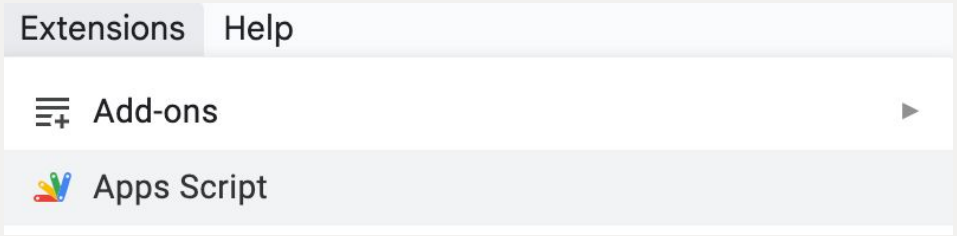
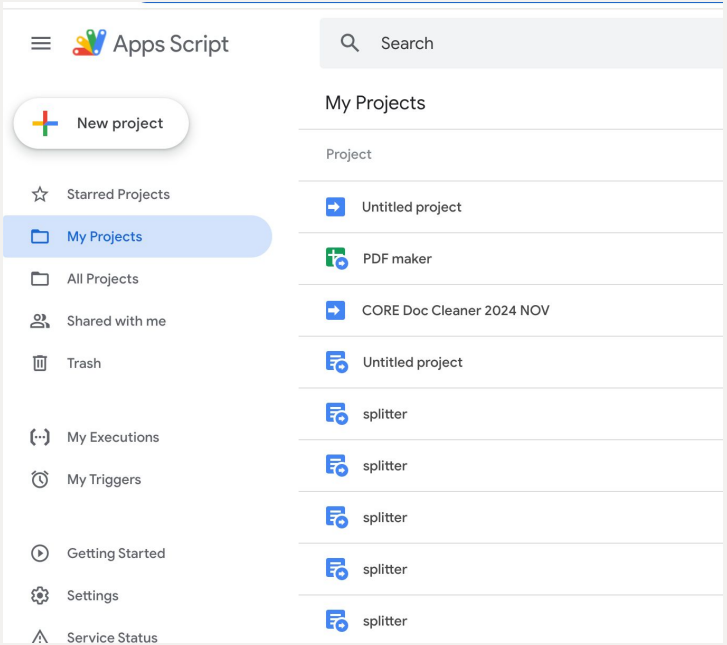
Start Scripting Today!

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

Open Script Editor

1. Bound Script
2. Stand Alone Script





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.

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.

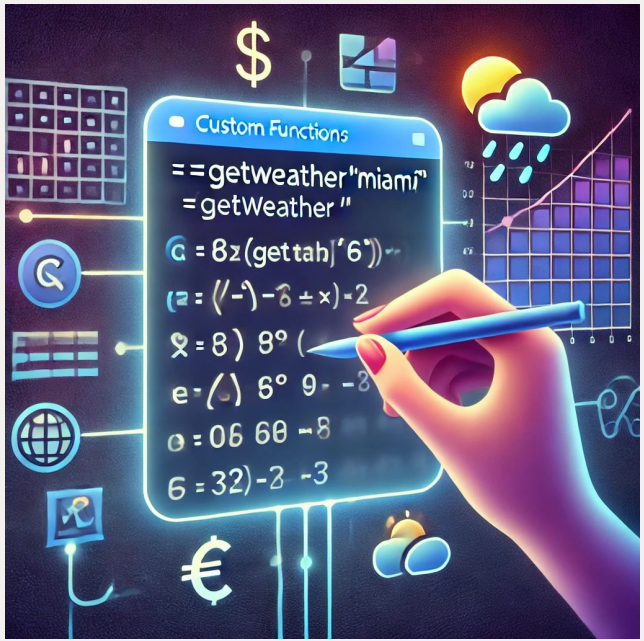
9. Custom Triggers ⏰
Run scripts on schedules.
Trigger on form submissions.

