



Google Apps Script



11-2025

Nhóm thực hiện

- Thân Hoàng Lộc
- Trần Anh Tuấn
- ...

Mục tiêu

- Giới thiệu tổng quan về Google Apps Script, và những khái niệm cơ bản
 - Một vài ví dụ ứng dụng Google Apps Script với Google Docs, Sheets



Nội dung

Phần 1

Giới thiệu tổng quan Google Apps Script
Các giới hạn
Các khái niệm cơ bản
Các ví dụ với Google Docs, Sheets

Phần 2

Triggers, các cơ chế kích hoạt
Giao diện người dùng – UI và Services
- Menu tùy chỉnh
- Sidebar tùy chỉnh

Phần 3

Google Apps Script với Firebase

Nội dung

Phần 1

Giới thiệu tổng quan Google Apps Script

Các giới hạn

Các khái niệm cơ bản

Các ví dụ với Google Docs, Sheets

Google Apps Script (GAS) là gì?

- Là một nền tảng phát triển ứng dụng nhanh (Rapid Application Development) dựa trên JavaScript.
- Cho phép bạn tự động hóa, tích hợp và mở rộng các ứng dụng trong Google Workspace.
- Hoạt động trên nền tảng đám mây của Google, không cần cài đặt.

The screenshot shows the Google Apps Script interface at <https://script.google.com>. The left sidebar contains icons for triggers (installTrigger, onFormSubmit), sheets (newSheet), and other functions. The main area displays a script:

```
function installTrigger() {
  ScriptApp.newTrigger('onFormSubmit')
    .forSpreadsheet(SpreadsheetApp.getActiveSpreadsheet())
    .onFormSubmit()
    .create();
}

function onFormSubmit(e) {
  let responses = e.namedValues;
  let email = responses['Email Address'][0].trim();
  let name = responses.Name[0].trim();
  let topicsString = responses.Topics[0].toLowerCase();
  let topics = Object.keys(topicUrls).filter(function(topic) {
    return topicsString.indexOf(topic.toLowerCase()) > -1;
  });
  let status = '';
  if (topics.length > 0) {
    MailApp.sendEmail({
      to: email,
      subject: EMAIL_SUBJECT,
      htmlBody: createEmailBody(name, topics),
    });
    status = 'Sent';
  } else {
    status = 'No topics selected';
  }
  let sheet = SpreadsheetApp.getActiveSheet();
  let row = sheet.getActiveRange().getRow();
  let column = e.values.length + 1;
  sheet.getRange(row, column).setValue(status);
}
```

Three circular callouts highlight specific parts of the code:

- A purple icon with three horizontal lines and a blue arrow points to the `onFormSubmit` trigger icon in the sidebar.
- A green icon with a grid and a blue arrow points to the `MailApp.sendEmail` function call.
- A yellow icon with a red and green 'M' and a blue arrow points to the `SpreadsheetApp.getActiveSheet()` function call.

Tại sao nên sử dụng Google Apps Script?

- **Tự động hóa** các tác vụ lặp đi lặp lại, tốn thời gian.
- **Tạo các quy trình làm việc** tùy chỉnh cho doanh nghiệp của bạn.
- **Xây dựng các giải pháp** mà không cần đầu tư vào cơ sở hạ tầng.
- **Tăng năng suất** và hiệu quả công việc.

The screenshot shows the Google Apps Script interface at <https://script.google.com>. The left sidebar contains icons for triggers (installTrigger, formSubmit), loops (for loop, while loop), conditions (if, else), and settings. The main area displays a script:

```
function installTrigger() {
  ScriptApp.newTrigger('onFormSubmit')
    .forSpreadsheet(SpreadsheetApp.getActiveSpreadsheet())
    .onFormSubmit()
    .create();
}

function onFormSubmit(e) {
  let responses = e.namedValues;
  let email = responses['Email Address'][0].trim();
  let name = responses.Name[0].trim();
  let topicsString = responses.Topics[0].toLowerCase();
  let topics = Object.keys(topicUrls).filter(function(topic) {
    return topicsString.indexOf(topic.toLowerCase()) > -1;
  });
  let status = '';
  if (topics.length > 0) {
    MailApp.sendEmail({
      to: email,
      subject: EMAIL_SUBJECT,
      htmlBody: createEmailBody(name, topics),
    });
    status = 'Sent';
  } else {
    status = 'No topics selected';
  }
  let sheet = SpreadsheetApp.getActiveSheet();
  let row = sheet.getActiveRange().getRow();
  let column = e.values.length + 1;
  sheet.getRange(row, column).setValue(status);
}
```

Three circular callouts point to specific parts of the code:

- A purple icon points to the `onFormSubmit` trigger icon in the sidebar.
- A green icon points to the `MailApp.sendEmail` function call.
- A blue icon points to the `SpreadsheetApp.getActiveSheet()` function call.

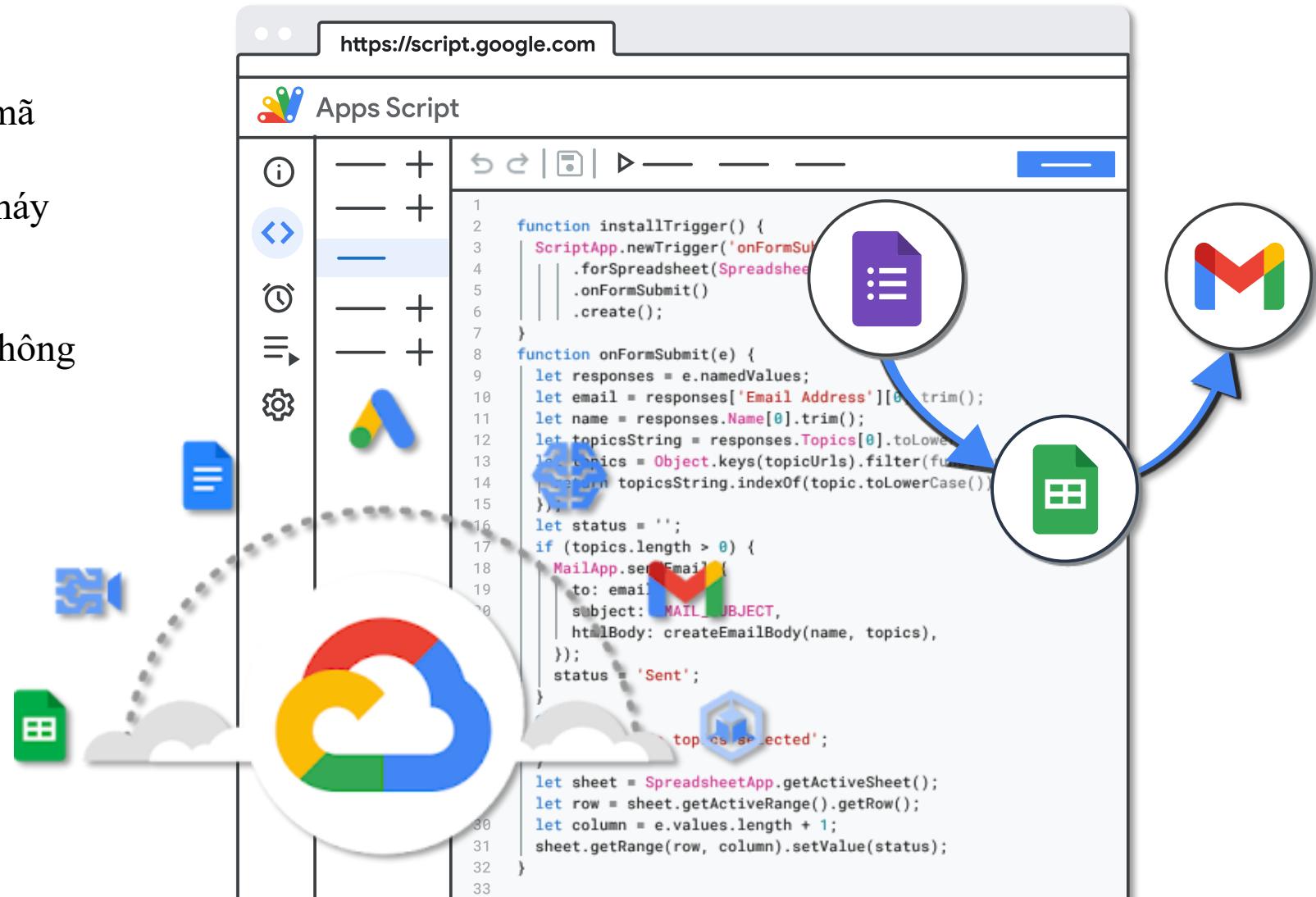
Google Apps Script - Đặc điểm chính

- **Dễ sử dụng:** Dựa trên ngôn ngữ JavaScript phổ biến.
 - **Tích hợp sẵn:** Kết nối dễ dàng với Gmail, Sheets, Docs, Drive, Calendar, v.v.
 - **Linh hoạt:** Có thể tạo ứng dụng web, add-on, hoặc các hàm tùy chỉnh.
 - **Miễn phí:** Với tài khoản Google, bạn có thể bắt đầu ngay mà không cần cài đặt.



