

Google Apps Script

Deploy as web app

4 มิ.ย. 2020

เรียบเรียงโดย
วสันต์ คุณติลกเศวต

พิมพ์ครั้งที่ 1 : 4 มิ.ย. 2020

Google Apps Script Deploy as web app

โดย
วสันต์ คุณดิลกเศวต

wasankds@gmail.com

Line ID : wasankds

08-1459-8343

www.poelclub.org

สารบัญ

คำนำ.....	7
บทที่ 1 บทนำ.....	9
บทที่ 2 Deploy as web app.....	13
2.1. สร้างโปรเจ็ค Google Apps Script แบบ Web App	14
2.2. การใช้งานโปรเจ็คแบบ Web App (Deploy as web app)	15
2.3. Web Apps ด้วย Google Apps Script	17
บทที่ 3 doGet() และ doPost().....	19
3.1. doGet() และ doPost()	20
3.1.ก.) Web App คืนค่าเป็น Text (21)	
3.1.ข.) Web App คืนค่าเป็น JSON (21)	
3.1.ค.) Web App คืนค่าเป็น HTML (22)	
3.1.ง.) Web App คืนค่าเป็น HTML โดยใช้เทมเพลต (23)	
บทที่ 4 สาธิตกระบวนการทำงานของ Web App.....	25
4.1. สาธิตกระบวนการทำงานของ Web App	26
4.1.ก.) สร้างโปรเจ็ค (26)	
4.1.ข.) โค้ดในไฟล์ .gs (26)	
4.1.ค.) โค้ดในไฟล์ .html (27)	
4.1.ง.) ใช้งานแบบ Web App (Deploy as web app) (27)	
4.2. ขั้นตอนการทำงาน	28
บทที่ 5 Scriptlets.....	31
5.1. Scriptlets	32
5.2. Standard scriptlet	32
5.3. Printing scriptlet	33
5.4. Force-printing scriptlet	34
5.5. โหลดข้อมูลจาก Google Sheets มาไว้ที่หน้าเว็บ	35
บทที่ 6 เซอร์วิส HTML.....	37
6.1. เซอร์วิส HTML	38
6.2. createHtmlOutput()	38
6.2.ก.) createHtmlOutput() (38)	
6.2.ข.) append() และ getContent() (39)	

6.3. createHtmlOutputFromFile()	40
6.4. คลาส HtmlTemplate	41
6.5. createTemplate() และ evaluate()	41
6.6. createTemplateFromFile()	42
6.6.ก.) ตัวอย่าง : แบนตัวแปรไปกับวัตถุ HtmlTemplate	(42)
6.6.ข.) ตัวอย่าง : การทำ Mail merge email	(44)
บทที่ 7 รันสคริปต์ฝั่ง Server-side	47
7.1. คลาส google.script.run	48
7.2. withSuccessHandler()	48
7.3. withFailureHandler()	49
7.4. withUserObject()	50
บทที่ 8 ใช้งาน Web App ใน iFrame	53
8.1. จับใส่ iframe	54
8.1.ก.) ฝัง iframe ที่หน้าเว็บ	(54)
8.1.ข.) แก้ไขโค้ดในไฟล์ .gs ของโปรเจ็ค	(54)
8.1.ค.) แก้ไขโค้ดในไฟล์ .html ของโปรเจ็ค	(56)
8.1.ง.) setXFrameOptionsMode()	(57)
8.1.จ.) setSandboxMode(mode)	(57)
8.2. จับลิงค์ของโปรเจ็คโดยใช้สคริปต์	58
บทที่ 9 Web App แบบหลายหน้า	61
9.1. เปลี่ยนหน้าโดยเขียนโค้ดเพื่อแทรกโค้ด HTML	62
9.1.ก.) โค้ดของโปรเจ็ค	(62)
9.1.ข.) ไฟล์ที่เรียกใช้ Web App ไปใช้แบบ iFrame	(64)
9.1.ค.) ผล	(64)
9.2. เปลี่ยนหน้าเว็บโดยส่งพารามิเตอร์เพื่อเปลี่ยนหน้า	65
9.2.ก.) โค้ดทั้งหมดของโปรเจ็ค Apps Script	(65)
9.2.ข.) ไฟล์ที่เรียกใช้ Web App ไปใช้แบบ iFrame	(67)
9.2.ค.) ผล	(67)
9.2.ง.) ข้อจำกัด	(67)

บทที่ 10 URLSession.....	71
10.1. URLSession	72
10.2. เมธอดที่สำคัญ	72
10.2.ก.) fetch() (72)	
10.2.ข.) getContent() (73)	
10.2.ค.) getContentText() (73)	
10.2.ง.) getResponseCode() (75)	
10.3. การใช้งาน Fixer API	76
10.3.ก.) ใช้บริการ Fixer (76)	
10.3.ข.) Apps Script พัฒนาการที่ 1 (78)	
10.3.ค.) Apps Script พัฒนาการที่ 2 (78)	
10.3.ง.) ขยายความบรรทัด obj = obj[keyPathArr[i]] (79)	
10.3.จ.) Apps Script พัฒนาการที่ 3 (80)	
10.4. การใช้งาน convertAPI	81
บทที่ 11 JSON.....	85
11.1. JSON คืออะไร	86
11.2. JSON Data Types	87
11.3. ตัวอย่างการเขียน JSON	88
11.3.ก.) ตัวอย่างที่ 1 (88)	
11.3.ข.) ตัวอย่างที่ 2 (88)	
11.3.ค.) ตัวอย่างที่ 2 - ต่อ (89)	
11.4. JSON และ Google Apps Script	90
11.4.ก.) ส่ง HTTP Request (90)	
11.4.ข.) HTTP Post Request (91)	
11.4.ค.) HTTP Get request (93)	
11.4.ง.) ส่ง JSON ติดไปกับ HTTP Request (94)	
11.4.จ.) บันทึก JSON ลง Google Sheets (94)	
11.4.ฉ.) แปลงข้อมูลจากตารางใน Google Sheets เป็น JSON (96)	
11.5. เซอร์วิส Content	97
11.5.ก.) createTextOutput() (97)	

คำนำ

หนังสือเล่มนี้ เป็นหนึ่งในชุด **การเขียนโปรแกรม Google Apps Script** โดยในเล่มนี้อธิบายการสร้าง Web Apps

ผู้เขียน เขียนหนังสือเล่มนี้ จุดประสงค์ดั้งเดิม ก็คือ **เก็บไว้อ่านเอง**

เมื่อผู้เขียนศึกษาเรื่องอะไร ก็จะไปเรียนรู้จากสื่อออนไลน์ในอินเทอร์เน็ต ทั้งคอร์สออนไลน์ วิดีโอ หรือ เอกสาร ทั้งในแบบฟรีและเสียเงิน

ในยุคปัจจุบันเราต้องเรียนรู้อะไรให้เร็ว โดยเฉพาะเรื่องของ IT ผู้เขียนจึงตั้งใจจะดูวิดีโอ หรือแปลเอกสารเพียงรอบเดียว จึงดูไป อ่านไป พิมพ์สรุปไป เวลาจำอะไรไม่ได้ มาดูจากที่พิมพ์สรุปไว้ง่ายกว่าการไปย้อนดูจากวิดีโอ นอกจากนี้ ก็ยังนำมาทบทวนได้ง่าย ในอนาคตสามารถเพิ่มเติมเสริมแต่งเนื้อหาได้เรื่อยๆด้วย

หนังสือเล่มนี้ ความตั้งใจดั้งเดิมของผู้เขียน ก็คือ **ตั้งใจเก็บไว้อ่านเองคนเดียว เหตุเพราะนำเนื้อหามาจากหลายแหล่ง แม้ผู้เขียนจะเขียนเพิ่มไปด้วยก็ตาม**

อย่างไรก็ดี อดสำหรับพิมพ์ไว้เป็นหนังสือแล้ว จะเก็บไว้อ่านคนเดียวก็รู้สึกเสียดาย ผู้เขียนจึงนำมาแบ่งปัน

เนื่องด้วย ผู้เขียนให้ความสำคัญกับประเด็นด้านลิขสิทธิ์มาก ฉะนั้นจึงขอแจ้งไว้ ณ ที่นี้ ตั้งแต่ต้นก็คือ

1. เนื้อหาในหนังสือ ผู้เขียนรวบรวมมาจากแหล่งต่างๆในอินเทอร์เน็ต ซึ่งจะพยายามให้มากที่สุด ที่จะบอกลิงค์หรือแหล่งที่มาในแต่ละหัวข้อ เพราะหนังสือเล่มนี้ถูกเขียนไว้นานแล้ว บางเรื่องลืมก็อปปีลิงค์มาแปะไว้
2. ผู้เขียนรวบรวมเนื้อหาจากหลายแหล่ง และเพิ่มเติมลงไปด้วย
3. หนังสือเล่มนี้แจกฟรี ผู้เขียนไม่มีรายได้จากหนังสือเล่มนี้

หนังสือเล่มนี้ยังไม่จบเสียทีเดียว หากผู้เขียนว่าง จะมาเขียนเพิ่มเติมเรื่อยๆ ให้ดูเวอร์ชันตามวันที่ที่ปล่อยหนังสือ

(ผู้เขียนไม่มีเวลาตรวจทาน หากมีข้อผิดพลาดประการใด ขออภัยไว้ ณ ที่นี้ด้วย)

วสันต์ คุณดิลกเศวต
wasankds@gmail.com
081-459-8343
Line ID : wasankds

บทที่ 1

บทนำ



จริงๆแล้วเนื้อในส่วนนี้ ควรจะอยู่ในคำนำ แต่เพราะผู้เขียนไม่อยากให้ทุกท่านเข้าไป จึงยกมาไว้ในบทที่ 1 ทั้งนี้ เพราะอยากจะทำให้เข้าใจเครื่องมือในการทำ Web Apps ก่อน ก่อนที่จะลุยเข้าไปอ่านเนื้อหาเกี่ยวกับเครื่องมือต่างๆในการพัฒนา Web Apps

Web App เป็นเว็บไซต์ที่มีการใช้งานเสมือนเป็นโปรแกรมคอมพิวเตอร์ตัวหนึ่ง เพียงแต่ใช้งานออนไลน์ มีการปฏิสัมพันธ์กับผู้ใช้งาน มีการบันทึก และโหลดข้อมูล เป็นต้น

การทำ Web App ต้องมีความรู้ที่กว้างและลึกในบางส่วน เพราะต้องใช้เครื่องมือหลายตัว ส่วนจะลึกแค่ไหนก็ขึ้นอยู่กับว่า เราจะทำ Web App เพื่อใช้งานอะไร ฉะนั้นถ้าทำคนเดียวก็จะหนักหน่อย

ในอดีตหลายปีก่อน ผู้เขียนเรียนทำ Web App โดยใช้เครื่องมือดังนี้ **PHP, Javascript, HTML, CSS** และ **MySQL** ซึ่งก็เรียนรู้มาอย่างละนิดละหน่อย เครื่องมือต่างๆในตอนนั้นก็เปลี่ยนไปจากเมื่อก่อนมาก Javascript สมัยก่อนกับสมัยนี้ต่างกันมาก สมัยนี้ Javascript เป็น OOP ที่เก่งมาก MySQL ก็ไปเกิดใหม่เป็น MariaDB แล้วเครื่องมืออื่นๆก็พัฒนาไปไกลไม่แพ้กัน

นี่ยังไม่รวมถึงโปรแกรมอื่นๆ และ ความสามารถอื่นๆ ที่เราต้องเป็นด้วย อย่างเช่น โปรแกรม Visual Studio, Dreamweaver ต้องทำกราฟิกได้ระดับหนึ่ง ต้องใช้โปรแกรม PhotoShop หรือ Gimp ได้ เป็นต้น

การทำ Web App จึงต้องมีความรู้ที่กว้างและลึกจริงๆ จึงไม่แปลกเลย ถ้าเราจ้างทำ Web App แล้ว ราคาก็จะสูง

หลังจากท่วงเห็นการทำ Web App ไปนาน ปัจจุบัน ผู้เขียนหันมาทำ Web App อีก แต่เครื่องมือเปลี่ยนไป

ผู้เขียนใช้ **Google Apps Script** เป็นสคริปต์ในฝั่ง **Server-side** แทน PHP และ ใช้ **Google Sheets** เป็นตัวเก็บข้อมูล แทน MySQL ส่วนเครื่องมือที่เหลือก็เหมือนเดิม **HTML, CSS** และ **Javascript** สำหรับโปรแกรมเสริมอื่นๆที่ใช้ในการพัฒนา Web App ก็หันมาใช้โอเพ่นซอร์สหรือซอฟต์แวร์ฟรีทั้งหมด เช่น LibreOffice, Visual Studio, Ubuntu, Gimp, Inkscape, Blender เป็นต้น

มาดูข้อดี ข้อเสีย ของเครื่องมือของ Google ที่ผู้เขียนเลือกใช้ในการทำ Web App

Google Sheets ที่ใช้เป็นตัวเก็บข้อมูล แม้จะเก็บข้อมูลไม่ได้มากมาย(จำกัดที่ 5 ล้านเซลล์) ประมวลผลไม่รวดเร็วเหมือน MySQL เพราะ **Google Sheets** ไม่ใช่โปรแกรมฐานข้อมูล แต่เป็นแอปตารางคำนวณ อย่างไรก็ตาม การใช้ **Google Sheets** นั้น เพียงพอต่อระบบงานที่ไม่ใหญ่โตนัก มีข้อดีที่ เข้าใจง่าย มองเห็นข้อมูลชัดเจน แก้ไขข้อมูลได้โดยตรง ไม่ต้องมีความรู้ด้านฐานข้อมูล ถ้าเคยใช้ Excel หรือ Calc มาก่อน จะใช้งาน **Google Sheets** ได้ไม่ยาก เพราะเครื่องมือคล้ายกัน

นอกจากนี้ ระบบของ Google นั้นใช้งานบนคลาวด์ ออนไลน์ เสถียร และ มีความปลอดภัยสูง มีจุดเด่นที่ใช้งานไฟล์ร่วมกันได้หลายยูสเซอร์พร้อมกัน และกำหนดสิทธิการใช้งานได้ด้วย

ที่สำคัญ [Google Sheets](#) และ [Google Apps Script](#) [ใช้งานได้ฟรี](#) [ไม่ต้องติดตั้งซอฟต์แวร์ใดๆ](#) เพียงแค่ Sign-in ด้วยบัญชี Google ไม่ว่าจะเป็น Gmail หรือ G Suite(มีทั้งฟรีและเสียเงิน) ก็ใช้งานได้แล้ว อย่างไรก็ตาม ก็ยังมีข้อจำกัดตามประเภทของบัญชี Google

เนื่องจากการทำ Web App ต้องมีความรู้ที่กว้างและลึก ฉะนั้น ในหนังสือเล่มนี้ ผู้เขียนจึงเน้นไปที่การอธิบายให้ครอบคลุมระบบโดยรวม และลงลึกในบางเรื่อง

พื้นฐานของบางเรื่อง อย่างเช่น Javascript ผู้เขียนไม่ได้อธิบายเลย ทั้งนี้ เพราะมีโครงสร้างการเขียนโปรแกรมเหมือนกับ Google Apps Script จึงไม่จำเป็นต้องปูพื้นฐานอีก ทุกคนสามารถหาอ่านได้จากหนังสือ ["หลักการเขียนโปรแกรม Google Apps Script"](#) เพียงแต่เราต้องมาเรียนรู้วัตถุดิบใหม่ๆ ของ Javascript เท่านั้นเอง เช่น วัตถุ document, NodeList, HTMLCollection เป็นต้น

สำหรับ Javascript ผู้เขียนจะเน้นไปที่การควบคุมวัตถุ HTML DOM Elements ซึ่งก็คือ แท็ก HTML นั่นเอง เนื่องจากใช้บ่อยในการทำ Web App

สำหรับ HTML ผู้เขียนก็ได้ปูพื้นฐานเหมือนกัน เพราะ HTML นั้นไม่ยาก โดยผู้เขียนจะเน้นไปที่แท็ก HTML สำหรับสร้างฟอร์มมากกว่า

สำหรับ CSS ผู้เขียนเน้นไปที่ การทำความเข้าใจและวิธีการใช้งาน ถ้าจะทำ Web App สักตัว ผู้เขียนจะใช้เครื่องมืออย่าง [BootStrap](#) หรือ [Materialize](#) ช่วยในการตกแต่งหน้าเว็บมากกว่า ที่จะมาออกแบบ CSS เอง

ขอให้ทุกท่านประสบความสำเร็จในการสร้าง Web Apps

บทที่ 2

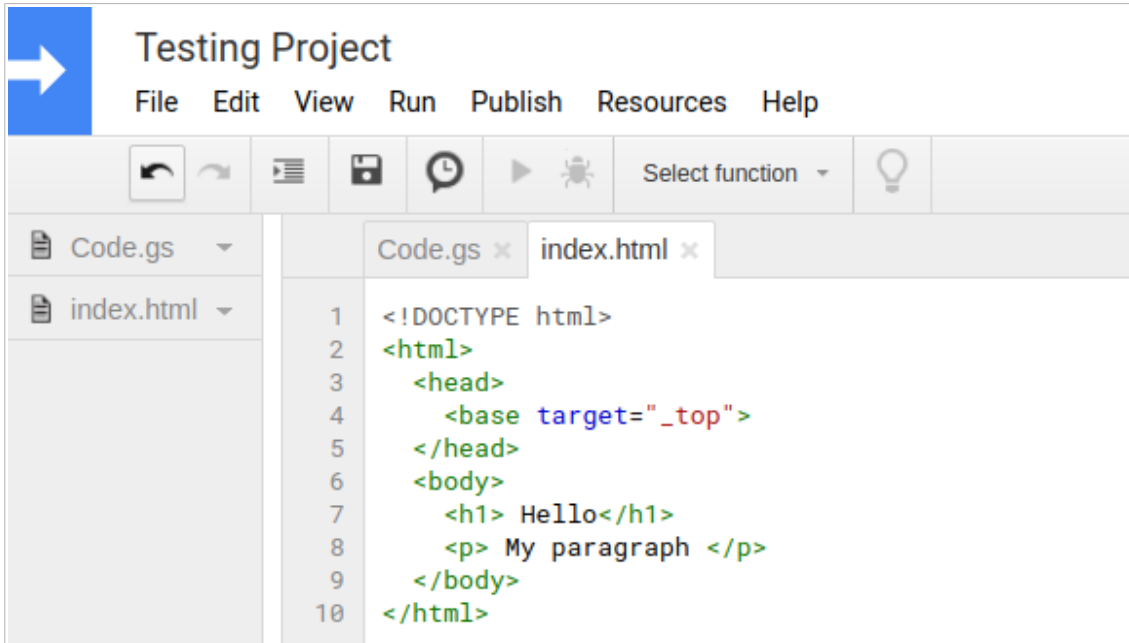
Deploy as web app



2.1. สร้างโปรเจ็ค Google Apps Script แบบ Web App

โปรเจ็ค Google Apps Script สามารถใช้งานแบบ Web App ได้ (Deploy as web app) ฉะนั้นเพื่อให้เข้าใจภาพรวมในเบื้องต้น ในบทนี้เราจะสร้างโปรเจ็คง่ายๆ แล้วใช้งานแบบ Web App

ก่อนอื่นๆ สร้างโปรเจ็ค Google Apps Script แบบ Standalone ที่มี 2 ไฟล์ ก็คือ [page.html](#) (File → New → HTML file → ตั้งชื่อไฟล์) และ [code.gs](#) (มีมาให้แล้วตั้งแต่ต้น) โดยแต่ละไฟล์ มีโค้ดดังต่อไปนี้



ไฟล์ [index.html](#) – เป็นหน้าต่างของ Web App

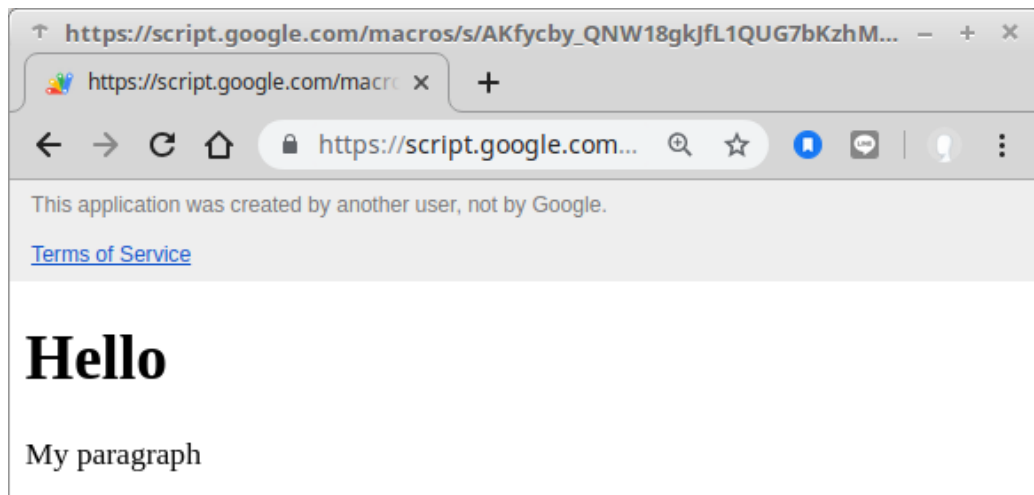
```
<html>
  <head>
    <base target="_top">
  </head>
  <body>
    <h1> Hello</h1>
    <p> My paragraph </p>
  </body>
</html>
```

ไฟล์ [code.gs](#) – เป็น Server-side script

```
function doGet(e) {
  Logger.log(e) ; // ดูผลที่ Logs ----- >
  return HtmlService.createHtmlOutputFromFile("page") ;
}
```

การใช้งานโปรเจ็ค จะรันปกติไม่ได้ (Run → Run function หรือกด <Ctrl><R>) ต้องใช้วิธี Deploy as web app (ดูเพิ่มเติมในข้อ 2.2 การใช้งานโปรเจ็คแบบ Web App (Deploy as web app) หน้า 15)

