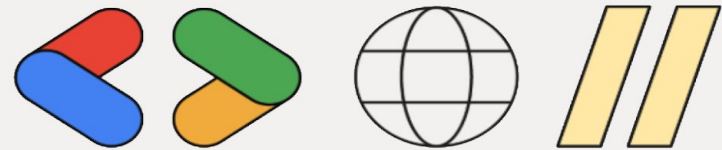


Start Scripting Today!



<https://script.google.com/home/>

Open Script Editor

1. Bound Script
2. Stand Alone Script

The screenshot shows a Google Sheets interface with a spreadsheet titled "AI chatbot Sheets". The menu bar includes File, Edit, View, Insert, Format, Data, Tools, Extensions, and Help. The toolbar shows various icons for search, undo, redo, print, and zoom, along with a zoom level of 100%. The spreadsheet has a table with the following data:

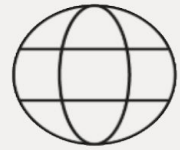
	A	B	C	D	E	F
1	Month	Sales				
2	Jan	1200				
3	Feb	1500				
4	Mar	2400				
5	Apr	1300				



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>

Apps Script



Apps Script can bring it all together

Automate & extend Google Workspace with simple code

Cloud-based JavaScript platform that lets you integrate with and automate tasks across Google products.

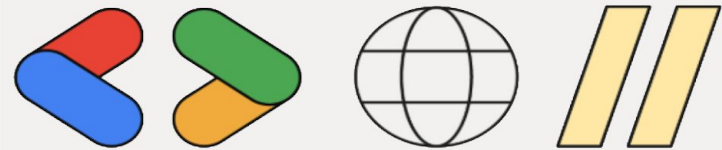
The JavaScript logo, consisting of the letters 'JS' in a bold, dark blue font on a yellow background.



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>

What you can do.....



1. Automate Tasks

Auto-generate reports (Sheets → PDF).

Schedule email reminders.

2. Custom Functions

=GETWEATHER("Miami") – live weather.

=CURRENCYCONVERT(100, "USD", "EUR")

3. Build Forms

Pre-fill Forms with Sheets data.

Auto-close Forms after deadlines.

4. Email Automation

Send emails for form submissions.

Notify users of tasks or deadlines.

5. Web Apps & Dashboards

Mini web apps for teams.

Internal tools with forms/buttons.

6. Data Analysis & Reports

Charts & PDF reports.

Analyze large datasets.

7. Workspace Integration

Sync Sheets, Docs, Calendar.

Update Calendar from Sheets.

8. File Management

Auto-create Drive folders/files.

Backup Docs/Sheets regularly.

9. Custom Triggers

Run scripts on schedules.

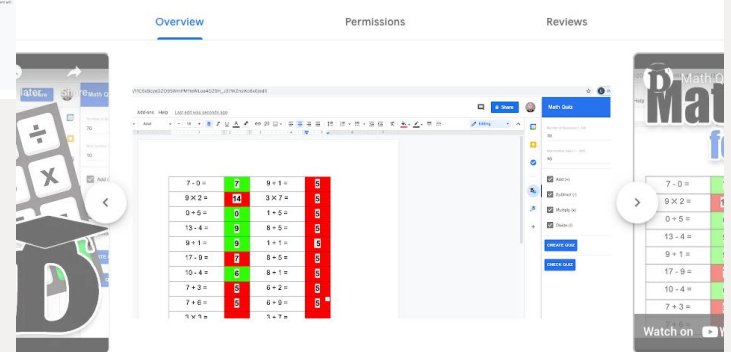
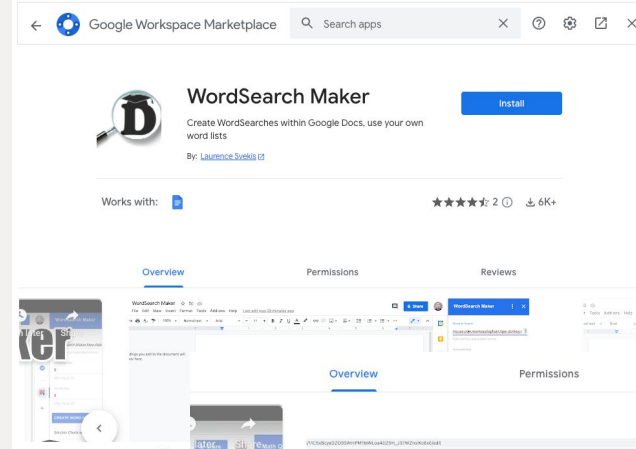
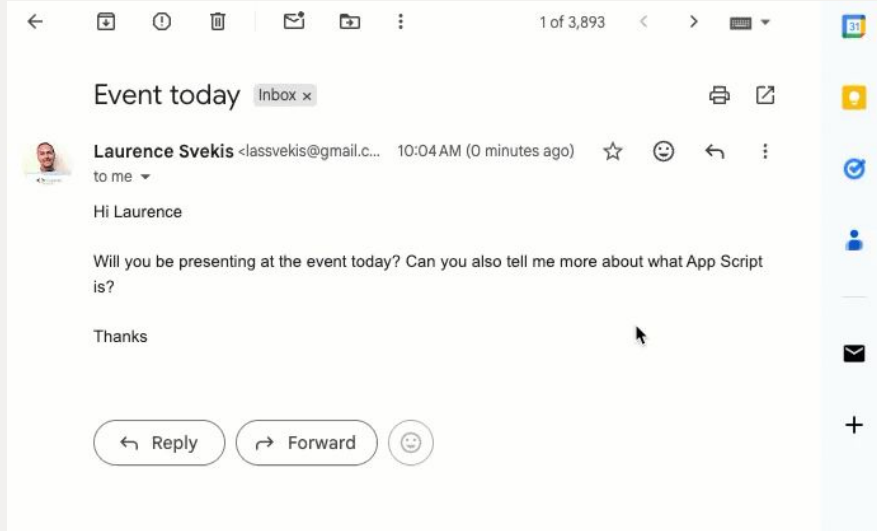
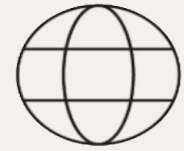
Trigger on form submissions.

