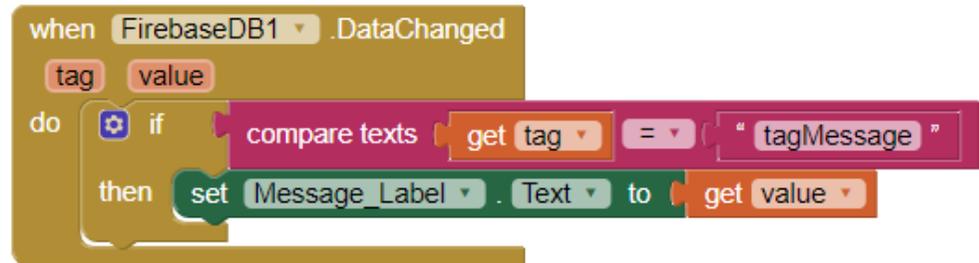


# Prg#3 從firebase取得資料

當資料庫資料有異動時

若資料tag名稱=tagMessage時

呼叫firebase，把抓出來的資料  
顯示在螢幕中。



```
when FirebaseDB1 .DataChanged
  tag value
do
  if
    compare texts get tag = "tagMessage"
  then
    set Message_Label . Text to get value
```

The image shows a Scratch code block for a Firebase database change event. The event is triggered when 'FirebaseDB1' has a '.DataChanged' event. The code block contains a 'do' loop with an 'if' condition. The 'if' condition is 'compare texts' with 'get tag' on the left, an equals sign in the middle, and 'tagMessage' in quotes on the right. If the condition is true, the 'then' block is executed, which is 'set Message\_Label . Text to get value'.

# Prg#4 從firebase清除資料

---

清除資料

呼叫firebase，把tag名稱=tagMessage的資料存入空白值。



# Step 7. 輸入程式碼

## 全部的程式碼

---

initialize global message to " "

```
when Send_Button .Click
do
  set global message to join
    Name_TextBox .Text
    "."
    Msg_TextBox .Text
  call FirebaseDB1 .GetValue
    tag "tagMessage"
    valueIfTagNotThere " "
```

```
when FirebaseDB1 .DataChanged
  tag value
do
  if
    compare texts
      get tag
      =
      "tagMessage"
  then
    set Message_Label .Text to get value
```

```
when FirebaseDB1 .GotValue
  tag value
do
  if
    compare texts
      get tag
      =
      "tagMessage"
  then
    set Msg_TextBox .Text to " "
    call FirebaseDB1 .StoreValue
      tag "tagMessage"
      valueToStore
        join
          get global message
          "\n"
          get value
```

```
when Clear_Button2 .Click
do
  call FirebaseDB1 .StoreValue
    tag "tagMessage"
    valueToStore " "
```

# Step #8. 在firebase檢視資料是否已經存入

App Inventor 2 裡面的設定

Database

資料 規則 備份 用量

https://app-inventor2-5a238.firebaseio.com/

您的安全性規則定義成公開狀態，因此任何人都能讀取或寫入您的資料庫。  
瞭解詳情 關閉

app-inventor2-5a238

- ChatRoom
  - tagMessage: "&quot;\&quot;Chih-Hung Wu : How are you\\nChih-Hung Wu
- firebase
  - myData: "&quot;\&quot;ABC Wu\"
  - mySharedData: "&quot;\&quot;Chih-Hung Wu