เมื่อ Deploy as web app แล้ว จะได้หน้า Web App ดังนี้



ผลที่ Logs

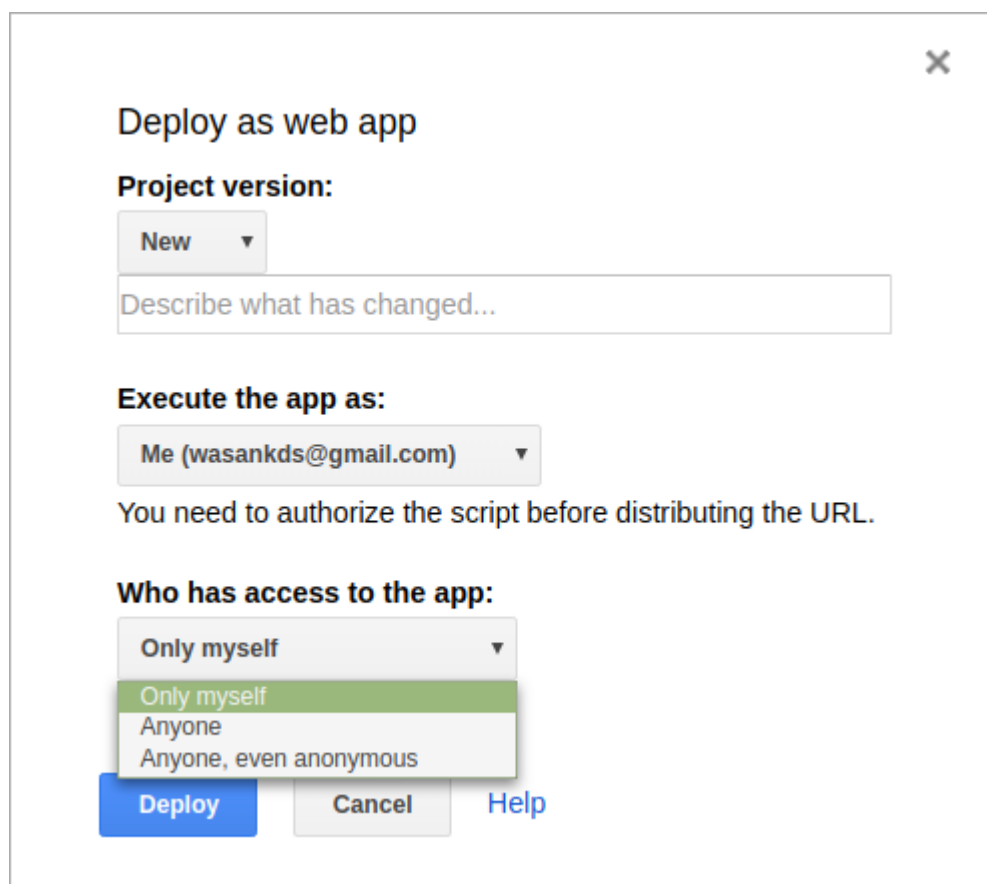
Logs

```
[ ] { parameter={}, contextPath=, contentLength=-1, queryString=, parameters={} }
```

2.2. การใช้งานโปรเจ็คแบบ Web App (Deploy as web app)

การใช้งานโปรเจ็คแบบ Web App ที่แอป Google Apps Script ให้ไปที่เมนู

[Publish](#) → [Deploy as web app](#) จะปรากฏหน้าต่างตามภาพ



ช่อง Project version : ใช้เลือกเวอร์ชันที่จะสร้างหรือใช้งาน หากต้องการสร้างเป็นเวอร์ชันใหม่ ให้เลือกเป็น new

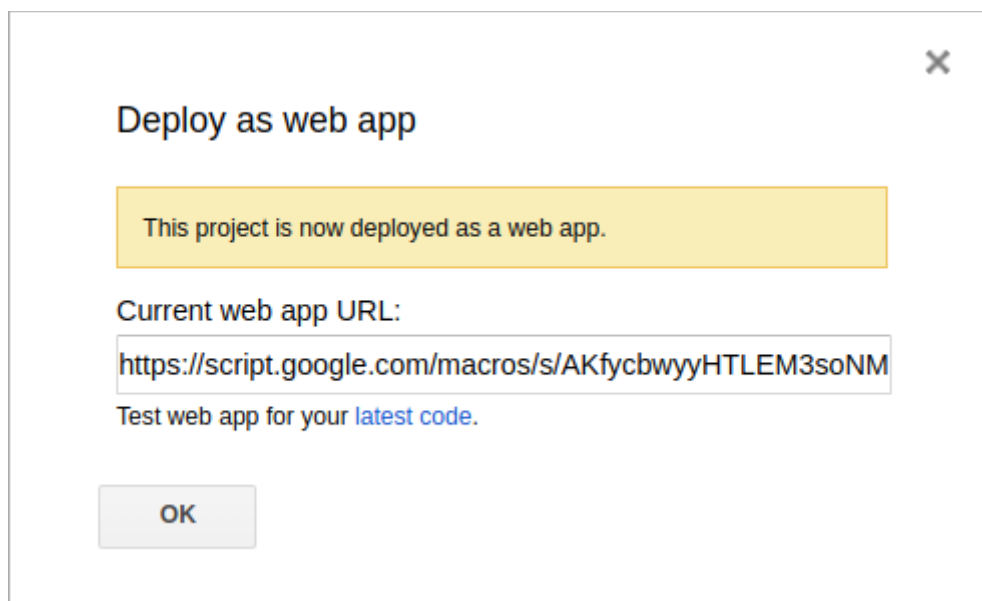
ช่อง Execute the app as : ใช้เลือกว่าจะรันแอปในฐานะใคร ตัวเลือกที่มี ก็คือ **Me** (เจ้าของโปรเจ็ค) และ **User accessing the web app** (ผู้ใช้งานจะต้อง Sign-in ด้วยบัญชี Google เพื่อใช้งาน Web App)

ช่อง Who has access to the app : เลือกว่าใครจะสามารถมาใช้นี้ได้ ตัวเลือกก็มี เราเท่านั้น(Only myself), ใครก็ได้(Anyone) และ ใครก็ได้ไม่ต้อง Sign-in(Anyone, even anonymous)

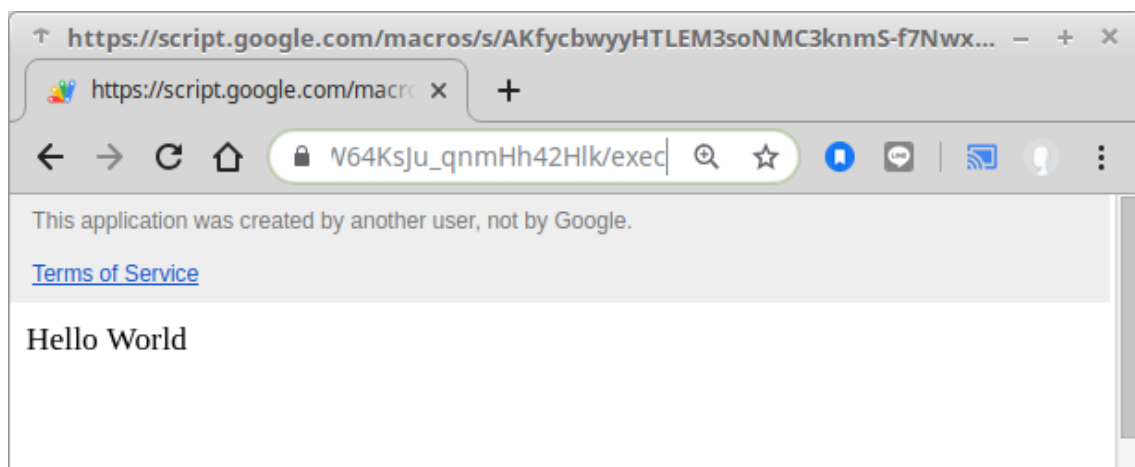
คลิกที่ปุ่ม **Deploy** / **Update** จะปรากฏหน้าต่าง Deploy as web app ตามภาพถัดไป และ **ได้ลิงค์** สำหรับใช้งาน Web App ที่ช่อง Current web app URL เช่น

<https://script.google.com/macros/s/ABCDEFGHJKLMNOPQRSTUVWXYZ/exec>

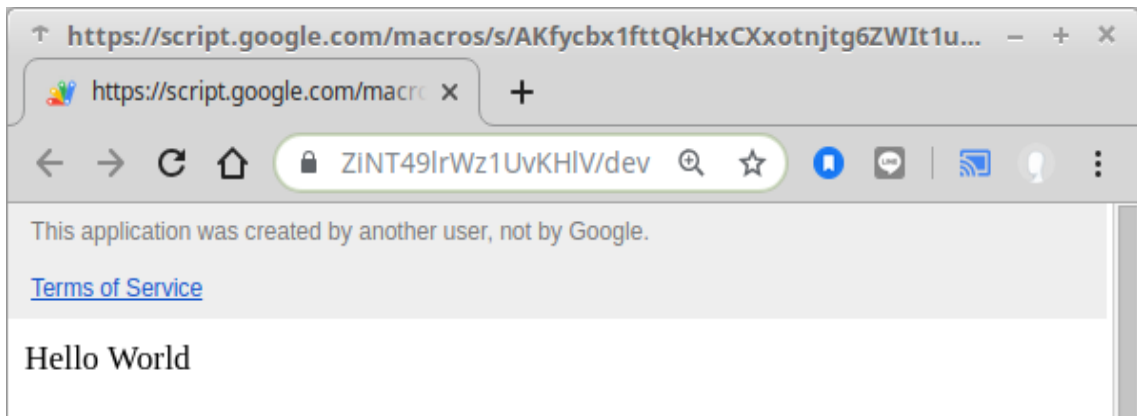
(ถ้าคลิกที่ **Lastest code** จะได้ลิงค์จะลงท้ายด้วย **/dev** เป็นการใช้งาน Web App ด้วยโค้ดล่าสุด)



เมื่อก๊อปปี้ลิงค์ Web App มาวางที่ Browser จะได้หน้า Web App ตามภาพถัดไป สังเกตที่ด้านหลังสุดของ URL ลงท้ายด้วย **/exec** ก็คือ เป็น Web app ที่ใช้งานจริง



ย้อนกลับไปหน้าต่าง Deploy as web app หากเราคลิกที่ [latest code](#) จะเข้าสู่หน้า Web App เหมือนกัน แต่จะใช้งานด้วยโค้ดล่าสุด เป็นโค้ดที่กำลังพัฒนา สังเกตที่ URL ด้านหลังสุดลงท้ายด้วย `/dev` (ผู้เขียนมักใช้แบบนี้ ขณะกำลังพัฒนา Web App)



2.3. Web Apps ด้วย Google Apps Script

Web Apps

<https://developers.google.com/apps-script/guides/web>

ถ้าเราสร้าง **Interface**(หน้าต่างของแอปที่สร้างด้วย HTML) สำหรับโปรเจ็ค Google Apps Script เราสามารถเผยแพร่โปรเจ็คของเราแบบ Web App ได้ เช่น การนัดหมายระหว่างยูสเซอร์กับทีมซัพพอร์ต จะดีที่สุดถ้าเราทำเป็น Web App ที่ยูสเซอร์สามารถเข้าถึงได้จาก Browser

แต่... สคริปต์จะเผยแพร่เป็น Web App ได้ มีเงื่อนไขต่อไปนี้

1. มีฟังก์ชัน **doGet(e)** หรือ **doPost(e)** – พารามิเตอร์ e จะมีหรือไม่มีก็ได้ แล้วแต่กรณี
2. ฟังก์ชันข้างต้น คืนค่าเป็นวัตถุ **HtmlOutput** (ก้อนของโค้ด HTML) หรือวัตถุ **TextOutput** (ก้อนของ Text ที่เกิดจากเซอร์วิส Content)

เมื่อยูสเซอร์เข้ามาดูเว็บ จะส่ง HTTP GET/POST Request มาที่ฟังก์ชัน **doGet(e)** หรือ **doPost(e)** แต่ไม่ว่าจะฟังก์ชันใดก็ตาม อาร์กิวเมนต์ **e** ที่ป้อนให้ฟังก์ชัน ก็คือ **event parameter** เป็นวัตถุที่เก็บข้อมูลเกี่ยวกับคำขอที่ส่งเข้ามา โดย **e** มี **Keys**(หรือ **Properties**) และ **Values** ดังตารางต่อไปนี้

Fields	
e.queryString	The value of the query string portion of the URL, or null if no query string is specified <code>name=alice&n=1&n=2</code>
e.parameter	An object of key/value pairs that correspond to the request parameters. Only the first value is returned for parameters that have multiple values. <code>{"name": "alice", "n": "1"}</code>
e.parameters	An object similar to e.parameter, but with an array of values for each key <code>{"name": ["alice"], "n": ["1", "2"]}</code>
e.contextPath	Not used, always the empty string.
e.contentLength	The length of the request body for POST requests, or -1 for GET requests 332

Fields	
<code>e.postData.length</code>	The same as <code>e.contentLength</code> 332
<code>e.postData.type</code>	The MIME type of the POST body text/csv
<code>e.postData.contents</code>	The content text of the POST body Alice,21
<code>e.postData.name</code>	Always the value "postData" postData

นอกจากนี้ **event parameter** แล้ว เรายังสามารถส่งผ่านพารามิเตอร์ที่เรากำหนดเองได้ด้วย อย่างเช่น **username** หรือ **age** ดังตัวอย่างต่อไปนี้ สังเกตที่หลังเครื่องหมาย "?" เป็นคู่ของ **พารามิเตอร์=ค่า** ที่ส่งไปกับ HTTP Request หากมีมากกว่า 1 คู่จะเชื่อมแต่ละคู่ด้วยเครื่องหมาย "&"

```
https://script.google.com/.../exec?username=jsmith&age=21
```

จากนั้น เราสามารถแสดงพารามิเตอร์ทั้งหมดในรูปแบบ JSON ได้ดังนี้

```
function doGet(e) {
  var params = JSON.stringify(e);
  return HtmlService.createHtmlOutput(params);
}
```

ผลจากโค้ดข้างต้น จะได้ JSON ดังต่อไปนี้

```
{
  "queryString": "username=jsmith&age=21",
  "parameter": {
    "username": "jsmith",
    "age": "21"
  },
  "contextPath": "",
  "parameters": {
    "username": [
      "jsmith"
    ],
    "age": [
      "21"
    ]
  },
  "contentLength": -1
}
```

หมายเหตุ

พารามิเตอร์ชื่อ **c** และ **sid** ถูกสงวนไว้สำหรับระบบเท่านั้น ห้ามตั้งชื่อพารามิเตอร์เป็นชื่อนี้ หากใช้จะปรากฏ Error HTTP 405 response ที่มีข้อความ "Sorry, the file you have requested does not exist." If possible, update your script to use different parameter names.

บทที่ 3

doGet() และ doPost()



3.1. doGet() และ doPost()

doGet and doPost Tutorial + 6 Web App Examples

<http://googleappscripting.com/doget-dopost-tutorial-examples/>

เมื่อทำ Web App เราจะต้องรู้จักกับ HTTP (Hypertext Transfer Protocol) ซึ่งเป็นข้อกำหนดในการสื่อสาร ทั้งรับและส่งข้อมูลระหว่างอุปกรณ์ที่แตกต่างกัน

HTTP รองรับการใช้งานหลายชนิด ที่นิยมกันมากก็คือ **GET** และ **POST**

ฟังก์ชัน **doGet()** และ **doPost()** ใน Google Apps Script เป็นตัวจัดการ HTTP GET Request และ HTTP POST request ที่ส่งมาจากอุปกรณ์อื่นๆ

GET Request

เกิดขึ้นตลอดเวลาที่เรากรอก URL ลงใน Browser เซิร์ฟเวอร์ของโดเมนที่ระบุใน URL จะรับคำร้องขอประมวลผล แล้วคืนค่ากลับไปเป็น HTML จากนั้น Browser ก็ประมวลผลให้เราเห็นเป็นหน้าเว็บ

GET Request สร้างมาจาก ชื่อโดเมน อย่างเช่น example.com และพาธเช่น [/fruits/apples/](http://example.com/fruits/apples/) และมักจะมี Query string (หรือพารามิเตอร์) อย่างเช่น [?variety=fuji&count=4](http://example.com/?variety=fuji&count=4) ติดตามด้วย ซึ่งมีลักษณะเป็น คู่ของตัวแปร และค่า หากมีหลายคู่ จะเชื่อมกันด้วยเครื่องหมาย &

สำหรับ Google Apps Script โดเมนเป็น script.google.com เสมอ พารามีลักษณะเป็น [/macros/s/AKf...ycb/exec](http://script.google.com/macros/s/AKf...ycb/exec) (หรือ [/dev](http://script.google.com/dev)) และตามด้วย Query string (หรือพารามิเตอร์) ซึ่งเป็นอะไรก็ได้แล้วแต่เราจะกำหนด

POST Request

เหมือนกับ GET แต่มีพลังมากกว่า แทนที่จะใช้สอบถามข้อมูล แต่ POST ถูกใช้ในการส่งข้อมูลจากอุปกรณ์หนึ่งไปยังอุปกรณ์หนึ่ง อย่างหนึ่งที่ POST ถูกใช้บ่อยก็คือ ส่งฟอร์ม เช่น Sign-up form, หน้า E-commerce checkout เป็นต้น POST

ไม่มีข้อกำหนดปริมาณการส่งข้อมูล จึงแบกน้ำหนักบรรทุก (Payload) ติดไปได้มหาศาล แยกข้อมูลได้หลากหลายประเภท เช่น ภาพ, เสียง และ อีกรวมหลายชนิด

3.1.ก.) Web App คืค่าเป็น Text

เมธอด `createTextOutput()` ของเซอร์วิส Content เป็นวิธีพื้นฐานที่ใช้คืนค่าให้กับ HTTP GET Request โดยจะคืนค่าเป็นวัตถุ `TextOutput` หรือเป็นก้อนของ Text (หรือ String) ค่าปริยายของชนิด Text ที่คืนกลับไปก็คือ text/plain อย่างไรก็ตามเราสามารถเซตเป็น Text ต่อไปนี้ได้ เช่น ATOM, CSV, ICAL, JAVASCRIPT, JSON, RSS, TEXT, VCARD และ XML

ตัวอย่าง

```
function doGet(){  
    textOutput = ContentService.createTextOutput("Hello World! Welcome to the web app.")  
    return textOutput  
}
```

เมื่อ Deploy as web app จะได้หน้าเว็บที่มีแต่ข้อความดังนี้

Hello World! Welcome to the web app.

3.1.ข.) Web App คืค่าเป็น JSON

JSON ก็คือ Text ที่มีรูปแบบที่เหมาะสมตามข้อกำหนดในการส่งข้อมูลระหว่างเซิร์ฟเวอร์ และสามารถแปลงเป็น Javascript object เพื่อนำไปประมวลผลต่อด้วยสคริปต์ในภาษาโปรแกรมมิ่งต่างๆ

การคืนค่าเป็น JSON สำคัญว่า เราต้องใช้เมธอด `setMimeType` (ของคลาส `TextOutput`) ระบุก้อนวัตถุ `TextOutput` ที่คืนค่ากลับไปว่ามี `Mime type` เป็น JSON

ตัวอย่าง

```
function doGet(){  
    // Javascript object  
    var appData = {  
        "heading": "Hello World!" ,  
        "body": "Welcome to the web app."  
    } ;  
  
    // สร้าง JSON String จาก Javascript object  
    //  
    var jsonString = JSON.stringify(appData) ;  
  
    // สร้างก้อนวัตถุ TextOutput จาก JSON String  
    //  
    var jsonOutput = ContentService.createTextOutput(jsonString) ;  
    jsonOutput.setMimeType(ContentService.MimeType.JSON) ;  
  
    return jsonOutput  
}
```

เมื่อ Deploy as web app จะได้หน้าเว็บที่มีแต่ข้อความดังนี้ ซึ่งก็คือ JSON String นั่นเอง

```
{"heading":"Hello World!","body":"Welcome to the web app."}
```

หมายเหตุ

การคืนค่าเป็น JSON ใช้กันมากในการสร้างหรือให้บริการ API(Application Program Interface) ที่รับคำร้องของ HTTP Request แล้วคืนค่ากลับไปเป็น JSON จากนั้นผู้ส่ง HTTP Request ก็จะนำ JSON ไปประมวลผลต่อไป ดูเพิ่มเติมจากลิงค์ต่อไปนี้

How to Make a Google Apps Script RESTful API or Service

<https://trevorfox.com/2015/03/rest-api-with-google-apps-script/>

ในหนังสือเล่มนี้ก็มีตัวอย่าง เช่น [การใช้งาน Fixer API](#) และ [การใช้งาน Content API](#)

3.1.ค.) Web App คืนค่าเป็น HTML

เมธอด createHtmlOutput() ของเซอร์วิส HTML มี 2 วัตถุประสงค์ **วัตถุประสงค์แรก** ก็คือ **คืนโค้ด HTML** ก่อนคืนกลับไปให้ Browser เช่น ตัดคอมเมนต์ออก เป็นต้น ซึ่งเมธอดนี้เซต Mime type ของก้อน Text เป็น text/html โดยปริยาย (ไม่ต้องระบุใดๆ)

และ **อีกวัตถุประสงค์** ก็คือ **สร้างหน้าเว็บ** ที่เราสามารถประกอบหน้าขึ้นมาเองจาก HTML String

ตัวอย่าง

```
function doGet(){  
  // สร้าง HTML String – โค้ด HTML  
  var HTMLString = "<style> h1,p {font-family: 'Helvetica', 'Arial'}</style>" +  
    "<h1>Hello World!</h1>" +  
    "<p>Welcome to the Web App" ;  
  
  // สร้างหน้าเว็บ  
  HTMLOutput = HtmlService.createHtmlOutput(HTMLString) ;  
  
  return HTMLOutput  
}
```

เมื่อ Deploy as web app จะได้หน้าเว็บดังนี้

Hello World!

Welcome to the Web App

3.1.ง.) Web App คีนค่าเป็น HTML โดยใช้เทมเพลต

เมทอด `createHtmlOutput()` สร้างหน้าเว็บจากโค้ด HTML แบบตรงๆ แต่ถ้าเราต้องการจะฝัง Scriptlets (ดูเพิ่มเติมบทที่ 5 หน้า 31) เพื่อใส่ตัวแปร ใส่โค้ด Google Apps Script ลงไป เพื่อให้สามารถปรับจูน เปลี่ยนเนื้อหาของหน้าเว็บ ตามข้อมูลต่างๆ เราสามารถทำได้โดยใช้เมทอดตระกูลสร้างเทมเพลต อย่างเช่น `createTemplate()` หรือ `createTemplateFromFile()` ซึ่งจะคืนค่าเป็นวัตถุ `HtmlTemplate` กลับมา

ตัวอย่าง

ตัวอย่างต่อไปนี้ ฟังก์ชัน `doGet(e)` รับพารามิเตอร์ `e` เข้ามาด้วย ซึ่ง `e` ก็คือ Event parameters ที่มีข้อมูลของ Event อยู่ข้างใน เช่นเมื่อเกิด Event HTTP Request จะสร้างพารามิเตอร์ `e` ป้อนให้กับ `doGet()` เป็นต้น โดยคีย์สำคัญที่อยู่ใน `e` ก็คือ คีย์ `e.parameter(s)` เก็บข้อมูล Query String เอาไว้

```
function doGet(e){
    // ใช้ CSS ของ Bootstrap
    var style = '<link rel="stylesheet"
                href="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">' ;

    // จับพารามิเตอร์ query string "greeting" หากไม่มี - กำหนดค่าปรัยายเป็น "Hello" ให้ตัวแปร
    var greeting = e.parameter.greeting || "Hello" ;

    // จับพารามิเตอร์ query string "name" หากไม่มี - กำหนดค่าปรัยายเป็น "World!" ให้ตัวแปร
    var name = e.parameter.name || "World" ;

    // สร้างวัตถุ HtmlTemplate - มี Scriptlet ด้วย
    var heading = HtmlService.createTemplate('<div class="container"><h1>
                                           <?= greeting ?><?= name ?>!</h1></div>')

    // เพิ่มคู่ Key และ Value ให้กับตัวแปรวัตถุ โดยใช้ Value ที่จับมาจาก Query String
    // เป็นการ Assign ค่าให้กับตัวแปร greeting และ name ใน Scriptlets ด้วย
    // ( ตัวอย่างนี้ใช้ชื่อ Key กับ e.parameter ชื่อเดียวกัน ระวัง งง !!! )
    heading.greeting = greeting ;
    heading.name = name ;

    // โค้ด HTML - เนื้อหาดายตัว
    var content = "<div class='container'><p>Welcome to the web app.</p></div>";

    var HTMLOutput = HtmlService.createHtmlOutput() ;           // สร้างวัตถุ HtmlOutput แบบว่างๆ
    HTMLOutput.append(style) ;                                   // แแนบ CSS

    // วัตถุ HtmlTemplate ต้องจบด้วยเมทอด evaluate() เพื่อประมวล Scriptlets และคลีนโค้ด Html
    // จากนั้นตามด้วย getContent() เพื่อแปลงเป็นโค้ด HTML String
    HTMLOutput.append(heading.evaluate().getContent()) ;
    HTMLOutput.append(content);                                  // แแนบ content - ข้อความต้อนรับ

    return HTMLOutput
}
```

เมื่อ Deploy as web app โดยลิงค์ที่ส่งพารามิเตอร์แบบกำหนดเองเพิ่มเติมเข้าไปด้วย ดังตัวอย่าง
https://script.google.com/macros/s/GAS_Project_URL/exec?greeting=Hi&name=Wasan

จะได้ผลลัพธ์ที่หน้า Web App ดังนี้

Hi Wasan!