Google Apps Script - Cách hoạt động

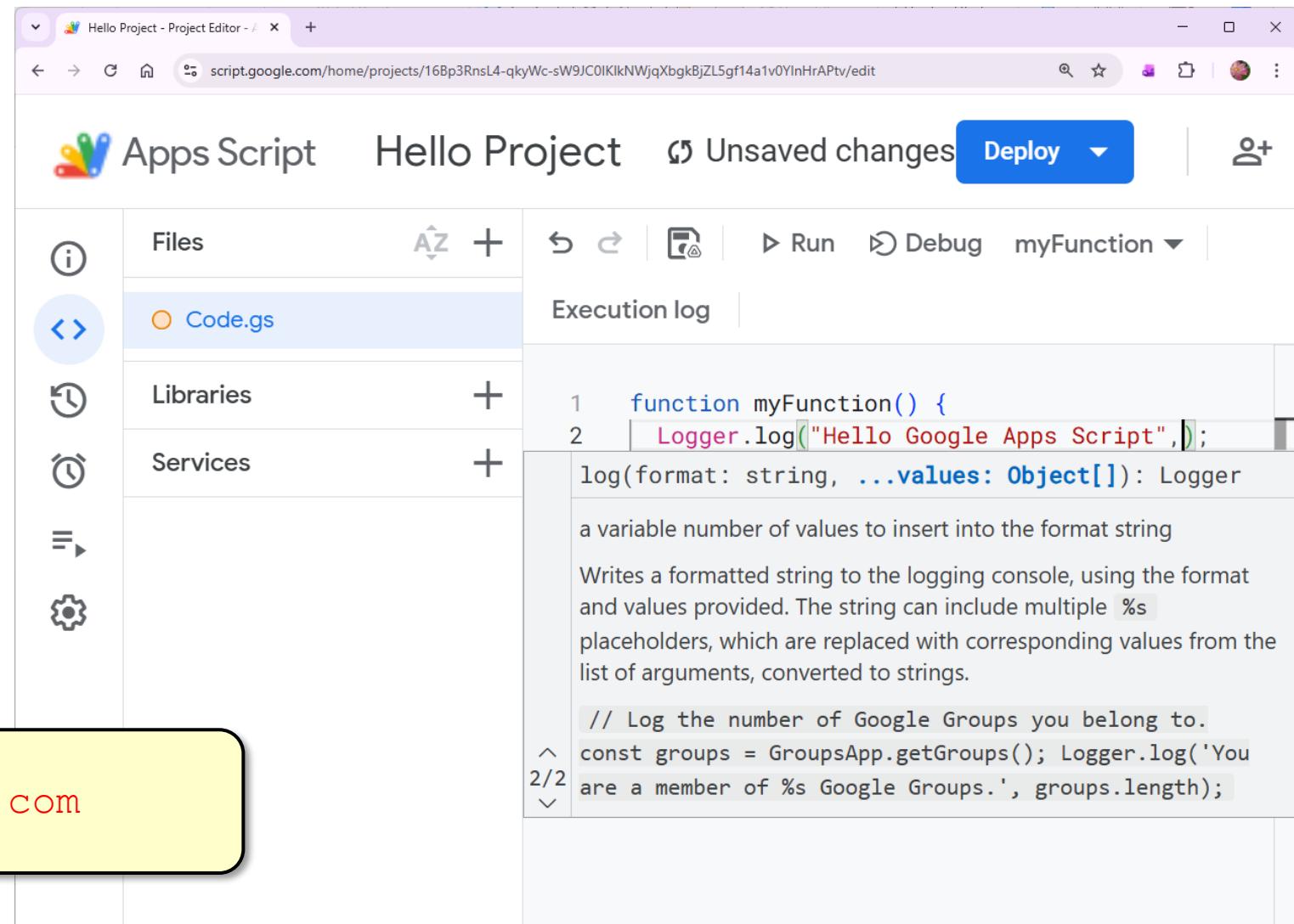
- **Viết mã** trong trình soạn thảo mã trực tuyến.
- Mã được lưu và **thực thi** trên máy chủ của Google.
- **Tương tác** với các dịch vụ của Google và các **API** bên ngoài thông qua các lệnh gọi hàm.



Môi trường phát triển, và JavaScript

- Sử dụng cú pháp **JavaScript tiêu chuẩn**.
- Cung cấp một bộ **API** phong phú để tương tác với các dịch vụ của Google.
- Trình soạn thảo mã dựa trên web (script.google.com).
- Gõ lỗi (**debugger**), ghi log (**logger**), và quản lý phiên bản.
- Có thể sử dụng công cụ dòng lệnh `clasp` để phát triển trên máy tính cá nhân.

script.google.com



The screenshot shows the Google Apps Script Project Editor interface. The title bar reads "Hello Project - Project Editor". The URL in the address bar is "script.google.com/home/projects/16Bp3RnsL4-qkyWc-sW9JC0IKlkNWjqXbgkBjZL5gf14a1v0YInHrAPtv/edit". The main area is titled "Hello Project" and shows "Unsaved changes". A "Deploy" button is visible. On the left, there's a sidebar with icons for Files, Libraries, and Services, and a "Code.gs" file listed. The right side has tabs for "Execution log" and "myFunction". The code editor contains the following code:

```
function myFunction() {
  Logger.log("Hello Google Apps Script");
}

// Log the number of Google Groups you belong to.
const groups = GroupsApp.getGroups(); Logger.log('You
are a member of %s Google Groups.', groups.length);
```

A tooltip for the `Logger.log` function is displayed, explaining its purpose: "Writes a formatted string to the logging console, using the format and values provided. The string can include multiple %s placeholders, which are replaced with corresponding values from the list of arguments, converted to strings."

Các giới hạn, giới hạn thực thi (run)

	Tài khoản Cá nhân/Miễn phí	Google Workspace (Trả phí)
Tổng thời gian chạy/lần	6 phút (360 giây)	6 phút (360 giây)
Tổng thời gian chạy/ngày	90 phút (1.5 giờ)	6 giờ
Số lần chạy đồng thời	30 lần	30 lần
Số lần gọi dịch vụ/ngày	20,000 lần	100,000 lần (hoặc cao hơn)

•Thời gian chạy/lần (6 phút):

Nếu script của bạn chạy lâu hơn 6 phút (ví dụ: xử lý lượng lớn dữ liệu Sheets), nó sẽ bị tự động dừng lại.

•Tổng thời gian chạy/ngày:

Nếu bạn có nhiều script chạy liên tục, tổng thời gian chúng chạy không được vượt quá giới hạn này.

Các giới hạn, giới hạn dịch vụ Google

A. Gmail (GmailApp / MailApp)

Hành động	Tài khoản Cá nhân/Miễn phí	Google Workspace (Trả phí)
Email người nhận/ngày	100 người nhận	1,500 - 2,000 người nhận
Tổng Email được gửi/ngày	100 email	1,500 - 2,000 email

B. Google Drive (DriveApp)

Hành động	Giới hạn hàng ngày (Xấp xỉ)
Tổng số lần gọi DriveApp/ngày	20,000 lần
Số lần đọc/ghi file/ngày	500,000 lần
Số lần tạo file/ngày	10,000 lần

Các giới hạn, giới hạn khác

A. Triggers

	Giới hạn
Tổng số triggers/người dùng	20 triggers
Thời gian chạy định kỳ tối thiểu	1 phút

B. Bộ nhớ Cache (Cache Service)

Kích thước dữ liệu/lần lưu:	100 KB
Thời gian lưu trữ tối đa:	6 giờ

C. Google Sheets

Tổng số cell / ô trong tài liệu, gồm: 10 triệu sheet x row x column	18,278
---	--------

D. Giới Hạn Mã (Code Limitations)

	Giới hạn
Kích thước dự án (Project Size)	50 MB
Dung lượng lưu trữ/script	10 MB
Thư viện bên ngoài	

Không thể sử dụng thư viện Node.js/npm trực tiếp.
Chỉ có thể sử dụng các Thư viện Apps Script khác.

E. UrlFetchApp (Kết nối bên ngoài/REST API)

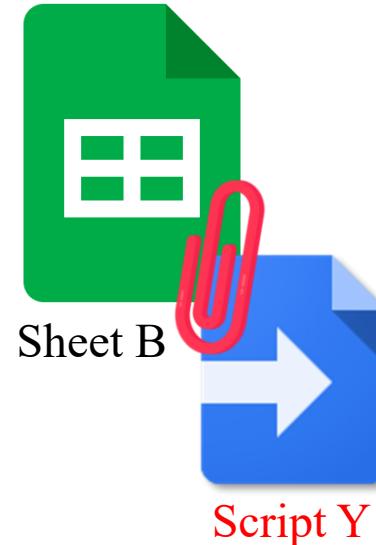
Tổng số lần gọi UrlFetch/ngày: (cho tài khoản miễn phí)	20,000 lần
Dịch vụ này được sử dụng khi bạn kết nối GAS với các dịch vụ bên ngoài như Firebase, Slack, hoặc các REST API khác	

Bound-Script, và Standalone-Script

Bound-Script (Gắn kết):

Là script **gắn kết** với một tài liệu cụ thể
ví dụ:

- *Script X được gắn kết với Doc A,*
- *Script Y được gắn kết với Sheet B*



Standalone-Script (Độc lập):

Là script **không gắn kết** với tài liệu cụ thể nào.



Bắt đầu

script.google.com

Danh sách các Project

Các thao tác với từng Project

Các hoạt động của các Project

New project

Hello Project

My Projects

Open project

Rename

Remove

Share

Locate in Drive

Triggers

Executions

Failed executions

Cloud logs

Bắt đầu

script.google.com

The screenshot shows the Google Apps Script interface. At the top, there's a purple header bar with the title "My Projects - Apps Script". Below it is a toolbar with icons for back, forward, search, and project details. The main area is divided into two panes: "My Projects" on the left and the "Project Editor" on the right.

Left Pane (My Projects):

- Header: Apps Script
- Buttons: New project, Starred Projects, My Projects (highlighted), All Projects
- Search bar: Search
- Table: My Projects

Project	Owner	Last modified
Hello Project	Me	5:08 PM

Right Pane (Project Editor):

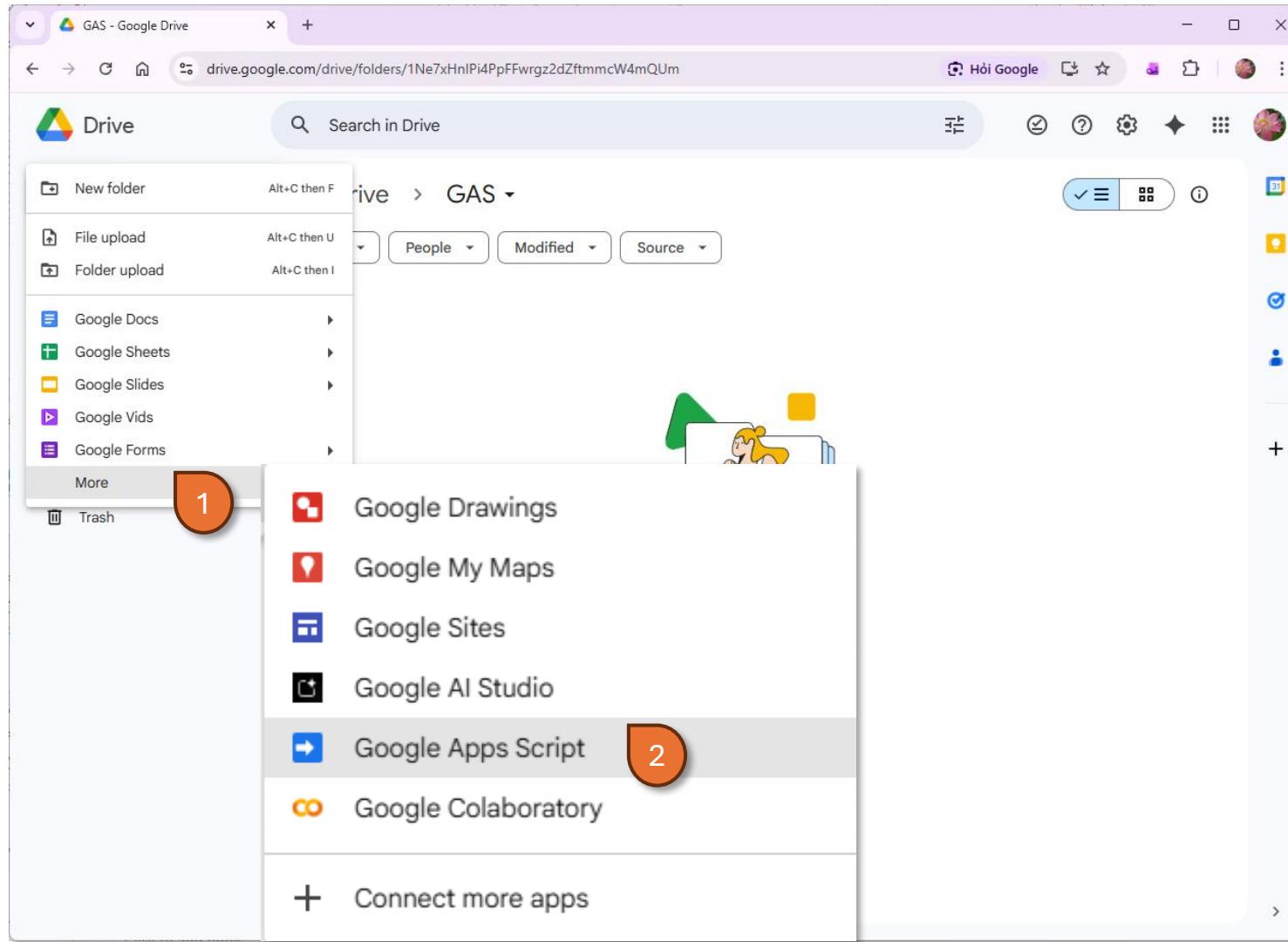
- Header: Hello Project
- Buttons: Deploy, Run, Debug, myFunction, Execution log
- File List: Files, Code.gs (selected), Libraries, Services
- Code Editor:


```
function myFunction() {
  Logger.log("Hello Google Apps Script");
}
```