資料表名稱：  
Chatroom

已經成功  
存入囉！

tag名稱：  
tagMessage

Properties

FirestoreDB1

FirebaseToken  
AlzaSyA\_h1tHDntnrskucbl

FirebaseURL  
https://app-inventor2-5a238

Use Default

Persist

ProjectBucket  
ChatRoom

# 雛形範例

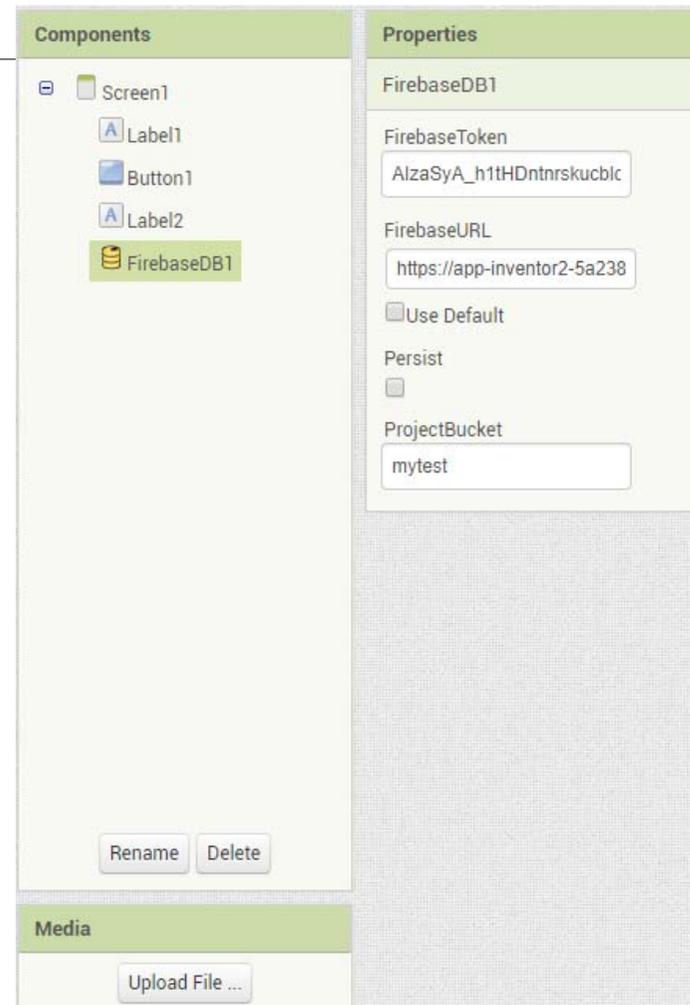
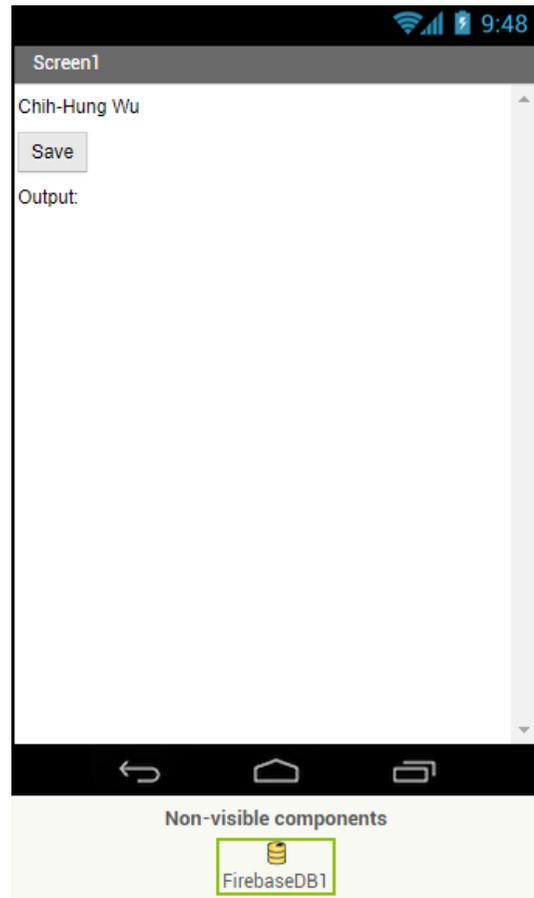
## 先從最簡單的程式來開始

---



# 雛形範例：儲存固定資料，然後將資料從firebase取出並顯示

螢幕設計



# 程式碼

---

```
when Button1 .Click
do
  call FirebaseDB1 .StoreValue
    tag "myData"
    valueToStore "Chih-Hung Wu Good!"
  call FirebaseDB1 .GetValue
    tag "myData"
    valueIfTagNotThere ""
```

```
when FirebaseDB1 .GotValue
  tag value
do
  if
  then set Label2 .Text to get value
```

# 資料已經儲存至Firebase 資料表mytest , tag=myData

The screenshot displays the Firebase console interface for a project named 'app inventor2'. The left sidebar shows navigation options under 'DEVELOP', 'STABILITY', 'ANALYTICS', and 'GROW'. The main content area is titled 'Database' and shows the 'Realtime Database' structure. A warning message at the top indicates that the security rules are set to public, allowing anyone to read or write to the database. The database structure is as follows:

```
app-inventor2-5a238
├── ChatRoom
├── firebase
│   ├── myData: "\"ABC Wu\"
│   └── mySharedData: "\"Chih-Hung Wu
└── mytest
    └── myData: "\"Chih-Hung Wu Good
```

The 'mytest' node and its 'myData' value are highlighted with a red dashed box.