Welcome to the web app.

หมายเหตุ :

ถ้าต้องการเว้นช่องว่างของค่าในพารามิเตอร์ที่ส่ง ให้ใช้ %20 แทน Space เช่น

https://script.google.com/macros/s/GAS_Project_URL/exec?greeting=Hi&name=Wasan%20Kds

กรณีไม่ส่งพารามิเตอร์แบบกำหนดเอง เข้าไปดังตัวอย่าง จะได้ผลลัพธ์ที่หน้า Web App ดังนี้

Hello World!

Welcome to the web app.

บทที่ 4
สถาปัตยกรรมการ
ทำงานของ
Web App



4.1. สาธิตกระบวนการทำงานของ Web App

Deploying a Google Apps Script Web Application PART 1

<https://www.youtube.com/watch?v=bwU1MSLi33Q&feature=youtu.be>

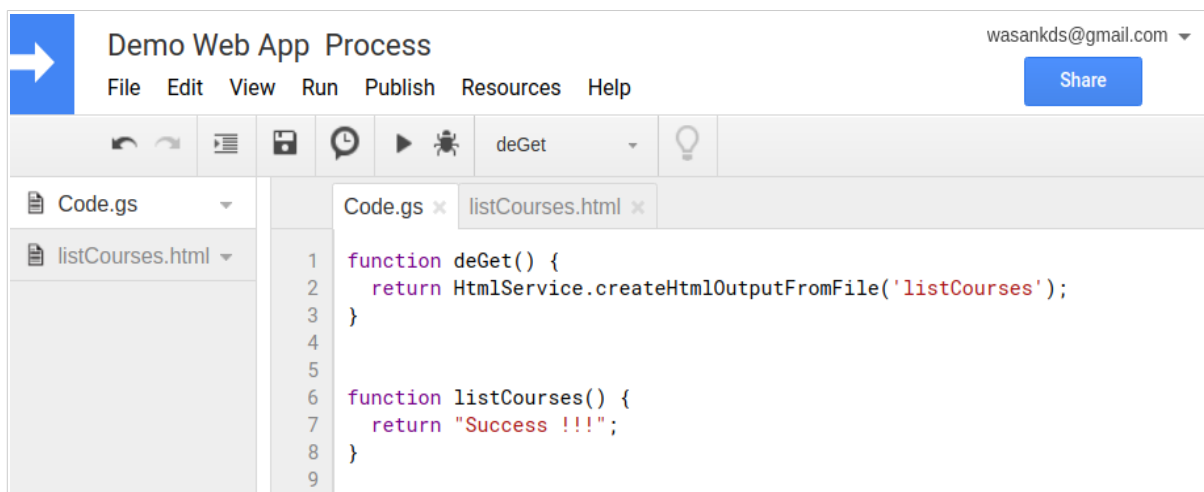
ในบทนี้เราจะสร้าง Web App ที่มีความซับซ้อนมากกว่าบทที่แล้ว มีการเรียกฟังก์ชันทั้งฝั่ง Client-side และ Server-side และ มีการจัดการ Success และ Failure โดยใช้เมธอดต่างๆ

จุดประสงค์เพื่อให้เข้าใจกระบวนการ การเรียกฟังก์ชันในแต่ละฝั่ง และเข้าใจเซอร์วิสและเมธอด ที่สำคัญในกระบวนการ

4.1.ก.) สร้างโปรเจ็ค

สร้างโปรเจ็ค Google Apps Script แบบ Standalone จากนั้นสร้าง ไฟล์ .gs และ ไฟล์ .html ตามภาพ ก็คือไฟล์ดังต่อไปนี้

ไฟล์ `code.gs` และ ไฟล์ `listCourses.html`



4.1.ข.) โค้ดในไฟล์ .gs

พิมพ์โค้ดต่างๆ ลงใน ไฟล์ .gs โดยโค้ดในไฟล์ .gs เป็น Server-side script คล้ายกับ PHP ที่รันในฝั่ง Server-side

ไฟล์ `code.gs`

```
function doGet() {  
    return HtmlService.createHtmlOutputFromFile('listCourses') ;  
}  
  
function listCourses() {  
    return "Success !!!" ;  
}
```

4.1.ค.) โค้ดในไฟล์ .html

พิมพ์โค้ดต่างๆ ลงใน **ไฟล์ .html** โดยจะเป็นส่วนของหน้าตา Web App และในบล็อก **<script>** **</script>** เป็น **Javascript** ที่ทำงานในฝั่ง Client-side แต่ก็สามารถเรียกรันฟังก์ชันในฝั่ง Server-side ได้โดยใช้ Javascript API `google.script.run`

ไฟล์ `listCourses.html`

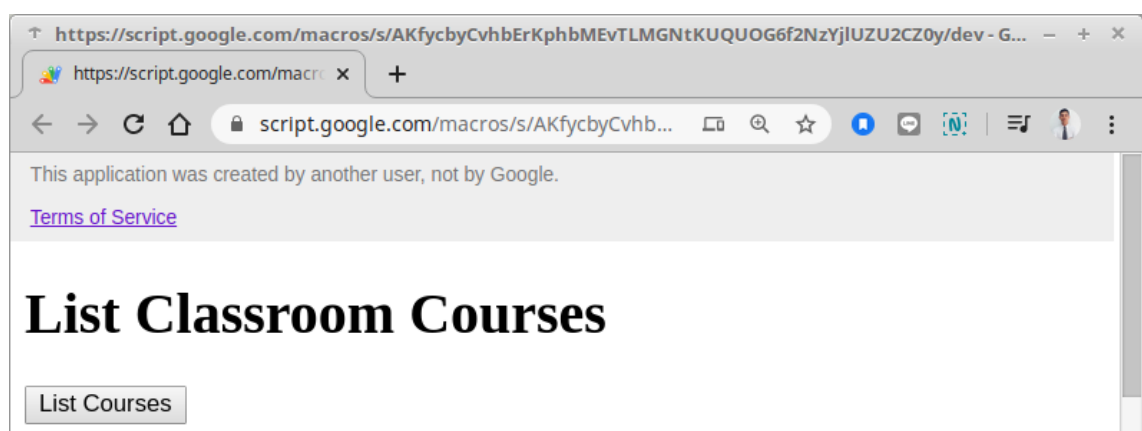
```
<!DOCTYPE html>
<html>
  <head>
    <base target="_self">
  </head>
  <body>
    <h1>List Classroom Courses</h1>

    <!-- สร้างปุ่ม "List Courses" -->
    <button onclick="listCourses()">List Courses</button>
    <ul id="courses"></ul>

    <script> <!-- Javascript -->
    function listCourses() {
      // google.script.run.listCourses() ; // แบบนี้ก็ได้ แต่ไม่มีการจัดการ Failure หรือ Success
      google.script.run
        .withSuccessHandler(function(response){ console.log(response) })
        .withFailureHandler(function(err){ console.log(err) })
        .listCourses() ;
    }
  </script>
</body>
</html>
```

4.1.ง.) ใช้งานแบบ Web App (Deploy as web app)

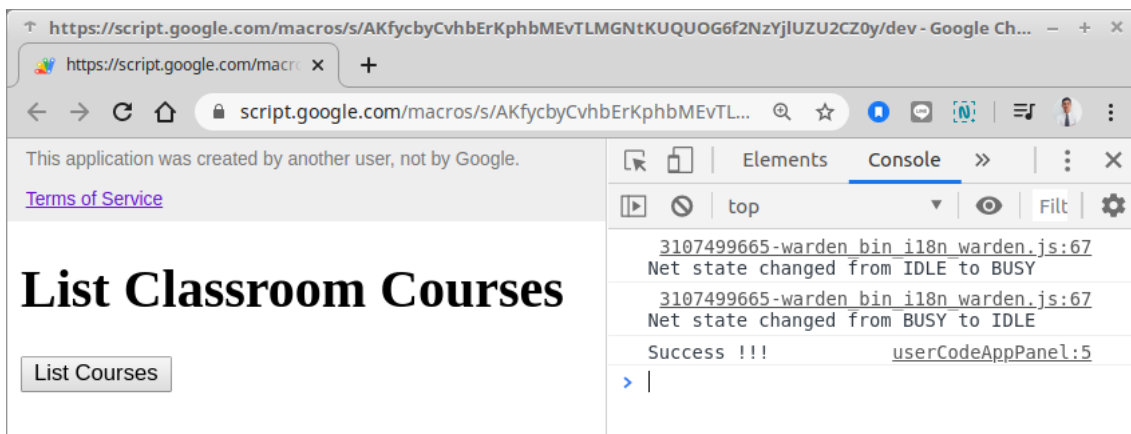
เมื่อใช้งานโปรเจกต์โดย Deploy as web app จะได้หน้า Web App ตามภาพ



เมื่อคลิกที่ปุ่ม **List Courses** จะปรากฏข้อความ **"Successes !!!"** ที่คอนโซลของ Browser ตามภาพถัดไป

ไป

(กด **<Ctrl><Shift><I>** = เปิดคอนโซลของ Browser)



4.2. ขั้นตอนการทำงาน

จากข้อก่อนหน้า โค้ดในไฟล์ต่างๆ มีขั้นตอนการทำงานดังนี้

ขั้นตอนการโหลดหน้า Web App

1. ที่ Browser

- 1.1 ผู้ใช้งานกรอกลิงค์เพื่อเปิด Web App ในช่อง URL ของ Browser เช่น
<https://script.google.com/macros/s/ABCDEFGHIJKLMNOPQRSTUVWXYZ/dev> หรือ /exec

2. ที่ Server

- 2.1 ฟังก์ชัน doGet() ในไฟล์ [code.gs](#) ทำงาน

doGet() ทำงานเมื่อมี HTTP GET Request หรือก็คือ มีการเรียกหน้าเว็บ นั้นเอง

- 2.2 ฟังก์ชัน doGet() สร้างโค้ด HTML จากไฟล์ [listCourses.html](#) จากคำสั่งบรรทัด

```
return HtmlService.createHtmlOutputFromFile('listCourses') ;
```

- 2.3 หากโค้ดในไฟล์ HTML มี Scriptlets ฝังอยู่ ก็คือ มีโค้ดในบล็อก `<? // Google Apps Script ?>`
(แต่ด้วยวิธีนี้ไม่มี) โค้ดจะถูกประมวลผลที่ Server-side แล้วแปลงเป็นโค้ด HTML
 (ดูเรื่อง Scriptlets ในบทที่ 5 หน้า 31)

- 2.4 Server ส่งโค้ด HTML กลับไปให้ Browser

3. ที่ Browser

- 3.1 ประมวลโค้ด HTML ที่ส่งกลับมาเป็นหน้าเว็บ
- 3.2 รวบรวมปฏิสัมพันธ์จากยูสเซอร์

List Classroom Courses

List Courses

จบขั้นตอนการโหลดหน้า Web App ถัดมาเป็นการใช้งานหน้า Web App หรือ การปฏิสัมพันธ์ระหว่าง ยูสเซอร์กับ Web App

ขั้นตอนการใช้งานหน้า Web App

1. ที่ Browser

1.1 เมื่อคลิก ปุ่ม List Courses

ฟังก์ชัน `listCourses()` ทำงาน (ฟังก์ชัน Javascript ในฝั่ง Client-side)
จากคำสั่งบรรทัด

```
<button onclick="listCourses()">List Courses</button>
```

1.2 ฟังก์ชัน `listCourses()` เรียกฟังก์ชัน `listCourses()` ในไฟล์ `code.gs` ให้ทำงาน ด้วยคำสั่งบรรทัด

```
google.script.run.(ฟังก์ชันในไฟล์ .gs ที่จะรัน) ;
```

2. ที่ Server

2.1 ฟังก์ชัน `listCourses()` (ฟังก์ชัน Google Apps Script ในไฟล์ `code.gs` ฝั่ง Server-side)

2.2 ถ้าฟังก์ชัน `listCourses()` ในไฟล์ `code.gs` ไม่มีการคืนค่ากลับมา (ไม่มี `return`) เขียนโค้ดเพียงเท่านี้ก็พอ ก็คือ

```
google.script.run.listCourses() ;
```

3. ที่ Browser (หากมี `return` จาก Server)

3.1 ถ้าฟังก์ชัน `listCourses()` ในไฟล์ `code.gs` มีการคืนค่า (มี `return`) ซึ่งในตัวอย่างนี้มี

```
function listCourses() {  
  return "Success !!!" ; // คืนค่าเป็น String ง่ายๆ  
}
```

ก.) (ทางเลือกที่ 1) เมธอด `withSuccessHandler` ทำงาน ถ้ามีค่าคืนกลับมา
จะรันฟังก์ชันที่ระบุในที่นี้ก็คือ `function(response)`

```
.withSuccessHandler(function(response){ console.log(response) } )
```

ข.) (ทางเลือกที่ 2) เมธอด `withFailureHandler` ทำงาน ถ้าไม่มีอะไรคืนกลับมา
จะรันฟังก์ชันที่ระบุในที่นี้ก็คือ `function(err)`

```
.withFailureHandler(function(err){ console.log(err) } )
```


บทที่ 5

Scriptlets



5.1. Scriptlets

HTML Service: Templated HTML

<https://developers.google.com/apps-script/guides/html/templates>

เราสามารถฝัง Scriptlets ลงไปในโค้ด HTML ที่เป็นส่วนหน้าตาของ Web App โดย Scriptlets เป็นบล็อกของสัญลักษณ์ `<? ?>` ภายใน Scriptlets เป็นโค้ด Google Apps Script ซึ่งคล้ายกับ PHP ที่เป็น Server-side script เหมือนกัน

โค้ดใน Scriptlets ถูกรันที่เซิร์ฟเวอร์เพียงครั้งเดียว แล้วแปลงเป็นโค้ด HTML ก่อนที่โค้ด HTML จะถูกส่งไปให้ Browser Scriptlets ถูกรันเพียงครั้งเดียว เหมือน PHP ที่เป็น Server-side script เหมือนกัน ต่างจาก Javascript ที่รันได้หลายครั้งในฝั่ง Client-side

Scriptlets มี 3 แบบดังต่อไปนี้

5.2. Standard scriptlet

ใช้สัญลักษณ์ `<? ... ?>` ครอบโค้ด Apps Script เพื่อประมวลโค้ดโดย ไม่พิมพ์ผลลัพธ์ออกมาในไฟล์ HTML อย่างไรก็ตาม โค้ดมีผลต่อเนื้อหาของ HTML

ตัวอย่าง

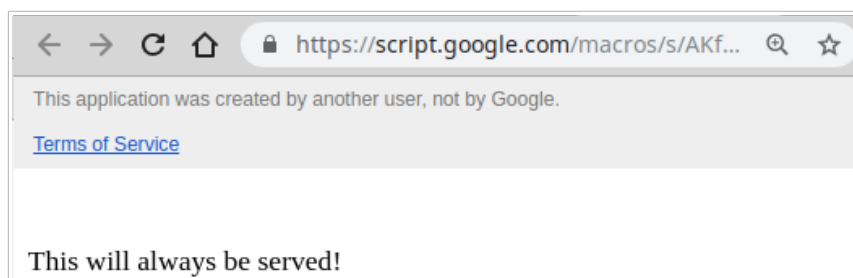
ไฟล์ Code.gs

```
function doGet() {  
  return HtmlService.createTemplateFromFile('Index').evaluate() ;  
}
```

ไฟล์ Index.html

```
<html>  
<head>  
  <base target="_top">  
</head>  
<body>  
  <? if (true) { ?> <!-- -->  
    <p>This will always be served!</p>  
  <? } else { ?>  
    <p>This will never be served.</p>  
  <? } ?>  
</body>  
</html>
```

ผล



5.3. Printing scriptlet

ใช้สัญลักษณ์ `<?= ... ?>` ครอบโค้ด Apps Script เพื่อพิมพ์ค่า เช่น พิมพ์ค่าของตัวแปรให้กับโค้ด HTML

ตัวอย่าง

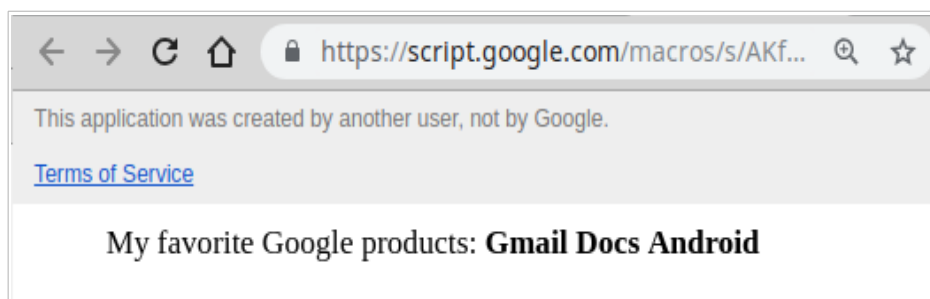
ไฟล์ Code.gs

```
function doGet() {  
  return HtmlService.createTemplateFromFile('Index').evaluate() ;  
}
```

ไฟล์ Index.html

```
<!DOCTYPE html>  
<html>  
  <head>  
    <base target="_top">  
  </head>  
  
  <body>  
  
    <!-- ถ้าใช้ <? ... ?> ข้อความจะไม่ถูกพิมพ์ -->  
    <?= 'My favorite Google products:' ?>                                <!-- พิมพ์ -->  
    <? var data = [ 'Gmail' , 'Docs' , 'Android' ] ;  
      for (var i = 0; i < data.length; i++) { ?>  
        <b><?= data[i] ?></b>                                <!-- พิมพ์ -->  
      <? } ?>  
  
  </body>  
</html>
```

ผล



5.4. Force-printing scriptlet

ใช้สัญลักษณ์ `<?!= ?>` ครอปโค้ด Apps Script เพื่อรัน String ในฐานะโค้ด HTML ให้ผลลัพธ์เหมือนรันโค้ด HTML ด้วย Browser

ตัวอย่าง

พัฒนาการที่ 1

```
<!-- ไฟล์ page.html -->
<? var title = "Hello !" ?>
<h1><? title; ?></h1>      <!-- รันแบบนี้ ผลลัพธ์จะไม่ออกมา -->
<h1><?= title; ?></h1>      <!-- รันแบบนี้ ใช้แสดงผลลัพธ์ -->
```

ผลที่ Browser

Hello !

พัฒนาการที่ 2 : ใส่แท็ก `` ครอป Hello ! เพื่อแปลงเป็น HTML String

```
<!-- ไฟล์ page.html -->
<? var title = "<b>Hello !</b>" ?> <!-- จับโค้ด HTML ใส่ตัวแปร -->
<h1><?= title; ?></h1>              <!-- รันแบบนี้ ใช้แสดงเป็น String ธรรมดา -->
                                   <!-- ไม่ประมวลผลแบบโค้ด HTML -->
```

ผลที่ Browser – ผลยังคงเหมือนเดิม อะไรที่อยู่ใน " " จะออกมาแบบนั้น

Hello !

พัฒนาการที่ 3 : ใช้เครื่องหมาย `<?!= ?>`

```
<!-- ไฟล์ page.html -->
<? var title = "<b>Hello !</b>" ?> <!-- จับโค้ด HTML ใส่ตัวแปร -->
<h1><?!= title; ?></h1>              <!-- รัน String แบบ Browser รันโค้ด HTML -->
```

ผลที่ Browser

Hello !

หมายเหตุ

พยายามหลีกเลี่ยงการใช้ `<?!= ?>` เพราะมีการรันสคริปต์ อาจเป็นปัญหาเมื่อยูสเซอร์ใส่อะไรแปลกๆ มาให้

5.5. โหลดข้อมูลจาก Google Sheets มาไว้ที่หน้าเว็บ

Calling Apps Script functions from a template

https://developers.google.com/apps-script/guides/html/templates#calling_apps_script_functions_from_a_template

ตัวอย่างนี้ เรียกใช้ฟังก์ชันในไฟล์ .gs ในฝั่ง Server-side ผ่านการฝัง Scriptlets ลงในโค้ด HTML ส่วนของ Scriptlets จะถูกรันและแปลงเป็นโค้ด HTML จากนั้นโค้ด HTML ทั้งก่อน จะถูกส่งไปที่ Browser

หมายเหตุ

มีอีกวิธีหนึ่งในการเรียกใช้ฟังก์ชันในไฟล์ .gs แม้หน้าเพจจะถูกโหลดแล้วก็ตาม ก็คือ ใช้โค้ด Javascript ที่ทำงานในฝั่ง Client-side เรียกใช้เซอร์วิส `google.script.run.(ฟังก์ชันในไฟล์ .gs)`

ไฟล์ Google Sheet - มีข้อมูลดังต่อไปนี้

	A	B	C	D
1	Fruit	Cost	Quantity	Total
2	Apple	15	200	3,000.00 บาท
3	Bananas	50	520	26,000.00 บาท
4	Mangoes	20	1,025	20,500.00 บาท
5	Durian	150	50	7,500.00 บาท
6	Longan	60	450	27,000.00 บาท
7	Total			84,000.00 บาท

ไฟล์ Code.gs

```
function doGet() {  
    return HtmlService.createTemplateFromFile('Index').evaluate();  
}  
  
// คำนวณกลับเป็นข้อมูลที่จับมาจากในชีต - เป็นอาเรย์ 2 มิติ  
//  
function getData() {  
    var id = 'Google Sheets File Id' ;  
    return SpreadsheetApp  
        .openById(id)  
        .getActiveSheet()  
        .getDataRange()           // จับเรนจ์ที่มีข้อมูลทั้งหมด  
        .getValues() ;           // จับข้อมูลในเรนจ์  
}
```

```

<!DOCTYPE html>
<html>
  <head>
    <base target="_top">
  </head>

  <body>

    <!-- เรียกใช้ฟังก์ชัน getData() ในไฟล์ Code.gs -->
    <!-- ได้ข้อมูลจาก Google Sheets จัปใส่ตัวแปร data -->
    <? var data = getData() ; ?>

    <!-- วนลูปพิมพ์ค่าของตัวแปร ลงในตาราง HTML -->
    <table border="1" cellpadding="5">
      <? for (var i = 0 ; i < data.length ; i++) { ?>
        <tr>
          <? for (var j = 0 ; j < data[i].length ; j++) { ?>
            <td><?= data[i][j] ?></td>      <!-- พิมพ์ข้อมูล -->
          <? } ?>
        </tr>
      <? } ?>
    </table>
  </body>
</html>

```

เมื่อ Deploy as web app จะได้หน้า Web App ตามภาพ

← → ↻ 🏠 🔒 https://script.google.com/macros/s/AKf... 🔍 ☆

This application was created by another user, not by Google.

