

# 中國文化大學教師教學創新暨教材研發獎勵成果報告書

## 壹、計畫名稱

物理－資訊跨學科場域學習與教學之教材研發

## 貳、實施課程、授課教師姓名

基礎科學跨學科教學之設計與實踐(四)、黃信健

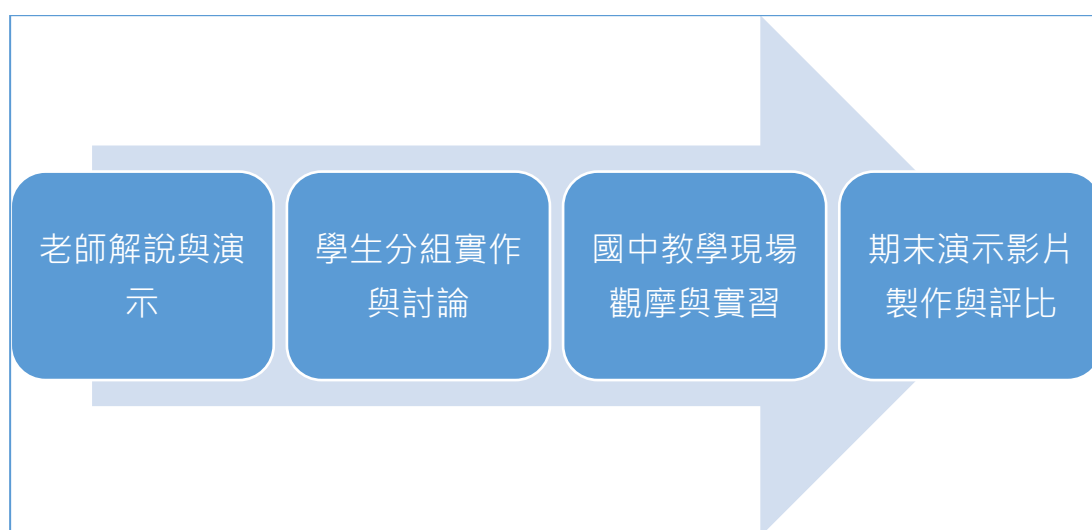
## 參、前言

本計畫的精神是延續我們歷年來致力於協助提振國高中物理教育的成果，進一步帶領物理及其他合作系所的同學們進入場域，以提升其敘事能力與專業能力，並培養利他精神。我們的場域包含大學校園及都會與偏鄉的國高中，在大學校園方面，我們希望讓跨院系的學生接觸超酷物理遊戲、促成校園科普推展。而對都會國高中而言，有關自然與生活科技的跨領域教學對國高中老師實是沉重的負擔，在此方面，我們能夠提供以物理為基礎的跨領域支援；至於偏鄉則更需要各種教育資源之挹注。此外，面對日益嚴重的少子化浪潮，大學端莫不鼓勵各系所與高中建立密切的合作關係，而高中端也更加重視如何善用大學資源，本計畫恰可扮演支援與協助相關系所的角色，共同推動與高中之跨學科實質交流，並在開發跨領域議題、擴大計畫影響力等方面，獲得很好的回饋。事實上，我們的計畫已推動多年，但以往課程內容大多仍以物理為主體，直到近年來我們逐步建構了跨物理、資訊傳播、地理及化學領域的教師社群，足以利用本校系所完整多樣的優勢，開始著力於跨學科的基礎科學學習與教學模組之發展。在本期計畫中，我們將以開發物理-資訊跨學科學習與教學模組為目標，在基礎文書軟體Google slides、Google Docs及Google sheets融入Javascript程式設計，藉此提升學生基本及專業的程式設計及實務能力，並透過服務學習場域培養自主學習與創新創意能力。

## 肆、計畫特色及具體內容

本計畫的主要創新處在於基礎文書軟體 Google slides、Google Docs 及 Google sheets 之融入 Javascript 程式設計，其優點有四：第一，藉由學生熟悉的簡報、文件及試算表中內建的指令碼編譯器來示範目前最常用的 Javascript 程式語言，因此免除安裝其他程式編譯器的麻煩；第二，由學生熟悉的文書應用著手，進而以簡單的程式提升其功能，排除一般學生對於編寫程式的恐懼，也能立即看到執行程式的成果，獲得