10. Enhance UI 🖥️
Add buttons, sidebars, menus.
Pop-ups for user input.

Laurence Svekis
https://basescripts.com/

Custom Functions in Sheet in action

```
function ADD_NUMBERS(a, b) { return a + b; }
function SUBTRACT_NUMBERS(a, b) { return a - b; }
function MULTIPLY_NUMBERS(a, b) { return a * b; }
function DIVIDE_NUMBERS(a, b) { if (b === 0) return
'Division by zero error'; return a / b; }
function MODULUS_NUMBERS(a, b) { if (b === 0) return
'Division by zero error'; return a % b; }
function POWER_NUMBERS(a, b) { return Math.pow(a, b); }
function MAX_NUMBERS(a, b) { return Math.max(a, b); }
function MIN_NUMBERS(a, b) { return Math.min(a, b); }
function ABSOLUTE_DIFFERENCE(a, b) { return Math.abs(a
- b); }
.....
```



Laurence Svekis
https://basescripts.com/

Sales Report to PDF

🎉 What Will This Script Do?
This Google Apps Script does the following:

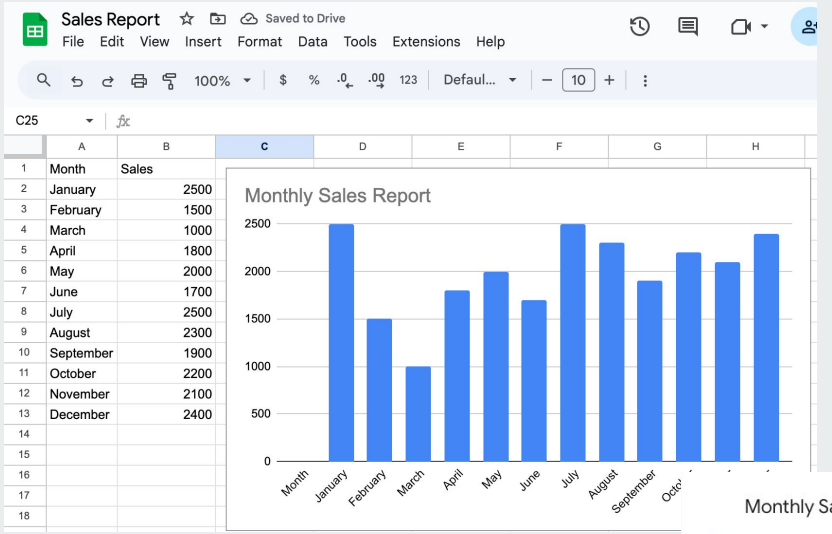
*Add Sample Sales Data to a Google Sheet.

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.

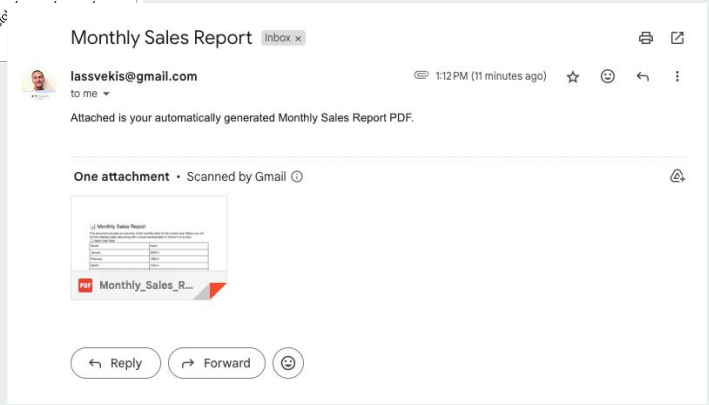
*Setup Trigger for the report

GitHub Code

Laurence Svekis
https://basescripts.com/



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).

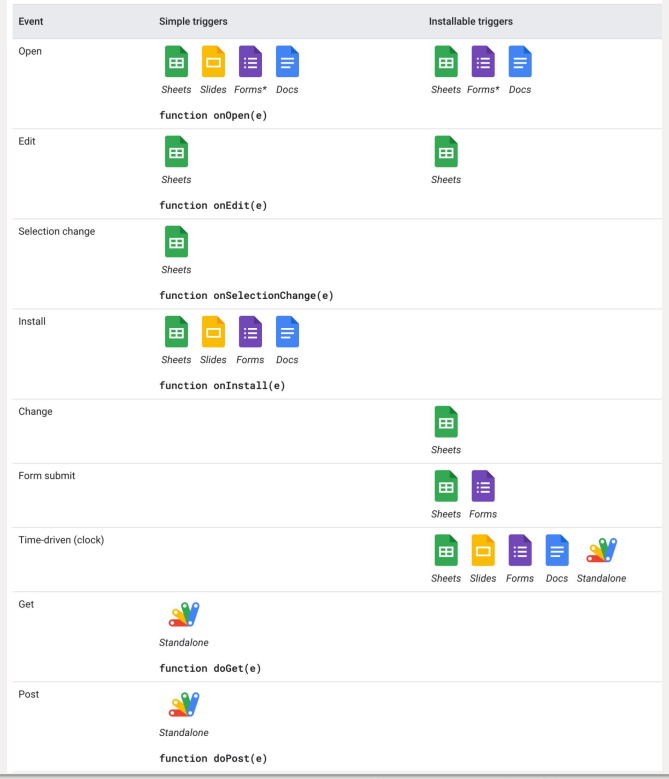
Laurence Svekis
https://basescripts.com/