[Terms of Service](#)

Fruit	Cost	Quantity	Total
Apple	15	200	3000
Bananas	50	520	26000
Mangoes	20	1025	20500
Durian	150	50	7500
Longan	60	450	27000
Total			84000

บทที่ 6

เซอร์วิส HTML



6.1. เซอร์วิส HTML

HTML Service

<https://developers.google.com/apps-script/reference/html>

เซอร์วิส HTML เป็นบริการที่ใช้คืนค่ากลับมาเป็น HTML โดยในโปรเจ็ค Google Apps Script สามารถมีไฟล์ HTML ได้ ซึ่งมักใช้เป็น Interface หรือหน้าต่างของโปรเจ็ค

เซอร์วิส HTML มีคลาสให้ใช้หลายตัว เช่น HtmlService, HtmlOutput, HtmlTemplate เป็นต้น

ดูเพิ่มเติม

HTML Service: Create and Serve HTML

<https://developers.google.com/apps-script/guides/html>

UI Service

<https://developers.google.com/apps-script/guides/ui-service>

6.2. createHtmlOutput()

6.2.ก.) createHtmlOutput()

คลาส HtmlOutput

<https://developers.google.com/apps-script/reference/html/html-output>

createHtmlOutput() - เมธอดในคลาส HtmlService

<https://developers.google.com/apps-script/reference/html/html-service#createhtmloutput>

createHtmlOutput(blob) - เมธอดในคลาส HtmlService

<https://developers.google.com/apps-script/reference/html/html-service#createhtmloutputblob>

createHtmlOutput(html) - เมธอดในคลาส HtmlService

<https://developers.google.com/apps-script/reference/html/html-service#createhtmloutputhtml>

createHtmlOutput() ใช้สร้างโค้ด HTML โดยจะคืนค่ากลับมาเป็นวัตถุ HtmlOutput ซึ่งเป็นก้อนของโค้ด HTML ที่ถูกคลีนแล้ว เช่น เอาคอมเมนต์ออก และมีความปลอดภัย พารามิเตอร์ที่ใส่ให้กับเมธอดนี้ สามารถเป็น **html** หรือ **blob** ก็ได้

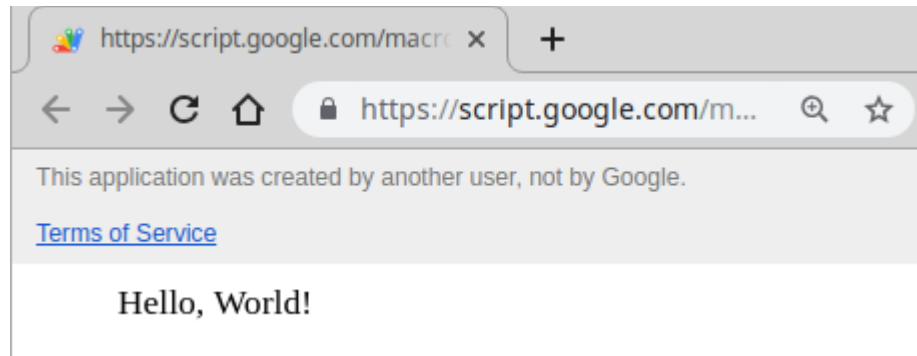
ตัวอย่าง - พารามิเตอร์ของ createHtmlOutput() เป็น String ที่เป็น โค้ด HTML

```
function doGet() {  
  return HtmlService.createHtmlOutput('<b>Hello, world!</b>');  
}
```

หมายเหตุ

doGet() เป็นฟังก์ชันสกรวน(Simple trigger) ทำงานเมื่อมี HTTP GET Request หรือ หน้าเว็บถูกเรียก หรือ รีเฟรชหน้าเว็บ

เมื่อใช้งานโปรเจ็คโดยวิธี Deploy as web app จะได้ผลลัพธ์ตามภาพ



6.2.ข.) append() และ getContent()

append(**addedContent**) – เมธอดในคลาส HtmlOutput

<https://developers.google.com/apps-script/reference/html/html-output#appendaddedcontent>

append() ใช้แนบเนื้อหาเพิ่มเติมให้กับวัตถุ HtmlOutput พารามิเตอร์ **addedContent** ก็คือ เนื้อหาซึ่ง

เป็น String ของโค้ด HTML

getContent() – เมธอดในคลาส HtmlOutput

<https://developers.google.com/apps-script/reference/html/html-output#getcontent>

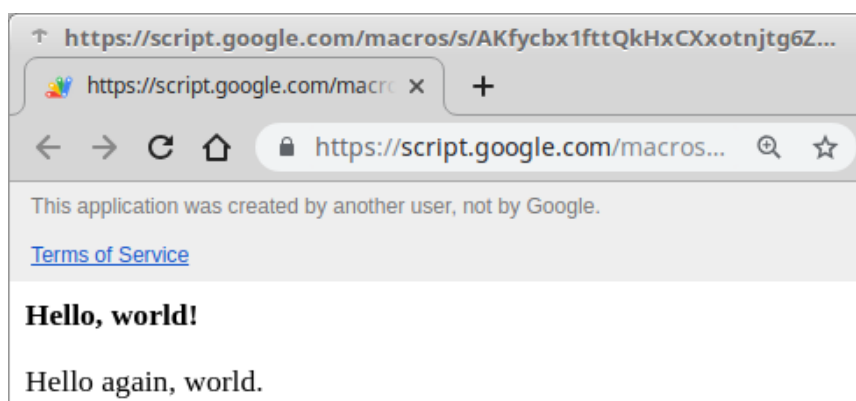
getContent() ใช้จับเนื้อหาในวัตถุ HtmlOutput ซึ่งก็คือโค้ด HTML นั้นเอง

ตัวอย่าง

ตัวอย่างนี้ มีการฝากเนื้อหาของ HtmlOutput ไว้ในตัวแปร Object แล้วทดสอบ Logs ออกมาดูว่า ค่าของ ตัวแปร Object เก็บอะไรไว้ จากนั้นจึงใช้เมธอด append() แนบเนื้อหาเพิ่มเข้าไป

```
function doGet() {  
  var output = HtmlService.createHtmlOutput('<b>Hello, world!</b>') ;  
  output.append('<p>Hello again, world.</p>') ;  
  Logger.log(output.getContent()) ;  
  // พิมพ์ : "<b>Hello, world!</b><p>Hello again, world.</p>"  
  return output ;    // ส่ง HtmlOutput ไปให้ Browser  
}
```

ผล - ได้หน้า Web App ตามภาพ



6.3. createHtmlOutputFromFile()

HTML Service: Create and Serve HTML

<https://developers.google.com/apps-script/guides/html>

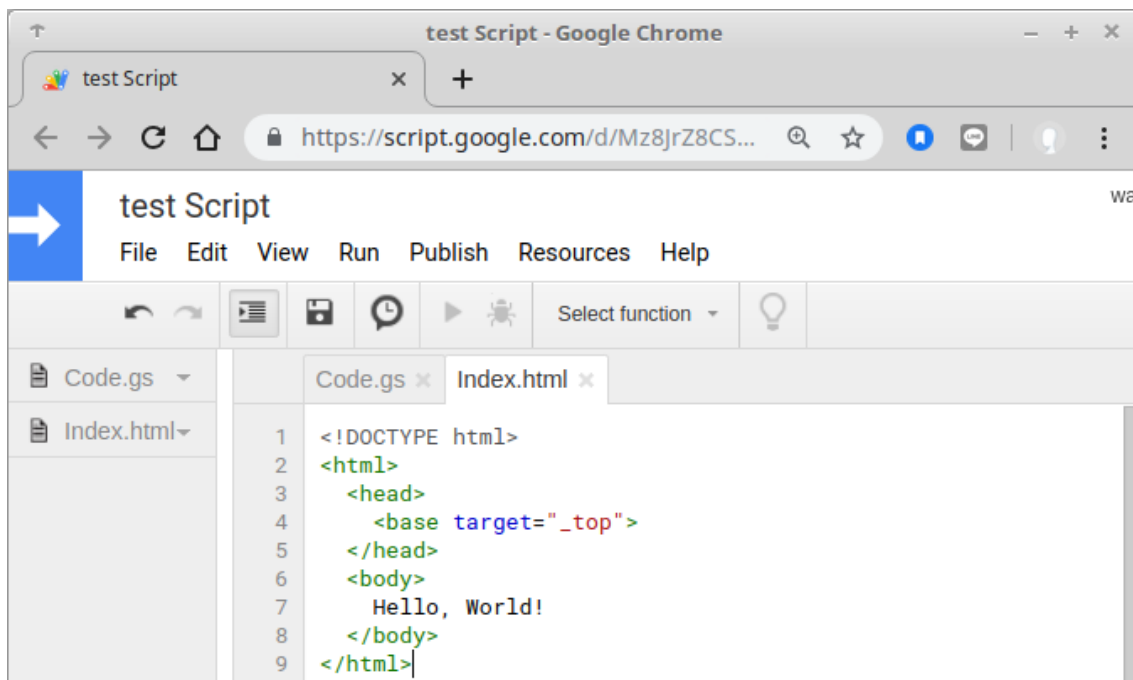
createHtmlOutputFromFile(filename) - เมธอดในคลาส HtmlService

<https://developers.google.com/apps-script/reference/html/html-service#createhtmloutputfromfilefilename>

createHtmlOutputFromFile() ใช้สร้างวัตถุ HtmlOutput ตัวใหม่ จากไฟล์ HTML ที่อยู่ในโปรเจ็ค

Google Apps Script

ตัวอย่าง - โปรเจ็ค Google Apps Script สร้างไฟล์ 2 ไฟล์ ดังต่อไปนี้



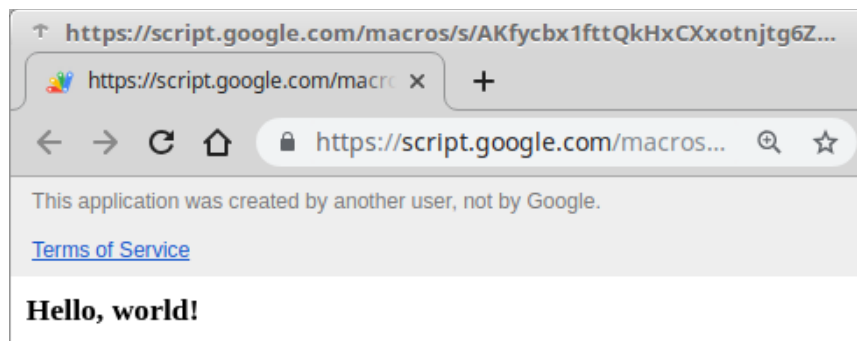
ไฟล์ Code.gs - มีโค้ดดังต่อไปนี้

```
function doGet() {
    // สร้างโค้ด Html จากไฟล์ Index(.html) ที่อยู่ในโปรเจ็ค
    return HtmlService.createHtmlOutputFromFile('Index') ;
}
```

ไฟล์ Index.html - มีโค้ด HTML ดังต่อไปนี้

```
<!DOCTYPE html>
<html>
  <head>
    <base target="_top">
  </head>
  <body>
    Hello, World!
  </body>
</html>
```


ผลลัพธ์เมื่อ Deploy as web app



6.4. คลาส HtmlTemplate

คลาส HtmlTemplate

<https://developers.google.com/apps-script/reference/html/html-template.html>

คลาส HtmlTemplate เป็นวัตถุ Template ที่ใช้สร้างโค้ด HTML ที่สามารถสร้างสรรค์เนื้อหาเพิ่มเติมได้ เช่น ฟังก์ชันแป้น เป็นต้น

6.5. createTemplate() และ evaluate()

createTemplate(html) – เมธอดในคลาส HtmlService

<https://developers.google.com/apps-script/reference/html/html-service#createTemplatehtml>

evaluate() – เมธอดในคลาส HtmlTemplate

<https://developers.google.com/apps-script/reference/html/html-template.html#evaluate>

เมธอด createTemplate() ใช้สร้างวัตถุ HtmlTemplate ตัวใหม่จากโค้ด HTML

เมธอด evaluate() ใช้สรุป HtmlTemplate หรือประมวลผลสุดท้าย ที่อาจมีการแต่งเติม เช่น *แบบตัวแปรติดไปได้ด้วย* โดยสรุปหรือคืนค่ากลับมาเป็นวัตถุ HtmlOutput ก่อนส่งให้ Browser

ตัวอย่างการใช้งาน

```
function doGet() {  
    var template = HtmlService.createTemplate('<?= foo ?>') ;  
    // template เป็นตัวแปรวัตถุอยู่แล้ว  
    // template.foo ก็คือ เพิ่ม Key ชื่อ foo พร้อมกับ Value ให้กับตัวแปรวัตถุเพิ่มเติม  
    template.foo = 'Hello World!' ;  
    Logger.log(template.getCode()) ;           // พิมพ์ : (โค้ด Javascript)  
    Logger.log(template.getRawContent()) ;      // พิมพ์ : <?= foo ?>  
    Logger.log(template.evaluate()) ;           // พิมพ์ : HtmlOutput *****  
    Logger.log(template.evaluate().getContent()) ; // พิมพ์ : Hello World!  
}
```

Scriptlet `<?= foo ?>` จะแสดงค่าของตัวแปร `foo` ผลลัพธ์ ก็คือ Hello World! ถ้า Deploy as web app จะได้หน้าเว็บที่มีข้อความ Hello World!

หมายเหตุ : เมธอดที่น่าสนใจ

`getCode()` – เมธอดในคลาส `HtmlTemplate`

<https://developers.google.com/apps-script/reference/html/html-template.html#getcode>

ใช้จับข้อความเป็นโค้ด Javascript ที่มาจากไฟล์เทมเพลตที่สามารถ Evaluate ได้

`getRawContent()` – เมธอดในคลาส `HtmlTemplate`

<https://developers.google.com/apps-script/reference/html/html-template.html#getrawcontent>

ใช้จับเนื้อหาในเทมเพลตที่ยังไม่ได้ประมวลผล

6.6. createTemplateFromFile()

`createTemplateFromFile(filename)` – เมธอดในคลาส `HtmlTemplate`

<https://developers.google.com/apps-script/reference/html/html-service#createTemplateFromFilefilename>

HTML Service: Templated HTML

<https://developers.google.com/apps-script/guides/html/templates>

`createTemplateFromFile` ใช้สร้างวัตถุ `HtmlTemplate` จากไฟล์ HTML

เมธอด `createTemplateFromFile()` ใช้สร้างวัตถุ `HtmlTemplate` ตัวใหม่จาก HTML ข้างต้นซึ่งต้องจบด้วยเมธอด `evaluate` ก่อนคืนค่ากลับไป เพื่อสรุปเป็น `HtmlOutput`

`createTemplateFromFile()` ใช้งานเหมือนกับ `createTemplate()` ต่างกันเพียงจะสร้าง `HtmlTemplate` จากไฟล์หรือจากโค้ด HTML เท่านั้น

6.6.ก.) ตัวอย่าง : แบนตัวแปรไปกับวัตถุ `HtmlTemplate`

สร้างโปรเจ็ค Google Apps Script และใช้งานแบบ Web App โดยมีไฟล์และโค้ดดังต่อไปนี้

ไฟล์ [page.html](#)

```
<html>
  <head>
    <base target="_top">
  </head>
  <body>
    <!-- ฟังก์ชัน Scriptlet รับตัวแปรจาก Apps Script มาใช้ใน HTML -->
    <h1><?= title ; ?></h1>
    <p> My paragraph </p>
  </body>
</html>
```

```
function doGet() {
    // สร้างวัตถุ HtmlTemplate หรือ ก่อนของโค้ด HTML จับใส่ตัวแปรไว้
    var tmp = HtmlService.createTemplateFromFile("page") ;

    // tmp เป็นตัวแปรวัตถุอยู่แล้ว
    // tmp.title ก็คือ เพิ่ม Key ชื่อ title ให้กับตัวแปรวัตถุเพิ่มเติม
    tmp.title = "My title" ;

    Logger.log(tmp) ; // ดูผลที่ Logs ----- > [ 01 ]
    Logger.log(tmp.getRawContent()) ; // ดูผลที่ Logs ----- > [ 02 ]
    Logger.log(tmp.evaluate()) ; // ดูผลที่ Logs ----- > [ 03 ]
    Logger.log(tmp.evaluate().getContent()) ; // ดูผลที่ Logs ----- > [ 04 ]

    return tmp.evaluate() ;
}
```

ผลที่ Logs

```
Logs

[ 01 ] {title=My title}
[ 02 ] <html>
        <head>
            <base target="_top">
        </head>
        <body>
            <h1><?= title ; ?></h1> <!-- ผัง Scriptlet รับตัวแปรจาก GAS มาใช้ใน HTML -->
            <p> My paragraph </p>
        </body>
    </html>
[ 03 ] HtmlOutput // วัตถุ HtmlOutput เป็นก่อนของโค้ด HTML + อื่นๆ
[ 04 ] <!DOCTYPE html>
        <html>
            <head>
                <base target="_top">
            </head>
            <body>
                <h1>My title</h1>
                <p> My paragraph </p>
            </body>
        </html>
```

ผลที่ Browser เมื่อ Deploy as web app

My title

My paragraph

6.6.ข.) ตัวอย่าง : การทำ Mail merge email

สร้างโปรเจ็ค Google Apps Script แบบ Web App โดยมีไฟล์และโค้ดดังต่อไปนี้

ไฟล์ Google Sheets มีข้อมูลในคอลัมน์ **อีเมล, ชื่อ, นามสกุล** และ **เบอร์โทรศัพท์**

	A	B	C	D
1	Email	Name	Last Name	Phone
2	info.poeclub@gmail.com	นภาพร	คุณติลกเศวต	081-111-1111
3	wasan@snpfood.com	wasan	snp	095-222-2222
4	wasankds.poeclub@gmail.com	PoE	Club	031-333-3333

ไฟล์ emailTmp.html

```
<html>

<head>
  <base target="_top">
</head>

<body>
  <p>เรียนคุณ <?= fn ?> <?= ln ?></p>
  <p>หมายเลขโทรศัพท์ของคุณ ที่เราบันทึกไว้คือ <?= phone ?></p>
  <p>กรุณาตอบกลับถ้าหากไม่ใช่</p>
  <p>ขอแสดงความนับถือ<br>ระบบส่งอีเมลอัตโนมัติ</p>
</body>

</html>
```

ไฟล์ .gs

```
function myFunction() {
  var email = 0 ;           //
  var first = 1 ;           //
  var last = 2 ;            //
  var phone = 3 ;           //

  // สร้างวัตถุ HtmlTemplate จาก HTML ข้างต้นในโปรเจ็ค
  var emailTemp = HtmlService.createTemplateFromFile("emailTmp") ;

  var ws = SpreadsheetApp.getActiveSpreadsheet().getSheetByName("DataEmail") ;
  var data = ws.getRange("A2:D"+ws.getLastRow()).getValues() ;

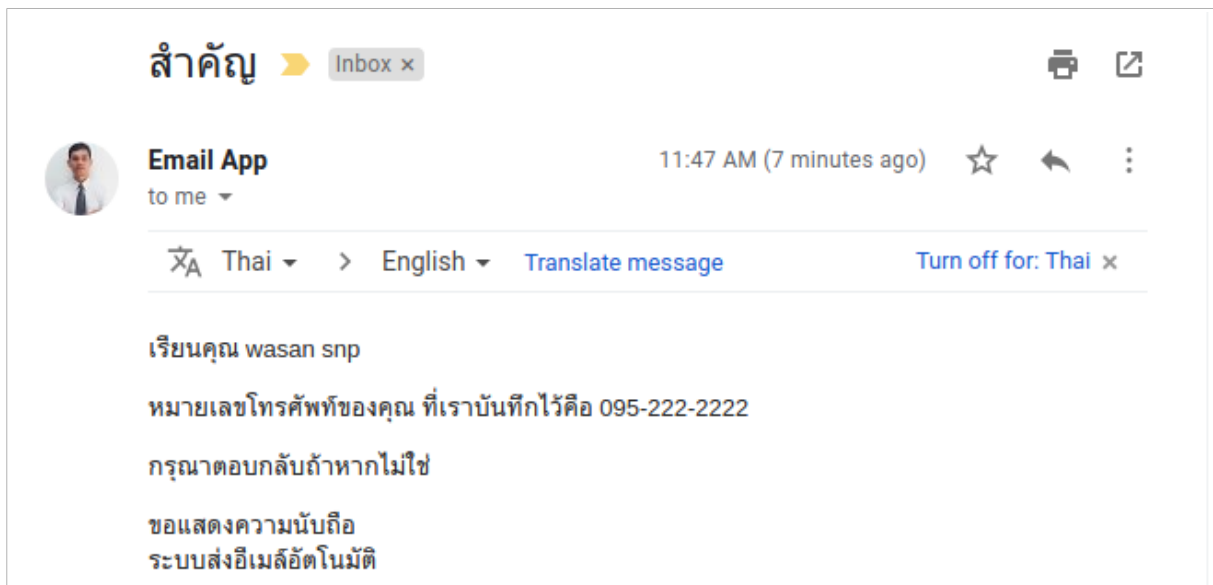
  // วนลูปส่ง ใส่ตัวแปรและค่าที่จะบรรทัด แแนบไปกับวัตถุ HtmlTemplate
  // และ ส่งแมสส์ที่ละบรรทัดด้วย
  data.forEach(function(row){

    emailTemp.fn = row[first] ;           // จับค่าของ ชื่อ ไปใส่
    emailTemp.ln = row[last] ;            // จับค่าของ นามสกุล ไปใส่
    emailTemp.phone = row[phone] ;        // จับค่าของ เบอร์โทร ไปใส่

    // Evaluate >> จับเนื้อหาในไฟล์ HTML ที่เป็น Template
    var htmlMessage = emailTemp.evaluate().getContent() ;
    Logger.log(htmlMessage) ;              // ดูผลที่ Logs ----- >
```

```
// ส่งเมล - อยู่ในลูป foreach จะวนส่งทีละ Record
GmailApp.sendEmail(
    row[email] ,
    "สำคัญ" ,
    "อีเมลของคุณไม่สนับสนุน HTML" ,
    { name : "Email App", htmlBody:htmlMessage }
) ; // End - sendMail
}); // End - foreach
}
```