Annotations:

- A yellow callout box labeled "Nội dung của Project" points to the "Code.gs" file in the file list.
- A red callout box labeled "Và các script có trong Project" points to the "Hello Project" entry in the "My Projects" table.
- A red arrow points from the text "Standalone-script không gắn với tài liệu nào" to the "Code.gs" file in the file list.
- A yellow callout box labeled "script.google.com" points to the URL bar at the top of the browser window.

Bound-Script, và Standalone-Script

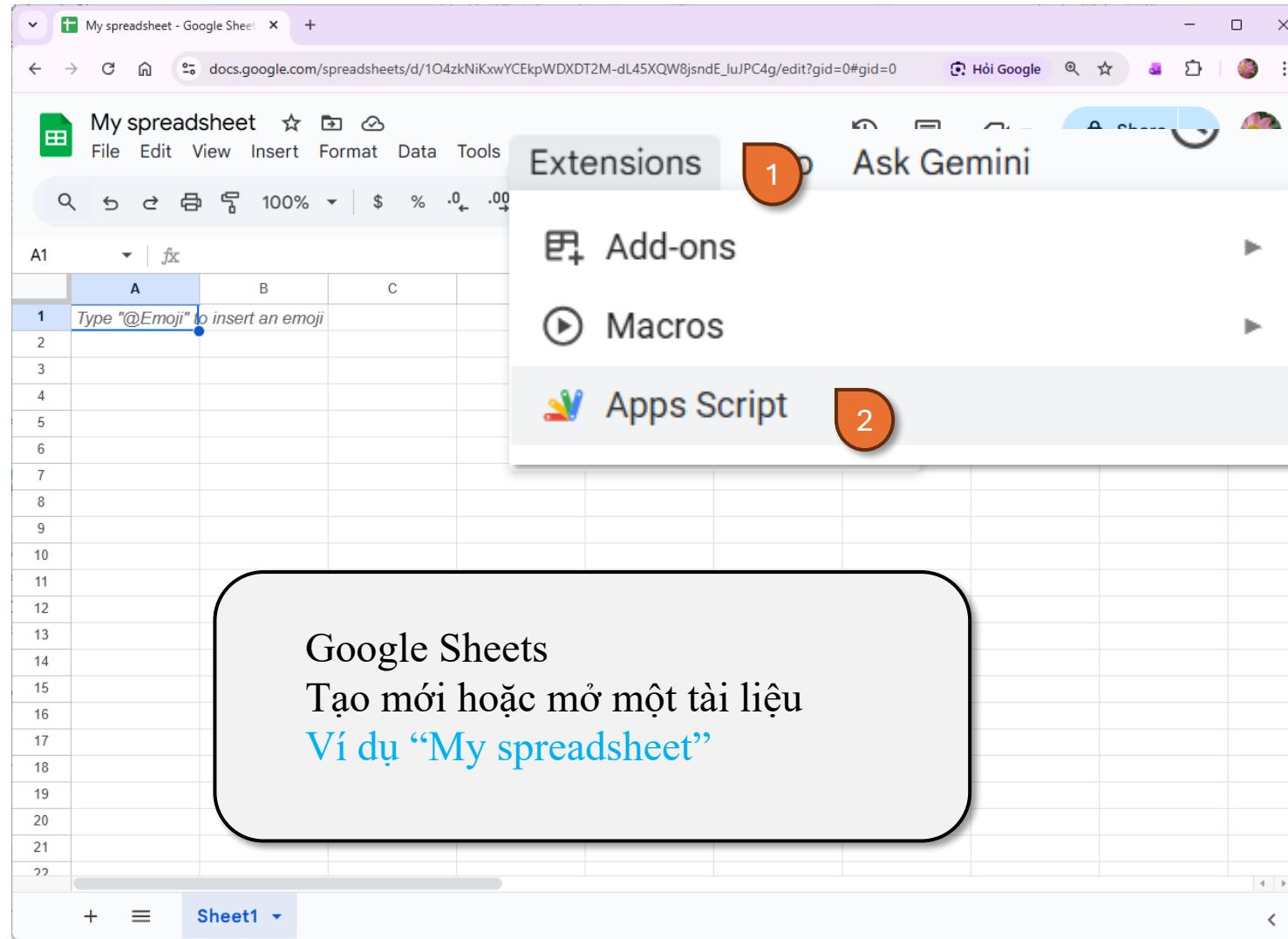


Tạo Standalone-Script (Độc lập):

Dùng dịch vụ Google Drive
 Chọn More từ “New File” và chọn tài liệu
 dạng “Google Apps Script”
 để mở Tab mới và bắt đầu tạo script

*Cách làm này gọi là tạo standalone-script.
 Nghĩa là script không gắn liền với tài liệu
 cụ thể nào*

Bound-Script, và Standalone-Script

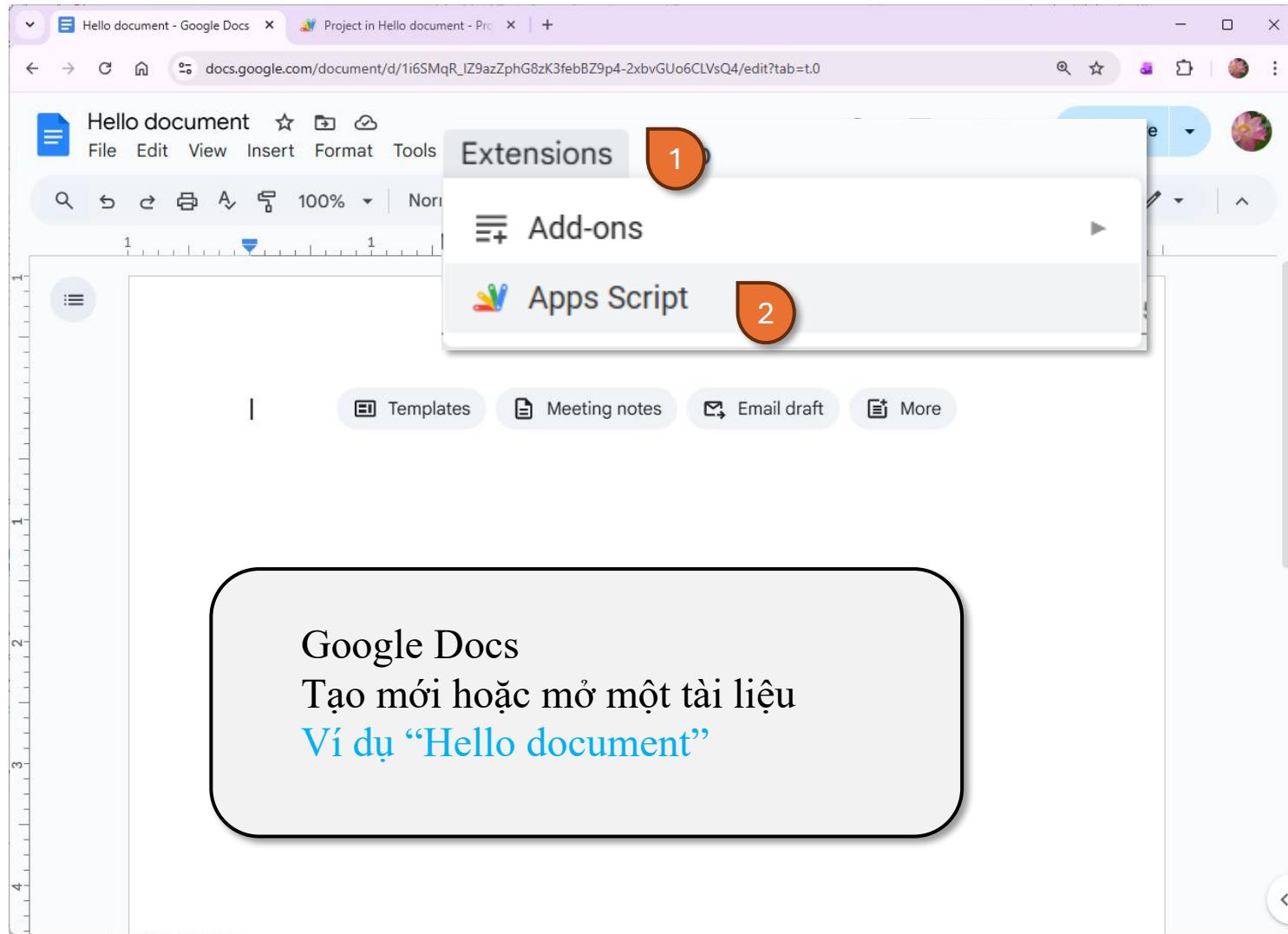


Tạo Bound-Script (Gắn kết):

Chọn menu “Apps Script”
để mở Tab mới và bắt đầu tạo script

*Cách làm này gọi là tạo bound-script.
Nghĩa là script được gắn liền với Sheet*

Bound-Script, và Standalone-Script



Tạo Bound-Script (Gắn kết):