# 也可以用這個函數

```
when Button1 .Click
do
  call FirebaseDatabase1 .StoreValue
  tag "myData"
  valueToStore "Chih-Hung Wu Good!"
  call FirebaseDatabase1 .GetValue
  tag "myData"
  valueIfTagNotThere ""
```

當偵測到資料有  
改變時，執行

```
when FirebaseDatabase1 .DataChanged
  tag value
do
  set Label2 . Text to get value
```

```
when FirebaseDatabase1 .GotValue
  tag value
do
  if get tag = "myData"
  then set Label2 . Text to get value
```



# 執行結果

按下Save按鈕後，

程式寫入Chih-Hung Wu Good至firebase。

然後讀取firebase中的資料，並顯示在螢幕上。



# 期中作品 思考一下，如何將 這個功能結合在水平儀遊戲呢？

將最高分數儲存在FIREBASE中

專案名稱：BALLOON\_DATABASE\_OK



# 考慮重點

---

將分數從firebase取出

若目前分數  $>$  firebase裡面的分數，則寫入新的最高分數

自己想一下要  
怎麼做呢？

---

盡量不要先往下看，先自己做做看吧！

# 先在firebase新增資料如下， 避免第一次玩沒有資料

The image shows two screenshots of the Firebase console. The top screenshot displays the root of the database for 'app-inventor2-5a238'. It shows a tree view with nodes: ChatRoom (tagMessage: "\Chih-Hung Wu : How are you\nChih-Hung Wu"), firebase (myData: "\ABC Wu", mySharedData: "\Chih-Hung Wu"), and mytest (myData: "\Chih-Hung Wu Good"). A modal is open to add a new node named 'gamedata' with a value of '0'. The bottom screenshot shows the 'gamedata' node selected, with a modal to add a new entry named 'highscore' with a value of '1'. Both screenshots include a security warning: '您的安全性規則定義成公開狀態，因此任何人都能讀取或寫入您的資料庫'.

app-inventor2-5a238

名稱 gamedata 值 0

取消 新增

ChatRoom

tagMessage: "\Chih-Hung Wu : How are you\nChih-Hung Wu

firebase

myData: "\ABC Wu\  
mySharedData: "\Chih-Hung Wu

mytest

myData: "\Chih-Hung Wu Good

https://app-inventor2-5a238.firebaseio.com/gamedata

您的安全性規則定義成公開狀態，因此任何人都能讀取或寫入您的資料庫

瞭解詳情 關閉

app-inventor2-5a238 > gamedata

gamedata: null

名稱 highscore 值 1

取消 新增

# 完成後資料會像這樣



```
app-inventor2-5a238
├── ChatRoom
│   └── tagMessage: "\nChih-Hung Wu : How are you\n\nChih-Hung Wu"
├── firebase
│   ├── myData: "\nABC Wu\n"
│   └── mySharedData: "\nChih-Hung Wu"
├── gamedata
│   └── highscore: 1
└── mytest
    └── myData: "\nChih-Hung Wu Good"
```

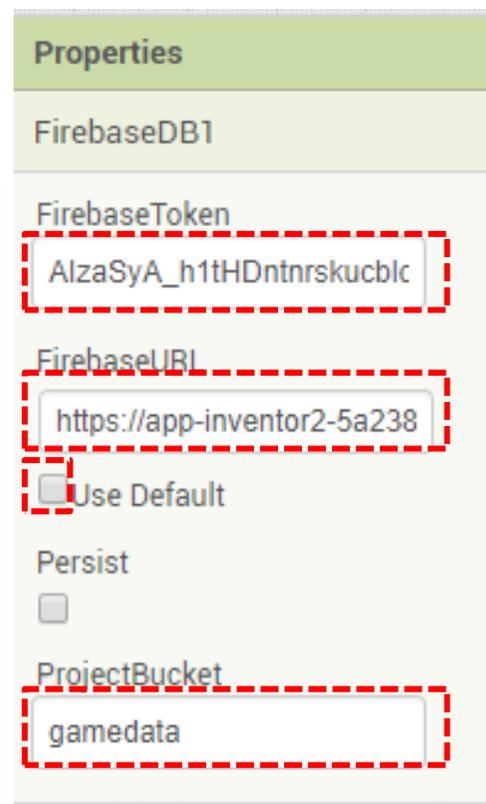
# 螢幕與元件屬性

螢幕設計

加入firebase元件



修改這些地方



# 加入讀取資料庫程式

---

一進入APP時，就先讀取最高成績

```
when Screen1.Initialize
do call FirebaseDatabase1.GetValue
    tag "highscore"
    valueIfTagNotThere ""

when FirebaseDatabase1.GotValue
tag value
do set Score_Label.Text to get value
```