ผล - อีเมลถูกส่งออกไป ผู้รับได้รับเมลลักษณะตามภาพ



ผลที่ Logs

```
Logs

[ 01 ]
<!DOCTYPE html>
<html>
<head>
  <base target="_top">
</head>

<body>
  <p>เรียนคุณ นภาพร คุณดิลกเสวต</p>
  <p>หมายเลขโทรศัพท์ของคุณ ที่เราบันทึกไว้คือ 081-111-1111</p>
  <p>กรุณาตอบกลับถ้าหากไม่ใช่</p>
  <p>ขอแสดงความนับถือ<br>ระบบส่งอีเมลอัตโนมัติ</p>
</body>
</html>
```

```
[ 02 ]
<!DOCTYPE html>
<html>
  <head>
    <base target="_top">
  </head>

  <body>
    <p>เรียนคุณ wasan snp</p>
    <p>หมายเลขโทรศัพท์ของคุณ ที่เราบันทึกไว้คือ 095-222-2222</p>
    <p>กรุณาตอบกลับถ้าหากไม่ใช่</p>
    <p>ขอแสดงความนับถือ<br>ระบบส่งอีเมลอัตโนมัติ</p>
  </body>
</html>
```

```
[ 03 ]
<!DOCTYPE html>
<html>
  <head>
    <base target="_top">
  </head>

  <body>
    <p>เรียนคุณ PoE Club</p>
    <p>หมายเลขโทรศัพท์ของคุณ ที่เราบันทึกไว้คือ 031-333-3333</p>
    <p>กรุณาตอบกลับถ้าหากไม่ใช่</p>
    <p>ขอแสดงความนับถือ<br>ระบบส่งอีเมลอัตโนมัติ</p>
  </body>
</html>
```

หมายเหตุ

การส่งเมลด้วย Apps Script มีข้อจำกัดตามประเภทของบัญชี Google เช่น บัญชี Gmail ส่งได้ 100 อีเมลต่อ 24 ชม. G Suite Basic ส่งได้ 1,500 อีเมลต่อ 24 ชม. เป็นต้น

บทที่ 7

รันสคริปต์ฝั่ง Server-side



เมื่อหน้า Web App ถูกโหลดจนจบแล้ว

ถ้าโค้ด HTML ของ Web App มี Scriptlets

Scriptlets ก็ถูกรันจนจบแล้วเช่นกัน

จากนั้น ถ้าเราต้องการรันโค้ดของฝั่ง Server-side

ก็คือโค้ด Google Apps Script

สามารถทำได้โดย

เขียน Javascript เรียกใช้ Javascript API

`google.script.run`

7.1. คลาส google.script.run

คลาส google.script.run (Client-side API)

<https://developers.google.com/apps-script/guides/html/reference/run>

คลาส `google.script.run` เป็นการสื่อสารทางเดียวจากฝั่ง Client-side เป็น JavaScript API และเป็นสิ่งที่ไม่มีในเซิร์ฟเวอร์ HTML ที่สามารถเรียกฟังก์ชัน Google Apps Script ในฝั่ง Server-side

HTML Service: Communicate with Server Functions

<https://developers.google.com/apps-script/guides/html/communication>

7.2. withSuccessHandler()

`withSuccessHandler(function)` – เมล็ดในคลาส `google.script.run`

<https://developers.google.com/apps-script/guides/html/reference/run#withsuccesshandlerfunction>

`withSuccessHandler()` เรียกฟังก์ชันในฝั่ง Client-side (พารามิเตอร์ `function`) ถ้าฟังก์ชันในฝั่ง Server-side คืนค่ากลับมา โดยค่าแรกที่คืนกลับมา จะถูกส่งผ่านให้ฟังก์ชันในฐานะอาร์กิวเมนต์ตัวที่ 1 และ `user object` (ถ้ามี) จะเป็นอาร์กิวเมนต์ตัวที่ 2

ตัวอย่าง – โปรเจ็คต่อไปนี้จะแสดงจำนวนอีเมลที่ยังไม่ได้อ่าน ไว้ที่หน้าเว็บ

ไฟล์ `.html`

```
<!DOCTYPE html>
<html>
  <head>
    <base target="_top">
    <script>

      function onSuccess(numUnread) {
        var div = document.getElementById('output') ;
        div.innerHTML = 'You have ' + numUnread
                        + ' unread messages in your Gmail inbox.' ;
      } // Close function
      google.script.run.withSuccessHandler(onSuccess).getUnreadEmails() ;

    </script>
  </head>

  <body>
    <div id="output"></div>
  </body>
</html>
```

ไฟล์ `.gs`

```
function doGet() {
  return HtmlService.createHtmlOutputFromFile('Index') ;
}

function getUnreadEmails() {
  return GmailApp.getInboxUnreadCount() ;
}
```


This application was created by another user, not by Google.

[Terms of Service](#)

You have 87 unread messages in your Gmail inbox.

7.3. withFailureHandler()

`withFailureHandler(function)` – เมธอดในคลาส `google.script.run`

<https://developers.google.com/apps-script/guides/html/reference/run#withfailurehandlerfunction>

เรียกรันฟังก์ชันในฝั่ง Client-side (พารามิเตอร์ `function` ที่ระบุ) ถ้าฟังก์ชันในฝั่ง Server-side เกิด Error วัตถุ `Error` จะถูกส่งผ่านให้ฟังก์ชันในฐานะอาร์กิวเมนต์ตัวที่ 1 และ `user object` (ถ้ามี) จะเป็นอาร์กิวเมนต์ตัวที่ 2 หากไม่มีตัวจัดการความผิดพลาด ข้อผิดพลาดจะแสดงที่ Console ของ Javascript เพื่อที่จะแทนที่ดังกล่าว ให้ใช้ `withFailureHandler(null)` เพื่อจัดการ Error โดยไม่มีอะไรเกิดขึ้นเลย

วัตถุ Error ของ Javascript

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Error

ตัวอย่าง

ไฟล์ .html

```
<!DOCTYPE html>
<html>
  <head>
    <base target="_top">
    <script>

      function onFailure(numUnread) {
        var div = document.getElementById('output') ;
        div.innerHTML = "ERROR: " + error.message ;
      }

      google.script.run.withSuccessHandler(onFailure).getUnreadEmails() ;

    </script>

  </head>
  <body>
    <div id="output"></div>
  </body>
</html>
```

ไฟล์ .gs

```
function doGet() {  
  return HtmlService.createHtmlOutputFromFile('Index') ;  
}  
  
function getUnreadEmails() {  
  // พิมพ์ชื่อเมลอดผิด จะเกิด Error  
  return GmailApp.getInboxUnreadCount() ;  
}
```

ผล

This application was created by another user, not by Google.

[Terms of Service](#)

ERROR: TypeError: GmailApp.getInboxUnreadCount is not a function

7.4. withUserObject()

withUserObject(object)

<https://developers.google.com/apps-script/guides/html/reference/run#withuserobjectobject>

สร้างตัวแปรวัตถุเพื่อส่งผ่านเป็นอาร์กิวเมนต์ตัวที่ 2 (พารามิเตอร์ object) ให้กับ withSuccessHandler หรือ withFailureHandler วัตถุนี้ จะไม่ส่งไปยัง Server-side ฉะนั้น ให้ฟังก์ชันในฝั่ง Client-side ตอบสนองเนื้อหา ที่ Client ติดต่อไปยัง Server

ไฟล์ .html

```
<!DOCTYPE html>  
<html>  
  <head>  
    <base target="_top">  
    <script>  
      function updateButton(email, button) {  
        button.value = 'Clicked by ' + email ;  
      }  
    </script>  
  </head>  
  <body>  
    <input type="button" value="Not Clicked"  
      onclick="google.script.run.withSuccessHandler(updateButton)  
        .withUserObject(this)  
        .getEmail()" />  
    <input type="button" value="Not Clicked"  
      onclick="google.script.run.withSuccessHandler(updateButton)  
        .withUserObject(this)  
        .getEmail()" />  
  </body>  
</html>
```

ไฟล์ .gs

```
function doGet() {  
  return HtmlService.createHtmlOutputFromFile('Index') ;  
}  
  
function getEmail() {  
  return Session.getActiveUser().getEmail() ;  
}
```

ผล – แรกเริ่มบนปุ่มปรากฏข้อความ "Not Clicked" แต่เมื่อคลิกปุ่มแล้วจะปรากฏข้อความ "Clicked by (email ของยูสเซอร์)"

This application was created by another user, not by Google.

[Terms of Service](#)

Not Clicked

Clicked by wasankds@gmail.com

บทที่ 8

ใช้งาน Web App ใน iFrame



8.1. จับใส่ iframe

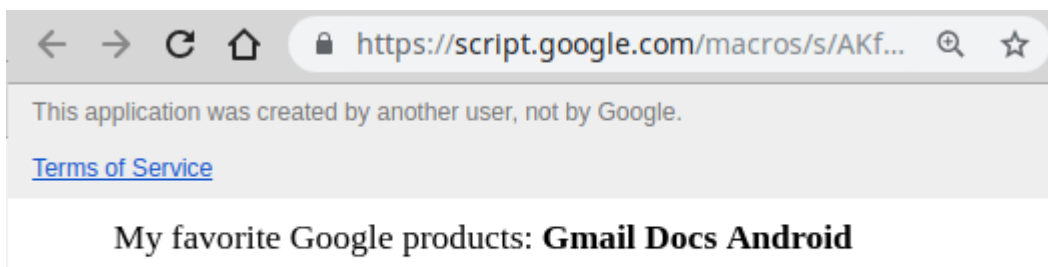
อีกวิธีในการนำ Web app ไปใช้งานก็คือ ฝังลิงค์ของโปรเจ็คไว้ใน iframe ด้วยวิธีนี้ จะทำให้เราสามารถนำโปรเจ็คไปใช้ในเว็บใดก็ได้ ใช้ในเว็บ Google Sites ที่ทำเองก็ได้

8.1.ก.) ฝัง iframe ที่หน้าเว็บ

ให้เรา [สร้างเว็บอีก 1 หน้า](#) สำหรับใช้ฝังโปรเจ็ค Apps Script ไว้ใน iframe ตัวอย่างโค้ดมีดังต่อไปนี้ นำลิงค์สำหรับใช้งาน Web App ใส่ลงในแท็ก iframe

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
  </head>
  <body>
    <!-- ลิงค์ใช้งานโปรเจ็ค -->
    <iframe src="https://script.google.com/macros/s/....exec"
      frameborder="0" width="100%" height="500px"></iframe>
  </body>
</html>
```

แต่ก่อนจะใช้งานหน้าเว็บข้างต้น ให้แก้โปรเจ็ค Apps Script ก่อน เพื่อเอากรอบด้านบนในหน้า Web App ที่มีข้อความจาก Google ออก (ตัวอย่างตามภาพ) และอนุญาตให้นำไปใช้ใน iframe



ให้เราแก้โปรเจ็ค Apps Script ดังต่อไปนี้

8.1.ข.) แก้ไขโค้ดในไฟล์ .gs ของโปรเจ็ค

การจับ Web App ใส่ iFrame จะใช้ Deploy as web app ในโหมด dev ไม่ได้ ถ้าใช้จะขึ้นว่า script.google.com refused to connect ต้องใช้โหมด exec เท่านั้น

นอกจากนี้ เมื่อใช้โหมด exec แล้ว ต้องใช้เมธอด setXFrameOptionsMode() กับหน้าเว็บที่ส่งกลับจาก doGet() ด้วย เพื่ออนุญาตให้นำไปใช้ใน iFrame

ตัวอย่าง

ไฟล์ .gs - ของโปรเจ็ค Apps Script

```
function doGet() {  
  return HtmlService.createTemplateFromFile('Test')  
    .evaluate()  
    // .setSandboxMode(HtmlService.SandboxMode.IFRAME)  
    .setXFrameOptionsMode(HtmlService.XFrameOptionsMode.ALLOWALL) ;  
}
```

หมายเหตุ :

การใช้หรือไม่ใช้ .setSandboxMode(HtmlService.SandboxMode.IFRAME) ไม่มีผลกับการนำไปใช้ใน iFrame

ไฟล์ Test.html - ของโปรเจ็ค Apps Script

```
<div>  
  <p><b>Username: </b><input type="text" name="username"/></p>  
  <p><b>Password: </b><input type="text" name="pssword"/></p>  
  <input type="button" id="loginBtn" value="Login"  
    onclick="this.value = 'Loading...';onLoginJS(this.parentNode)"/>  
</div>
```

เมื่อ Deploy as web app จะได้หน้าเว็บตามภาพ



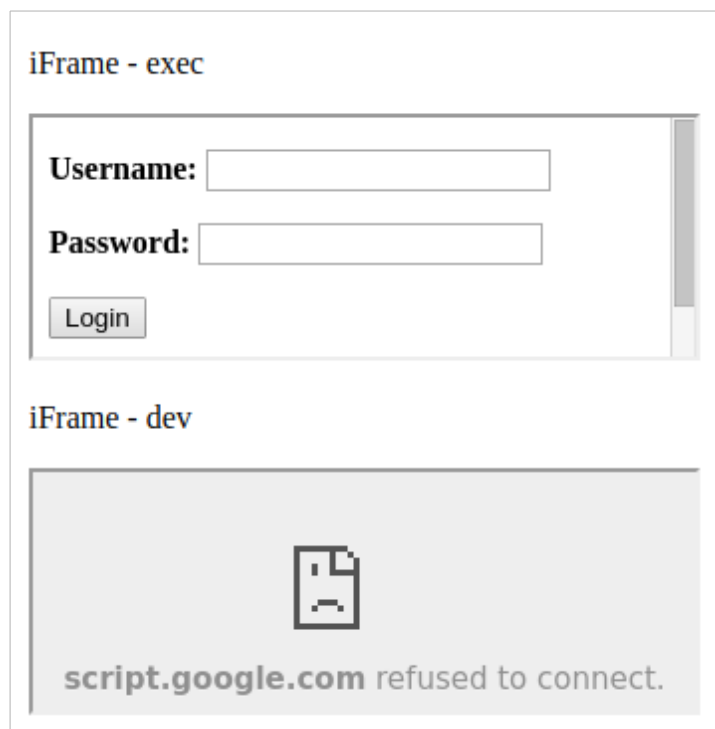
The screenshot shows a web form with two text input fields. The first field is labeled 'Username:' and the second is labeled 'Password:'. Below these fields is a button labeled 'Login'.

เมื่อโค้ดต่างๆของโปรเจ็ค App Script เสร็จแล้ว จากนั้น เราสามารถนำไปใช้ในเว็บอื่น ได้โดยใช้ iFrame
ดังนี้

ไฟล์ CallPoint.Html - ใช้ Web App โดยการใช้ iFrame

```
<html>  
  <body>  
    <p>iFrame - exec</p>  
    <iframe src= "https://script.google.com/macros/s/(Project_Deployment_URL)/exec"  
      frameborder="1" width="50%" height="120px"></iframe>  
  
    <p>iFrame - dev</p>  
    <iframe src= "https://script.google.com/macros/s/(Project_Deployment_URL)/dev"  
      frameborder="1" width="50%" height="120px"></iframe>  
  </body>  
</html>
```

ผล – สังเกตว่า โหมด exec ทำงานได้ แต่ dev ไม่ทำงาน



8.1.ค.) แก้ไขโค้ดในไฟล์ .html ของโปรเจ็ค

ต่อจากข้อก่อนหน้า เราสามารถนำโปรเจ็คไปใช้ใน iframe ได้แล้ว

แต่ว่า... เมื่อเรากดปุ่มใดๆ ในหน้า Web App เพื่อจะประมวลผลใดๆ Browser มักจะโหลดลิงค์ของ Web App มาใช้ สรุปก็คือ ไม่ได้ใช้ iframe อีกแล้ว

ทั้งนี้ เป็นที่การตั้งค่าที่ไฟล์ Html ของโปรเจ็ค Apps Script

วิธีแก้ ไฟล์ Test.html (ของโปรเจ็ค Apps Script) ให้แก้เป็น `<base target="_self">` ดังนี้ ซึ่งปกติใช้เป็น `_top`

ไฟล์ Test.html (ของโปรเจ็ค Apps Script)

```
<html>
  <head>
    <base target="_self">
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  </head>
  <body>
    ....
```


8.1.จ.) setXFrameOptionsMode()

setXFrameOptionsMode(mode) – เมธอดในคลาส HtmlOutput

[https://developers.google.com/apps-script/reference/html/html-output#setXFrameOptionsMode\(XFrameOptionsMode\)](https://developers.google.com/apps-script/reference/html/html-output#setXFrameOptionsMode(XFrameOptionsMode))

เมธอด setXFrameOptionsMode() ใช้เพื่อสถานะของ X-Frame-Options header ของหน้าเว็บ ซึ่งใช้ป้องกัน clickjacking

พารามิเตอร์ **mode** ก็คือ Enum setXFrameOptionsMode (ดูตารางด้านล่าง)

การตั้งเป็น XFrameOptionsMode.ALLOWALL เป็นการอนุญาตให้หน้าเว็บไปใช้ใน iframe ซึ่งผู้พัฒนาจะต้องไปเขียนป้องกัน clickjacking เอง

ถ้าเราไม่ใช้เมธอด setXFrameOptionsMode() กับหน้าเว็บของเรา Apps Script จะเซตเป็นค่าปริยายให้ก็คือ XFrameOptionsMode.DEFAULT ซึ่งจะไม่สามารถนำไปใช้ใน iframe ได้

ตัวอย่าง

```
function doGet(){
  var tmp = HtmlService.createTemplateFromFile('index') ;
  return tmp.evaluate()
    .setXFrameOptionsMode(HtmlService.XFrameOptionsMode.ALLOWALL) ;
}
```

Enum XFrameOptionsMode

<https://developers.google.com/apps-script/reference/html/x-frame-options-mode>

เข้าถึงโดย **HtmlOutput.setXFrameOptionsMode(mode)**.

Property	Type	Description
ALLOWALL	Enum	No X-Frame-Options header will be set. This will let any site iframe the page, so the developer should implement their own protection against clickjacking.
DEFAULT	Enum	Sets the default value for the X-Frame-Options header, which preserves normal security assumptions. If a script does not set an X-Frame-Options mode, Apps Script uses this mode as the default.

8.1.จ.) setSandboxMode(mode)

setSandboxMode(mode)

<https://developers.google.com/apps-script/reference/html/html-output#setsandboxmodemode>

Enum SandboxMode – ตอนนี้ใช้ได้เฉพาะ IFRAME

<https://developers.google.com/apps-script/reference/html/sandbox-mode>

เพื่อป้องกันยูสเซอร์จาก HTML หรือ Javascript ที่ปองร้าย Apps Script ใช้ iFrames ทำ Sandbox

```
function doGet() {
  var template = HtmlService.createTemplateFromFile('top');
  return template.evaluate().setSandboxMode(HtmlService.SandboxMode.IFRAME);
}
```

ไฟล์ .html

```
<!DOCTYPE html>
<html>
  <body>
    <div>
      <a href="http://google.com" target="_top">Click Me!</a>
    </div>
  </body>
</html>
```

เปลี่ยนอแอทธิวิธ target ในแท็ก base จาก _top เป็น _blank ก็ได้

```
<head>
  <base target="_top">
</head>
```

8.2. จับลิงค์ของโปรเจ็คโดยใช้สคริปต์

เซอร์วิส Script

<https://developers.google.com/apps-script/reference/script>

เป็นบริการที่ใช้เข้าถึง Trigger และ การเผยแพร่สคริปต์

คลาส ScriptApp – ส่วนหนึ่งของเซอร์วิส Script

<https://developers.google.com/apps-script/reference/script/script-app>

ใช้เข้าถึงและจัดการการเผยแพร่ Script และ Triggers อนุญาตให้ยูสเซอร์สร้าง Triggers และ ควบคุม

การเผยแพร่สคริปต์ในแบบการให้บริการ (Script as a service)

getService() – เมธอดในคลาส ScriptApp

<https://developers.google.com/apps-script/reference/script/script-app#getservice>

จับวัตถุ(คืนค่าเป็น **คลาส Service**) ที่ถูกใช้เป็นตัวควบคุมการเผยแพร่สคริปต์แบบ Web App

คลาส Service

<https://developers.google.com/apps-script/reference/script/service>

ใช้เข้าถึงและจัดการการเผยแพร่สคริปต์

getURL() – เมธอดในคลาส Service

<https://developers.google.com/apps-script/reference/script/service#geturl>

คืนค่าเป็น String ที่เป็น URL ของ Web App ถ้าเปิดใช้งานอยู่(Deployed) แต่ถ้าไม่ได้เปิดใช้งานจะคืน

ค่าเป็น **null**

เราสามารถจับลิงค์ของโปรเจ็ค ได้โดยใช้โค้ดดังต่อไปนี้

```
function myFunction() {
  var projectURL = ScriptApp.getService().getURL() ;
  Logger.log(projectURL) ;    // ผลลัพธ์ Logs ----- >
}
```

ผล

Logs

[] <https://script.google.com/macros/s/ABCDEFGHIJKLMNOPQRSTUVWXYZ/exec>

บทที่ 9
Web App
แบบหลายหน้า



9.1. เปลี่ยนหน้าโดยเขียนโค้ดเพื่อแทรกโค้ด HTML

(ประยุกต์มาจาก) [More than one HTML or script file in the the same App Project and add Login?](https://stackoverflow.com/questions/31638429/google-app-more-than-one-html-or-script-file-in-the-the-same-app-project-and-a)

<https://stackoverflow.com/questions/31638429/google-app-more-than-one-html-or-script-file-in-the-the-same-app-project-and-a>

เทคนิคการเปลี่ยนหน้า Web App ในข้อนี้ ใช้วิธีเขียนโค้ด Javascript เพื่อสร้างโค้ด HTML จากไฟล์หรือจาก HTML String จากนั้น ก็เขียนลงไปในอิลีเมนต์ HTML ที่เตรียมไว้สำหรับแสดงผลลัพธ์

ตัวอย่างที่ยกมานี้ ภาพรวมของโปรเจ็ค เริ่มด้วยการสร้างหน้า Login(ตามภาพ) และเมื่อ Login สำเร็จ จะมีหน้าข้อมูลของยูสเซอร์ปรากฏ แต่ถ้า Login ไม่สำเร็จ จะปรากฏข้อความแจ้งความผิดพลาด ซึ่งจากนั้นสามารถ Login ใหม่ได้เลย

กรอกชื่อ รหัสยูสเซอร์(12345) จากนั้นคลิกปุ่ม "Login" (ห้ามกด Enter)

User id:

Login

9.1.ก.) โค้ดของโปรเจ็ค

ไฟล์ Code.gs