10. Enhance UI

Add buttons, sidebars, menus.

Pop-ups for user input.

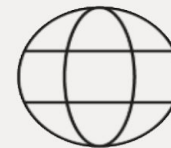
Add Ons Marketplace or Local



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>

Use Marco it creates Apps Script



AI chatbot Sheets ☆ 📁 ☁

File Edit View Insert Format Data Tools Extensions Help ...

🔍 ↶ ↷ 🖨 🗣 100% ▾ | \$ % .0 .00 123 | Default... ▾ - 10 + | B I ⌵ A 🔍 🏠 📄 ▾ | :

C2 ▾ | $\sum$

	A	B	C	D	E	F	G	H	I
1	Month	Sales							
2	Jan	1200							
3	Feb	1500							
4	Mar	2400							
5	Apr	1300							
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									



Google Developer Experts

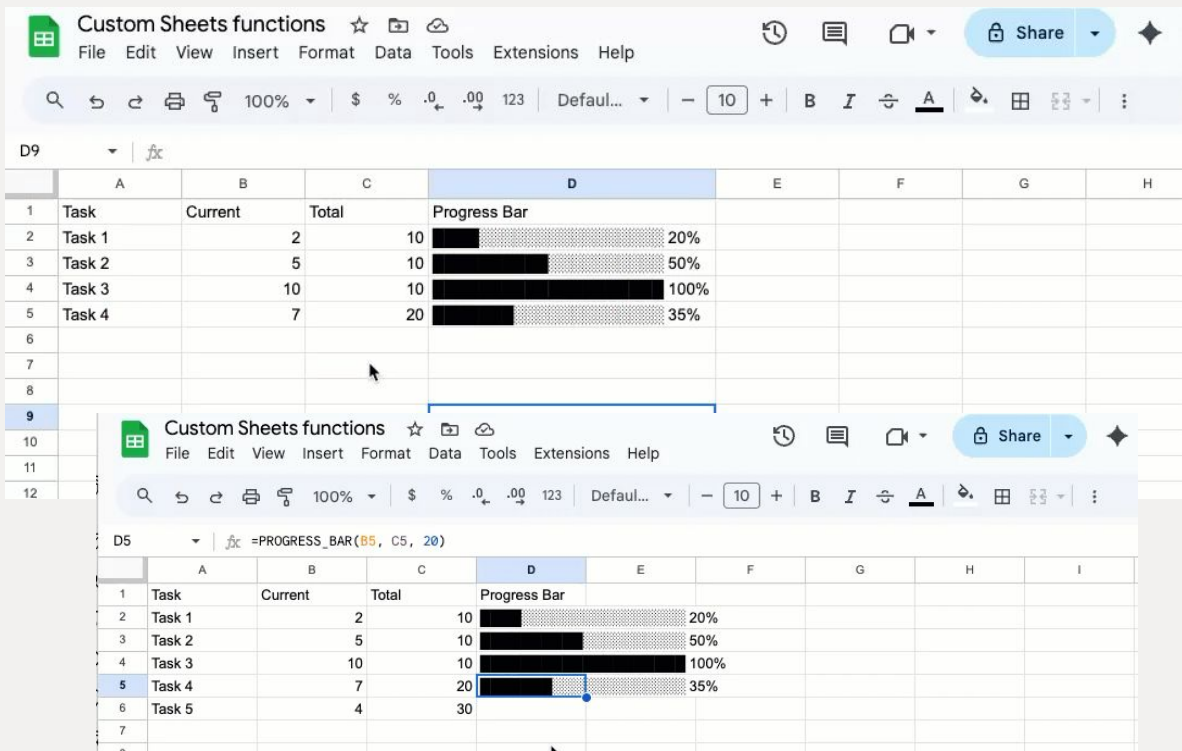
Laurence Svekis
<https://basescripts.com/>

Custom Functions in Sheet in action



1. In your Sheet: Extensions → Apps Script
2. Delete any placeholder code.
3. Paste everything from the code block below into [Code.gs](#).
4. Save → go back to the Sheet → use functions like `=TOUPPER_RANGE(A1:A5)`.

*You can select and drag the formula to other cells just like any other formula.



Sales Report to PDF



What Will This Script Do?

This Google Apps Script does the following:

*Add Sample Sales Data to a Google Sheet.

1. Create a New Google Doc for the report.
2. Insert a Data Table directly from the sheet into the Google Doc.
3. Generate a Dynamic Chart from the sheet's data.
4. Embed the Chart as an Image in the Google Doc.
5. Export the Google Doc as a PDF.
6. Send the PDF Report via Email to the person who runs the script.