The image shows two blocks of Swift code. The first block is a 'when' block for 'Screen1.Initialize'. Inside the 'do' block, there is a 'call' block for 'FirebaseDatabase1.GetValue'. This 'call' block has two arguments: 'tag' with the value 'highscore' and 'valueIfTagNotThere' with the value '' (empty string). The second block is a 'when' block for 'FirebaseDatabase1.GotValue'. It has a 'tag' block with the value 'value'. Inside the 'do' block, there is a 'set' block for 'Score\_Label.Text' to 'get value'.

# 正常的話APP應該會像這樣

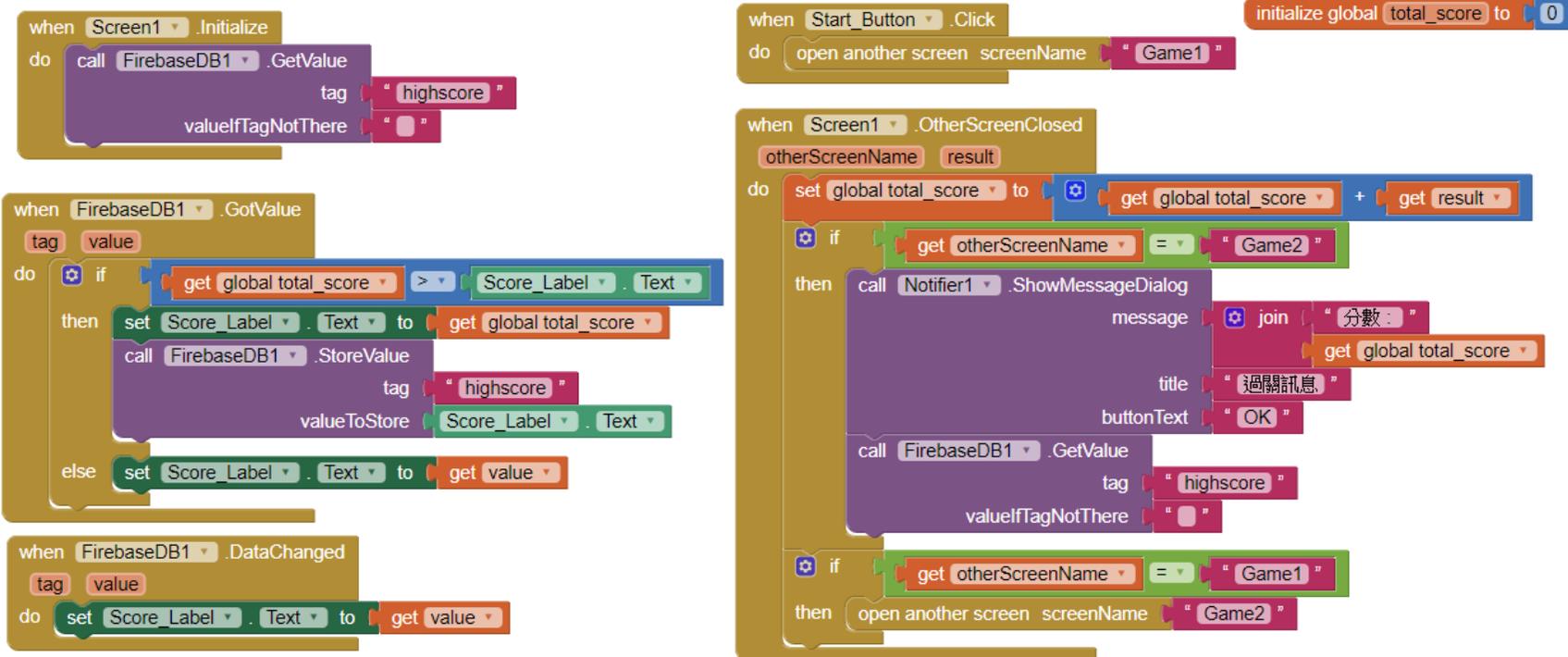
最高分應該是1分。



# 加入判斷最高分的程式

```
when Screen1 .OtherScreenClosed
  otherScreenName result
do
  set global total_score to get global total_score + get result
  if get otherScreenName = "Game2"
  then
    call Notifier1 .ShowMessageDialog
      message join "分數：" get global total_score
      title "過關訊息"
      buttonText "OK"
    call FirebaseDB1 .GetValue
      tag "highscore"
      valueIfTagNotThere ""
    if get global total_score > Score_Label . Text
    then
      set Score_Label . Text to get global total_score
  if get otherScreenName = "Game1"
  then
    open another screen screenName "Game2"
```