Chọn menu “Apps Script”
để mở Tab mới và bắt đầu tạo script

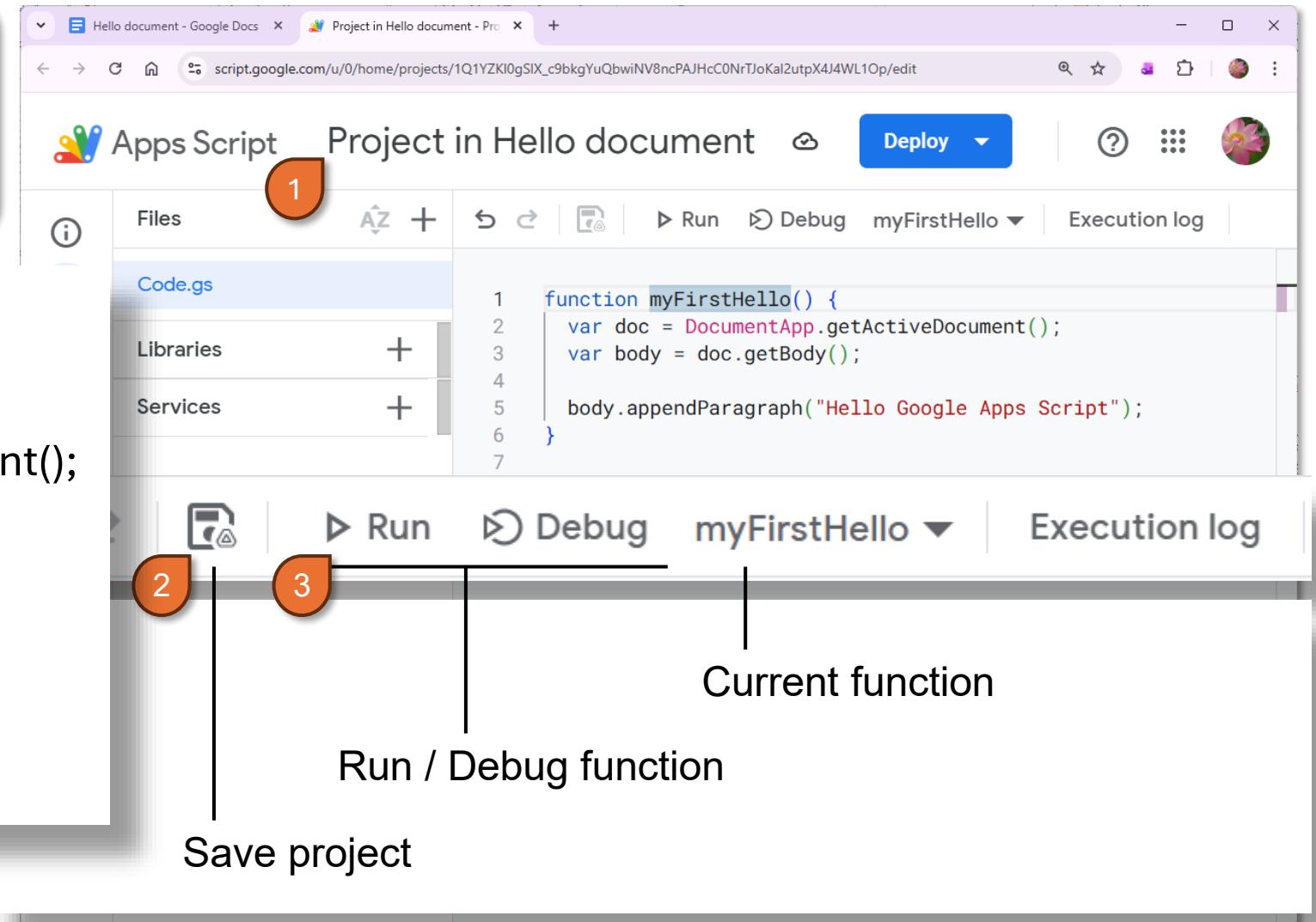
*Cách làm này gọi là tạo bound-script.
Nghĩa là script được gắn liền với Doc*

Ví dụ tạo Bound-Script, từ Google Docs

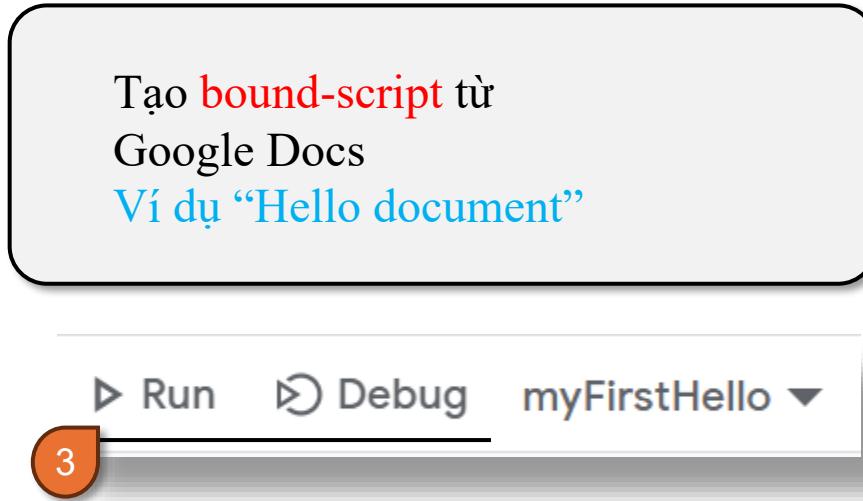
Tạo bound-script từ
Google Docs
Ví dụ “Hello document”

```
function myFirstHello() {
  var doc =
    DocumentApp.getActiveDocument();
  var body = doc.getBody();

  body.appendParagraph(
    "Hello Google Apps Script");
}
```

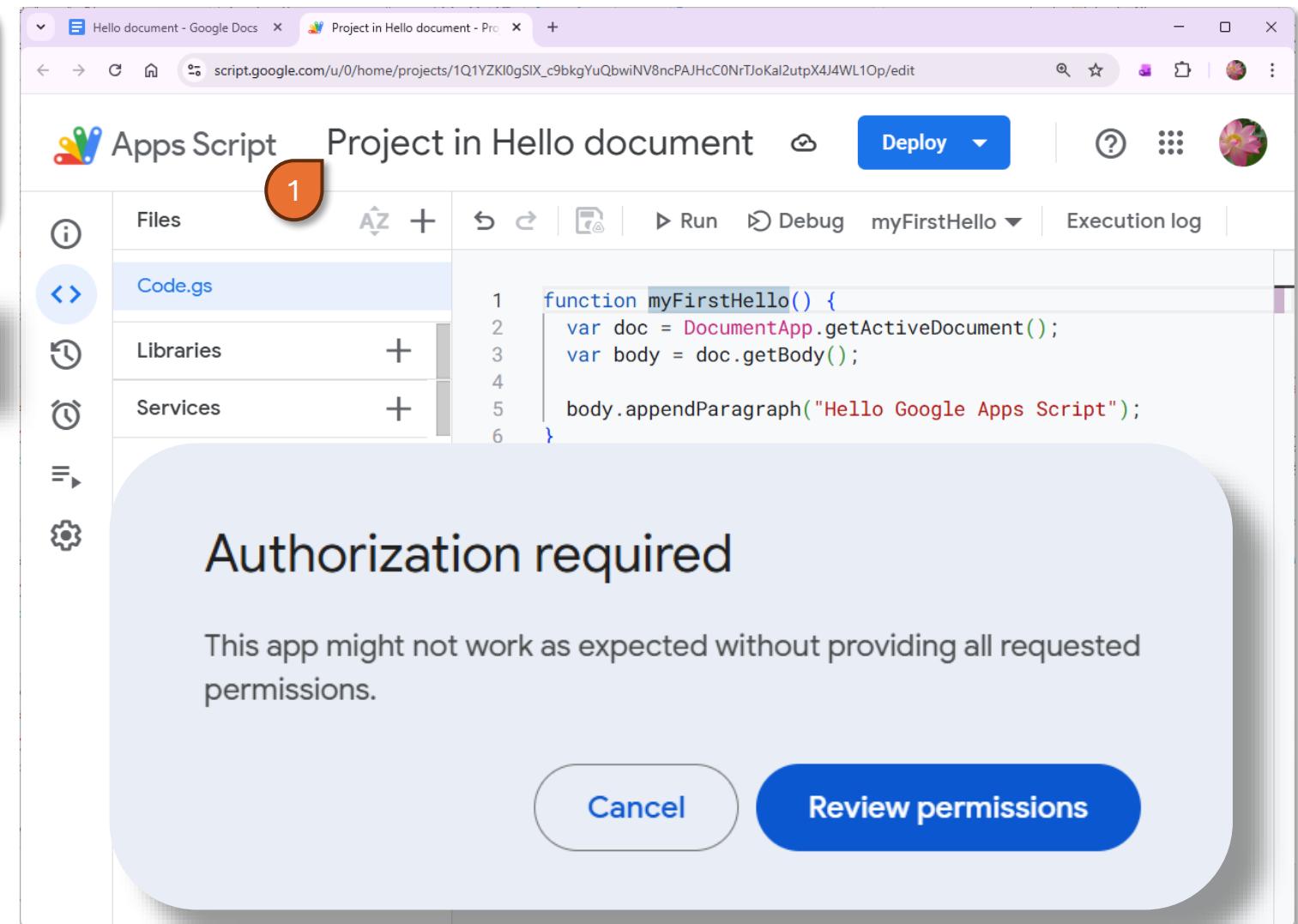


Ví dụ tạo Bound-Script, từ Google Docs



Run và Review permissions

khi Run lần đầu, cần cấp quyền
cho Project (chưa script)



Ví dụ tạo Bound-Script, từ Google Docs

Tạo bound-script từ Google Docs
Ví dụ “Hello document”

The screenshot illustrates the process of creating a bound script in Google Apps Script from a Google Document. On the left, a Google Doc titled "Hello document" contains the text "Hello Google Apps Script.". On the right, the Google Apps Script editor shows a project named "Project in Hello document". The "Code.gs" file contains the following code:

```
function myFirstHello() {  
  var doc = DocumentApp.getActiveDocument();  
  var body = doc.getBody();  
  
  body.appendParagraph("Hello Google Apps Script.");  
}
```

A red arrow points from the text in the Google Doc to the "Execution log" panel, which shows the execution results:

Time	Message	Details
1:57:54 PM	Notice	Execution started
1:57:55 PM	Notice	Execution completed