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



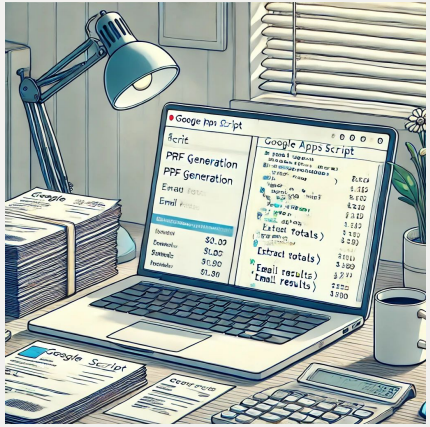
Laurence Svekis
https://basescripts.com/

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.

- Generate sample PDFs with mock invoice data.
- Extract key details, like totals, from PDFs using OCR.
- Email a summary of the extracted information directly to your inbox.



GitHubCode

Laurence Svekis
https://basescripts.com/

```
73    try {
74      const blob = file.getBlob().setContentType('application/pdf'); // Ensure it is a PDF blob
75
76      // Convert PDF to Google Doc using OCR
77      const text = extractTextFromPDF(blob);
78
79      const totalValue = extractTotalFromText(text); // Extract the total value from the text
80      Logger.log(`File: ${fileName} - Total Value: ${totalValue}`); // Log the total value for
        each file
81      emailBody += `File: ${fileName} - Total Value: ${totalValue}\n`; // Add total to email body
82    } catch (e) {
83      Logger.log(`Failed to extract text from file: ${fileName}. Error: ${e}`);
84      emailBody += `File: ${fileName} - Error extracting total\n`;
85    }
86  }
```

Execution log

4:07:58 PM	Notice	Execution started
4:07:58 PM	Info	Processing file: Sample Invoice 3.pdf
4:08:06 PM	Info	File: Sample Invoice 3.pdf - Total Value: 300.00
4:08:06 PM	Info	Processing file: Sample Invoice 2.pdf
4:08:10 PM	Info	File: Sample Invoice 2.pdf - Total Value: 150.00
4:08:10 PM	Info	Processing file: Sample Invoice 1.pdf
4:08:16 PM	Info	File: Sample Invoice 1.pdf - Total Value: 75.00

1 of 2,528

Extracted Totals from PDF Files

Inbox ×

@gmail.com 4:08 PM (2 minutes ago)

to me

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.

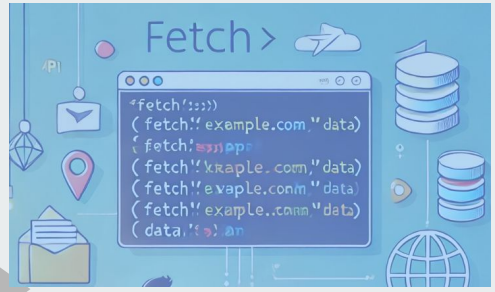
Key Steps:

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.

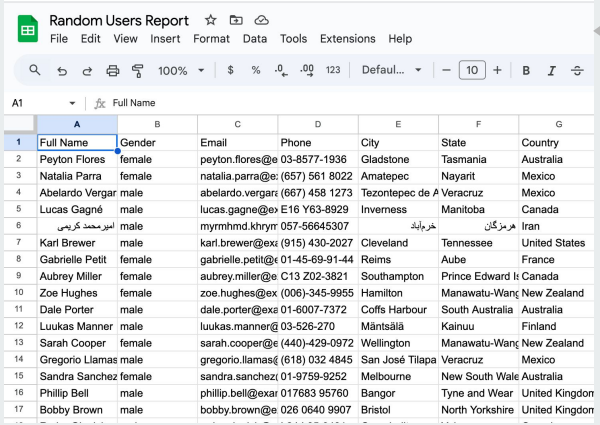
Use Case: Generate sample data for testing, analysis, or API integration practice.