# 全部程式 (Screen1)



The image displays three distinct code snippets for a screen named 'Screen1', likely created using a visual programming tool like Xcode or Swift Playgrounds. The snippets are as follows:

- Initialization:** A 'when Screen1 .Initialize' block that calls 'FirebaseDB1 .GetValue' with a tag of 'highscore' and a 'valueIfTagNotThere' of an empty string.
- Score Update Logic:** A 'when FirebaseDB1 .GotValue' block with a 'tag' of 'value'. It contains an 'if' statement: if 'get global total\_score' is greater than 'Score\_Label . Text', it sets 'Score\_Label . Text' to 'get global total\_score', stores this value in 'FirebaseDB1 .StoreValue' with tag 'highscore', and otherwise sets 'Score\_Label . Text' to 'get value'.
- Real-time Updates:** A 'when FirebaseDB1 .DataChanged' block with a 'tag' of 'value' that simply sets 'Score\_Label . Text' to 'get value'.

On the right side, there are two more snippets:

- Start Button Click:** A 'when Start\_Button .Click' block that initializes a global variable 'total\_score' to 0 and then opens another screen named 'Game1'.
- Other Screen Closed:** A 'when Screen1 .OtherScreenClosed' block that receives 'otherScreenName' and 'result'. It sets 'global total\_score' to the sum of 'get global total\_score' and 'get result'. It then checks if 'get otherScreenName' is 'Game2'. If true, it shows a message dialog with the message '分數: ' + 'get global total\_score', title '過關訊息', and button text 'OK'. It also calls 'FirebaseDB1 .GetValue' with tag 'highscore'. If 'get otherScreenName' is 'Game1', it opens another screen named 'Game2'.

# 全部程式 (Game1)

```
initialize global time1 to 30
initialize global score1 to 30

when Clock1.Timer
do
  set global time1 to get global time1 - 1
  set Time1_Label.Text to get global time1
  set global score1 to get global score1 - 1
  set Score1_Label.Text to get global score1
  if get global time1 = 0
  then call re_start

when OrientationSensor1.OrientationChanged
  azimuth pitch roll
do
  call Balloon_imageSprite.MoveTo
    x Balloon_imageSprite.X - OrientationSensor1.Roll / 5
    y Balloon_imageSprite.Y - OrientationSensor1.Pitch / 5

when Goal_Ball.CollidedWith
  other
do
  set Clock1.TimerEnabled to false
  set OrientationSensor1.Enabled to false
  call Balloon_imageSprite.MoveTo
    x 0
    y 320
  call Player2.Start

when Player2.Completed
do
  close screen with value result get global score1
```

```
when ImageSprite2.CollidedWith
  other
do
  call re_start
```

```
when ImageSprite1.CollidedWith
  other
do
  call re_start
```

```
to re_start
do
  set OrientationSensor1.Enabled to false
  set Balloon_imageSprite.Picture to "boom.png"
  call Player1.Start
```

```
when Player1.Completed
do
  set Balloon_imageSprite.Picture to "ball.png"
  call Balloon_imageSprite.MoveTo
    x 5
    y 10
  set OrientationSensor1.Enabled to true
```