```
function doGet() {
  return HtmlService.createTemplateFromFile('Index')
    .evaluate()
    .setXFrameOptionsMode(HtmlService.XFrameOptionsMode.ALLOWALL) ;
}

function onLogin(form) { // (1.) < ----- รับเมื่อคลิกปุ่ม Login

  Logger.log(form) ; // { userid=12345 } <-- สิ่งที่ได้จากการคลิกปุ่ม

  if (form.userid == "12345") { // ถ้า Login ถูกต้อง
    var template = HtmlService.createTemplateFromFile('Response') ;

    template.userid = "รหัสผู้ใช้งาน : " + form.userid ;
    template.username = "ชื่อผู้ใช้งาน(User name) : wasankds" ;
    template.firstname = "ชื่อ: วสันต์ " ;
    template.lastname = "นามสกุล : คุณดิลกเสวต" ;

    return template.evaluate().getContent() ; // คืนกลับไปเป็นโค้ด Html ----- > (2.1)
  }
  else // ถ้า Login ผิดพลาด
  {
    // คืนค่าเป็น Error ที่เรากำหนดรายละเอียดเองได้
    throw " : ไม่มีข้อมูลในฐานข้อมูล โปรดลองใหม่อีกครั้ง " ; // คืนกลับไปเป็น Error --- > (2.2)
  }
}
```

```

<head>
  <base target = "_self">
  <style> p b {width: 100px ; display: inline-block ;} </style>
</head>

<body>
<!-- ----- ใช้สำหรับแสดง Error ----- -->
<div id="errors" style="color:red ;"></div><p></p>
<!-- ----- ใช้สำหรับแสดง Error ----- -->

<!-- ----- ส่วนที่ปรากฏ ----- -->
<div style="color:blue;">กรอกชื่อ รหัสยูสเซอร์(12345) จากนั้นคลิกปุ่ม "Login" (ห้ามกด Enter)</div>

<div id="content"> <!-- สำหรับแสดงผลลัพธ์ จาก Response.html -->
  <form> <!-- parentNode ของปุ่ม Login ก็คือ ก้อน Form -->
    <p><b>User id: </b><input type="text" name="userid"/></p>
    <input type="button" id="loginBtn" value="Login"
      onclick="this.value = 'Loading...';onLoginJS(this.parentNode)"/>
  </form> <!-- parentNode ของปุ่ม Login ก็คือ ก้อน Form -->
</div> <!-- สำหรับแสดงผลลัพธ์ จาก Response.html -->
<!-- ----- ส่วนที่ปรากฏ ----- -->
<script>

// รันเมื่อคลิกปุ่ม Login – รับพารามิเตอร์เป็นก้อน <form> มา แล้วส่งต่อไปให้ฝั่ง Server ประมวลผลต่อ
function onLoginJS(parentnode) {
  document.getElementById('loginBtn').disabled = true ; // Disable ปุ่ม
  google.script.run.withSuccessHandler(loadPage) // ----- > (2.1)
    .withFailureHandler(onLoginFailure) // ----- > (2.2)
    .onLogin(parentnode) ; // ----- > (1)
}

function loadPage(htmlOut) { // (2.1) < -----
  console.log('loadPage ran!') ;
  console.log('htmlOut: ' + htmlOut) ;

  var div = document.getElementById('content') ; // จับไปที่ <div id="content">
  div.innerHTML = htmlOut ; // เขียนเนื้อหาที่มาจากไฟล์ Response.html
  document.getElementById('errors').innerHTML = "" ; // ล้าง <div id="error">
}

// Error ที่มาจากโค้ดในฝั่ง Server เช่น ReferenceError: dd is not defined
//
function onLoginFailure(error) { // (2.2) < -----
  var loginBtn = document.getElementById('loginBtn') ;

  // เช็ทปุ่ม Login – หลังรันสคริปต์แล้ว
  loginBtn.disabled = false ;
  loginBtn.value = 'Login' ;

  var errors = document.getElementById('errors') ;
  errors.innerHTML = 'ผิดพลาด => ' + error.message ;
}
</script>
</body>

```

```
<div style="width:100%; height:200px; border:1px solid red;">
  <p></p>
  <p><em>Response form Response.html</em></p>
  <p><?= userid ?><br>
    <?= username ?><br>
    <?= firstname ?><br>
    <?= lastname ?></p>
</div>
```

9.1.ข.) ไฟล์ที่เรียกใช้ Web App ไปใช้แบบ iFrame

ไฟล์ Test.html เป็นหน้าเว็บที่นำ Web App ไปใช้แบบ iFrame

```
<!DOCTYPE html>
<html>
  <head>
    <base target = "_top">
  </head>
  <body>
    <iframe src="https://script.google.com/macros/s/(Project_Deployment_URL)/exec"
      frameborder="0" width="100%" height="1000px"></iframe>
  </body>
</html>
```

9.1.ค.) ผล

เมื่อเปิดไฟล์ Test.html จะได้ผลลัพธ์ตามภาพ

กรอกชื่อ รหัสยูสเซอร์(12345) จากนั้นคลิกปุ่ม "Login" (ห้ามกด Enter)

User id:

Login

เมื่อยูสเซอร์กรอกรหัสผิด ปรากฏข้อความสีแดงเตือน แต่ยังสามารถกรอก User Id เพื่อ Login ได้เลย

ผิดพลาด => : ไม่มีข้อมูลในฐานข้อมูล โปรดลองใหม่อีกครั้ง

กรอกชื่อ รหัสยูสเซอร์(12345) จากนั้นคลิกปุ่ม "Login" (ห้ามกด Enter)

User id:

Login

ปุ่มมีการตอบสนองเมื่อคลิก โดยเมื่อคลิกปุ่ม Login จะเปลี่ยนจาก Login เป็น Loading และเมื่อประมวลผลเสร็จจะกลับมาเป็น Login อีกครั้ง



เมื่อกรอกรหัสถูก หน้าเว็บจะเปลี่ยนเป็นตามภาพ

กรอกชื่อ รหัสยูสเซอร์(12345) จากนั้นคลิกปุ่ม "Login" (ห้ามกด Enter)

Response form Response.html

รหัสผู้ใช้งาน : 12345

ชื่อผู้ใช้งาน(User name) : wasankds

ชื่อ: วสันต์

นามสกุล : คุณดิลกเศวต

9.2. เปลี่ยนหน้าเว็บโดยส่งพารามิเตอร์เพื่อเปลี่ยนหน้า

เทคนิคการเปลี่ยนหน้า Web App ในข้อนี้ ใช้วิธีส่งพารามิเตอร์ต่อท้าย URL เพื่อเลือกหน้าที่ต้องการดู หรือ ใช้สร้างปุ่มเพื่อเลือกหน้าที่จะลิงค์ไปหา

9.2.ก.) โค้ดทั้งหมดของโปรเจ็ค Apps Script

ไฟล์ Code.gs

```
function doGet(e) {  
  Logger.log(e); // {queryString=v=form, parameter={v=form}, contextPath=,  
                  _parameters={v=[form]}, contentType=-1.0}  
  if(e.parameters.v=='form')  
  {  
    // ถ้าพารามิเตอร์เป็น  
    return loadForm(); // http://(Project_Deployment_URL)?v=form  
  }  
  else // อื่นๆ เข้าหน้า Home  
  {  
    return HtmlService.createTemplateFromFile("Home")  
      .evaluate()  
      .setXFrameOptionsMode(HtmlService.XFrameOptionsMode.ALLOWALL) ;  
  }  
}
```

```
function loadForm() {

    var options = '<option value="1">ตัวเลือก 1</option>' +
                  '<option value="2">ตัวเลือก 2</option>' +
                  '<option value="3">ตัวเลือก 3</option>' ;

    var tmp = HtmlService.createTemplateFromFile("Form") ;
    tmp.choices = options ;

    return tmp.evaluate()
    //      .setSandboxMode(HtmlService.SandboxMode.IFRAME)
    //      .setXFrameOptionsMode(HtmlService.XFrameOptionsMode.ALLOWALL) ;
}
```

ไฟล์ Home.html

```
<!DOCTYPE html>
<html>
  <head>
    <base target="_self">
  </head>
  <body>
    <div>
      <h1>Welcome !</h1>
      <a href="<?= ScriptApp.getService().getUrl();?>?y=form">
        <button id="btn">Form</button>
      </a>
    </div>
  </body>
</html>
```

ไฟล์ Form.html

```
<!DOCTYPE html>
<html>
  <head>
    <base target="_self"> <!-- เปิดลิงค์ในกรอบ iFrame จึงไม่มีข้อความจาก Google -->
  </head>
  <body>
    <h1>Form</h1>
    <p>กรุณาเลือก</p>
    <select>
      <?!= choices ?>
    </select>
  </body>
</html>
```

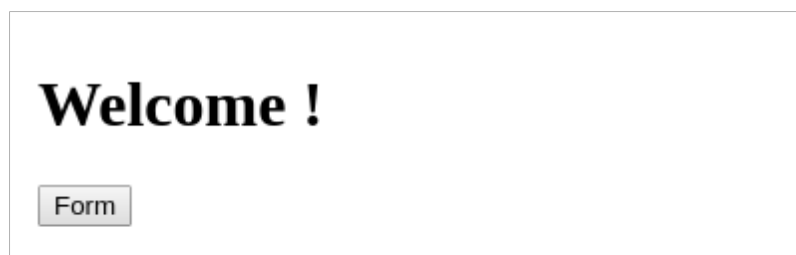
9.2.ข.) ไฟล์ที่เรียกใช้ Web App ไปใช้แบบ iFrame

ไฟล์ **Test.html** เป็นหน้าเว็บที่นำ Web App ไปใช้แบบ iFrame

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <iframe src="https://script.google.com/macros/s/(Project_Deployment_URL)/exec"
            frameborder="0" width="100%" height="1000px"></iframe>
  </body>
</html>
```

9.2.ค.) ผล

เมื่อเปิดไฟล์ **Test.html** จะได้หน้าเว็บตามภาพ ซึ่งเป็นหน้าที่มาจากไฟล์ Home.html ของโปรเจ็ค Apps Script



จากนั้น เมื่อคลิกที่ปุ่ม **Form** หน้าเว็บจะเปลี่ยนไปเป็นตามภาพ ซึ่งมาจากไฟล์ Form.html ของโปรเจ็ค Apps Script



9.2.ง.) ข้อจำกัด

การใช้งาน Web App ใน iFrame มีข้อจำกัดที่การ Redirect เช่น

จากหน้าเริ่มต้น Home เราคลิกปุ่ม Form เพื่อเปลี่ยนมาหน้า Form สามารถทำได้ปกติอย่างที่อธิบายไปในข้อก่อนหน้า

แต่ ... ที่หน้า Form ไม่ว่าจะเปลี่ยนโค้ดยังไง เพื่อให้ Redirect ย้อนไปหน้า Home ไม่สามารถทำได้เลย
เช่น เพิ่มบรรทัดต่อไปนี้ลงไปไฟล์ Form.html

```
<a href="<?= ScriptApp.getService().getUrl();?>">กลับสู่หน้า Home</a>
```

การทำลิงค์เพื่อ Redirect กลับไปหน้า Home ข้างต้น ทำได้เฉพาะการใช้งาน Web App โดยใช้ลิงค์ตรง
ที่ได้จากการ Deployment as web app เท่านั้น

วิธีแก้ไข(ชัด) 1 : โหลดเอกสาร HTML ใหม่ทั้งหมด

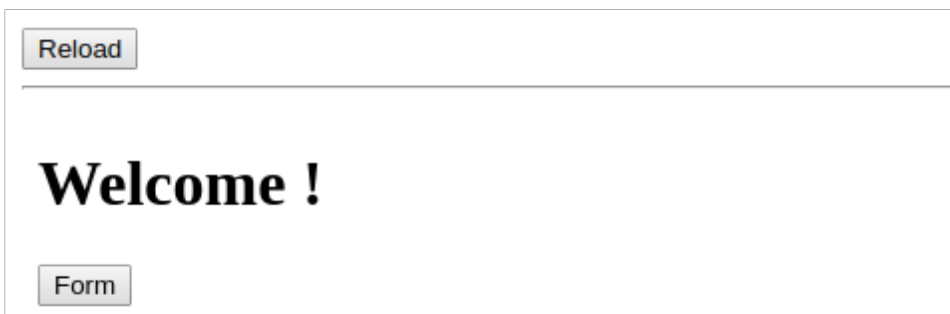
ไฟล์ Test.html ที่เรียกใช้ Web App ใน iFrame เพิ่มปุ่มโหลดหน้าเว็บลงไป

```
<!DOCTYPE html>
<html>
  <head>
  </head>

  <body>
    <!-- เพิ่มปุ่ม Reload หน้าเว็บลงไป -->
    <button id="reload" onclick="document.location.reload();">Reload</button>
    <br><hr>
    <iframe src="https://script.google.com/macros/s/(Project_Deployment_URL)/exec"
      frameborder="0" width="100%" height="1000px"></iframe>

  </body>
</html>
```

ผล



วิธีแก้(ขัด) 2 : โหลดเฉพาะส่วนของ iFrame ใหม่

ไฟล์ **Test.html** ที่เรียกใช้ Web App ใน iFrame เพิ่มปุ่มลงไป และเขียนฟังก์ชัน Javascript ผูกไว้ ตัวอย่าง

```
<!DOCTYPE html>
<html>
  <head>
  </head>

  <body>
    <button id="reload" onclick="reloadframe();">Reload</button>
    <br><br>
    <iframe src="https://script.google.com/macros/s/(Project_Deployment_URL)/exec"
      frameborder="0" width="100%" height="1000px"
      id="fm" ></iframe>
    <br><br>
    <p>ส่วนล่างนอก iFrame</p>
  </body>

  <script>
    function reloadframe(){
      var frame = document.getElementById("fm")
      var webAppURL = frame.src ;    // ได้ URL มา
      frame.src = webAppURL ;
    }
  </script>
</html>
```

ผล - เมื่อคลิกที่ปุ่ม Reload สังเกตว่า ส่วนที่ถูกรีโหลดใหม่ เกิดเฉพาะส่วนของ iFrame



บทที่ 10

URLFetchApp



10.1. URLFetchApp

คลาส URLFetchApp

<https://developers.google.com/apps-script/reference/url-fetch/url-fetch-app>

ใช้จับเนื้อหา และ ใช้สื่อสารกันระหว่างเครื่องคอมพิวเตอร์(host) ในอินเทอร์เน็ต

10.2. เมธอดที่สำคัญ

10.2.ก.) fetch()

คลาส HTTPResponse

<https://developers.google.com/apps-script/reference/url-fetch/http-response.html>

เป็นคลาสที่ให้ผู้สเซอร์เข้าถึงข้อมูลใน HTTP Response หรือ ข้อมูลตอบกลับที่เกิดจากการส่งคำขอแบบ HTTP/HTTPS

fetch(url, params) – เมธอดในคลาส URLFetchApp

<https://developers.google.com/apps-script/reference/url-fetch/url-fetch-app#fetchurl,-params>

fetch() แปลว่า "นำมา" หรือ "เอามา" ใช้สร้างคำขอแบบ HTTP/HTTPS เพื่อไปเอาเนื้อหาจาก URL ที่ระบุ โดยจะคืนค่ากลับมาเป็นวัตถุ [HTTPResponse](#)

พารามิเตอร์

Name	Type	Description
url	String	The URL to fetch.
params	Object	(ไม่จำเป็น) Javascript object specifying advanced parameters as defined below. (ดูตาราง แอดวานซ์พารามิเตอร์)

แอดวานซ์พารามิเตอร์ (Advanced parameters)

Name	Type	Description
contentType	String	the content type (defaults to ' application/x-www-form-urlencoded '). Another example of content type is ' application/xml; charset=utf-8 '.
headers	Object	a Javascript key/value map of HTTP headers for the request
method	String	the HTTP method for the request: get , delete , patch , post , or put . The default is get .
payload (น้ำหนักบรรทุก)	String	the payload (that is, the POST body) for the request. Certain HTTP methods (for example, GET) do not accept a payload. It can be a string, a byte array, a blob, or a Javascript object. A Javascript object is interpreted as a map of form field names to values, where the values can be either strings or blobs.
useIntranet	Boolean	Deprecated. This instructs fetch to resolve the specified URL within the intranet linked to your domain through (deprecated) SDC
validateHttpsCertificates	Boolean	If false the fetch ignores any invalid certificates for HTTPS requests. The default is true.
followRedirects	Boolean	If false the fetch doesn't automatically follow HTTP redirects; it returns the original HTTP response. The default is true.

Name	Type	Description
<code>muteHttpExceptions</code>	Boolean	If <code>true</code> the fetch doesn't throw an exception if the response code indicates failure, and instead returns the <code>HTTPResponse</code> . The default is <code>false</code> . (ถ้ามีปัญหาจะ Error ระหว่างรัน แต่ถ้าตั้งเป็น true จะไม่แสดง Error แม้จะไม่มี Response กลับมา แต่เราเอาเนื้อหา Err มาแสดงได้)
<code>escaping</code>	Boolean	If <code>false</code> reserved characters in the URL aren't escaped. The default is <code>true</code> .

10.2.ข.) `getContent()`

`getContent()` - เมธอดในคลาส `HTTPResponse`

<https://developers.google.com/apps-script/reference/url-fetch/http-response.html#getContent>

ใช้จับข้อมูล Binary จากการตอบคำขอแบบ HTTP/HTTPS โดยจะคืนค่ากลับมาเป็น `Byte[]` - Binary ในอาเรย์ มีลักษณะ เช่น `[60.0, 33.0, 100.0, 111.0, 99.0, ... , ...]`

10.2.ค.) `getTextContent()`

`getTextContent()` - เมธอดในคลาส `HTTPResponse`

<https://developers.google.com/apps-script/reference/url-fetch/http-response.html#getTextContent>

ใช้จับข้อมูลจากการตอบกลับคำขอแบบ HTTP/HTTPS และเปลี่ยนหรือแปลงเป็นข้อความ (String)

ตัวอย่างที่ 1 - โค้ดต่อไปนี้ Logs โค้ด HTML ของหน้าแรก Google ออกมาดู

```
var response = URLFetchApp.fetch("http://www.google.com/");
Logger.log(response.getTextContent());
```

ตัวอย่างที่ 2 - ส่ง HTTP Request ไปที่เว็บ <https://httpbin.org/post> โดยมี Payload เป็น Javascript object ติดไปด้วย

```
// Make a POST request with form data.
// ส่วนของ Payload ทั้งหมด - มีทั้ง Text และ ไฟล์ Text
var resumeBlob = Utilities.newBlob('Hire me!', 'text/plain', 'resume.txt');
var formData = {
    'name' : 'Bob Smith',
    'email' : 'bob@example.com',
    'resume' : resumeBlob // Javascript Object
};

// Because payload is a Javascript object, it is interpreted as form data.
// ( Payload ที่ส่งไปเป็น Javascript Obj. จะถูกแปลงเป็นข้อมูล Form data-ส่งจากฟอร์ม )
// ( No need to specify contentType; it automatically defaults to either
// 'application/x-www-form-urlencoded' or 'multipart/form-data' )
var options = {
    'method' : 'post',
    'payload' : formData
};

var response = URLFetchApp.fetch('https://httpbin.org/post', options);
Logger.log(response.getTextContent()); // ดูผลที่ Logs ----- >
```

Logs

```
{
  "args": {} ,
  "data": "" ,
  "files": {
    "resume": "Hire me!"
  },
  "form": {
    "email": "bob@example.com",
    "name": "Bob Smith" } ,
    "headers": {
      "Accept-Encoding": "gzip,deflate,br" ,
      "Content-Length": "469", "Content-Type": "multipart/form-data; boundary=\"-----BRDt3Odb06PhSFmuuM0nfZRgHkOmG25AR0E78eogERcutObLfr\"", "Host": "httpbin.org",
      "User-Agent": "Mozilla/5.0 (compatible; Google-Apps-Script; beanserver; +https://script.google.com; id: UAEmdDd8rFddydAVRBVLJCiBePMCfpI4BT5g)",
      "X-Amzn-Trace-Id": "Root=1-5e884d17-d2511b928f8716c2b5247a40"
    },
    "json": null ,
    "origin": "159.192.217.999, 107.178.194.999" ,
    "url": "https://httpbin.org/post"
  }
}
```

หมายเหตุ :

เว็บ <https://httpbin.org/> สำหรับทำสอบ HTTP Request และ Response Service



[Base URL: <httpbin.org/>]

A simple HTTP Request & Response Service.