GitHubCode

Laurence Svekis
https://basescripts.com/



GitHubCode



	Full Name	Gender	Email	Phone	City	State	Country
2	Peyton Flores	female	peyton.flores@e	03-8577-1936	Gladstone	Tasmania	Australia
3	Natalia Parra	female	natalia.parra@e	(657) 561 8022	Amatepec	Nayarit	Mexico
4	Abelardo Vergar	male	abelardo.vergar	(667) 458 1273	Tezontepec de A	Veracruz	Mexico
5	Lucas Gagné	male	lucas.gagne@ex	E16 Y63-8929	Inverness	Manitoba	Canada
6	امیرمحمد کریمی	male	myrmhmd.khrym	057-56645307	خرم‌آباد	هرمزگان	Iran
7	Karl Brewer	male	karl.brewer@exa	(915) 430-2027	Cleveland	Tennessee	United States
8	Gabrielle Petit	female	gabrielle.petit@e	01-45-69-91-44	Reims	Aube	France
9	Aubrey Miller	female	aubrey.miller@e	C13 Z02-3821	Southampton	Prince Edward Is	Canada
10	Zoe Hughes	female	zoe.hughes@ex	(006)-345-9955	Hamilton	Manawatu-Wang	New Zealand
11	Dale Porter	male	dale.porter@exa	01-6007-7372	Coffs Harbour	South Australia	Australia
12	Luukas Manner	male	luukas.manner@	03-526-270	Mäntsälä	Kainuu	Finland
13	Sarah Cooper	female	sarah.cooper@e	(440)-429-0972	Wellington	Manawatu-Wang	New Zealand
14	Gregorio Llamas	male	gregorio.llamas@	(618) 032 4845	San José Tilapa	Veracruz	Mexico
15	Sandra Sanchez	female	sandra.sanchez@	01-9759-9252	Melbourne	New South Wale	Australia
16	Phillip Bell	male	phillip.bell@exar	017683 95760	Bangor	Tyne and Wear	United Kingdom
17	Bobby Brown	male	bobby.brown@e	026 0640 9907	Bristol	North Yorkshire	United Kingdom

```javascript
function createNewSpreadsheetAndFetchUsers() {
  const spreadsheet = SpreadsheetApp.create('Random Users Report');
  const sheet = spreadsheet.getActiveSheet();
  const url = 'https://randomuser.me/api/?results=50';
  const response = UrlFetchApp.fetch(url);
  const jsonData = JSON.parse(response.getContentText());
  const users = jsonData.results;
  const headers = ['Full Name', 'Gender', 'Email', 'Phone', 'City', 'State', 'Country'];
  const userData = users.map(user => [
    `${user.name.first} ${user.name.last}`,  // Full Name
    user.gender,                             // Gender
    user.email,                              // Email
    user.phone,                              // Phone
    user.location.city,                      // City
    user.location.state,                     // State
    user.location.country                    // Country
  ]);
  sheet.getRange(1, 1, 1, headers.length).setValues([headers]); // Set headers
  sheet.getRange(2, 1, userData.length, userData[0].length).setValues(userData);
  const spreadsheetUrl = spreadsheet.getUrl();
}
```

Call Gemini

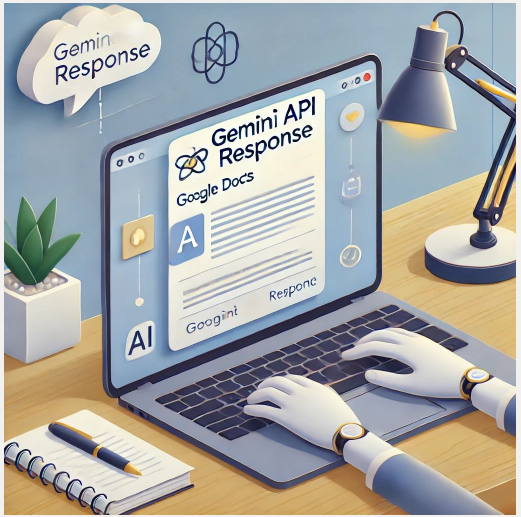
```
function callGemini_(prompt, modelId = 'gemini-2.5-flash') {
  const apiKey = PropertiesService.getScriptProperties().getProperty('GEMINI_API_KEY');
  if (!apiKey) {
    throw new Error('GEMINI_API_KEY is not set in Script Properties.');
  }
  const url = `https://generativelanguage.googleapis.com/v1beta/models/${modelId}:generateContent`;
  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 || '';
}
```