*Setup Trigger for the report



[GitHub Code](#)



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>

Monthly Sales Report Inbox x



lassvekis@gmail.com

to me ▾

1:12 PM (11 minutes ago)



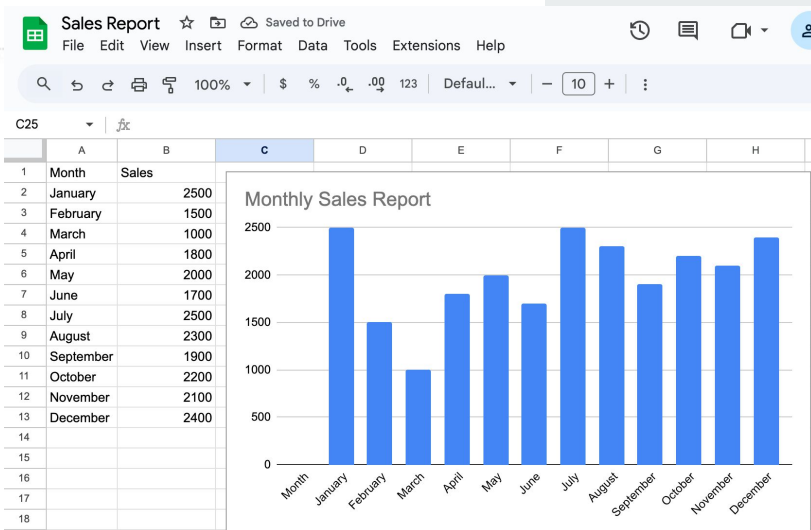
Attached is your automatically generated Monthly Sales Report PDF.

One attachment • Scanned by Gmail ⓘ



← Reply

→ Forward



GitHub Code Sales Report Sheet

Triggers in Apps Script



Triggers allow you to create event-driven workflows. For example:

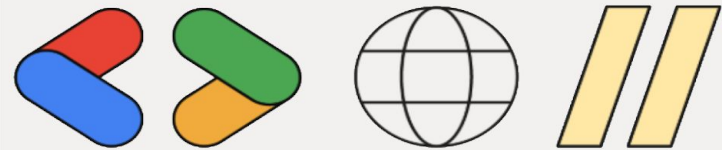
- Run a script every day at 8 AM.
- Automatically send an email when a new row is added to a Google Sheet.
- Notify users when a Google Form is submitted.
- Run clean-up scripts when a document is closed.

There are two main types of triggers in Apps Script:

1. **Simple Triggers:** Run automatically without any manual setup. Examples include `onOpen`, `onEdit`, and `onFormSubmit`.
2. **Installable Triggers:** Manually installed via the Triggers Dashboard and provide more flexibility (like specifying exact times and events).



Setting up a Trigger



How to Set It Up:

1. Open the Triggers Dashboard.
2. Click + Add Trigger.
3. Select the sendDailyReport function.
4. Choose Time-driven → Day timer → 8:00 AM.

Trigger Type	Use Case
onOpen	Add custom menu to Sheets/Docs/Forms
onEdit	Run scripts on specific cell edits (like checkboxes)
onFormSubmit	Send thank-you emails after form submissions
Time-Driven	Schedule daily reports or clean up old files
Custom Add-ons	Configure add-ons during install

Event	Simple triggers	Installable triggers
Open	    Sheets Slides Forms* Docs function onOpen(e)	   Sheets Forms* Docs
Edit	 Sheets function onEdit(e)	 Sheets
Selection change	 Sheets function onSelectionChange(e)	
Install	    Sheets Slides Forms Docs function onInstall(e)	
Change		 Sheets
Form submit		  Sheets Forms
Time-driven (clock)		     Sheets Slides Forms Docs Standalone
Get	 Standalone function doGet(e)	
Post	 Standalone function doPost(e)	



Custom UI menu Item



```
1. function onOpen() {  
2.   var ui =  
   SpreadsheetApp.getUi();  
3.   ui.createMenu('🔧 Custom  
   Tools')  
4.     .addItem('Create Progress  
   Test Data',  
   'createProgressBarTestData')  
5.     .addItem('Show Welcome  
   Message',  
   'showWelcomeMessage')  
6.     .addSeparator()  
7.     .addItem('Reload Menu',  
   'onOpen')  
8.     .addToUi();  
9. }
```

The screenshot shows the Google Sheets interface for a file named 'AI chatbot Sheets'. The custom menu 'Custom Tools' is visible in the top toolbar. The spreadsheet contains the following data:

	A	B	C	D	E	F	G	H	I	J
1	Month	Sales								
2	Jan	1200								
3	Feb	1500								
4	Mar	2400								
5	Apr	1300								
6										
7										
8										
9										
10										
11										
12										



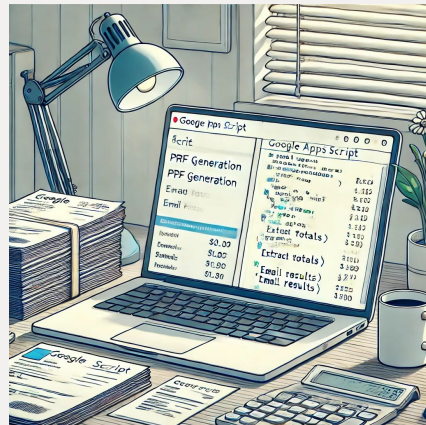
Get Text from a PDF



Challenge: Extracting specific information from PDF files, such as invoice totals.

Solution: Get data from PDF in a folder, send the summary of the data to an email.

- Extract key details, like totals, from PDFs using OCR.
- Email a summary of the extracted information directly to your inbox.



[GitHubCode](#)



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>



1 of 2,528

Extracted Totals from PDF Files Inbox x



@gmail.com

to me ▾

4:08 PM (2 minutes ago)



Here are the total values extracted from the PDF files in your folder:

File: Sample Invoice 3.pdf - Total Value: 300.00

File: Sample Invoice 2.pdf - Total Value: 150.00

File: Sample Invoice 1.pdf - Total Value: 75.00

↩ Reply

➦ Forward



My Drive > PDF ▾

Type ▾

People ▾

Modified ▾

Name ↑

Owner



Sample Invoice 1.pdf



me



Sample Invoice 2.pdf



me



Sample Invoice 3.pdf



me

[GitHubCode](#)

Fetch & Store API Data



Objective: Automate data fetching and storage.