Run locally: `$ docker run -p 80:80 kennethreitz/httpbin`

[the developer - Website](#)
[Send email to the developer](#)

ตัวอย่าง

```
// Make a POST request with a JSON payload.
var data = {
    'name' : 'Bob Smith',
    'age' : 35 ,
    'pets' : [ 'fido' , 'fluffy' ]
};

var options = {
    'method' : 'post' ,
    'contentType': 'application/json' ,
    // Convert the Javascript object to a JSON string.
    'payload' : JSON.stringify(data)
};

var response = URLFetchApp.fetch('https://httpbin.org/post', options) ;
Logger.log(response.getContentText()) ;
```

10.2.ง.) getResponseCode()

getResponseCode()

<https://developers.google.com/apps-script/reference/url-fetch/http-response.html#getresponsecode>

จับรหัสสถานะ HTTP Response ถ้าคืนค่าเป็น 200 ถือว่า OK

ตัวอย่าง

```
var response = URLFetchApp.fetch("http://www.google.com/") ;
Logger.log(response.getResponseCode()) ;
```

10.3. การใช้งาน Fixer API

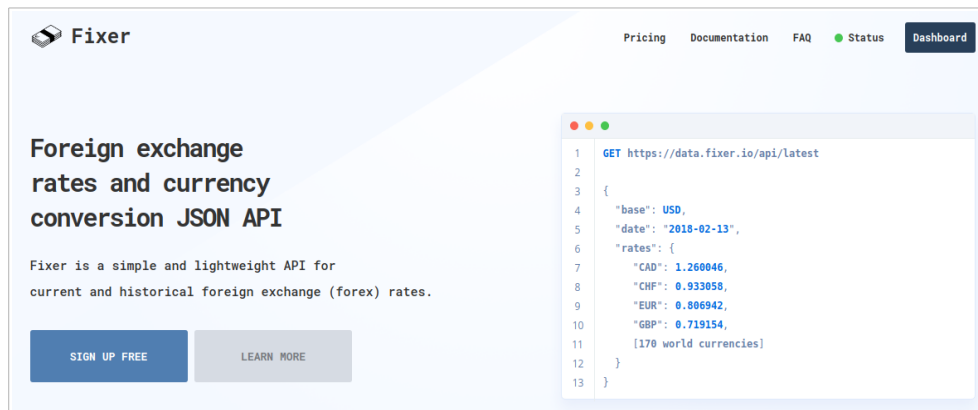
ดัดแปลงจาก Apps Script URLFetchApp API, Get JSON data, Build Google Sheets Function, Advanced Tutorial
<https://www.youtube.com/watch?v=kOsu6345KDI>

10.3.ก.) ใช้บริการ Fixer

Foreign exchange rates and currency conversion JSON API

<https://fixer.io/>

Fixer เป็นเว็บ API สำหรับให้ข้อมูลอัตราแลกเปลี่ยนเงินตราต่างประเทศ (forex)



หลังจาก Sign up แล้ว เราจะได้ **API Access Key** สำหรับใช้งาน API ตัวอย่างเช่น

API Access Key = 123456789abcdefghijklmnopqrstuvwxyz ซึ่ง API Access Key ตัวนี้ จะใช้สำหรับใช้งาน API ด้วย ตัวอย่างตามลิงค์ต่อไปนี้

http://data.fixer.io/api/latest?access_key=123456789abcdefghijklmnopqrstuvwxyz

เมื่อเข้าตามลิงค์ข้างต้น จะเห็นข้อมูล JSON ดังต่อไปนี้ที่หน้าเว็บ

JSON

```
{
  "success":true,
  "timestamp":1586762946,
  "base":"EUR",
  "date":"2020-04-13",
  "rates":{
    "AED":4.026292,
    "AFN":82.904484,
    "ALL":126.975615,
    "AMD":544.60603,
    "ANG":1.94965,
    "AOA":613.318555,
    ...
    ...
    ...
  }
}
```

ข้างต้นสกุลเงินที่เป็น **base** ก็คือ **EUR** (1 EUR มีค่าเท่ากับเงินสกุลอื่น เป็นตัวเลขเท่าไรก็ว่าไป) หากต้องการใช้ **base** เป็นสกุลอื่น ให้ส่งพารามิเตอร์ **base** เพิ่มเติมเข้าไปในลิงค์ **แต่ตอนนี้ base ใช้ฟรีไม่ได้แล้ว** ถ้าใช้จะได้ JSON ต่อไปนี้

`http://data.fixer.io/api/latest?access_key=123456789abcdefghijklmnopqrstuvwxyz&base=USD&format=1`

ผล – Error ถูกจำกัดการเข้าถึง

JSON

```
{
  "success":false,
  "error":{
    "code":105,
    "type":"base_currency_access_restricted"
  }
}
```

อย่างไรก็ดี ยังพอมีพารามิเตอร์ **symbols** ให้ใช้ทดสอบ พารามิเตอร์ **symbols** ใช้ดูเฉพาะสกุลเงินที่ระบุ ซึ่งจะระบุก็ได้ ขึ้นโดย "," ตัวอย่างดังนี้ (**base** เป็น **EUR** เสมอ)

`http://data.fixer.io/api/latest?access_key=123456789abcdefghijklmnopqrstuvwxyz&symbols=USD,AUD,CAD,PLN,MXN&format=1`

JSON

```
{
  "success":true,
  "timestamp":1586762946,
  "base":"EUR",
  "date":"2020-04-13",
  "rates":{
    "USD":1.096173,
    "AUD":1.725593,
    "CAD":1.52801,
    "PLN":4.553666,
    "MXN":25.594313
  }
}
```

10.3.ข.) Apps Script พัฒนาการที่ 1

หลังจากทดสอบดูข้อมูล จากการพิมพ์ลิงค์ URL โดยตรงแล้ว เราจะมาเขียนสคริปต์เพื่อนำข้อมูลมาใช้งาน โดยขั้นแรก ทดสอบจับค่าที่ได้จากการตอบสนองจาก Fixer ออกมาดูก่อน

```
var key = '123456789abcdefghijklmnopqrstuvwxyz';
var currencyArr = ['USD','AUD','THB'] // ดูเฉพาะสกุลเงิน 3 ตัวนี้เท่านั้น เทียบกับ EUR
var currencyJoin = currencyArr.join(',') // USD,AUD,THB

var url = 'http://data.fixer.io/api/latest?access_key=' + key +
          '&symbols='+ currencyJoin + '&format=1'

var fixerResponse = URLFetchApp.fetch(url); // ไปเอาข้อมูลจาก Fixer
Logger.log(fixerResponse) /* {
                                "success":true,
                                "timestamp":1586776146,
                                "base":"EUR",
                                "date":"2020-04-13",
                                "rates": {
                                    "USD":1.09123,
                                    "AUD":1.722269,
                                    "THB":35.727522
                                }
                            } */

var obj = JSON.parse(fixerResponse) // แปลงเป็น Javascript Obj
var base =obj["base"]; // EUR
var date =obj["date"]; // 2020-04-13
var rates =obj["rates"]; // {THB=35.767743, USD=1.09298, AUD=1.722626}
var ratesTHB =obj["rates"]["THB"]; // 35.7538
```

10.3.ค.) Apps Script พัฒนาการที่ 2

โค้ดในข้อก่อนหน้า เป็นการทดสอบดูค่า คีย์(Keys) และ ค่า(Values) ของ JSON ที่ส่งมาจาก fixer โดยในข้อนี้ เราจะทำให้ยืดหยุ่นกว่าเดิม โดยสร้างฟังก์ชัน แล้วส่ง String ที่เป็น Keypath เข้าไปจับ Values มา

Keypath(คีย์พาท) ก็คือ พาทไปยังคีย์ เพื่อจะเข้าถึงค่าต่างๆของคีย์ บางคีย์ไม่มีคีย์ย่อย(ไม่มีโหนดใน) บางคีย์มีคีย์ย่อย(มีโหนดใน) อย่างเช่น คีย์พาท **rates > THB** จะเข้าถึงค่า 35.7538

```
function call() {
    var a = fixerAPI('base'); // key = base
    Logger.log(a); // value = EUR

    var b = fixerAPI('date'); // key = date
    Logger.log(b); // value = 2020-04-13

    var c = fixerAPI('rates/THB'); // key = rates > THB
    Logger.log(c); // 35.727522

    var d = fixerAPI('rates'); // key = rates
    Logger.log(d); // [Object Object] ----- > ไปแก้ในข้อ 10.3.จ (หน้า 80)
}
```

```

function fixerAPI(keyPath) { // ระบุพารามิเตอร์เป็น Keypath ในลักษณะ key1/key2...

    var key = '123456789abcdefghijklmnopqrstuvwxyz' ;
    var currencyArr = ['USD','AUD','THB'] ;
    var currencyJoin = currencyArr.join(',') ; // USD,AUD,THB

    var url = 'http://data.fixer.io/api/latest?access_key=' + key +
              '&symbols=' + currencyJoin + '&format=1' ;
    var fixerResponse = URLFetchApp.fetch(url) ;
    var obj = JSON.parse(fixerResponse) ;

    var keyPathArr = keyPath.split('/') ;
    // สมมุติถ้า keyPath = 'rates/THB' จะได้ผลลัพธ์เป็น [rates,THB]

    // ขยายความในข้อถัดไป ===== >
    //
    for( let i = 0 ; i < keyPathArr.length ; i++){

        obj = obj[keyPathArr[i]] ;
        Logger.log('Loop ' + i + ' ' + obj + ' = ' + typeof obj ) ;

        // ตัวอย่าง
        // ถ้า keyPath = 'rates/THB' (มีโหนดใน)
        // Loop 0 [object Object] = object
        // Loop 1 35.73684 = number < ---- return obj เป็นตัวนี้

        // ถ้า keyPath = 'date' (ไม่มีโหนดใน)
        // Loop 0 2020-04-13 = string < ---- return obj เป็นตัวนี้
    }

    return obj ;
}

```

10.3.ง.) ขยายความบรรทัด obj = obj[keyPathArr[i]]

บรรทัด `obj = obj[keyPathArr[i]]` ; เป็นการกำหนด(Assign) คีย์และค่า 1 คู่ ให้กับตัวแปรวัตถุตัวเดิม ผลที่ได้คือ วัตถุตัวนั้นจะเหลือแค่ คีย์และค่า 1 คู่ ตัวอย่างต่อไปนี้

```

// ไม่มีโหนดใน
var obj = {
    name : 'Wasan' ,
    lastname : 'kds' ,
    phone : '0814598343' ,
    line : 'wasankds' ,
}
Logger.log(obj) ; // {line=wasankds, phone=0814598343, name=Wasan, lastname=kds}

obj = obj["name"] // **** Assign คีย์และค่า 1 คู่ ให้กับตัวเดิม
Logger.log(obj) ; // Wasan (type เป็น String)

```

แต่ถ้ามีการรวมรูป กำหนดค่าให้ตัวแปรเดิม แบบนี้ใช้ได้เฉพาะกับคีย์ที่มีโหนดในเท่านั้น แบบอื่นจะและทะ ดั่งตัวอย่างต่อไปนี้

```

var obj = {
    name: "Wasan" ,
    lastname: "kds",
    phone: "0814598343" ,
    line: "wasankds" ,
    pet : {                // มีโหนดใน
        dog1 : "Mafia" ,
        dog2 : "Yakuza" ,
        dog3 : "Kaopod" ,
    }
}

var arr = ["pet" , "dog1"] ;    // เป็นอาเรย์ที่เก็บ Keypath จาก pet > dog1

for( var i = 0 ; i < arr.length ; i++){
    obj = obj[arr[i]] ;
    Logger.log('Loop ' + i + ' : ' + obj + ' = ' + typeof obj )    // ดู Logs --- > [01], [02]
}

Logger.log(obj + ' = ' + typeof obj ) ;                          // ดู Logs --- > [03]

```

ผล

Logs

```

[ 01 ] Loop 0 : [object Object] = object
[ 02 ] Loop 1 : Mafia = string
[ 03 ] Mafia = string

```

10.3.จ.) Apps Script พัฒนาการที่ 3

ย้อนกลับไปในข้อ 10.3.ค Apps Script พัฒนาการที่ 2 หน้า 78 เมื่อเรียกใช้ฟังก์ชันโดยระบุ Keypath เป็น **rates** ก็คือ fixerAPI('rates') ผลที่ได้กลับมามีค่าคือ [object Object] ซึ่งยังไม่ OK ทั้งนี้เป็นเพราะคีย์ **rates** มีโหนดใน หรือเก็บค่าที่เป็นก้อนวัตถุไว้อีกชั้นหนึ่ง

เราจะมาแก้ตรงนี้ต่อ โดยแปลงโหนดวัตถุทั้งก้อน เป็นอาเรย์ แล้วแสดงหรือเขียนลงใน Google Sheets ทั้งก่อน

```

function call() {
    var a = fixerAPI('base') ;    Logger.log(a) ;    // key = base    // value = EUR
    var b = fixerAPI('date') ;    Logger.log(b) ;    // key = date    // value = 2020-04-13
    var c = fixerAPI('rates/THB') ; Logger.log(c) ; // key = rates > THB    // 35.727522
    var d = fixerAPI('rates') ;    // key = rates
    Logger.log(d) ;                // [[USD, 1.090852], [AUD, 1.713518], [THB, 35.692579]]
    var e = fixerAPI('xxx') ;      // key = ไม่มีคีย์นี้
    Logger.log(e) ;                // Node Not Avialable
}

```



```

function fixerAPI(keyPath) {
  var key = '123456789abcdefghijklmnopqrstuvwxyz' ;
  var currencyArr = ['USD','AUD','THB']
  var currencyJoin = currencyArr.join(',') // USD,AUD,THB
  var url = 'http://data.fixer.io/api/latest?access_key=' + key +
    '&symbols=' + currencyJoin + '&format=1'
  var fixerResponse = URLFetchApp.fetch(url) ;
  var obj = JSON.parse(fixerResponse) ;

  var keyPathArr = keyPath.split('/') ;
  for( let i = 0 ; i < keyPathArr.length ; i++){
    obj = obj[keyPathArr[i]] ;
  }

  if(typeof(obj) === "undefined") { // ถ้าไม่มี Keypath ที่ระบุ
    return "Node Not Available" ;
  }
  else if(typeof(obj) === "object"){ // ถ้า Keypath เป็น Object – จับทั้งวงเป็นอาเรย์ 2 มิติ
    var tempArr = [] ;
    for(var key in obj){
      tempArr.push([key,obj[key]]) ;
    } // xlose - for
    Logger.log(tempArr) ; // [[USD, 1.090852], [AUD, 1.713518], [THB, 35.692579]]
    return tempArr ;
  } else { // ถ้า Keypath เป็นอย่างอื่น – String, Number – มีค่าเดียว
    return obj ;
  }
}

```

ทดสอบเรียกใช้ฟังก์ชันที่เขียนเองใน Google Sheets จะได้ผลตามภาพ

fx	=fixerAPI("rates/THB")	
	A	B
1	35.692579	
2		
3		

fx	=fixerAPI("rates")	
	A	B
1	USD	1.090852
2	AUD	1.713518
3	THB	35.692579

10.4. การใช้งาน convertAPI

ConvertAPI

<https://www.convertapi.com/pdf-to-merge>

การรวมไฟล์ PDF หลายไฟล์ให้เป็นไฟล์เดียว บริการของ Google Apps Script ทำยากมาก ใช้บริการ API ของเจ้าอื่นง่ายกว่า ซึ่งในที่นี้ขอใช้บริการของ ConvertAPI

เมื่อสมัครใช้งานแล้ว จะได้ลิงค์สำหรับส่ง API Request ตัวอย่างเช่น

<https://v2.convertapi.com/pdf/to/merge?secret=ABCDEFGHIJKLMNOPQRSTUVWXYZ>

ลิงค์นี้เราต้องใช้ในการเขียน Google Apps Script เพื่อส่งโหลด(Payload) เป็นไฟล์ PDF ไปประมวลผล แล้วเอาตัวที่รวมแล้วกลับมาสร้างเป็นไฟล์ใน Google Drive

บริการของ **ConvertAPI** นั้นไม่ฟรี อย่างไรก็ตาม มีโควต้าให้ใช้ฟรี เพื่อทดสอบได้ปริมาณหนึ่ง (1500 วินาที ประมวลผล) หากโควต้าแล้วก็ต้องเสียเงิน

ค่าใช้จ่ายมีทั้งแบบ **Subscription Plans** และ **Prepaid packages**

Subscription Plans หรือ แบบจ่ายรายเดือน ต่ำสุด **6000 วินาที 30US\$/เดือน** (0.00500\$ ต่อวินาที)

แบบ **Prepaid packages** หรือ แบบเป็นครั้งๆ ซื้อครั้งหนึ่งได้เวลาประมวลผลกี่วินาทีก็ว่าไป เช่น **15,000 วินาที ราคา 10US\$** (0.00667\$ ต่อวินาที)

ดูโดยรวมแล้ว แบบ **Subscription Plans** ถูกกว่า 25%

Merge Multiple PDF's into one PDF

<https://stackoverflow.com/questions/15414077/merge-multiple-pdfs-into-one-pdf>

โค้ด Google Apps Script สำหรับใช้บริการ ConvertAPI มีดังนี้

```
function merge() {
  // โฟลเดอร์ที่มีไฟล์ PDF
  var folder = DriveApp.getFolderById('1Vo9aFom7CPMK8NmAlX7Q8LZf_mVXIQH3');
  var files = folder.GetFilesByType(MimeType.PDF);

  var formData = {};
  var index = 0;
  while(files.hasNext()) {
    var file = files.next();
    formData['Files[' + index + ']'] = file.getBlob(); // วนลูปจับ Blob ของไฟล์ใส่ formData
    index++;
  }

  Logger.log(formData); // formData = { Files[2]=Blob, Files[1]=Blob, Files[0]=Blob }
```

```

var options = {
    'method' : 'post' ,
    'payload' : formData ,
    'muteHttpExceptions': true // สำหรับ Log ดู Error reponse
};












// ส่ง Request ไปยัง ConvertApi พร้อมรับค่ากลับมาใส่ตัวแปร response
// URL ที่ได้มาจากการสมัครใช้ ConvertAPI
var response = URLFetchApp.fetch('https://v2.convertapi.com/pdf/to/merge?Secret=
                                ABCDEFGHIJKLMNOPQRSTUVWXYZ', options) ;
// Logger.log(response.getContentText()) ; // ดูว่ามีอะไรส่งมาบ้าง

if(response.getResponseCode() == 200) {
    var contentText = JSON.parse(response.getContentText()) ;
    var blob = Utilities.base64Decode(contentText.Files[0].FileData) ;

    folder.createFile(Utilities.newBlob(blob, 'application/pdf', 'merge.pdf')) ;
}
}












```

ตัวอย่าง – ในโฟลเดอร์ มีไฟล์ pdf 3 ไฟล์ และก็มีไฟล์อื่นๆสมอยู่

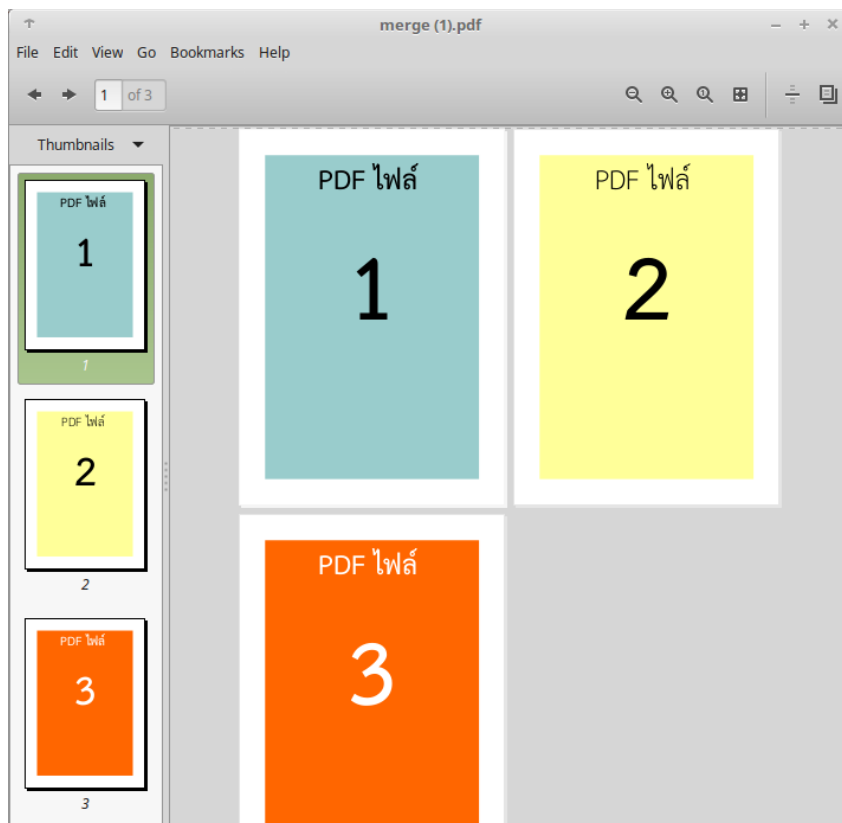
My Drive > 99 Images and PDFs ▾	
Name ▾	Owner
 Subfolder	me
 pdf_File_3.pdf 	me
 pdf_File_2.pdf 	me
 pdf_File_1.pdf 	me
 man_3_pic.jpg 	me
 man_3_doc.jpg 	me

หลังจากรันโค้ด จะได้ไฟล์ pdf ชื่อ merge.pdf เพิ่มมาอีกไฟล์ เป็นไฟล์ที่รวมแล้วนั่นเอง

ตัวอย่างตามภาพ

My Drive > 99 Images and PDFs ▾	
Name ▾	Owner
 Subfolder	me
 pdf_File_3.pdf 	me
 pdf_File_2.pdf 	me
 pdf_File_1.pdf 	me
 merge.pdf 	me
 man_3_pic.jpg 	me

ตัวอย่างในไฟล์ [merge.pdf](#) มีดังต่อไปนี้



บทที่ 11

JSON



11.1. JSON คืออะไร

JSON - Introduction

https://www.w3schools.com/js/js_json_intro.asp

Enum MimeType

<https://developers.google.com/apps-script/reference/content/mime-type>

เมื่อต้องส่งข้อมูลระหว่าง Server หรือ Browser ข้อมูลต้องเป็น Text เท่านั้น

JSON (Javascript Object Notation) คือ **Text** ใน Google Apps Script เราสามารถแปลงจาก Javascript Object ไปเป็น JSON และส่ง JSON ไปที่ Server ได้

และเช่นเดียวกัน เราสามารถแปลงจาก JSON ที่ได้รับจาก Server ไปเป็น Javascript object เพื่อนำมาใช้ต่อใน Google Apps Script ได้ด้วย

การส่งข้อมูล

```
// Javascript object
var myObj = {
    name: "John",
    age: 31,
    city: "New York"
};

// แปลงเป็น JSON ----- > ส่งไปให้ Server
var myJSON = JSON.stringify(myObj) ;

// สังเกตที่ Logs มีเครื่องหมาย " " ครอบที่คีย์ทั้งหมด - คีย์เป็น String
Logger.log(myJSON) ; /* {
    "name":"John" ,
    "age":31 ,
    "city":"New York"
} */
```

การรับข้อมูล

```
// JSON ใน Single quotes – มี Backtick ครอบ ก็คือ เป็น Text ทั้งก้อน
var myJSON = ' {
    "name":"John" ,
    "age":31 ,
    "city":"New York"
} ' ;

// แปลงเป็น Javascript object ----- > นำไปใช้ใน Google Apps Script
var myObj = JSON.parse(myJSON) ;

// สังเกตที่ Logs ไม่มีเครื่องหมาย " " ครอบที่คีย์ - คีย์เป็นตัวแปร
Logger.log(myObj) ; /* {
    city=New York,
    name=John,
    age=31
} */
```

11.2. JSON Data Types

JSON Data Types

https://www.w3schools.com/js/js_json_datatypes.asp

ข้อมูลใน JSON เป็นอะไรได้หลากหลาย ดังต่อไปนี้

JSON Strings

```
{  
  "name":"John"  
}
```

JSON Numbers

```
{  
  "age":30  
}
```

JSON Objects

```
{  
  "employee" :  
  {  
    "name":"John" ,  
    "age":30 ,  
    "city":"New York"  
  }  
}
```

JSON Arrays

```
{  
  "employees":[ "John", "Anna", "Peter" ]  
}
```

JSON Booleans

```
{  
  "sale":true  
}
```

JSON null

```
{  
  "middlename":null  
}
```

11.3. ตัวอย่างการเขียน JSON

11.3.ก.) ตัวอย่างที่ 1

สร้างไฟล์ .json สามารถสร้างโดยใช้ Visual Studio หรือใช้ Text Editor

```
// user.json

{
  "name": "Kyle" ,
  "favoriteNumber": "3" ,
  "isProgrammer": true ,
  "hoibies": [ "Weight Lifting" , "Bowling" ] ,
  "friends": [ {
    "name": "Joey" ,
    "favoriteNumber": "100" ,
    "isProgrammer": false
  } ]
}
```