立即的回饋；第三，課程採用逐步示範、分段完成的方式進行，能提高學生的專注度與學習效率；第四，課程採用具有極大選擇彈性的題材，讓學生易於觸類旁通，發展出自己的創意，提高及成就感。因此，我們將能透過本教材，讓學生在課堂先學會及演練文書應用融入程式設計的操作與實務、繼而發揮創意、開發新的應用，達成自主學習與創意學習，最終能在服務學習場域靈活發揮所學、指導國高中生及跨科系的同學，從中培養利他與服務精神，而整個培訓過程，也將為學生在開發應用數位技的領域奠定基礎。本計畫教材包含 Google suite 中 Google slides、Google Docs 及 Google sheets 的 Javascript 程式融入應用。其中關於 Google slides 的部分是在簡報內設置一簡報進度條，可自動讀取簡報頁數呈現簡報進度，此單元需時約 45–90 分鐘。至於 Google Docs 則是設置了一個自動編譯器，可在文件內找出特定贅字，並加上底色標示，此單元需時約 4–6 小時。而在 Google sheets 的部分是從特定欄位讀取地址資料、標示於地圖上，再以附件形式由 Gmail 寄出，此單元需時約 45–90 分鐘。這三個模組都具有極大的彈性與擴充性，可以讓學生自行發揮創意，在各服務場域中進行跨領域學習與教學。此外，本計畫採行經驗證有效的國高中教學現場服務學習方案，使學生的學習過程包含老師解說與演示、學生分組實作與討論、國中教學現場觀摩與實習以及期末演示影片製作與評比四個階段(圖一)。在此流程下，學生被投進一個主動的動態學習迴圈，他們必須向老師、同學及國中小朋友證明自己四年所學，因此展現出與其他靜態課堂完全不一樣的主動學習態度。



圖一：「基礎科學跨學科教學之設計與實踐」課程的學習流程

本計畫主要在設計一套可供物理系四年級學生用於服務學習場域，協助國高中生提升科技領域之資訊科技水準的教材。有別於我們在 1092 學期「基礎科學跨學科教學之設計與實踐 IV」課程所教的樂高機器人圖形化程式，在此我們採用進階的 Javascript 程式，並透過 Google suite 內建的指令碼編譯器來執行，以此賦予學生熟悉的簡報、文件及試算表等文書應用更為強大的功能。藉此，學生將可體驗執行程式能帶給他們習以為常的文書處理作業始末料及的強大功能，而藉由我們設計教材的逐步演示引導，也可降低學生學習程式的高門檻，從而提升學習意願與成效。

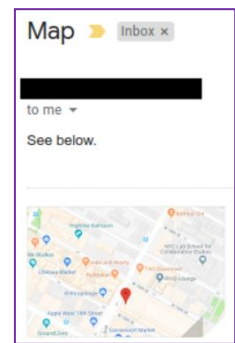
選擇 Google suite 的主要原因除了它是免費的之外，還在於 Google app script 其實是基於 Javascript，其在學生最常接觸的網頁應用方面，較之於 Microsoft Office 所採用的 Visual Basic script 更為普遍。在 Google suite 之中，我們先選擇了 Google slides、Google Docs 及 Google sheets 這三項最主要、也最普遍的應用。關於 Google slides 的部分是在引導學生進入 Google 簡報後，開啟 script editor，在簡報內設置一簡報進度條，如圖二。這個 script 雖算簡單，但會用到序列化(sequencing)、陣列(arrays)、迴圈(loops)、函式(functions)這四個程式設計的主要概念，又可讓學生在指導者逐步引領之下完成操作，是一個很適合程式初學者的學習單元，需時約 45 – 90 分鐘，由於程式的彈性，學生學會後可自由選擇不同的圖形與顏色，增加學習趣味與成就。至於 Google Docs 則是利用 script 針對英文中常見的贅字進行搜尋，再加上底色標示，如圖三；接著，把這個操作序列設置為函式，就可以針對其他贅字同時搜尋，再加上不同的底色。這個文件應用很適合兼顧提升學生英文能力的跨域課程及強調雙語教學特色的國高中，此單元需時約 4 – 6 小時。而在 Google sheets 的部分是從特定欄位讀取地址資料，再標示於地圖上，最後以附件形式由 Gmail 寄出，如圖四。這個單元除了呈現試算表處理資料的特色，還整合了地圖及 email 的應用，很適合於大量數據的進階處理，如客戶維護及中大型會議管理等，整個單元需時約 45 – 90 分鐘。



圖二：簡報進度條



圖三：自動編譯器



圖四：寄送地圖

Google 推廣應用數位技(Google applied digital skills)多年，建有充足的教學影片及自學資料庫，適合學生補強資訊及英文能力。因此我們在其中挑選簡報、文件及試算表融入 Javascript 的單元編寫教材，如表一。對物理系的大四學生來說，他們有 Visual Fortran 的基礎，學習 Javascript，恰可觸類旁通，加強程式能力；而對其他初學之學員來說，廣泛採用於網頁應用的 Javascript 也更易上手，更為風行。