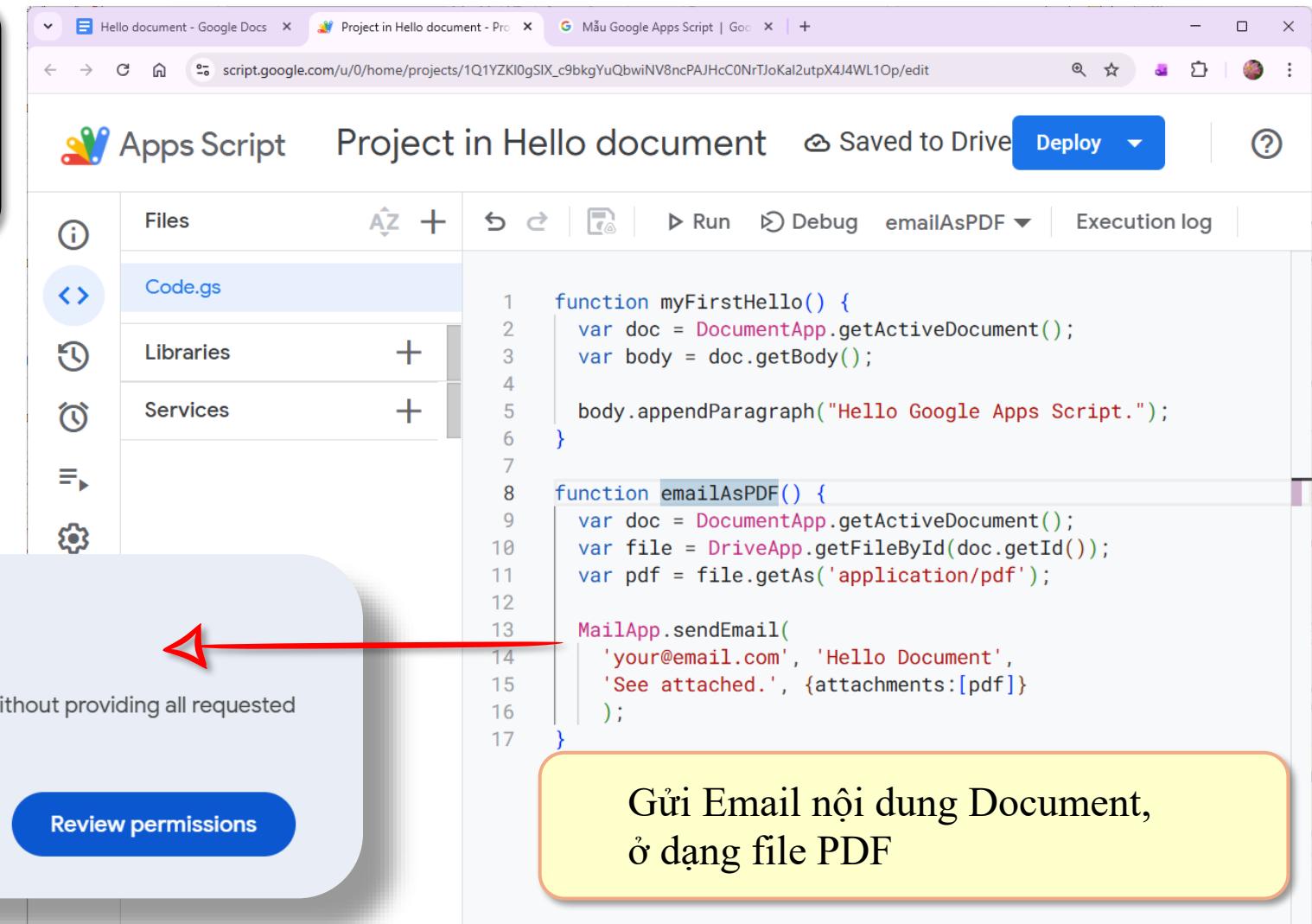
A yellow callout box highlights the "Execution log" panel with the text "Kết quả thực thi (Run) script".

Ví dụ tạo Bound-Script, từ Google Docs

Tạo bound-script + gửi email từ
Google Docs
Ví dụ “Hello document”

Lưu ý

Các dịch vụ có sử dụng trong script
như DocumentApp, DriveApp, MailApp
Cần được **cấp quyền** để Script có thể sử
dụng và thực thi được



```

function myFirstHello() {
  var doc = DocumentApp.getActiveDocument();
  var body = doc.getBody();

  body.appendParagraph("Hello Google Apps Script.");
}

function emailAsPDF() {
  var doc = DocumentApp.getActiveDocument();
  var file = DriveApp.getFileById(doc.getId());
  var pdf = file.getAs('application/pdf');

  MailApp.sendEmail(
    'your@email.com', 'Hello Document',
    'See attached.', {attachments:[pdf]}
);
}

```

Authorization required

This app might not work as expected without providing all requested permissions.

Cancel **Review permissions**

Gửi Email nội dung Document,
ở dạng file PDF

Các ví dụ Apps Script với Docs & Sheets, ...



100+Apps-Script-Examples.pdf



GoogleSheet+Apps-Script-Examples.pdf



Apps+Script+Code+Snippets+V2.pdf

Nội dung

Phần 2

Triggers, các cơ chế kích hoạt
Giao diện người dùng – UI và Services

- Menu tùy chỉnh
- Sidebar tùy chỉnh

Dịch vụ và APIs

- * **Dịch vụ Google Workspace:** `SpreadsheetApp`, `GmailApp`, `DocumentApp`, `DriveApp`, v.v.
- * **Dịch vụ nâng cao:** BigQuery, Google Analytics, YouTube Analytics, v.v.
- * **Dịch vụ tiện ích:** `UrlFetchApp` (gọi API ngoài), `CacheService`, `PropertiesService`.

Triggers (Các cơ chế kích hoạt)

Triggers

- * Cho phép script chạy tự động khi một sự kiện xảy ra.
- * - **Simple Triggers:** `onOpen()`, `onEdit()`, `onInstall()`.
- * - Installable Triggers: Có thể cài đặt để chạy theo thời gian hoặc khi có sự kiện (ví dụ: khi form được gửi).

Time-driven Triggers

Chạy script vào một thời điểm cụ thể hoặc theo một lịch trình lặp lại (hàng giờ, hàng ngày, hàng tuần).

Ví dụ:

- Gửi báo cáo hàng ngày
- Dọn dẹp dữ liệu hàng tuần.

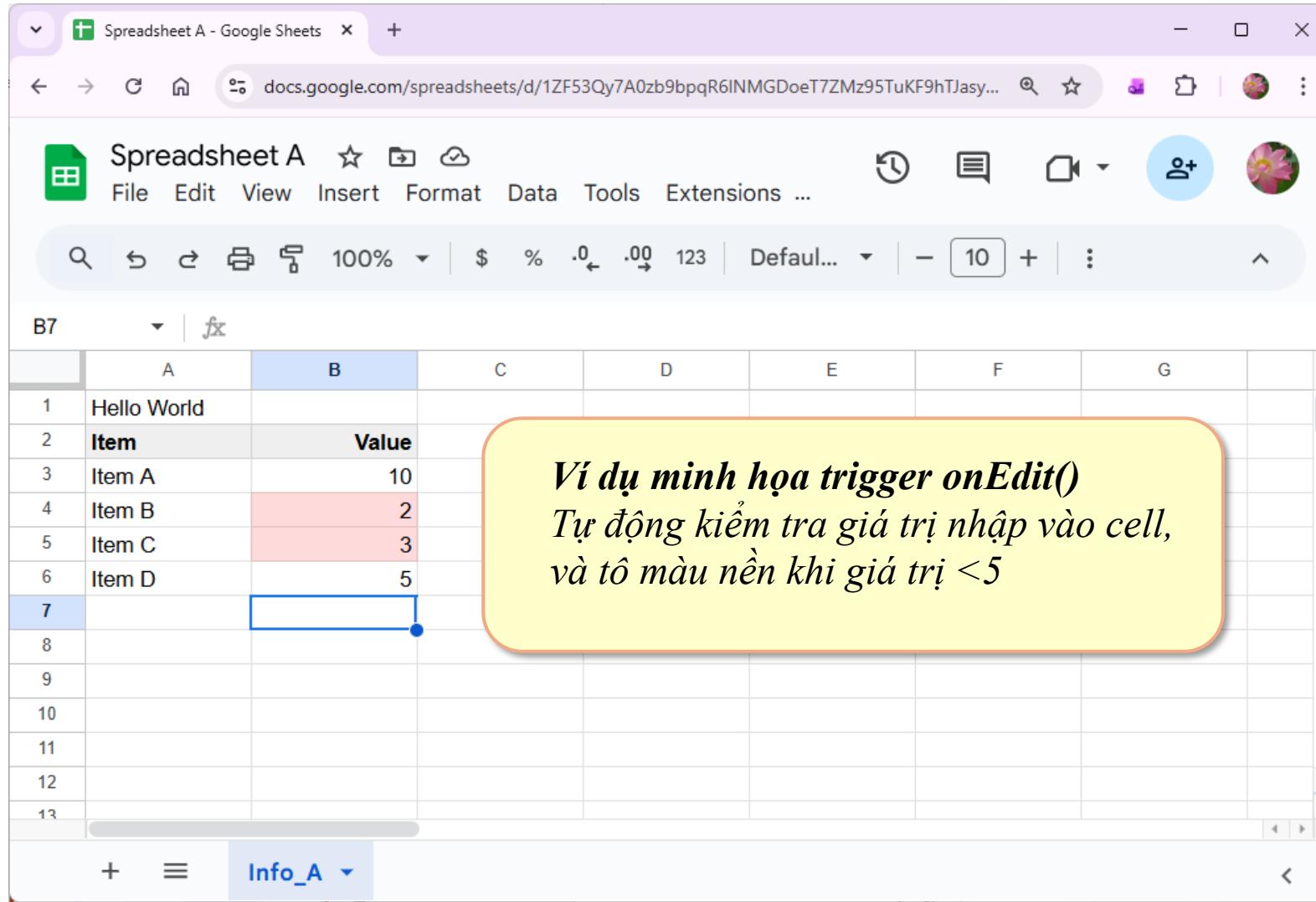
Event-driven Triggers

Chạy script khi có một sự kiện xảy ra trong một ứng dụng Google.

Ví dụ:

- Khi một hàng mới được thêm vào Google Sheets
- Khi một email mới được nhận
- Khi một sự kiện lịch được tạo.

Triggers, `onEdit()` (ví dụ Google Sheets)



The screenshot shows a Google Sheets interface with a table titled 'Item' and 'Value'. The 'Value' column uses conditional formatting where cells containing values less than 5 are highlighted with a pink background. A yellow callout box contains the text: 'Ví dụ minh họa trigger `onEdit()`. Tự động kiểm tra giá trị nhập vào cell, và tô màu nền khi giá trị <5'.