# 全部程式 (Game2)

```
when Clock1.Timer
do
  set global time2 to (get global time2) - 1
  set Time2_Label.Text to (get global time2)
  set global score2 to (get global score2) + 1
  set Score2_Label.Text to (get global score2)
  if (get global time2) = 0
  then call re_start2

initialize global time2 to 30
initialize global score2 to 30

when OrientationSensor1.OrientationChanged
  azimuth pitch roll
do
  call Ballon_ImageSprite.MoveTo
  x (Ballon_ImageSprite.X) - (OrientationSensor1.Roll) / 5
  y (Ballon_ImageSprite.Y) - (OrientationSensor1.Pitch) / 5

when ImageSprite1.CollidedWith
  other
do call re_start2

when ImageSprite2.CollidedWith
  other
do call re_start2

when Goal_Ball.CollidedWith
  other
do
  set Clock1.TimerEnabled to false
  set OrientationSensor1.Enabled to false
  call Player2.Start

to re_start2
do
  set OrientationSensor1.Enabled to false
  set Ballon_ImageSprite.Picture to "boom.png"
  call Player1.Start

when Player2.Completed
do close screen with value result (get global score2)

when Player1.Completed
do
  set Ballon_ImageSprite.Picture to "ball.png"
  call Ballon_ImageSprite.MoveTo
  x 5
  y 10
  set OrientationSensor1.Enabled to true
```

# 完成畫面



有最高分的  
紀錄了！

# 最高分資料也正確寫入了



https://app-inventor2-5a238.firebaseio.com/

您的安全性規則定義成公開狀態，因此任何人都能讀取或寫入您的資料庫 [瞭解詳情](#) [關閉](#)

```
app-inventor2-5a238
├── ChatRoom
│   └── tagMessage: "\"Chih-Hung Wu : How are you\\nChih-Hung Wu"
├── firebase
│   ├── myData: "\"ABC Wu\"
│   └── mySharedData: "\"Chih-Hung Wu"
├── gamedata
│   └── highscore: "\"51\"
└── mytest
    └── myData: "\"Chih-Hung Wu Good"
```