Key Steps:

1. Create a new Google Spreadsheet.
2. **Fetch 50 random users via the Random User API.**
3. Extract key details: Name, Gender, Email, Phone, Location.
4. Populate the spreadsheet automatically.



[GitHubCode](#)

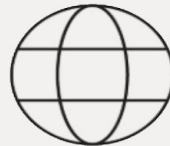
Use Case: Generate sample data for testing, analysis, or API integration practice.



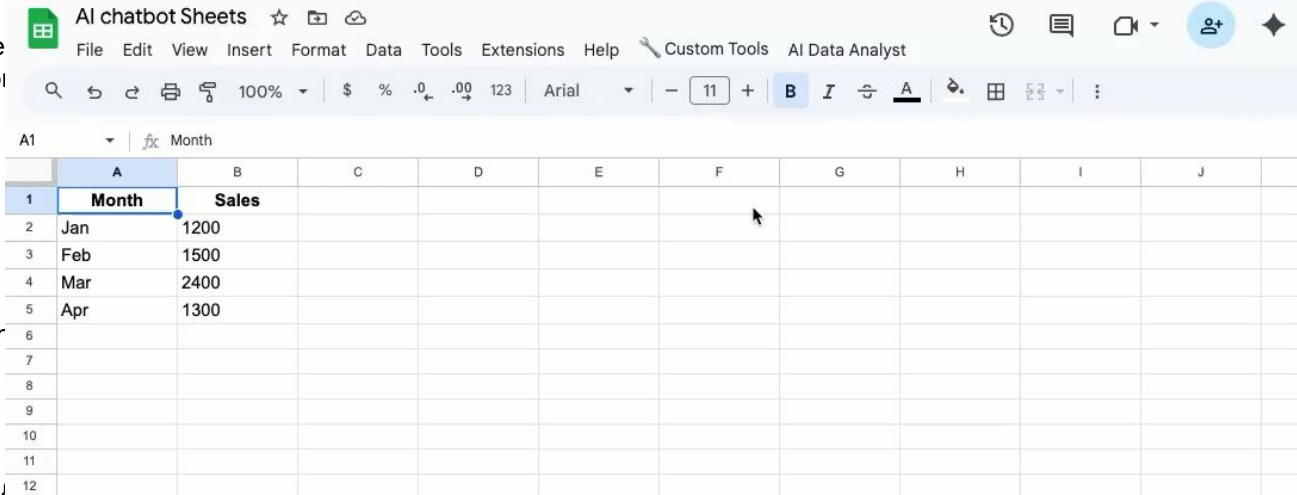
Google Developer Experts

Laurence Svekis
<https://basescripts.com/>

Call Gemini



```
function callGemini_(prompt, modelId = 'gemini-2.5-flash') {
  const apiKey = PropertiesService.getScriptProperties().getProperty('GEMINI_API_KEY');
  if (!apiKey) {
    throw new Error('GEMINI_API_KEY is not set in Script Properties.'):
  }
  const url = `https://generativelanguage.google
  const payload = { contents: [{ parts: [{ text: prompt
  const options = {
    method: 'post',
    contentType: 'application/json',
    headers: { 'x-goog-api-key': apiKey },
    muteHttpExceptions: true,
    payload: JSON.stringify(payload)
  };
  const response = UrlFetchApp.fetch(url, options);
  const text = response.getContentText();
  const code = response.getResponseCode();
  if (code !== 200) {
    console.error('Gemini error', code, text);
    throw new Error(' Gemini API error: ${code} ${text} ');
  }
  const data = JSON.parse(text);
  return data.candidates?.[0]?.content?.parts?.[0]?.text || '';
}
```



	A	B	C	D	E	F	G	H	I	J
1	Month	Sales								
2	Jan	1200								
3	Feb	1500								
4	Mar	2400								
5	Apr	1300								
6										
7										
8										
9										
10										
11										
12										



Gemini Create a Doc



1. Get your API Key <https://aistudio.google.com/app/apikey>
2. Create new file utilis.gs and add a key value and add model path <https://deepmind.google/technologies/gemini/>
3. Make request to the model and do something with the response

```
const geminiApiKey = 'AIz*****iI';  
const geminiModel =  
  `https://generativelanguage.googleapis.com/v1beta/models/gemini-1.5-  
  flash-latest:generateContent?key=${geminiApiKey}`;
```



```
function callGemini(q, temperature=0) {
  const payload = { contents: [{ parts: [{ text: q }] }] };
  const options = {
```



Gemini API Response



File Edit View Insert Format Tools Extensions Help



```
: 'application/json', 'payload': JSON.stringify(payload)
```

```
h(geminiModel, options);
```

```
getContentText();
```

```
][0][ "content" ][ "parts" ][ 0 ][ "text" ];
```

Search Undo Redo Print Bold Italic Underline Text Color Link

1 2 3 4 5 6



****Prompt**:**

What is the future of AI

****Gemini Response**:**

Predicting the future of AI is inherently speculative, but based on current trends and developments, we can outline several likely trajectories:

****Near-Term Future (Next 5-10 years):****

****Increased Automation:**** AI will continue to automate more tasks across various industries, impacting jobs but also creating new ones. This will include more sophisticated robotic process automation (RPA), self-driving vehicles becoming more prevalent, and AI-powered tools enhancing productivity in fields like healthcare and finance.

****Improved Natural Language Processing (NLP):**** We'll see more natural and nuanced interactions with AI systems. Chatbots will become more sophisticated, capable of handling complex conversations and providing more helpful assistance. AI-powered translation will improve significantly, breaking down language barriers.