表一：Google 數位應用技教材目錄

序號	教材中文名稱	教材英文名稱	課程時間
1	Google Slides: 簡報進度條	Google Slides: Program a progressive bar	45 - 90 minutes
2	Google Docs: 自動編譯器	Google Docs: auto editor	4 - 6 hours
3	Google Sheets: 寄送地圖	Google Sheets: Hello maps!	45 - 90 minutes

從 1051「物理敘事與演示 I」到 1092「基礎科學跨學科教學之設計與實踐 IV」，我們主要是為國中的物理及自然學科教學提供教學協助，補其不足，也為我們參與服務學習的同學提供訓練敘事力、發揮利他精神的教學場域。但從 1092 學期開始，我們改為針對資訊科技領域規劃與執行行動計畫，借重夥伴學校以特色計畫採購的樂高教育機器人進行合作，一方面協助他們補強師資方面不足以照顧到七、八年級十六個班近 500 位學生的困難，另方面更是為我們物理系的服務學習及專題教育開闢新方向。就服務學習而言，以往我們大部分與物理、自然學科合作，但這次則是資訊學科，對重視資訊知能的計算物理領域而言，深具意義。在專題教育面，物理系已有利用 Visual Fortran 及 EJS 於跨域整合應用的兩個專題組，引入 Google 應用數位技正可提供另一個

足以讓學生發揮所學的新舞台。為配合夥伴學校的課程安排，及節省師生舟車勞頓，我們是以機動調課方式實施校外教學，並集中於一週內密集進行。最後。我們還安排了為期三週的華岡校園策展，也是機動調課方式於三天內密集進行，除作為教學成果期末策展，給予學生發揮所學的舞台，也希望為校園內跨科系的同學引入不同的視角與體驗。1092 學期我們在 4.12-4.14 實施中山國中七、八年級的場域實踐、在 4.27-4.29 實施「樂高機器人華岡校園策展：Google 應用數位技」，兩者都大受歡迎，如圖五-圖八。因此，1101 學期我們也將採用相同模式實施，表二為 1092 學期「基礎科學跨學科教學之設計與實踐 IV」實施之課程表。



圖五：1092 中山國中帶動中小場域實踐



圖六：助教指導學生分組實作



圖七：華岡校園策展- Google 數位應用技



圖八：跨院系學生參加數位技闖關活動

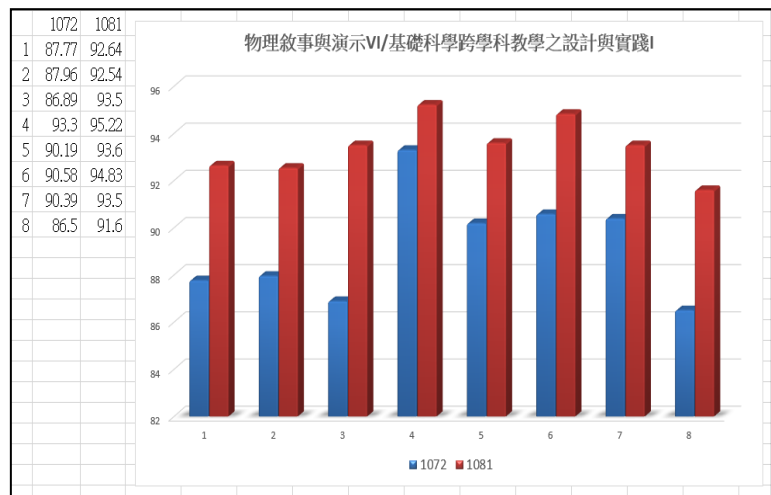
表二：基礎科學跨學科教學之設計與實踐(三)課程表

週次	內容
1	基礎科學跨學科教學教材教法－ Google Slides: 簡報進度條
2	基礎科學跨學科教學教材教法－ Google Docs: 自動編譯器 I
3	基礎科學跨學科教學教材教法－ Google Docs: 自動編譯器 II
4	基礎科學跨學科教學教材教法－ Google Sheets: 寄送地圖
5	基礎科學跨學科教學教材教法一場域一:中山國中七年級
6	基礎科學跨學科教學教材教法一場域二:中山國中七年級
7	基礎科學跨學科教學教材教法一場域三:中山國中七年級
8	基礎科學跨學科教學教材教法一場域四:中山國中七年級
9	期中成果報告
10	基礎科學跨學科教學教材教法一場域五:中山國中八年級
11	基礎科學跨學科教學教材教法一場域六:中山國中八年級
12	基礎科學跨學科教學教材教法一場域七:中山國中八年級
13	基礎科學跨學科教學教材教法一場域八:中山國中八年級
14	基礎科學跨學科教學教材教法一場域九:華岡校園策展
15	基礎科學跨學科教學教材教法一場域十:華岡校園策展
16	基礎科學跨學科教學教材教法一場域十一:華岡校園策展
17	期末成果報告
18	期末考週