# 補充說明

---

經過測試後，發現以AI2 Companion方式測試時，每一關的分數無法正確的回傳到主程式。

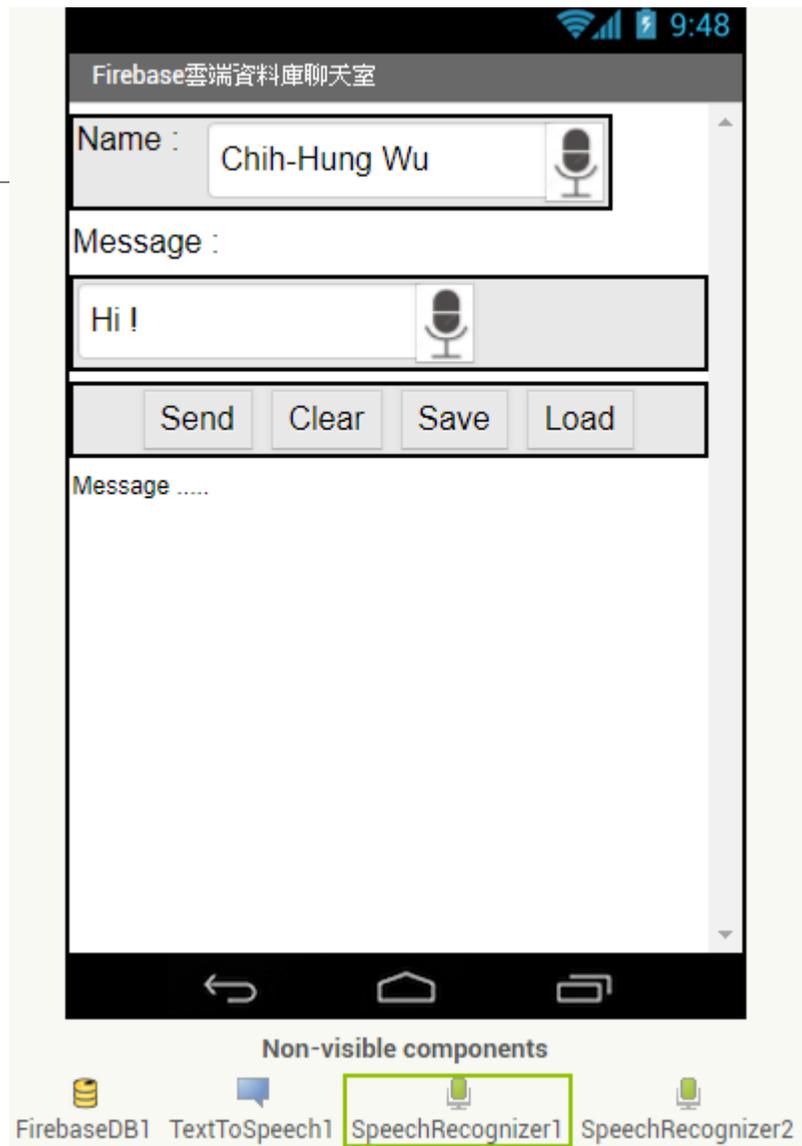
但變成apk之後，安裝在手機後，程式執行結果就正確了！

# 聊天室新增語音辨識功能

---



螢幕設計



新增兩個語音辨識按鈕

新增兩個語音辨識器

# 加入語音辨識功能

---

兩個按鈕的

語音辨識程式碼

```
when Spk_Button1 .Click  
do call SpeechRecognizer1 .GetText
```

```
when SpeechRecognizer1 .AfterGettingText  
result  
do set Name_TextBox .Text to get result
```

```
when Spk_Button2 .Click  
do call SpeechRecognizer1 .GetText
```

```
when SpeechRecognizer2 .AfterGettingText  
result  
do set Msg_TextBox .Text to get result
```

# 加上 訊息 Save & Load功能

---

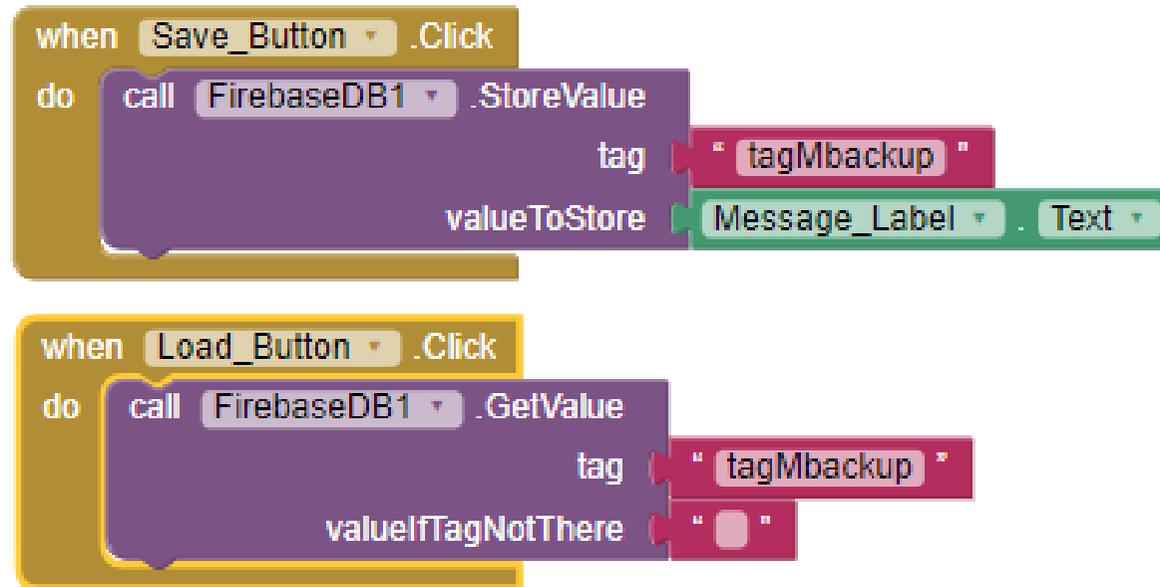
自己試試看應該如何做到吧。



# 新增訊息儲存與載入功能

---

## Save & Load



# 修改資料庫載入部分

新增讀入備份訊息tag=tagMessage\_backup

```
when FirebaseDB1 .GotValue
  tag value
do
  if compare texts get tag = "tagMessage"
  then
    set Msg_TextBox . Text to ""
    call FirebaseDB1 .StoreValue
      tag "tagMessage"
      valueToStore join get global message
                    "
                    get value
  if compare texts get tag = "tagMbackup"
  then
    set Message_Label . Text to get value
    call FirebaseDB1 .StoreValue
      tag "tagMessage"
      valueToStore Message_Label . Text
```

The image shows a Visual Basic code snippet for a Firebase database. The code is structured as follows:

- when FirebaseDB1 .GotValue**: The main event handler.
- tag value**: A variable declaration.
- do**: A loop structure.
- if compare texts get tag = "tagMessage"**: A conditional check.
- then**: A block of code executed if the condition is true:
  - set Msg\_TextBox . Text to ""**: Clears the text box.
  - call FirebaseDB1 .StoreValue**: A function call with two arguments:
    - tag "tagMessage"**: The tag for the stored value.
    - valueToStore join get global message**: A concatenation of the global message and the current value.
- if compare texts get tag = "tagMbackup"**: A conditional check for a backup tag.
- then**: A block of code executed if the condition is true:
  - set Message\_Label . Text to get value**: Sets the text of the label to the current value.
  - call FirebaseDB1 .StoreValue**: A function call with two arguments:
    - tag "tagMessage"**: The tag for the stored value.
    - valueToStore Message\_Label . Text**: The text of the message label.