Laurence Svekis
https://basescripts.com/

Gemini Create a Doc

1. Get your API Key https://aistudio.google.com/app/apikey
2. Create new file utlis.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}`;
```



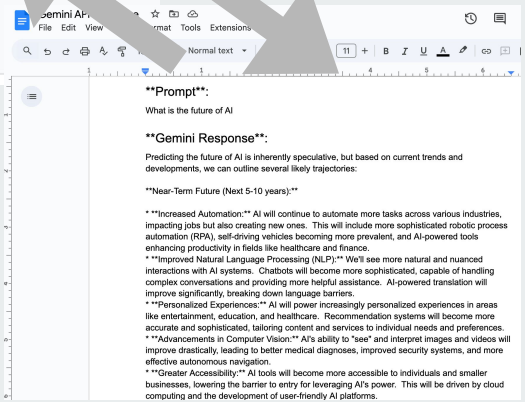
Laurence Svekis
https://basescripts.com/

```gs
function callGemini(q, temperature=0) {
  const payload = { contents: [{ parts: [{ text: q }] }] };
  const options = {
    'method' : 'post','contentType': 'application/json', 'payload': JSON.stringify(payload)
  };
  const response = UrlFetchApp.fetch(geminiModel, options);
  const data = JSON.parse(response.getContentText());
  const content = data["candidates"][0]["content"]["parts"][0]["text"];
  return content;
}

function createDocWithGeminiResponse() {
  const prompt = "What is the future of AI"; // The prompt for the Gemini API
  const geminiResponse = callGemini(prompt); // Get the response from Gemini API
  const doc = DocumentApp.create('Gemini API Response');
  const docId = doc.getId();
  const body = doc.getBody();
  body.appendParagraph('**Prompt**:').setHeading(DocumentApp.ParagraphHeading.HEADING2);
  body.appendParagraph(prompt); // Add the prompt
  body.appendParagraph('**Gemini Response**:').setHeading(DocumentApp.ParagraphHeadING2);
  body.appendParagraph(geminiResponse); // Add the response from the Gemini API
  const docUrl = doc.getUrl();
  Logger.log(`New document created: ${docUrl}`);
}
```

GitHubCode

Gemini



Gemini Doc Summaries

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

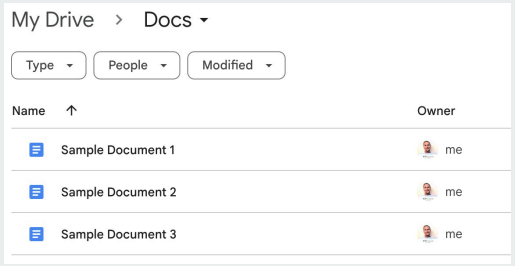
Key Steps:

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]

Laurence Svekis
https://basescripts.com/



Doc Summaries with Gemini GitHub

Gemini

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

@gmail.com 4:29 PM (11 minutes ago)

to me

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.

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

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.

	A	B	C	D
1	File Name	File URL	MIME Type	Description
2	image4.JPG	https://drive.google.cor	image/jpeg	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.
3	image3.JPG	https://drive.google.cor	image/jpeg	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.
4	image2.JPG	https://drive.google.cor	image/jpeg	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.
5	image1.JPG	https://drive.google.cor	image/jpeg	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.

Gemini

[Generate Descriptions Gemini AI GitHub]

What You'll Build 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

Laurence Svekis
https://basescripts.com/

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.

Google Developer Experts

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.

Laurence Svekis
https://basescripts.com/

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)
}
```


Laurence Svekis
https://basescripts.com/

Core Helper: callGemini_

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;
}
```


Laurence Svekis
https://basescripts.com/

Example 1: Test with a Logger

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

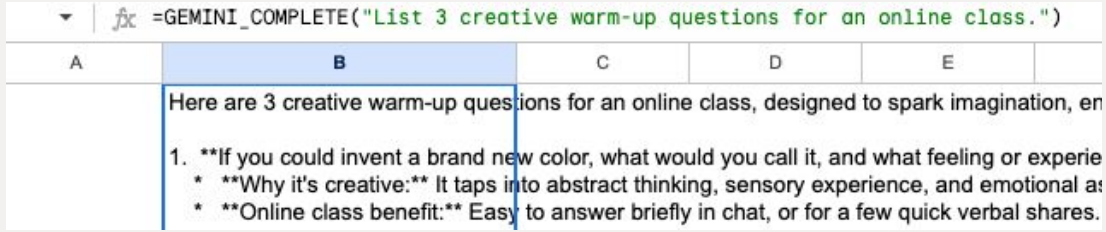
1. Add this function to Code.gs.
2. Select testGeminiSimple from the dropdown.
3. Click Run.
4. Authorize the script when prompted.
5. Open View → Logs to see the response!