本計畫是我們持續多年努力成果的具現，因此從 105-107 學年的「物理敘事與演示 I-VI」系列，一直到 108 學年的「基礎科學跨學科教學之設計與實踐 I-II」系列，都有一些推動的方法和成果可以參照。例如由教師教學學生意見調查表(附件二)結果可看出學生對創新學模式的接受度及學習成效，由場域參與回饋問卷(附件三)可看出參與者對課程與活動的綜合評價。以下表三是 1061-1072「物理敘事與演示 III」-「物理敘事與演示 VI」課程及 1081「基礎科學跨學科教學之設計與實踐 I」課程的調查結果，足以顯示學生對課程評價明顯高出其他課程。右邊圖四的直方圖則呈現我們在中山國中場域所獲得的回饋。

學期	總參考 評分	校平均 參考評分
----	-----------	-------------

1061	93.8	89.9
1062	95.2	90.3
1071	94	89.75
1072	94.8	90.01
1081	92.6	89.51



表三: 1061-108 教學意見調查

圖四：1072-1081 中山國中場域回饋表

前述回饋問卷前四題是關於學員對各場四項實驗滿意度的個別調查，後四題則分別為對老師的講解、助教的協助、關於課程時間安排，及對未來學習的幫助。由圖中可看出各項滿意度百分比都在八成六以上，且 1081 的各項分數都高於 1072，顯示我們在選材及操作策略方面，都能汲取經驗、持續改進之成果。本計畫將繼續採用這些措施，不僅讓計畫能順利執行，並希望獲得充分的回饋，以確保良好的動態修正與改善機制及執行成效。至於 1092 學期則因新冠肺炎疫情再起，幸好我們超前佈署，在五月前已完成所有場域活動；相對於 1091 順應疫情而把場域活動延遲至六月畢業考後方能實施，對於情勢之靈活掌控，實堪稱道。

#### 伍、實施成效及影響（量化及質化，且說明是否達到申請時所期之學習目標與預期成效）

1. 藉由具備高度應用趣味性與啟發性的數位技能教材以提升學生學習興趣與學習成效。
2. 提升學生基本與專業的程式設計及實務能力。
3. 在設計不同的簡報、文件與試算表呈現的過程中，落實學生創新創意能力之培養。
4. 強化學生計算思維敘事與演示的能力。
5. 透過國中資訊領域教學現場的合作與帶動，協助解決國中資訊教學的困難、提高學習成效。
6. 透過物理科學營隊的辦理與高中進行實質交流，讓學生藉此增進多元敘事力，培養

利他精神，並強化本系與高中密切的合作關係、提升高教市場競爭力。

綜合上述六項，本計畫已圓滿達成預期之學習目標與預期成效。

## 陸、 結論

本計畫藉由設計十三篇樂高圖形化程式操控智慧型機器人的教材，帶領學生透過國高中教學現場及大學跨學門全人學習場域的服務學習，培養其敘事力與利他心，在提升教學成效之同時，也帶動合作國高中的資訊科學教學，並協助其國際交流活動，對於鞏固與各校的合作關係、因應少子化浪潮，也發揮了正面的作用，此外，在華岡校園場域中面對跨學門學生時，我們也發現這套Google應用數位技教材及教法的吸引力，及其在計算思維融入學習活動中，所能發揮的巨大影響力。

## 柒、 附件：

講義一：「Google Slides: 簡報進度條教材」



講義二：「Google Docs: 自動編譯器教材」

講義三：「Google Sheets: 寄送地圖教材」

## Program a progressive bar

### 在簡報內設置進度條

1.1 用 Google 簡報新增(new)簡報

1.2 為簡報命名

1.3 新增(add)四頁簡報

1.4 在工具欄(tool)開啟指令碼編輯器(Script Editor)

```
Code.gs x
1 function myFunction() {
2
3 }
```

1.5 輸入變數以儲存你的簡報

```
1 function myFunction() {
2   var slides = SlidesApp.getActivePresentation().getSlides()
3 }
```

1.6 存檔後執行程式(需授權)

```
14 /**
15  * @OnlyCurrentDoc
16  */
```

2.1 在第一頁[0]簡報右下角插入一個太陽圖形(SUN Shape)

```
1 function myFunction() {
2   var slides = SlidesApp.getActivePresentation().getSlides()
3   slides[0].insertShape(SlidesApp.ShapeType.SUN, 0, 395, 10, 10)
4 }
```