The second **if** block is highlighted with a red dashed border.

# 全部程式

initialize global message to ""

when Send\_Button .Click  
do call send\_message

to send\_message  
do set global message to join Name\_TextBox .Text  
Msg\_TextBox .Text  
call FirebaseDatabase1 .GetValue  
tag tagMessage  
valueIfTagNotThere ""

when Clear\_Button2 .Click  
do call FirebaseDatabase1 .StoreValue  
tag tagMessage  
valueToStore ""  
set Message\_Label .Text to ""

when FirebaseDatabase1 .GetValue  
tag value  
do if compare texts get tag = tagMessage  
then set Msg\_TextBox .Text to ""  
call FirebaseDatabase1 .StoreValue  
tag tagMessage  
valueToStore join get global message  
"\n"  
get value  
if compare texts get tag = tagMbackup  
then set Message\_Label .Text to get value  
call FirebaseDatabase1 .StoreValue  
tag tagMessage  
valueToStore Message\_Label .Text

when Spk\_Button1 .Click  
do call SpeechRecognizer1 .GetText

when SpeechRecognizer1 .AfterGettingText  
result  
do set Name\_TextBox .Text to get result

when Spk\_Button2 .Click  
do call SpeechRecognizer2 .GetText

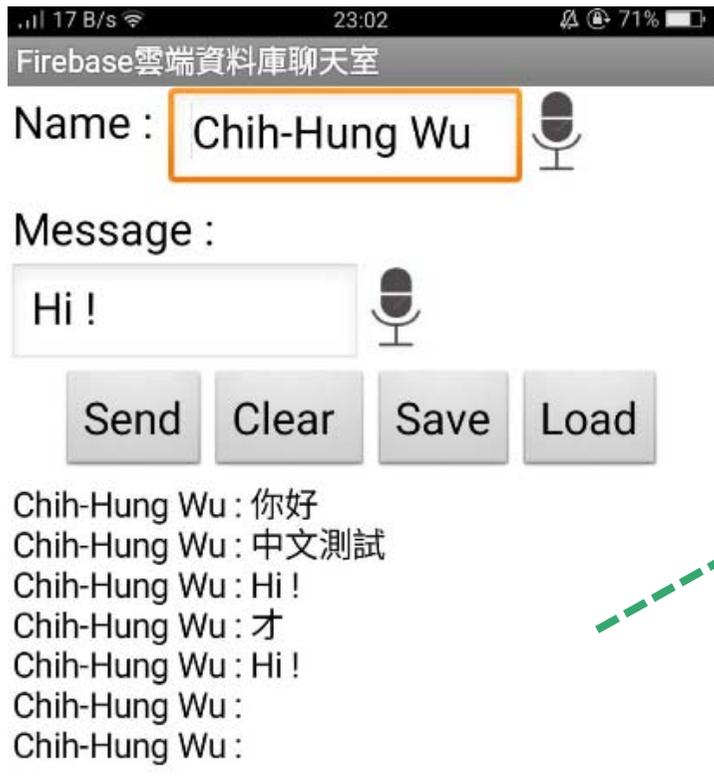
when SpeechRecognizer2 .AfterGettingText  
result  
do set Msg\_TextBox .Text to get result  
call send\_message

when Save\_Button .Click  
do call FirebaseDatabase1 .StoreValue  
tag tagMbackup  
valueToStore Message\_Label .Text

when Load\_Button .Click  
do call FirebaseDatabase1 .GetValue  
tag tagMbackup  
valueIfTagNotThere ""

when FirebaseDatabase1 .DataChanged  
tag value  
do if compare texts get tag = tagMessage  
then set Message\_Label .Text to get value

# 存入firebase的資料表結構與內容



Firestore

```
app-inventor2-5a238
└── ChatRoom
    ├── tagMbackup: "\"Chih-Hung Wu : 你好\\nChih-Hung Wu : 中文測試\\nCh
    ├── tagMessage: "\"Chih-Hung Wu : 你好\\nChih-Hung Wu : 中文測試\\nCh
    └── tagMessage_backup: "\"\""
```