11.3.ข.) ตัวอย่างที่ 2

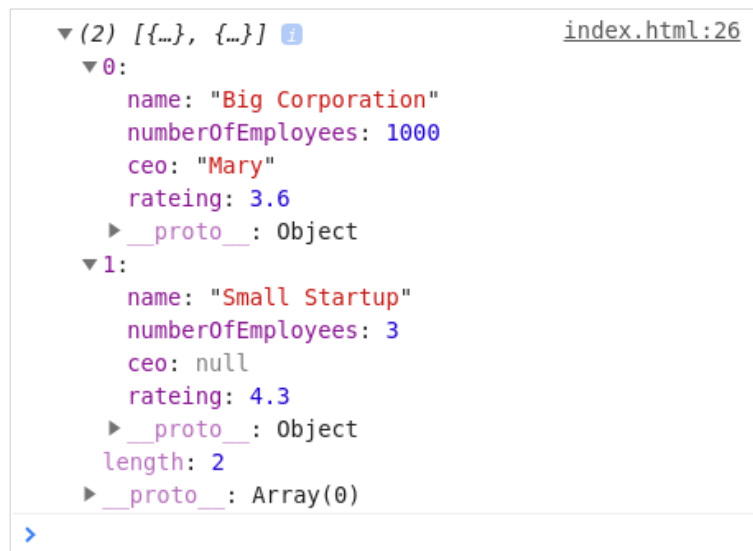
ไฟล์ index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>JSON Example</title>
</head>
<body>
  <script>

    // companies เป็นอาเรย์ 1 แถว ที่มีสมาชิกเป็น JSON เรียงต่อกันไป
    let companies = [ {
      "name":"Big Corporation" ,
      "numberOfEmployees":1000 ,
      "ceo":"Mary" ,
      "rateing":3.6
    } ,
    {
      "name":"Small Startup" ,
      "numberOfEmployees":3 ,
      "ceo":null ,
      "rateing":4.3
    } ]

    console.log(companies) ; // ----- > ดูผลที่คอนโซล

  </script>
</body>
</html>
```

11.3.ค.) ตัวอย่างที่ 2 - ต่อ

ไฟล์ index.html

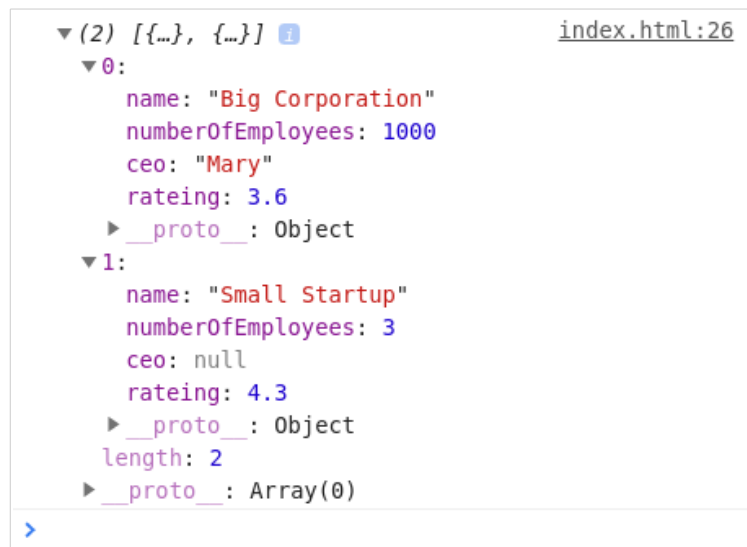
```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>JSON Example</title>
</head>
<body>
  <script>

    // companies เป็นก้อน Text - มี `` (Back tick) ครอบทั้งก้อน
    // ข้างในก้อน Text เป็นอาเรย์ 1 แถว ที่มีสมาชิกเป็น JSON เรียงต่อกันไป
    let companies = `[
      "name":"Big Corporation" ,
      "numberOfEmployees":1000,
      "ceo":"Mary",
      "rateing":3.6
    ] ,
    [
      "name":"Small Startup" ,
      "numberOfEmployees":3 ,
      "ceo":null ,
      "rateing":4.3
    ]
  ]`

    console.log(JSON.parse(companies)) ; // ----- > ดูผลที่คอนโซล

  </script>
</body>
</html>
```

ผล - ได้ผลเหมือนก่อนหน้า



11.4. JSON และ Google Apps Script

ดัดแปลงมาจาก How to Get, Parse, Query, and Return JSON
<http://googleappscripting.com/json/>

11.4.ก.) ส่ง HTTP Request

เวลาเราใช้งาน API โดยส่ง HTTP Request เข้าไป จะได้รับ JSON กลับมา ตัวอย่าง URL ที่ส่ง HTTP Request ดังต่อไปนี้

<http://api.icndb.com/jokes/random>

ผลที่ได้รับกลับมา หรือ ที่หน้าเว็บ จะแสดงข้อมูลเป็น JSON ดังต่อไปนี้
(JSON ด้านล่าง จัดรูปแบบให้เหมาะสำหรับพิมพ์ แต่จริงๆแล้ว Server ส่งกลับมาไม่สวยแบบนี้)

```
{
  "type": "success",
  "value": {
    "id": 149 ,
    "joke": "Chuck Norris proved that we are alone in the universe. We weren't before his first space expedition.",
    "categories": []
  }
}
```

ตัวอย่างข้างต้น เป็นการส่ง HTTP Request ผ่าน การพิมพ์ URL ตรงๆที่ Browser แต่ถ้าจะโปรแกรม ด้วย Google Apps Script จะใช้เมธอดเฉพาะ ดังนี้

```
// ส่ง HTTP Get request ไปที่เว็บ API – Default เป็น GET
var chuckNorrisJSON = URLFetchApp.fetch("http://api.icndb.com/jokes/random") ;

// เมื่อ Logs จะได้ผลลัพธ์เหมือนกับ JSON ข้างต้น
Logger.log(chuckNorrisJSON)
```

11.4.ข.) HTTP Post Request

ฟังก์ชัน doPost() ของในโปรเจ็ค Google Apps Script จะรันทุกครั้งที่มี HTTP POST Request ส่งเข้ามาโดย doPost() จะรับอากิวเมนต์ที่ส่งเข้ามา(ถ้ามี) ไปประมวลผลต่อไป จากนั้นก็คืนค่ากลับไป ตัวอย่างดังนี้

```
function doPost(request){

    // get the sting value of the POST data
    var postJSON = request.postData.getDataAsString() ;

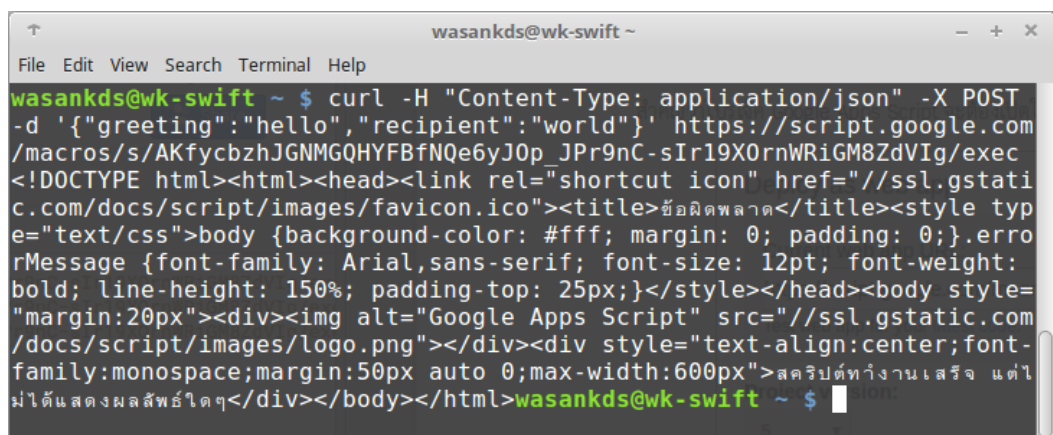
    // save that JSON to a file
    DriveApp.createFile('post.json', postJSON) ;

}
```

ให้ทดสอบส่ง HTTP POST Request ไปที่โปรเจ็ค Google Apps Script โดยใช้คำสั่งต่อไปนี้ (Linux Terminal) หรือใช้ Post Man ก็ได้

```
curl -H "Content-Type: application/json" -X POST -d
'{"greeting":"hello","recipient":"world"}' https://script.google.com/yourscriptURL
```

ตัวอย่างตามภาพ

A screenshot of a terminal window titled 'wasankds@wk-swift ~'. The terminal shows a command being executed: 'curl -H "Content-Type: application/json" -X POST -d '{"greeting":"hello","recipient":"world"}' https://script.google.com/macros/s/AKfycbzhJGNMGQHYFBfNQe6yJ0p_JPr9nC-sIr19X0rnWRiGM8ZdVIg/exec'. The output is a large block of HTML code, indicating that the server has successfully received the POST request and is returning an HTML response. The terminal text is as follows:
wasankds@wk-swift ~ \$ curl -H "Content-Type: application/json" -X POST
-d '{"greeting":"hello","recipient":"world"}' https://script.google.com
/macros/s/AKfycbzhJGNMGQHYFBfNQe6yJ0p_JPr9nC-sIr19X0rnWRiGM8ZdVIg/exec
<!DOCTYPE html><html><head><link rel="shortcut icon" href="//ssl.gstatic
c.com/docs/script/images/favicon.ico"><title>ข้อผิดพลาด</title><style typ
e="text/css">body {background-color: #fff; margin: 0; padding: 0;}.erro
rMessage {font-family: Arial,sans-serif; font-size: 12pt; font-weight:
bold; line-height: 150%; padding-top: 25px;}</style></head><body style=
"margin:20px"><div></div><div style="text-align:center;font-
family:monospace;margin:50px auto 0;max-width:600px">สคริปต์ทำงานเสร็จ แต่ไ
ม่ได้แสดงผลใดๆ</div></body></html>wasankds@wk-swift ~ \$

สำคัญว่าการ Deploy as web app ของโปรเจ็ค Google Apps Script จะต้องเปิดให้รันสคริปต์ เหมือนเจ้าของรันเอง ตามภาพ

Deploy as web app

Current web app URL:

Disable web app

https://script.google.com/macros/s/AKfycbzhJGNMGQH

Test web app for your latest code.

Project version:

5

Execute the app as:

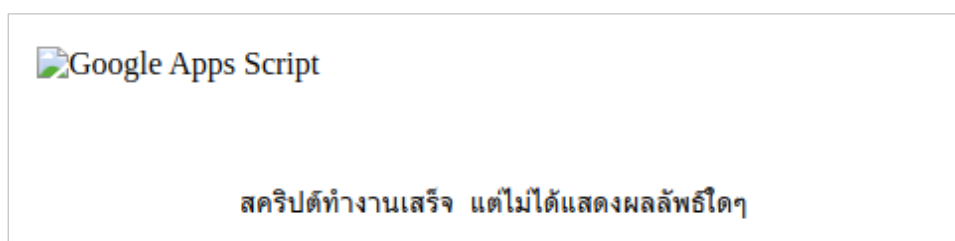
Me (wasankds@gmail.com)

You need to authorize the script before distributing the URL.

Who has access to the app:

Anyone, even anonymous

หลังรันคำสั่ง (ใน Linux Terminal) จะมีการตอบกลับ เป็นโค้ด HTML ถ้านำมาสร้างเป็นหน้าเว็บจะได้ตามภาพ



ผลจากฟังก์ชัน doGet() จะสร้างไฟล์ JSON ไว้ที่ Google Drive ตามภาพ

My Drive ▾				
Name ↑		Owner	Last modified by ...	
post.json		me	11:12 AM	

เนื้อหาในไฟล์ ก็คือ

```
{"greeting":"hello","recipient":"world"}
```

จากนั้น ถ้าเราจะเข้าถึงไฟล์ JSON ที่สร้างไว้ เพื่อไปเอาเนื้อหาข้างใน แล้วแปลงเป็น Javascript object เพื่อจะเอาไปทำอะไรต่อใน Google Apps Script สามารถทำได้ดังนี้

```
function getJSONfromFile(){  
    // จับไฟล์ตามชื่อไฟล์ - ได้มาเป็นวัตถุ FilIterator - ต้องวนลูปเข้าไปเอา  
    var iter = DriveApp.getFilesByName("post.json") ;  
  
    var json ;  
  
    while (iter.hasNext()) {  
        var file = iter.next() ; // จับไฟล์  
        var jsonFile = file.getAs('application/json') // จับเนื้อหาเป็น JSON - ได้ Blob  
        json = jsonFile.getDataAsString() // จับ Text จาก Blob - ได้ String  
  
        Logger.log(json) ; // {"greeting":"hello","recipient":"world"}  
  
        // แปลงเป็น Javascript Obj. - ถ้านำมาใช้ต่อใน Apps Script  
        var obj = JSON.parse(json) ;  
        Logger.log(obj) ; // {recipient=world, greeting=hello}  
        Logger.log(obj.greeting) ; // hello  
        Logger.log(obj.recipient) ; // world  
    }  
}
```

11.4.ค.) HTTP Get request

doGet() ก็คล้ายกับ doPost() เพียงแต่ใช้กับ HTTP Get Request โค้ดต่อไปนี้จะตอบกลับเป็น JSON

```
function doGet(request){ // ไม่ได้เอา request มาใช้ในโค้ดนี้  
  
    var myDog = { "name": "Rhino", "breed": "pug", "age": 8 }  
    var myJSON = JSON.stringify(myDog) ;  
  
    // คืนค่ากลับเป็น JSON  
    var a = ContentService.createTextOutput(myJSON)  
        .setMimeType(ContentService.MimeType.JSON) ;  
  
    Logger.log(a) ; // TextOutput  
    Logger.log(a.getContent()) ; // {"name":"Rhino","breed":"pug","age":8}  
  
    return a ;  
}
```

ผล - เมื่อเข้าถึง Deploy as web app ของโปรเจ็ค Google Apps Script จะได้ผลลัพธ์ที่หน้าเว็บดังนี้

```
{"name": "Rhino", "breed": "pug", "age": 8}
```

11.4.ง.) ส่ง JSON ติดไปกับ HTTP Request

โค้ดต่อไปนี้ ส่ง Payload เป็น JSON ติดไปกับ HTTP Request ที่สร้างโดยเมธอด `URLFetchApp`

```
var options = {
    'method' : 'post',
    'contentType': 'application/json' ,
    'payload' : myJSON
};

// send the request
URLFetchApp.fetch('https://example.com/post', options) ;
```

11.4.จ.) บันทึก JSON ลง Google Sheets

ตัวอย่างต่อไปนี้ มี 2 โปรเจ็ค

โปรเจ็คแรก เป็นตัวยิง HTTP Request ไปหาโปรเจ็คที่สอง

โปรเจ็คที่สอง นำ Payload(JSON) ที่โปรเจ็คแรกส่งมา ไปเขียนลง Google Sheets

โปรเจ็คที่ 1 - มีโค้ดดังต่อไปนี้

```
function call() {
    var myDog = { name: "Mafia", breed: "Tshizu", age: 10 } ;
    var myJSON = JSON.stringify(myDog) ;

    var options = {
        'method' : 'post',
        'contentType': 'application/json',
        'payload' : myJSON // ส่ง payload เป็น JSON
    };

    var url = 'https://script.google.com/macros/s/AppScriptProject_URL' ;

    // ส่ง HTTP Request ไปที่ URL ข้างต้น โดยแนบ payload ส่งไปด้วย
    // และ จักรการตอบกลับมาใส่ตัวแปรไว้
    //
    var response = URLFetchApp.fetch(url, options) ;

    // Logs การตอบกลับออกมาดู
    // โปรเจ็คที่สอง ตอบกลับมาเป็น JSON ง่ายๆ
    Logger.log(response) ; // {"response":"success"}
    Logger.log(response.getContentText()) ; // {"response":"success"}
}
```

โปรเจ็คที่ 2 - มีโค้ดดังต่อไปนี้

เมื่อได้รับ JSON เราสามารถบันทึกลง Google Sheets ได้ดังนี้

```
function doPost(request){  
    // postData – เป็น Event parameters  
    // https://developers.google.com/apps-script/guides/web  
    // เป็นส่วนของ Payload ที่มาจาก POST Request  
    var json = request.postData.getDataAsString() ;  
    var obj = JSON.parse(json) ;  
  
    // จับคีย์ทั้งหมดของ Javascript Obj ใส่อาเรย์  
    var headerRow = Object.keys(obj) ; // [name, breed, age]  
  
    // วนลูป ส่งคีย์ที่จับมาก่อนหน้า ไปเอาค่าจาก Javascript Obj  
    var row = headerRow.map(function(key){ // [Rhino, pug, 8.0]  
        return obj[key]  
    }) ;  
  
    // จับ หัวตาราง + ข้อมูล ใส่อาเรย์ - จะได้เป็นอาเรย์ 2 มิติพร้อมเซตลง Google Sheets  
    var contents = [ headerRow , row ] ; // [[name, breed, age],  
                                           // [Rhino, pug, 8.0]]  
  
    // เซตลง Google Sheets  
    var id = 'Google Sheets Id' ;  
    var sheet = SpreadsheetApp.openById(id).getSheetByName('Sheet Name');  
    var rng = ss.getActiveSheet().getRange(1, 1, contents.length, headerRow.length )  
    rng.setValues(contents)  
  
    // ตอบกลับเป็น JSON แบบง่าย แค่เพียงทดสอบ  
    //  
    var successObj = { response: "success" } ;  
  
    return ContentService.createTextOutput(JSON.stringify(successObj))  
        .setMimeType(ContentService.MimeType.JSON) ;  
}
```

ผลที่ไฟล์ Google Sheets

	A	B	C
1	name	breed	age
2	Mafia	Tshizu	10

11.4.จ.) แปลงข้อมูลจากตารางใน Google Sheets เป็น JSON

Creating a JSON object from Google Sheets

<https://stackoverflow.com/questions/47555347/creating-a-json-object-from-google-sheets>

```
function test_getJsonArrayFromData() {
  var data = [
    ['Planet', 'Mainland', 'Country', 'City'] ,
    ['Earth', 'Europe', 'Britain', 'London'] ,
    ['Earth', 'Europe', 'Britain', 'Manchester'] ,
    ['Earth', 'Europe', 'Britain', 'Liverpool'] ,
    ['Earth', 'Europe', 'France', 'Paris'] ,
    ['Earth', 'Europe', 'France', 'Lion']
  ] ;

  Logger.log(getJsonArrayFromData(data)) ;

  // => [{Mainland=Europe, Country=Britain, Planet=Earth, City=London}, {Mainland=Europe,
Country=Britain, Planet=Earth, City=Manchester}, {Mainland=Europe, Country=Britain,
Planet=Earth, City=Liverpool}, {Mainland=Europe, Country=France, Planet=Earth, City=Paris},
{Mainland=Europe, Country=France, Planet=Earth, City=Lion}]
}

function getJsonArrayFromData(data) {

  var obj = {} ;
  var result = [] ;
  var headers = data[0] ;      // ชับอาเรย์แถวแรกมาเป็น header - ใช้เป็นคีย์ด้วย
  var cols = headers.length ;
  var row = [] ;

  for (var i = 1 , i = data.length; i < data.length; i++) {

    // get a row to fill the object
    row = data[i] ;

    // clear object
    obj = {} ;

    for (var col = 0 ; col < cols ; col++) {
      // fill object with new values
      obj[headers[col]] = row[col] ;
    }

    // add object in a final result
    result.push(obj) ;
  }

  return result ;
}
```

11.5. เซอร์วิส Content

เซอร์วิส

<https://developers.google.com/apps-script/reference/content>

คลาส `TextOutput`

<https://developers.google.com/apps-script/reference/content/text-output>

เซอร์วิส Content ใช้สร้างและเสริมข้อมูลแบบ Text ในหลากหลายรูปแบบ เช่น Text, XML, JSON

11.5.ก.) `createTextOutput()`

`createTextOutput(content)` - Method ในคลาส `ContentService`

<https://developers.google.com/apps-script/reference/content/content-service#createTextOutputContent>

ใช้สร้างวัตถุ `TextOutput` ตัวใหม่ โดยสร้างจากพารามิเตอร์ที่ใส่เข้าไป โดยพารามิเตอร์สามารถเว้นไว้ได้ โดยจะคืนค่ากลับมาเป็นวัตถุ `TextOutput`

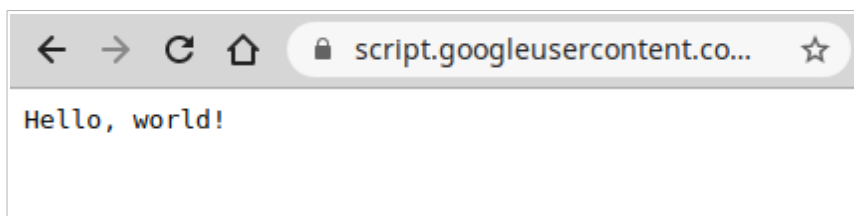
ตัวอย่างที่ 1 - แบบไม่มีพารามิเตอร์

```
function doGet() {  
    var output = ContentService.createTextOutput() ;  
    output.append("Hello world!") ; // แแนบ Text เข้าไปในวัตถุ TextOutput  
    return output ;  
}
```

ตัวอย่างที่ 2 - แบบมีพารามิเตอร์

```
function doGet() {  
    return ContentService.createTextOutput('Hello, world!') ;  
}
```

เมื่อเปิดลิงค์ที่ใช้ Deploy as web app จะได้ผลลัพธ์ตามภาพ



ตัวอย่างที่ 3

Content Service - Guide

<https://developers.google.com/apps-script/guides/content>

โค้ดต่อไปนี้จะคืนค่ากลับมาเป็น JSON แจ้ง **true** หรือ **false** ว่ามี Event ในช่วงเวลาดังกล่าวหรือไม่

```
function doGet(request) {  
  Logger.log(request) ;    // ดูผลที่ Logs ----- > [ 01 ]  
  
  var events = CalendarApp.getEvents(  
    new Date(Number(request.parameters.start) * 1000) ,  
    new Date(Number(request.parameters.end) * 1000)  
  ) ;  
  
  var result = {  
    available : events.length == 0  
  } ;  
  
  Logger.log(result) ;      // ดูผลที่ Logs ----- > [ 02 ]  
  
  return ContentService.createTextOutput(JSON.stringify(result))  
    .setMimeType(ContentService.MimeType.JSON) ;  
}
```

เมื่อเปิดลิงค์ที่ใช้ Deploy as web app จากนั้นต่อท้ายลิงค์ Web App ด้วยพารามิเตอร์ ตัวอย่างดังต่อไปนี้

<https://script.google.com/macros/s/Script Id?start=1548954000&end=1585674000>

เพื่อส่งตัวแปร start และ end ให้กับ doGet()

Logs

```
[ 01 ] {contextPath=, parameters={start=[1580490000], end=[1585674000]},  
contentLength=-1.0, queryString=start=1580490000&end=1585674000,  
parameter={start=1580490000, end=1585674000}}
```

```
[ 02 ] {available=true}
```

ที่หน้า Web App ได้ผลลัพธ์ดังต่อไปนี้

```
{"available":false}
```

หมายเหตุ : ศึกษาเพิ่มเติมที่

doGet and doPost Tutorial + 6 Web App Examples

<http://googleappscripting.com/doget-dopost-tutorial-examples/>