2.2 執行



2.3 利用 for 迴圈，插入五個太陽(因有五頁簡報)，但目前只插入第一頁，且重疊

```
function myFunction() {
  var slides = SlidesApp.getActivePresentation().getSlides()
  for (var i = 0; i < slides.length; ++i) {
    slides[0].insertShape(SlidesApp.ShapeType.SUN, 0, 395, 10, 10)
  }
}
```

## 2.4 檢視五個重疊的太陽，再予刪除

### 3.1 利用第二個 for 迴圈，插入二十五個太陽(都重疊在第一頁)

```
1 function myFunction() {
2   var slides = SlidesApp.getActivePresentation().getSlides()
3   for (var i = 0; i < slides.length; ++i) {
4     for (var j = 0; j < slides.length; ++j) {
5       slides[0].insertShape(SlidesApp.ShapeType.SUN, 0, 395, 10, 10)
6     }
7   }
8 }
```

### 3.2 檢視二十五個重疊的太陽，再予刪除

### 3.3 把 0 改成 i，在每頁插入五個太陽(但重疊)

```
1 function myFunction() {
2   var slides = SlidesApp.getActivePresentation().getSlides()
3   for (var i = 0; i < slides.length; ++i) {
4     for (var j = 0; j < slides.length; ++j) {
5       slides[i].insertShape(SlidesApp.ShapeType.SUN, 0, 395, 10, 10)
6     }
7   }
8 }
9
```

### 3.4 把 0 改成 10\*i，讓太陽不重疊

```
1 function myFunction() {
2   var slides = SlidesApp.getActivePresentation().getSlides()
3   for (var i = 0; i < slides.length; ++i) {
4     for (var j = 0; j < slides.length; ++j) {
5       slides[i].insertShape(SlidesApp.ShapeType.SUN, 10*i, 395, 10, 10)
6     }
7   }
8 }
```

### 4.1 利用條件指令，讓第 n 頁簡報有 n 個太陽變成有顏色



### 4.2 顏色用 RGB 值來定

```
1 function myFunction() {
2   var slides = SlidesApp.getActivePresentation().getSlides()
3   for (var i = 0;i<slides.length;++i){
4     for (var j = 0;j<slides.length;++j){
5       slides[i].insertShape(SlidesApp.ShapeType.SUN, 10*j, 395, 10, 10)
6       if (j<=i){
7         var shapes=slides[i].getShapes();
8         var lastshape = shapes[shapes.length-1];
9         lastshape.setFill().setSolidFill(230, 152, 18)
10      }
11    }
12  }
13 }
```

## 5.0 改用不同的形狀和顏色

### 自動編譯器(Auto Editor) I

關鍵字: 變數(variable)、迴圈(loop)、函數(functions)

目標: 自動編譯器(Auto Editor) I

1. 利用自動編譯器在文件裡找出某一個贅字(very)
2. 把這個贅字用底色標示
3. 把文件裡所有的這個贅字都用底色標示
4. 把以上功能寫成一個可重複使用的函數(function)

1.1 由以下連結開啟一份 google 簡報 :

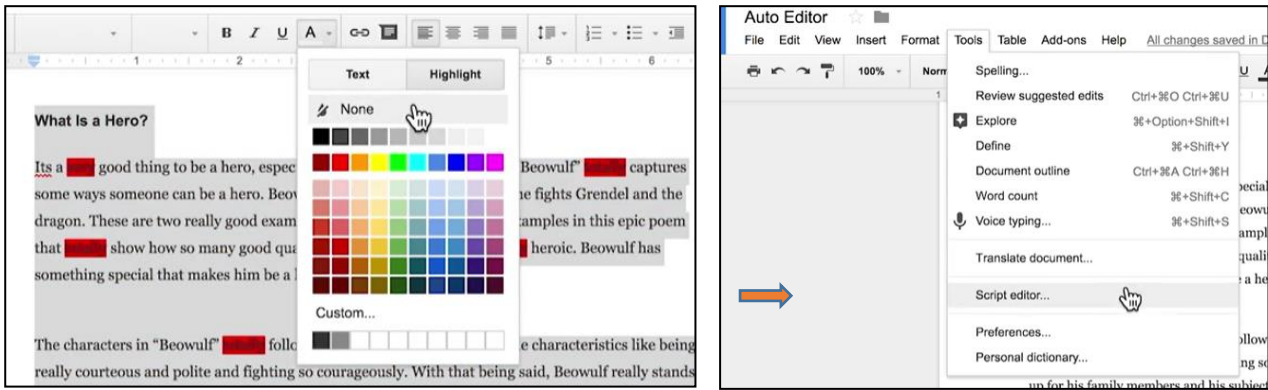
<https://docs.google.com/document/d/1laeUX6KAiVAXZEA0JaOC85AZgNaaoWy3oiCgGavXOco/edit?resourcekey=0-LvCX5KfuuN9OmCokOMp5OQ>

1.2 重新命名 · 例如 Auto Editor

1.3 清除文件內的顏色標示

編輯=>全選





1.4 建立函數：點選指令編譯器，出現如圖之程式框架：

```

Code.gs
1  function myFunction() {
2
3

```

1.5 尋找字串：輸入 DocumentApp. 再點選 getActiveDocument(). 再點選 getBody().