****Personalized Experiences:**** AI will power increasingly personalized experiences in areas like entertainment, education, and healthcare. Recommendation systems will become more accurate and sophisticated, tailoring content and services to individual needs and preferences.

****Advancements in Computer Vision:**** AI's ability to "see" and interpret images and videos will improve drastically, leading to better medical diagnoses, improved security systems, and more effective autonomous navigation.

****Greater Accessibility:**** AI tools will become more accessible to individuals and smaller businesses, lowering the barrier to entry for leveraging AI's power. This will be driven by cloud computing and the development of user-friendly AI platforms.

```
e() {
```

```
e of AI"; // The prompt for the Gemini API
```

```
.(prompt); // Get the response from Gemini API
```

```
 Gemini API Response');
```

```
'.setHeading(DocumentApp.ParagraphHeading.HEADING2);
```

```
Add the prompt
```

```
sponse**').setHeading(DocumentApp.ParagraphHeading.HEADING2);
```

```
se); // Add the response from the Gemini API
```

```
${docUrl}`);
```

[GitHubCode](#)

Gemini Doc Summaries



Objective: Automate doc summarization, and email delivery **using Gemini AI**

Key Steps:

1. Extract all Google Docs from a specific folder.
2. **Use Gemini AI to generate summaries of the content.**
3. Email the combined summaries to the user.

Use Case: Ideal for executive summaries, reports, and quick document overviews.



[Doc Summaries with Gemini GitHub](#)



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>

Summary of Google Docs (Generated by Gemini AI) Inbox x



@gmail.com

to me ▾

4:29 PM (11 minutes ago) ☆ 😊

Here is a summary of the contents from the Google Docs in your folder (powered by Gemini AI):

Sample Document 3

This is a short introductory paragraph for a document explaining a data aggregation process. It successfully sets the stage by:

* **Clearly stating the document's purpose:** Extracting data from multiple Google Docs and combining them into a single summary.

* **Highlighting the benefit:** Useful for reports, overviews, and executive summaries.

* **Using concise and accessible language:** Easy to understand for a wide audience.

The only potential improvement would be to briefly mention *how* the process will be demonstrated (e.g., "using [specific tool/method]"). This would further enhance reader expectation and engagement.

Sample Document 2

This short document highlights the benefits of cloud automation, specifically mentioning how Google Apps Script can be used to create custom workflows and boost workplace productivity. The core message is that automation saves time and effort.

My Drive > Docs ▾

Type ▾

People ▾

Modified ▾

Name ↑

Owner



Sample Document 1



me



Sample Document 2



me



Sample Document 3



me

Doc Summaries with Gemini GitHub

Image Descriptions with Gemini

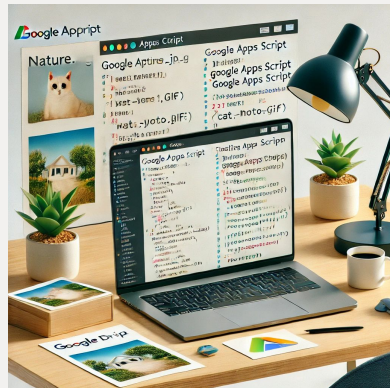


Objective: Automate the process of listing and describing image files.

Key Steps:

1. List all image files from a Google Drive folder (name, URL, MIME type).
2. Convert images to Base64 for AI processing.
3. **Use Gemini AI to generate descriptions for each image.**
4. Store image details and descriptions in a Google Sheet.

Use Case: Simplifies cataloging images with descriptions for reference or documentation.



[Generate Descriptions](#)
[Gemini AI GitHub](#)



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>



File Name	Description
image4.JPG	A cheerful fox and skunk dance in a sunny forest clearing. The fox, orange with a white-tipped tail, and the skunk, black with a white bushy tail and a pink nose, stand on their hind legs, facing each other, happily dancing. Light filters through the trees. Mushrooms and flowers dot the forest floor around them.
image3.JPG	A fluffy cat places its paws on a vintage rotary phone. A glowing fringed lamp sits behind on a table, next to stacked books and a comfortable chair. The scene is set in a cozy room with a window and curtains in the background.
image2.JPG	A dog in an astronaut suit sits at the helm of a spaceship, gazing out at a swirling galaxy through the front window. The canine captain wears a helmet and a jacket with an American flag patch. The control panel glows with complex instruments.
image1.JPG	A majestic male lion sits at a desk in a modern office, diligently typing on a keyboard. The incongruity of the wild animal in a corporate setting creates a humorous and surreal image.

Build with Gemini Today



Call **Gemini from Apps Script** using the REST API

Securely store your API key (out of your code)

Build a **simple logger demo** to test the connection

Create a **Sheets custom function**: =GEMINI_COMPLETE()

Construct a **Sheets sidebar** helper "Prompt Pad"

Make a **Docs summarizer** for selected text

Draft a basic **Gmail reply** helper



One-Time Setup (3 Steps)



1. Get Gemini API Key

Go to **Google AI Studio**. Create a new project (or choose one) and click 'Create API key'. Copy this key.



2. Create Apps Script

Open a Google Sheet or Doc. Go to **Extensions** → **Apps Script**. This will create a new, bound script project.



3. Store API Key

In Apps Script, go to **Project Settings** (⚙️). Under **Script Properties**, add a property named **GEMINI_API_KEY** and paste your key as the value.



Secure Your API Key



Why use Script Properties?