	A	B	C	D	E	F	G
1	Hello World						
2	Item	Value					
3	Item A	10					
4	Item B	2					
5	Item C	3					
6	Item D	5					
7							
8							
9							
10							
11							
12							
13							

Tạo bound-script
từ Google Sheets, sử dụng
`trigger onEdit()`

Lưu ý

Có thể sử dụng tính năng có sẵn Format \ **Conditional formatting** để thay thế cho Script + trigger `onEdit()`

→ Nên ưu tiên sử dụng các tính năng có sẵn để tối ưu hiệu suất

Triggers, onEdit (ví dụ Google Sheets)

Tạo bound-script
từ Google Sheets, sử dụng
trigger onEdit()

	A	B
1	Hello World	
2	Item	Value
3	Item A	10
4	Item B	2
5	Item C	3
6	Item D	5
7		

The screenshot shows the Google Apps Script editor interface. On the left, there's a sidebar with icons for Files, Libraries, and Services. The main area shows a script named 'Code.gs' with the following content:

```

1 /**
2 * Hàm onEdit(e) là một Simple Trigger, tự động chạy khi người dùng chỉnh sửa Cell.
3 * Tham số 'e' (Event Object) chứa tất cả thông tin về sự kiện chỉnh sửa.
4 *
5 * Lưu ý: ví dụ sau có thể dùng Conditional formatting, mà không cần tạo Apps Script.
6 */
7 function onEdit(e) {
8     /** 1. Kiểm tra xem sự kiện có hợp lệ không (ví dụ: người dùng đang chỉnh sửa một ô) */
9     if (!e || !e.range) {
10         Logger.log("Sự kiện chỉnh sửa không hợp lệ.");
11         return;
12     }
13
14     const range = e.range;
15     const newValue = e.value;
16
17     // Chúng ta chỉ muốn xử lý khi người dùng chỉnh sửa MỘT ô DUY NHẤT.
18     if (range.getNumRows() > 1 || range.getNumColumns() > 1) {
19         Logger.log("Chỉnh sửa nhiều ô, bỏ qua.");
20         return;
21     }
22
23     /** 2. Chuyển giá trị mới về dạng số (nếu có thể) */
24     const numberValue = parseFloat(newValue);
25
26     // LƯU Ý: Nếu người dùng xóa nội dung ô (trở thành rỗng), newValue sẽ là undefined.
27
28     /** 3. Thực hiện Logic: Kiểm tra giá trị */
29     if (!isNaN(numberValue) && numberValue < 5) {
30         // Giá trị là số và NHỎ HƠN 5
31         // Tô màu nền ĐỎ NHẠT (#FFDADA)
32         range.setBackground('#FFDADA');
33     } else {
34         // Giá trị >= 5, hoặc không phải là số, hoặc ô bị xóa (rỗng)
35         // Xóa định dạng nền (trở về màu trắng hoặc mặc định)
36         range.setBackground(null);
37     }
38 }

```

A red arrow points from the text "Lưu ý: ví dụ sau có thể dùng Conditional formatting, mà không cần tạo Apps Script." to the explanatory note in the code. Another red arrow points from the value '5' in cell B6 of the Google Sheets table to the logic in the script where it checks if the value is less than 5.

Giao diện người dùng (UI)

- * **Hộp thoại (Dialogs):** `SpreadsheetApp.getUi().alert()` và `prompt()`.
- * **Thông báo (Toasts):** `SpreadsheetApp.getActiveSpreadsheet().toast()`.
- * **Menu tùy chỉnh:** Thêm các mục menu mới vào giao diện người dùng của Google Sheets, Docs, Forms.
- * **Thanh bên (Sidebars):** Tạo các thanh bên tùy chỉnh với nội dung HTML.

Giao diện người dùng (UI)

The screenshot shows a Google Sheets interface with a custom menu. The menu is titled "Thông tin người dùng" (User Information) and is part of the "Menu của tôi" (My Menu). The menu item is highlighted in blue. The spreadsheet contains a table with data:

Item	Value
Item A	10
Item B	2
Item C	3
Item D	5

A yellow callout bubble with an orange border contains the following text:

Ví dụ minh họa
 Script tạo menu, do người dùng tự định nghĩa

Menu được tự động thêm vào Sheets mỗi khi Sheets được mở lên, bằng cách sử dụng trigger `onOpen()`

Tạo **bound-script** từ Google Sheets, sử dụng trigger `onOpen()`

Để tạo **menu tùy chỉnh**, và **Hộp thoại trình bày thông tin người dùng**

Giao diện người dùng (UI)

Tạo **bound-script**
từ Google Sheets, sử dụng
trigger onOpen()

Để tạo **menu tùy chỉnh**, và
**Hộp thoại trình bày thông
tin người dùng**

Menu của tôi

Thông tin người dùng

```

1 /**
2  * Hàm onOpen() chạy tự động khi Google Sheet được mở.
3  * Nó tạo ra một menu tùy chỉnh để người dùng dễ dàng chọn chức năng.
4  */
5 function onOpen() {
6   const ui = SpreadsheetApp.getUi();
7   ui.createMenu('Menu của tôi') // Tên menu
8     .addItem('Thông tin người dùng', 'getUserInfo') // Tên mục menu và hàm gọi
9     .addToUi();
10}
11
12
13
14 /**
15  * Hàm getUserInfo: Lấy và hiển thị thông tin của người dùng hiện tại.
16  * * LUU Y QUAN TRONG:
17  * - Session.getActiveUser().getEmail() chỉ trả về email nếu tài khoản là Google Workspace
18  * HOẶC nếu script được chạy dưới dạng Web App.
19  * - Đối với tài khoản cá nhân, nó có thể trả về một chuỗi rỗng trừ khi script được
20  * chạy với quyền cao hơn (như khi người dùng bấm vào menu).
21 */
22 function getUserInfo() {
23   const ui = SpreadsheetApp.getUi();
24
25   try {
26     ...
27   } catch (e) {
28     ...
29   }
30 }
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66

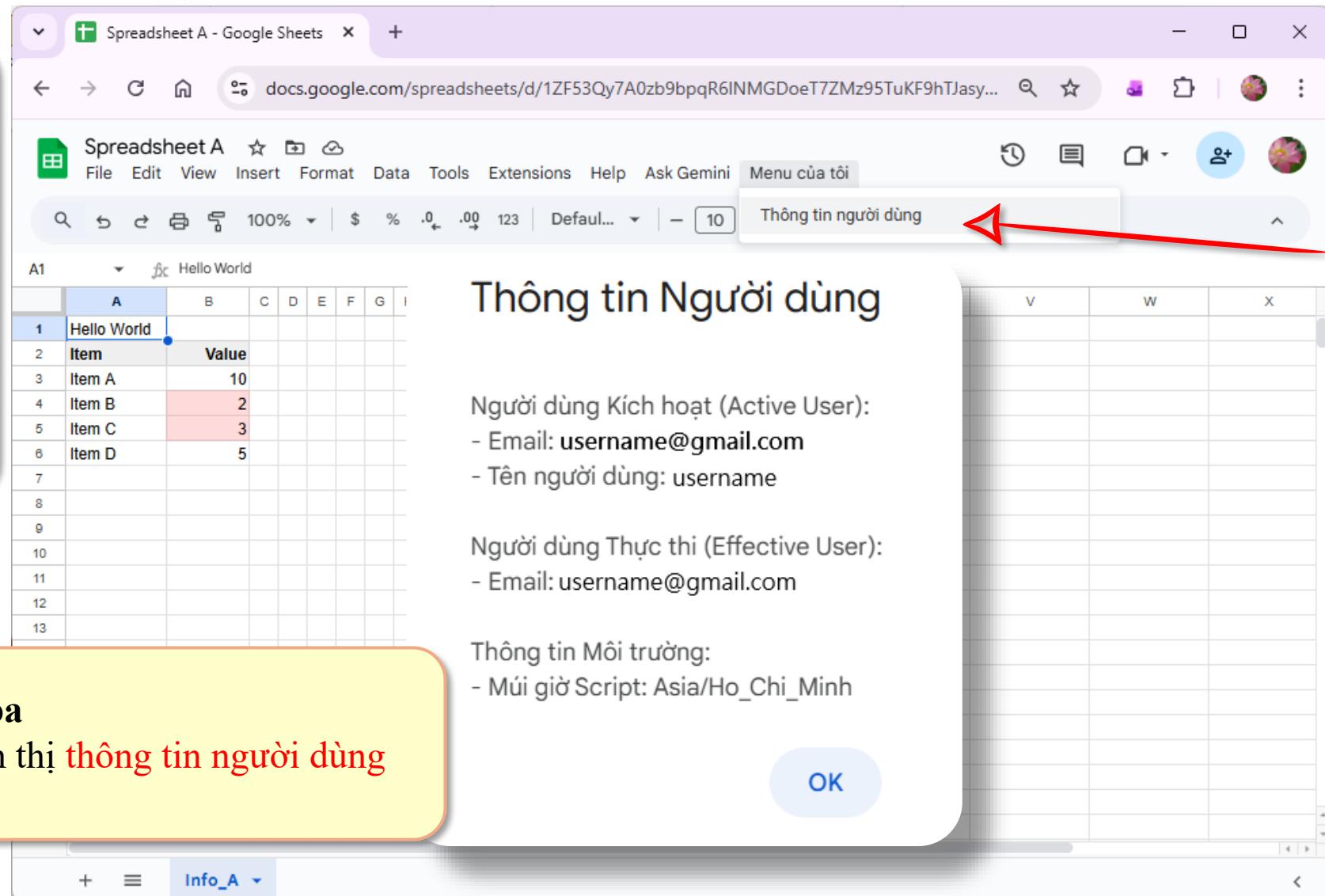
```

Giao diện người dùng (UI)

Tạo **bound-script**
từ Google Sheets, sử dụng
trigger onOpen()

Để tạo **menu tùy chỉnh**, và
Hộp thoại trình bày **thông
tin người dùng**

Ví dụ minh họa
Hộp thoại, hiển thị **thông tin người dùng**



Giao diện người dùng (UI)

Tạo bound-script
từ Google Sheets, sử dụng
trigger onOpen()

Để tạo menu tùy chỉnh, và
Hộp thoại trình bày thông
tin người dùng

The screenshot shows the Google Apps Script Project Editor interface. On the left, there's a sidebar with 'Files', 'Code.gs' (selected), 'Libraries', and 'Services'. The main area displays the 'Code.gs' script:

```

function getUserInfo() {
  const ui = SpreadsheetApp.getUi();

  try {
    // Lấy thông tin người dùng đang thực thi script
    const user = Session.getActiveUser();
    const userEmail = user.getEmail() || 'Không có sẵn (Có thể là tài khoản cá nhân)';
    const userName = user.getUsername() || 'Không có sẵn';

    // Lấy múi giờ của script
    const scriptTimezone = Session.getScriptTimeZone();

    // Lấy tên người dùng đang sử dụng
    // (Nếu không có email, có thể dùng cách này để thử lấy tên chung)
    const effectiveUser = Session.getEffectiveUser();
    const effectiveEmail = effectiveUser.getEmail() || 'Không có sẵn (Quyền thực thi)';

    // Tạo nội dung hộp thoại
    const infoMessage = `
      Người dùng Kích hoạt (Active User):
      - Email: ${userEmail}
      - Tên người dùng: ${userName}

      Người dùng Thực thi (Effective User):
      - Email: ${effectiveEmail}

      Thông tin Môi trường:
      - Múi giờ Script: ${scriptTimezone}
    `;
    ui.alert('Thông tin Người dùng', infoMessage, ui.ButtonSet.OK);
  } catch (e) {
    // Bắt lỗi nếu script không có quyền truy cập thông tin email
    Logger.log('Lỗi khi lấy thông tin người dùng: ' + e.toString());
    ui.alert('Lỗi', 'Không thể lấy thông tin người dùng. Vui lòng đảm bảo script đã được cấp quyền truy cập email.', ui.ButtonSet.OK);
  }
}