再點選 findText("very")



1.6 執行後，如未打錯字，將順利返回 (return type)

1.7 加入變數 searchResult 以儲存搜尋結果；加入記錄函數 Logger.log( searchResult)以

儲存搜尋結果

```

1  function myFunction() {
2
3      var searchResult
4
5      searchResult = DocumentApp.getActiveDocument().getBody().findText("very")
6
7      Logger.log(searchResult)
8
9  }

```

1.8 檢視搜尋結果： `view=>Logs` 如順利找到，會返回 `RangeElement`，如果找不到，會返回 `null`;



1.9 用雙斜線加入註解：

```
1 function myFunction() {
2   //create the variable to show the results of finding the text
3   var searchResult
4
5   //search for the text stored in the parameter item
6   searchResult = DocumentApp.getActiveDocument().getBody().findText("very")
7
8   //log the results of the search for testing
9   Logger.log(searchResult)
10
11 }
```

2.1 用設定底色 `setBackgroundColor` 函數把文件內的 `very` 都用紅色底色標示：

```
Logger.log(searchResult)

searchResult.getElement().asText().setBackgroundColor(startOffset, endOffsetInclusive, color)
}
```

2.2 設定範圍參數

```
searchResult.getElement().asText().setBackgroundColor(searchResult.getStartOffset(), searchResult.getEndOffsetInclusive(), color)
}
```

2.3 執行後，會出現未設定顏色之錯誤訊息

2.4 在 `color` 處輸入紅色的代碼 “`##FF0000`”

3.1 用 `while loop` 把文件裡所有的 `very` 都用紅色標示：

3.2 把標示紅色之指令加入迴圈：`while (searchResult !== null)` (搜尋結果不為零)

```

1 function myfunction() {
2 //create the variable to show the results of finding the text
3 var searchResult
4 // search for the text stored in the parameter item
5 searchResult = DocumentApp.getActiveDocument().getBody().findText("very")
6
7
8 //log the results of the search for testing
9 Logger.log(searchResult)
10
11 //while the search results for very are not empty do the following
12 while (searchResult != null) {
13
14 //change the background color of the search result text to red
15 searchResult.getElement().asText().setBackgroundColor(searchResult.getStartOffset(), searchResult.getEndOffsetInclusive(), "#FF0000")
16
17 //end of the loop
18 }
19
20
21 }

```

3.3 加上再次搜尋指令，這會把文件裡所有的 very 都用紅色標示

```

function myfunction() {
//create the variable to show the results of finding the text
var searchResult
// search for the text stored in the parameter item
searchResult = DocumentApp.getActiveDocument().getBody().findText("very")

//log the results of the search for testing
Logger.log(searchResult)

//while the search results for very are not empty do the following
while (searchResult != null) {

//change the background color of the search result text to red
searchResult.getElement().asText().setBackgroundColor(searchResult.getStartOffset(), searchResult.getEndOffsetInclusive(), "#FF0000")

//find text again
searchResult = DocumentApp.getActiveDocument().getBody().findText("very", searchResult)

//end of the loop
}
}

```

4.1 把以上功能寫成一個可重複使用的函數(function)，以找尋其他贅字

4.2 把 myfunction 改成 findText

```

1 function findText() {
2 //create the variable to show the results of finding the text

```

4.3 在程式後再加一個函數 highlightProblem

```

25 function highlightProblem(){
26
27 }

```

4.4 加入欲搜尋字串("very" and "totally")

```
function highlightProblem(){
  findText("very")
  findText("totally")
}
```

#### 4.5 把 findText 的字串改成 item

```
1 function findText(item) {
```

#### 4.6 加入 logger.log(item) 執行後，從 view=>log 檢視結果

```
3 Logger.log(item)
```

Logs

```
[17-10-10 11:31:57:834 BST] very
[17-10-10 11:31:57:936 BST] RangeElement
[17-10-10 11:31:57:946 BST] totally
[17-10-10 11:31:57:948 BST] RangeElement
```