This is the recommended way to keep "secrets" like API keys **out of your source code**. This prevents you from accidentally sharing your key.

```
function getGeminiApiKey_() {  
  const scriptProps = PropertiesService.getScriptProperties();  
  const key = scriptProps.getProperty('GEMINI_API_KEY');  
  if (!key) {  
    throw new Error('GEMINI_API_KEY not set');  
  }  
  return key;  
}
```

How to read the key:

Use the PropertiesService to read the key you just stored. We'll add this to a helper function.



Core Helper: callGemini_



Part 1 - The Request

This reusable function is the core of our project. It builds the request and calls the Gemini REST API.

```
function callGemini_(prompt) {  
  const modelId = 'gemini-1.5-flash';  
  const url = `.../models/${modelId}:generateContent`;  
  const apiKey = getGeminiApiKey();  
  
  // 1. Build the payload  
  const payload = {  
    contents: [{  
      parts: [{ text: prompt }]  
    }]  
  };  
  
  // 2. Set Fetch Options  
  const options = {  
    method: 'post',  
    contentType: 'application/json',  
    headers: { 'x-goog-api-key': apiKey },  
    muteHttpExceptions: true,  
    payload: JSON.stringify(payload)  
  };  
  
  // ... (fetch and parse response)  
}
```





Part 2 - The Response

After setting up the request, we use `UrlFetchApp` to make the call and then parse the JSON response.

```
// ... (inside callGemini_ function)

// 3. Make the API call
const response = UrlFetchApp.fetch(url, options);
const code = response.getResponseCode();
const text = response.getContentText();

// 4. Check for errors
if (code !== 200) {
  console.error('Gemini error', code, text);
  throw new Error('API Error: ' + text);
}

// 5. Parse the JSON and extract text
const data = JSON.parse(text);
const candidates = data.candidates || [];
if (!candidates.length) { return ''; }

const parts = candidates[0].content?.parts || [];
const resultText = parts.map(p => p.text || '').join('');

return resultText;
}
```



Example 1: Test with a Logger



This is the simplest way to check if your API key and helper function are working correctly.

1. Add this function to Code.gs.
2. Select testGeminSimple from the dropdown.
3. Click Run.
4. Authorize the script when prompted.
5. Open View → Logs to see the response!

```
function testGeminSimple() {  
  const prompt = 'Write a two-line motivational quote ' +  
    'about learning to code.';  
  const reply = callGemin_(prompt);  
  
  // Check the logs!  
  Logger.log(reply);  
}
```



Sheets Custom Function



=GEMINI_COMPLETE("List 3 creative warm-up questions for an online class.")					
A	B	C	D	E	
	Here are 3 creative warm-up questions for an online class, designed to spark imagination, en				
	1. **If you could invent a brand new color, what would you call it, and what feeling or experie				
	* **Why it's creative:** It taps into abstract thinking, sensory experience, and emotional as				
	* **Online class benefit:** Easy to answer briefly in chat, or for a few quick verbal shares.				

=GEMINI_COMPLETE("Write a haiku")

We can expose our helper function directly in Sheets by creating a custom function.

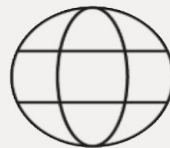
Wrap the API call in a try/catch block to show errors in the cell.

```
/**
 * @param {string} prompt
 * @return {string}
 * @customfunction
 */
function GEMINI_COMPLETE(prompt) {
  if (!prompt) {
    return 'Please provide a prompt.';
  }

  try {
    // Note: convert prompt to string
    const result = callGemini_(String(prompt));
    return result;
  } catch (err) {
    console.error(err);
    return 'Error: ' + err.message;
  }
}
```



Sheets Sidebar Architecture



1. HTML File

Create **Sidebar.html**. This file contains the HTML for the form (textarea, button) and client-side JavaScript to handle clicks. It uses `google.script.run` to call backend functions.



2. Backend Functions

In **Code.gs**, add functions to be called by the sidebar:

- `sidebarAskGemini(prompt)`
- `insertResultIntoActiveCell(text)`



3. Custom Menu

Use a standard `onOpen()` function to add a custom menu to the Sheet UI. This menu will have an item that runs `showGeminiSidebar()`.



Sheets Sidebar



Sheets sidebar “Prompt Pad”

Let’s build a small UI inside Sheets so beginners can type a prompt and paste the result into a cell.

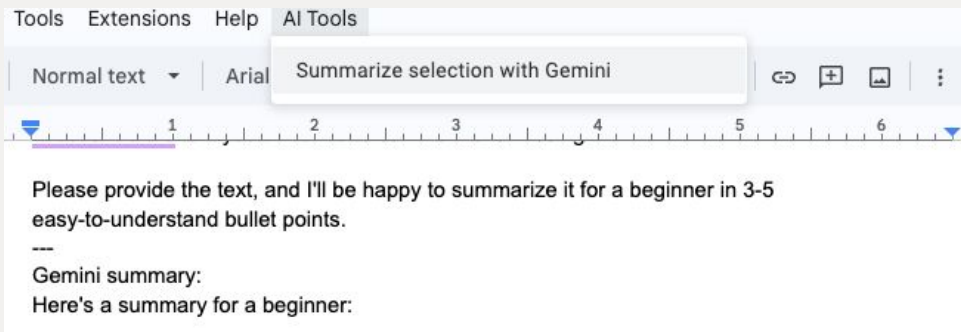
Create the sidebar HTML

In Apps Script:

- Click + → HTML.
- Name it Sidebar.html.

A screenshot of a web application titled "Gemini Prompt Pad" with a close button (X) in the top right corner. The main heading is "Gemini Prompt Pad". Below it is a label "Prompt:" followed by a text input field containing the text "Explain VLOOKUP in simple terms". The input field has a small red squiggly line under "VLOOKUP". Below the input field are two buttons: "Ask Gemini" and "Insert into Sheet". A vertical scrollbar is visible on the right side of the input field.