```

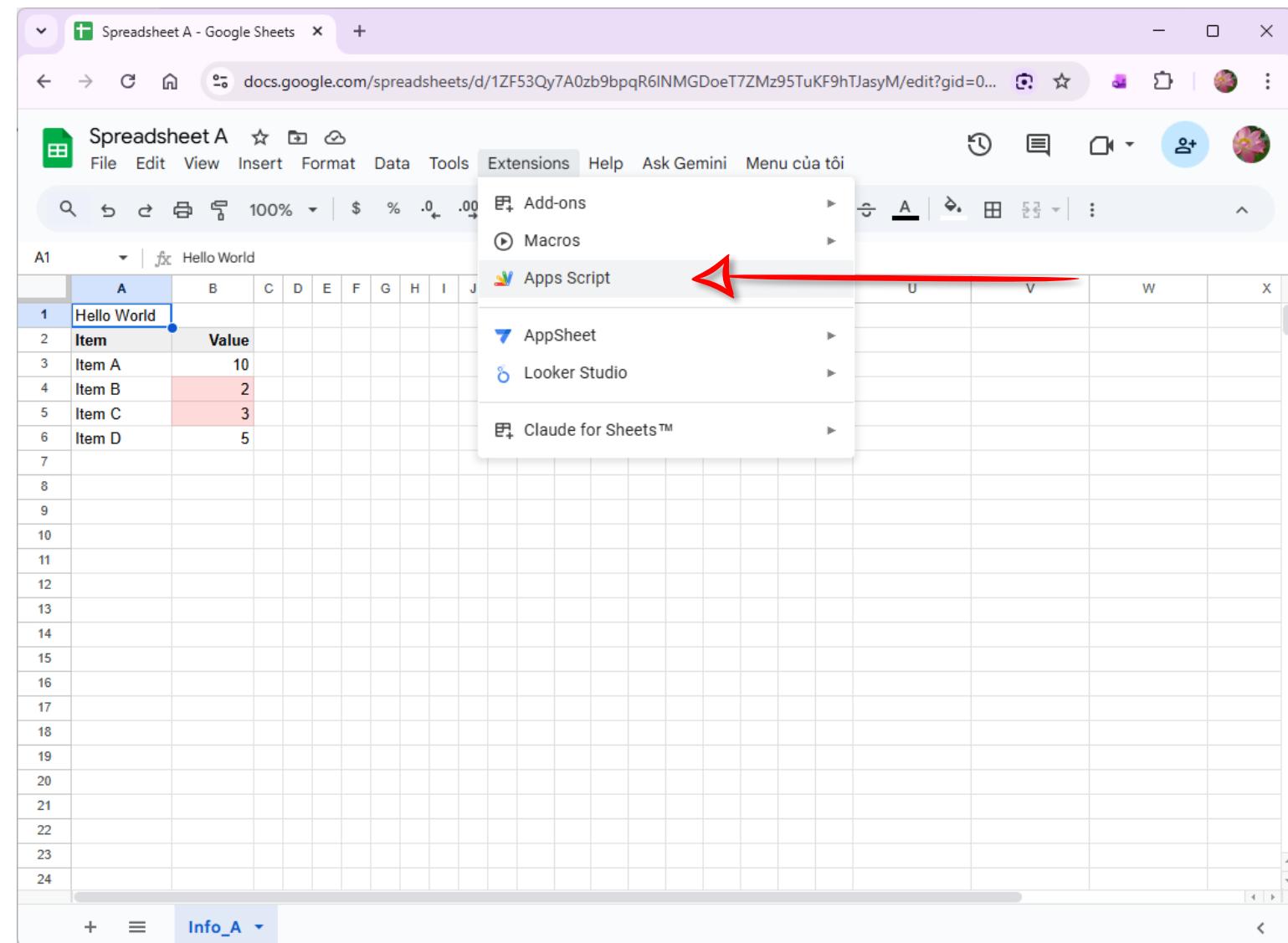
A modal dialog titled 'Thông tin Người dùng' is displayed in the foreground, containing the following information:

- Người dùng Kích hoạt (Active User):
 - Email: username@gmail.com
 - Tên người dùng: username
- Người dùng Thực thi (Effective User):
 - Email: username@gmail.com
- Thông tin Môi trường:
 - Múi giờ Script: Asia/Ho_Chi_Minh

An 'OK' button is at the bottom right of the dialog.

Giao diện người dùng (UI), Sidebar + HTML service

Tạo Sidebar từ Google Sheets

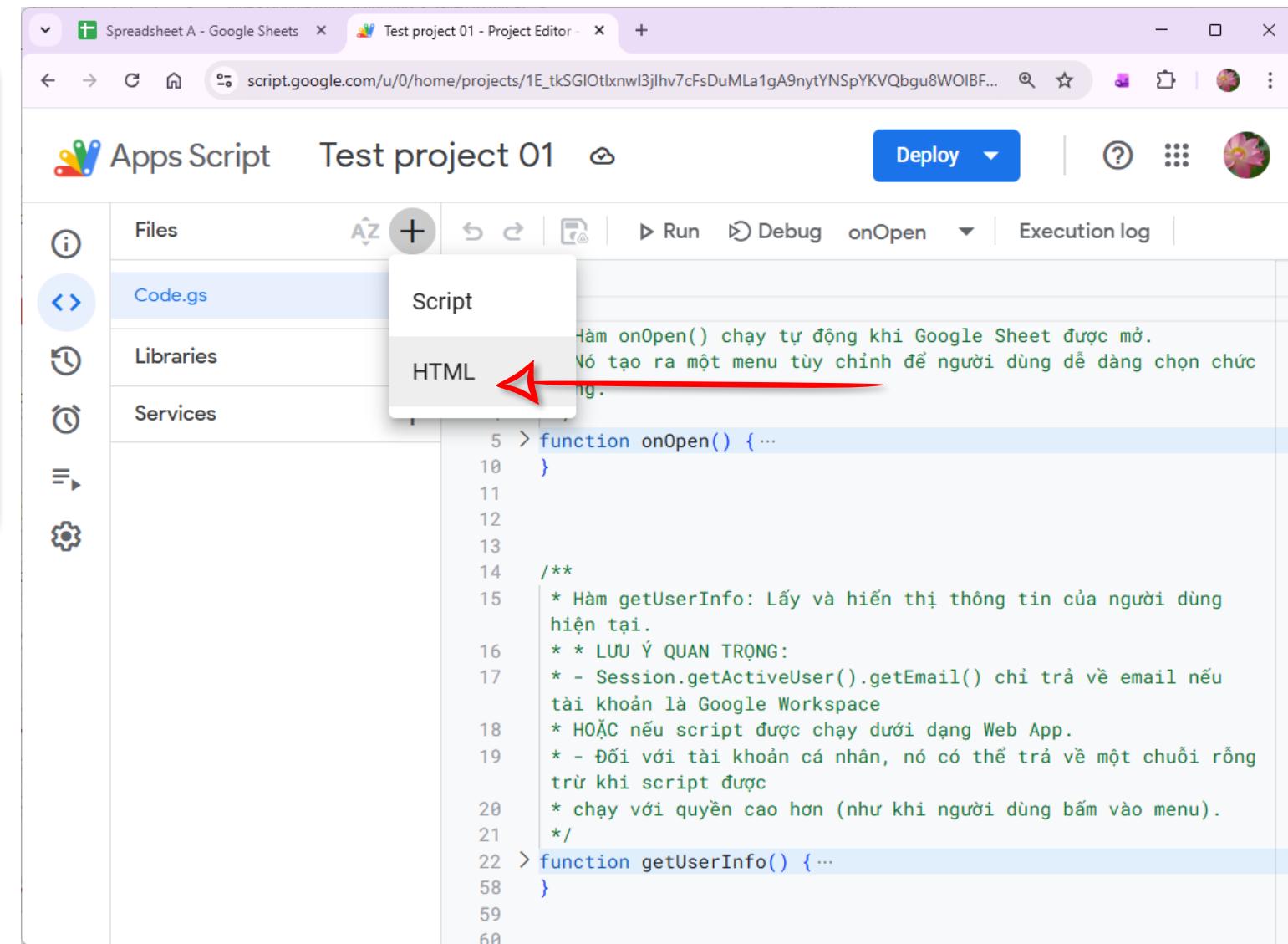


Giao diện người dùng (UI), Sidebar + HTML service

Tạo Sidebar từ Google Sheets

Nội dung gồm 2 file:

- File **Code.gs**: để lưu nội dung code để thực thi
- File **Sidebar.html**: để mô tả chi tiết các button (nút) trong sidebar, và liên kết với các function trong file Code.gs



Giao diện người dùng (UI), Sidebar + HTML service

Tạo Sidebar từ Google Sheets

Nội dung gồm 2 file:

1. File **Code.gs**: để lưu nội dung code để thực thi
2. File **Sidebar.html**: để mô tả chi tiết các button (nút) trong sidebar, và liên kết với các function trong file Code.gs

The screenshot shows the Google Apps Script Project Editor interface. On the left, there's a sidebar with icons for files, libraries, and services. A red arrow points from the 'Sidebar.html' file listed in the sidebar to the code editor on the right. The code editor contains the following HTML code:

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <base target="_top">
5   </head>
6   <body>
7
8   </body>
9 </html>
10

```

A yellow callout box with a red border and arrow points to the sidebar section, containing the following text:

Sidebar
Sử dụng **HTML + Tailwind CSS + JavaScript**
Để mô tả Sidebar và liên kết với Apps Script Code

Giao diện người dùng (UI), Sidebar + HTML service

Tạo Sidebar từ Google Sheets

Nội dung gồm 2 file:

1. File **Code.gs**: để lưu nội dung code để thực thi
2. File **Sidebar.html**: để mô tả chi tiết các button (nút) trong sidebar, và liên kết với các function trong file Code.gs



Code.gs



Sidebar.html

The screenshot shows a Google Sheets spreadsheet titled "Spreadsheet A". The sidebar on the right contains two buttons: "Lấy Thông tin Người dùng" and "Chức năng khác ...". A red arrow points from the "Code.gs" icon on the left towards the sidebar area.

	A	B	C	D	E
1	Hello World				
2	Item	Value			
3	Item A	10			
4	Item B	2			
5	Item C	3			
6	Item D	5			
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					
21					
22					
23					
24					
25					
26					
27					
28					
29					
30					
31					
32					

Nội dung

Phần 3

Google Apps Script với Firebase



Apps Script + Firebase

Firebase console

console.firebaseio.google.com/?pli=1

Hello, Tuan
Welcome back to Firebase!

Get started

- Create a new Firebase project**
Integrate Firebase products to super-charge your app
- Start coding an app**
Create a new app from one of the Firebase Studio templates
- Build an AI-powered Flutter app**
Deploy a sample app that showcases how the Gemini Live API, multimodal prompts, and image creation with Nano Banana all work in Flutter
- Try an agentic barista app**
Deploy a sample app that uses Firestore, Authentication, and function calling in Firebase AI Logic. Explore the code in Firebase Studio

Let's start with a name for your project^①

Project name: AppSheet-demo

appsheet-demo-2025

I accept the [Firebase terms](#)

Already have a Google Cloud project?
[Add Firebase to Google Cloud project](#)

Continue

Language: English (United States)

Support | Terms | Privacy Policy

console.firebaseio.google.com

Tạo Apps Script và Firebase
với Google Sheets

- Bước 1.1: Tạo Project**
từ Firebase Console
- Bước 1.2: Tạo Firestore Database**
- Bước 1.3: Lấy thông tin Project**
- Project ID
 - Web API key



Apps Script + Firebase \ Firestore Database

The screenshot shows the Firebase Console's Project Overview page for a project named "AppSheet-demo". The sidebar on the left includes categories like Firebase, Project Overview, Product categories, Build, Run, Analytics, and AI. The main area displays a welcome message: "Hello, Tuan" and "Welcome to your Firebase project!". Below this, there are sections for "Next steps with Gemini" (Tell us about your app, Build a backend, Monetize your app) and "Related development tools" (Firebase Studio). A red arrow points from the project name "AppSheet-demo" to the "Build" sidebar. In the "Build" sidebar, which is expanded, a red box highlights the "Firestore Database" option.

Tạo Apps Script và Firebase
với Google Sheets

Bước 1.1: Tạo Project
từ Firebase Console

Bước 1.2: Tạo Firestore Database

Bước 1.3: Lấy thông tin Project

- Project ID
- Web API key



Apps Script + Firebase \ Firestore Database

The screenshot shows the 'Create a database' wizard for Firestore Database. On the left, a sidebar lists various Google services: Build, App Check, App Hosting, Authentication, Data Connect, Extensions, **Firebase Database** (which is highlighted with a red arrow), Functions, Hosting, Machine Learning, Realtime Database, and Storage. The main area is titled 'Create a database' and shows step 1: 'Select edition'. It offers two options: 'Standard edition' (selected) and 'Enterprise edition'. A note below says, 'Not sure which edition is right for you? [Compare editions](#)'. Step 2: 'Database ID & location' and Step 3: 'Configure' are also listed.

Bước 1.2: Tạo Firestore Database

- a: Chọn Standard edition
- b: DatabaseID & Location
- c: Database Configure



Apps Script + Firebase \ Firestore Database

Build

- App Check
- App Hosting
- Authentication
- Data Connect
- Extensions
- Firebase Database** →
- Functions
- Hosting
- Machine Learning
- Realtime Database
- Storage

Create a database

1 Select edition

2 Database ID & location

Database ID
(default)

Location
asia-southeast1 (Singapore)

Your location setting is where your Cloud Firestore data will be stored

After you set this location, you cannot change it later. [Learn more](#)

[Next](#)

Configure

Bước 1.2: Tạo Firestore Database

- a: Chọn Standard edition
- b: DatabaseID & Location
- c: Database Configure



Apps Script + Firebase \ Firestore Database

Build

- App Check
- App Hosting
- Authentication
- Data Connect
- Extensions
- Firebase Database**
- Functions
- Hosting
- Machine Learning
- Realtime Database
- Storage

× Create a database

- 1 Select edition
- 2 Database ID & location
- 3 Configure

After you define your data structure, you will need to write rules to secure your data. [Learn more](#)

Start in **production mode**

Your data is private by default. Client read/write access will only be granted as specified by your security rules.

Start in **test mode**

Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

```
rules_version = '2';

service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if false;
    }
  }
}
```

i All third party reads and writes will be denied

Cancel **Create**

Bước 1.2: Tạo Firestore Database

- a: Chọn Standard edition
- b: DatabaseID & Location
- c: Database Configure



Apps Script + Firebase \ Project \ Project ID

Firebase

- Project Overview
- Project shortcuts
- Firestore Database
- Product categories
- Build
- Run
- Analytics
- AI

AppSheet-demo ▾

Project settings

General Cloud Messaging Integrations Service accounts Data privacy Users and permissions

Your project

Project name	AppSheet-demo
Project ID	appsheet-demo-2025

Project ID

A user-assigned unique identifier for your Firebase project. This identifier may appear in URLs or names for some Firebase resources, but it should generally be treated as a convenience alias to reference your project.

Tạo Apps Script và Firebase với Google Sheets

Bước 1.1: Tạo Project từ Firebase Console

Bước 1.2: Tạo Firestore Database

Bước 1.3: Lấy thông tin Project

- Project ID
- Web API key



Apps Script + Firebase \ Project \ Web API key

Tạo Apps Script và Firebase với Google Sheets

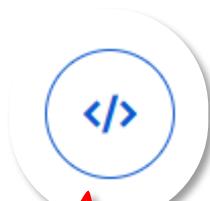
Bước 1.1: Tạo Project từ Firebase Console
 Bước 1.2: Tạo Firestore Database
Bước 1.3: Lấy thông tin Project

- **Project ID**
- **Web API key**

Web App \ Web API key
 Tạo Web API key từ ứng dụng Web App



Apps Script + Firebase \ Project \ Web API key



Tạo Web App

Add Firebase to your web app

1 Register app

App nickname ?

Also set up **Firebase Hosting** for this app. [Learn more](#)

Hosting can also be set up later. There is no cost to get started anytime.

Register app

2 Add Firebase SDK

Add Firebase to your web app

1 Register app

2 Add Firebase SDK

Use npm Use a <script> tag

```
<script type="module">
  // Your web app's Firebase configuration
  const firebaseConfig = {
    apiKey: "*****",
    authDomain: "appsheet-demo-2025.firebaseio.com",
    projectId: "appsheet-demo-2025",
  };
  // Others configuration
</script>
```

Continue to console

Bước 1.3: Lấy thông tin Project

- Project ID
- Web API key



Apps Script + Firebase \ Project \ Scripts

The screenshot shows a Google Sheets window titled "Input_Data_Sheet". The sheet contains a table with columns "Item" and "Value". The data is as follows:

	A	B	C	D
1	Item	Value		
2	Item A	100		
3	Item B	20		
4	Item C	30		
5	Item D	50		
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				

Below the table, there is a "Firestore Input" sidebar. It has a status message "Trạng thái: Sẵn sàng kết nối." and a red button labeled "Ghi Dữ liệu A2:B5 vào Firestore". Below the button, it says "Collection đích: sheet_input_data" and lists "Cột A: Item (stringValue)" and "Cột B: Value (doubleValue)".

Bước 2: Tạo Sheets và các Scripts, gồm

- Các thông tin cấu hình
- Logic đọc và ghi dữ liệu vào Firestore

Tạo Sheets với nội dung minh họa như hình bên, gồm 2 cột dữ liệu

Tạo các Scripts để :

- Thêm menu “Firebase Tools”
- Sidebar với chức năng đọc và ghi dữ liệu của Sheet vào Firestore Database (đã tạo ở bước 1.2)



Apps Script + Firebase \ Project \ Scripts

```

Input_Data_Sheet - Google Sheets GASandFirebase_project - Projects script.google.com/u/0/home/projects/18KfsAEF336COChrUsCj0-A9aRa1ORlk8iBdZg0jSahbrESRxxuwAEG/edit
Deploy Files Execution log
Code.gs
Utils.gs
Libraries Services
Code.gs

/*
 * HÀM CHẠY KHI MỞ FILE VÀ HIỂN THỊ SIDEBAR
 */
function onOpen() {
  // ...
}

/*
 * Hiển thị Sidebar (Thanh bên) sử dụng file FirebaseSidebar.html
 */
function showFirebaseSidebar() {
  // ...
}

// -----
// 2. HÀM XỬ LÝ LƯU DỮ LIỆU (SERVER-SIDE FUNCTION)
// -----

/*
 * Hàm đọc dữ liệu từ Google Sheet (vùng A2:B5) và gửi mỗi hàng thành một document Firestore.
 * @return {string} Trả về thông báo thành công hoặc thất bại.
 */
function saveDataToFirestore() {
  // ...
}

```

Bước 2: Tạo Sheets và các Scripts, gồm

- Các thông tin cấu hình
- Logic đọc và ghi dữ liệu vào Firestore



Code.gs



Utils.gs



FirebaseSidebar.html

Các **Scripts** gồm:

- **Code.gs:** file chính, chứa logic đọc / ghi dữ liệu và các function khác
- **Utils.gs:** file lưu thông tin cấu hình chính như Project, API key, data
- **FirebaseSidebar.html:** mô tả nội dung của sidebar và nút chức năng đọc và ghi dữ liệu



Apps Script + Firebase \ Database Rule

Ví dụ Database Rule

```

1 rules_version = '2';
2
3 service cloud.firestore {
4     match /databases/{database}/documents {
5         match /sheet_input_data/{document} {
6             allow read: if true;
7             allow write: if true;
8             //     allow write: if request.auth != null;
9         }
10    }
11 }
```

Bước 3: Thiết lập Database Rule

Lưu ý:

- Rule : là quy tắc của Firebase Database thiết lập các quyền, permission, để có thể đọc / ghi dữ liệu.
- Các thiết lập này (Web API key + Rule) **phù hợp với ứng dụng nội bộ**, và đơn giản.
- Sử dụng thiết lập **Service Account**, thay cho Web API key, để nâng cao độ bảo mật hơn



Apps Script + Firebase \ Project \ Run

Input_Data_Sheet - Google Sheets

docs.google.com/spreadsheets/d/1JRi-MzcWnwO3f8XeyoBlmXG9CFS-jebZdjmIrps-VYk/edit?gid=0#gid=0

Input_Data_Sheet

File Edit View Insert Format Data Tools Extensions Help Ask Gemini Firebase Tools

A1

	A	B	C	D
1	Item	Value		
2	Item A	100		
3	Item B	20		
4	Item C	30		
5	Item D	50		
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				

Sheet1



Bước 4: Ghi dữ liệu vào Firestore

Kết quả

- Thực thi các Scripts thành công.
- Đã ghi dữ liệu từ Sheet vào Firestore

Tham khảo

1. Tổng quan về Google Apps Script

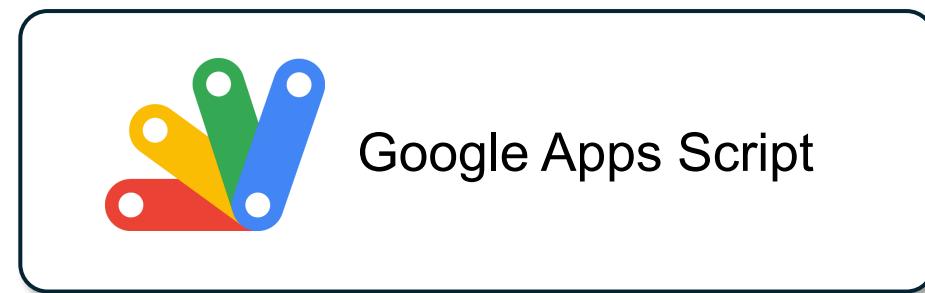
<https://developers.google.com/apps-script/overview?hl=vi>

2. Khóa học “Google Apps Script Complete Course Beginner to Advanced”

<https://fpl.udemy.com/course/apps-script-course/learn/lecture/10208194#overview>

3. Các ví dụ mẫu

<https://developers.google.com/apps-script/samples?hl=vi>



TRÂN TRỌNG CẢM ƠN

11-2025
Nhóm thực hiện
Thân Hoàng Lộc
Trần Anh Tuấn
...