#### 4.7 加入註解

```
1 function findText(item) {
2   //add the parameter to the log to test that it is working as expected
3   Logger.log(item)
4
5   //create the variable to show the results of finding the text
6   var searchResult
7   // search for the text stored in the parameter item
8   searchResult = DocumentApp.getActiveDocument().getBody().findText("very")
9
10
11  //log the results of the search for testing
12  Logger.log(searchResult)
13
14  //while the search results for very are not empty do the following
15  while (searchResult !== null) {
16
17    //change the background color of the search result text to red
18    searchResult.getElement().asText().setBackgroundColor(searchResult.getStartOffset(), searchResult.getEndOffsetInclusive(), "#FF0000")
19
20    //find text again
21    searchResult = DocumentApp.getActiveDocument().getBody().findText("very", searchResult)
22
23  //end of the loop
24  }
25 }
26
27
28 function highlightProblem(){
29   //highlight the word "very"
30   findText("very")
31   //highlight the word "totally"
32   findText("totally")
33 }
```

#### 4.8 目前雖有搜尋到 “very” and ”totally”，但只有標示 view，所以要把程式內的 very 也改成 item

#### 4.9 先清除文件內的紅色標示，並更新註解後再執行

```
//find text again that is stored in the item variable
searchResult = DocumentApp.getActiveDocument().getBody().findText(item, searchResult)
```

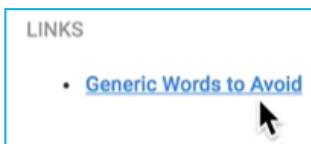
#### 5.1 利用陣列搜尋多個贅字：如果想一次搜尋多個贅字，可重複先前的步驟：

```
function highlightProblem(){
  //highlight the word "very"
  findText("very")
  //highlight the word "totally"
  findText("totally")
  findText("really")
  findText("by")
  findText("with that being said")
  findText("who")
  findText("its")
  findText("it's")
}
```

5.2 如果想要更精簡，並避免出錯，可利用陣列 (words)：把陣列元素加上雙引號，並用[] 刮起來：

```
28 function highlightProblem(){
29   //create an array to store all of the words that will be search for
30   var words = ["very", "so", "totally", "really", "for", "to"]
}
```

5.3 關於應避免的贅字，可參考此連結之清單



Overused Words to Avoid		
a lot	in other words	same exact
any	is	so
are	its (instead of it's for "it is")	some
awesome	just	something
bad	like	thing
be	lots	totally
by the same token	maybe	truly unique
even	nice	ultimately
everything	pretty	very

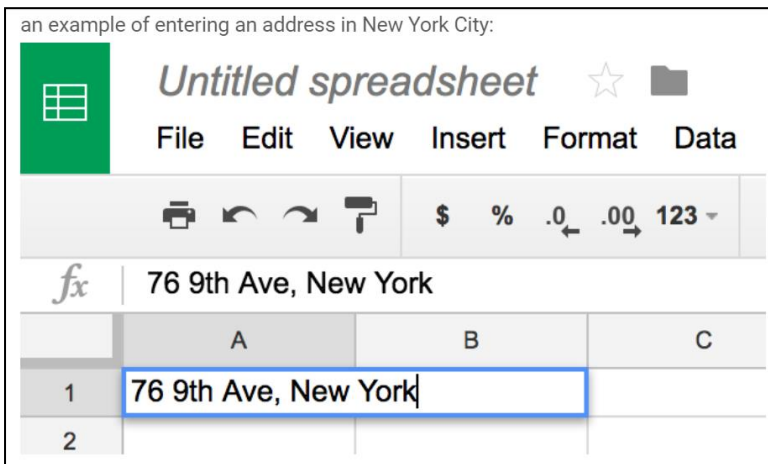


## 地圖快遞(Hello Maps!)

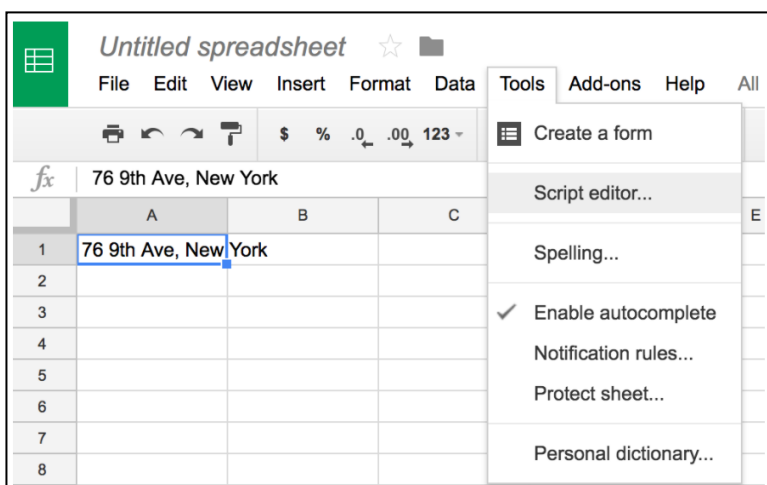
關鍵字: Google Apps Script、Google Sheets、Google Map、Gmail