Docs Summarizer Logic



This script runs in a Google Doc and adds a summary of the selected text to the end of the document.

1. `onOpen()` creates an "AI Tools" menu.
2. `summarizeSelectionWithGemini()` is called.
3. Get the selection: `DocumentApp.getActiveDocument().getSelection()`.
4. Loop through elements to get the text.
5. Create a prompt: "Summarize: \n\n" + text.
6. Call `callGemini_()` with the prompt.
7. Append the summary: `doc.getBody().appendParagraph(summary)`.



Gmail Reply Drafter



This script (standalone or in Gmail) drafts a reply to the most recent email in your inbox.

1. Get the newest thread: `GmailApp.getInboxThreads(0, 1)`.
2. Get the last message's body, subject, and sender.
3. Build a prompt: "You are a polite assistant. Read the email below and write a reply...\n\n" + body.
4. Call `callGemini_()` with the full prompt.
5. Create a draft: Video: An AI-generated video that visually represents the concept of extracting totals from PDF files and emailing them as a summary.
6. `GmailApp.createDraft(...)`.

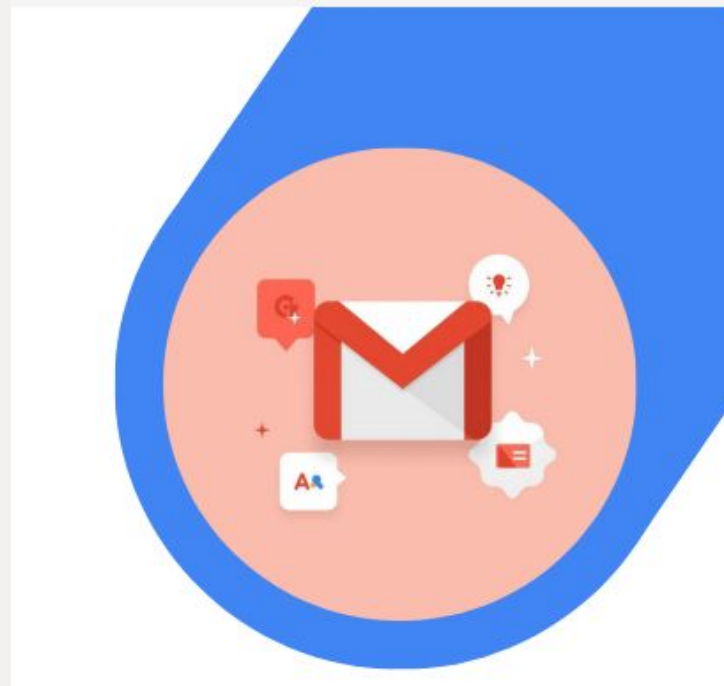


Gmail Smart Reply Assistant

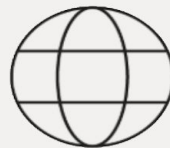


Gmail Smart Reply Assistant (Apps Script + Gemini)

- A Gmail add-on built with Apps Script + Gemini
- Reads the email thread you have open
- Summarizes the conversation
- Generates a reply in a chosen tone (Professional, Friendly, Concise)



How It Works



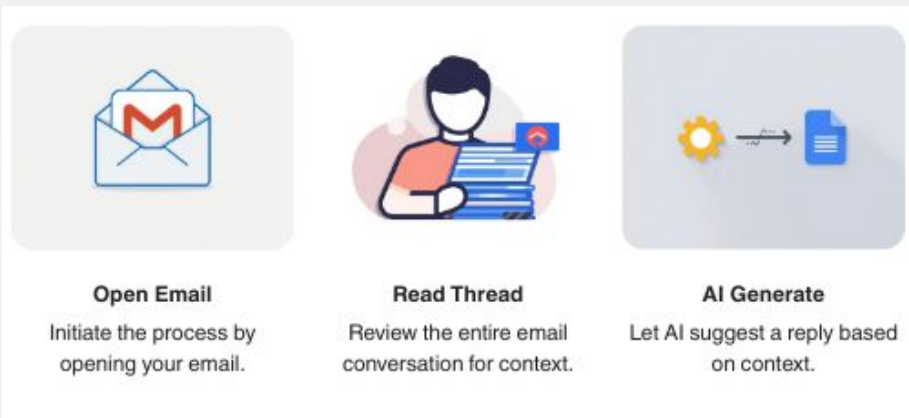
User opens an email

- Add-on loads UI inside Gmail
- User selects tone + optional instructions

AI generates the reply

- Script extracts thread text
- Sends context to Gemini
- Gemini returns:
 - Summary
 - Key points
 - Full suggested reply

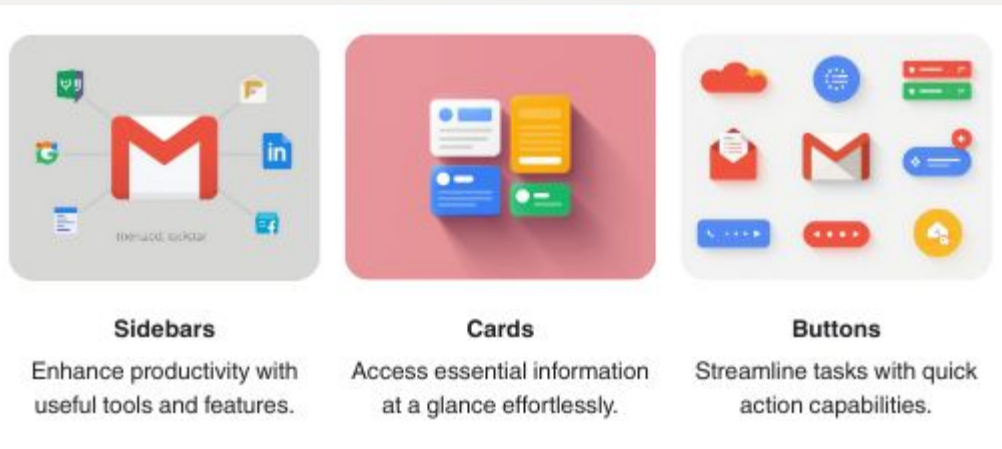
Results appear directly in the add-on card



What the Add-on Can Do



- Show AI-generated summary + reply
- Allows user to insert the suggested reply as a Gmail draft
- Ensures replies are consistent, fast, and tailored in tone
- Works entirely inside Gmail — no external tools needed



Turn List to Bullets

Give Gemini a messy list and ask it to format it as clear bullet points.

```
const prompt = `Turn this list into  
clear bullet points: ...`
```

Explain Code

Paste a block of code and ask Gemini to explain it step-by-step for a beginner.

```
const prompt = `Explain this code: \n` + code;
```

Generate Quiz

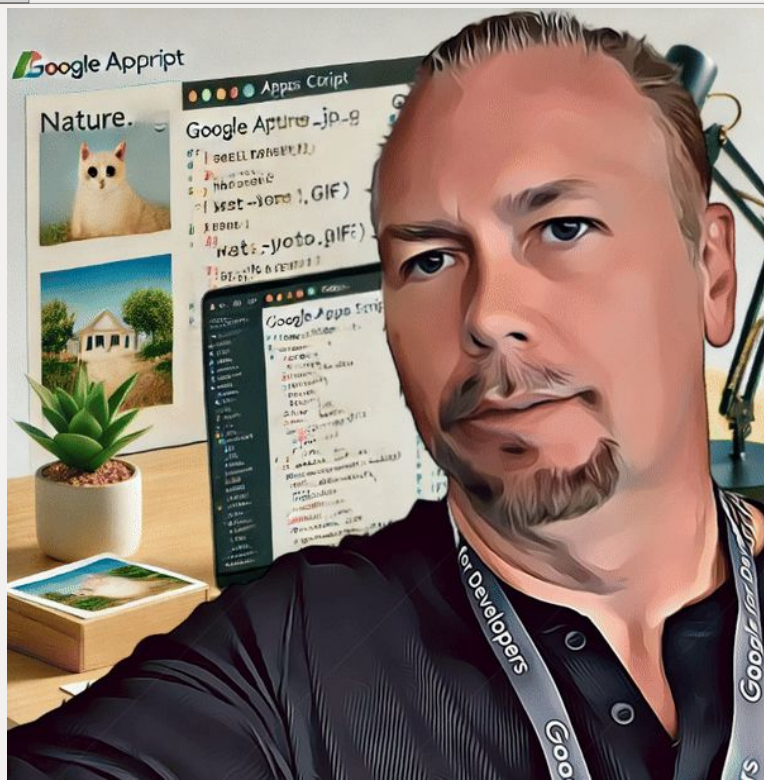
Ask Gemini to create multiple-choice questions about a specific topic.

```
const prompt = `Create 5 multiple-choice  
questions about...`;
```

How I got Started Apps Script



- 2015 was looking for a solution to capture tweets in a Google Sheet
- No Server!
- Found apps Script
- Added Twitter API
- Captured Tweet object and output into spreadsheet



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>

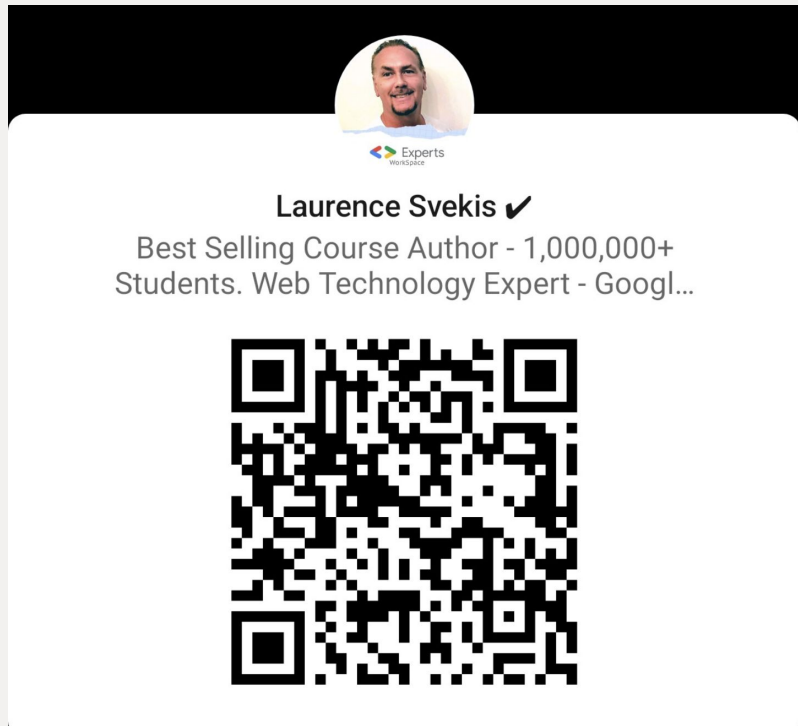
Connect with Laurence



BaseScript <https://basescripts.com/>

GitHub <https://github.com/lsvekis>

LinkedIn <https://www.linkedin.com/in/svekis/>



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>



Questions?

Get more Apps Script Content at:

<https://basescripts.com/>



Google Developer Experts

Laurence Svekis
<https://basescripts.com/>