目標：

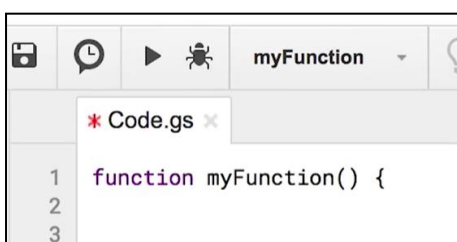
5. 從 Google Sheet 讀取地址
6. 新增(new) 一份空白 Google 試算表
7. 在 A1 欄位輸入一個有效地址，如下圖紐約的地址



8. 在工具欄(tool)開啟指令碼編輯器(Script Editor)



9. 點選指令編譯器，出現如下圖之程式框架：



10. 貼上下圖程式碼，把 `YOUR_EMAIL_ADDR` 改成你的 email，存檔。

```

/** @OnlyCurrentDoc */

function sendMap() {

```

```

var sheet = SpreadsheetApp.getActiveSheet();

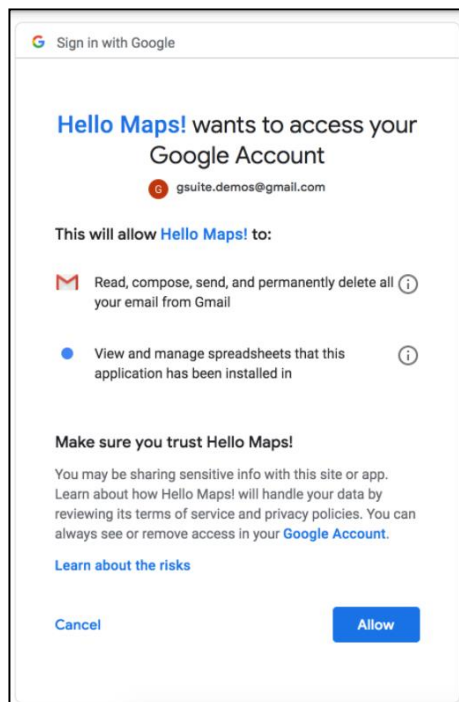
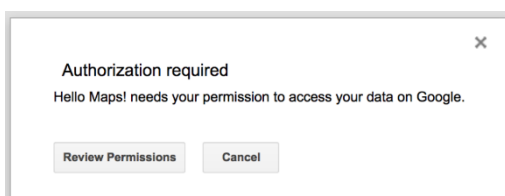
var address = sheet.getRange('A1').getValue();

var map = Maps.newStaticMap().addMarker(address);

GmailApp.sendEmail('YOUR_EMAIL_ADDR', 'Map', 'See below.',
{attachments:[map]});
}

```

## 11. 執行程式，授權。



## 12. Application detailed explanation

1. This is a normal JavaScript function declaration for `sendMap()`.

```
function sendMap() {
```

- The first line of code calls the [Spreadsheet Service](#) accessible from Apps Script via the [SpreadsheetApp object](#). The returned sheet is assigned to a variable of the same name. The [getActiveSheet\(\)](#) method does exactly what it says it does—it returns a "handle" to the current sheet that is active in the user interface (UI).

```
var sheet = SpreadsheetApp.getActiveSheet();
```

- With the [sheet](#) object, reference the cell range (of a single cell) in [A1 notation](#) with [getRange\(\)](#). A "range" is a group of cells, including just a single one like ours... cell [A1](#), the one we entered the address in. Now let's fetch what's *inside* that range of cells with the [getValue\(\)](#) call, and assigned to the address variable upon return. Try adding more addresses and reading from different cells.

```
var address = sheet.getRange('A1').getValue();
```

- The 3rd line of code connects to the Google [Maps Service](#) via the [Maps object](#). As soon as we have access to the Maps Service, we request a new static map be created via [newStaticMap\(\)](#). You can then put a "pin" dropped on the address we pulled from the Sheet by using the [addMarker\(\)](#) method.

```
var map = Maps.newStaticMap().addMarker(address);
```

- The last line uses the [Gmail Service](#) (via the [MailApp object](#)), calling its [sendEmail\(\)](#) method, to send the email which includes both the text "See below." and the map image as an attachment.

```
GmailApp.sendEmail('friend@example.com', 'Map', 'See below.',  
{attachments:[map]});  
  
}
```