```
function testGeminiSimple() {
  const prompt = 'Write a two-line motivational quote ' +
                 'about learning to code.';
  const reply = callGemini_(prompt);

  // Check the logs!
  Logger.log(reply);
}
```


Laurence Svekis
https://basescripts.com/

Sheets Custom Function



=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;
  }
}
```


Laurence Svekis
https://basescripts.com/

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().

Google Developer Experts

Laurence Svekis
https://basescripts.com/

Sheets Sidebar

Sheets sidebar “Prompt Pad”

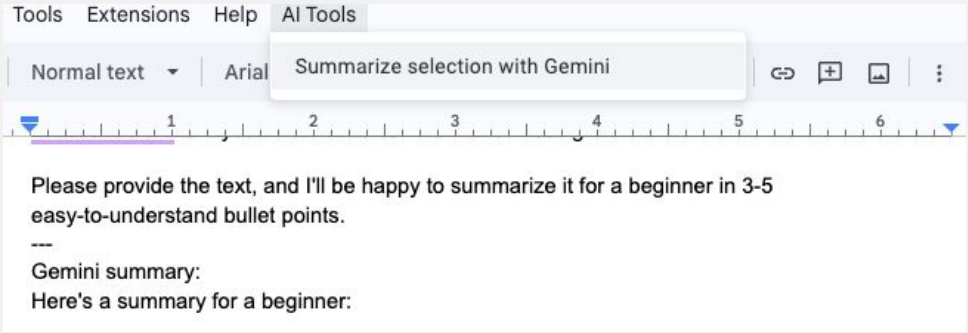
Let’s build a small UI inside Sheets so beginners can type a prompt and paste the result into a cell.

Create the sidebar HTML
In Apps Script:

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

Laurence Svekis
https://basescripts.com/

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).

Laurence Svekis
https://basescripts.com/

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(...).

Laurence Svekis
https://basescripts.com/

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

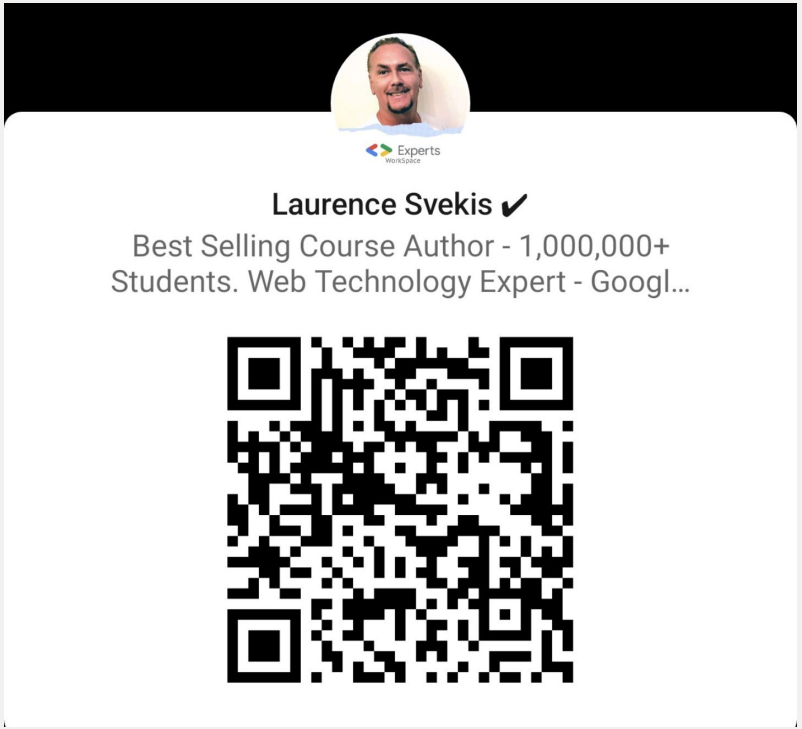
Laurence Svekis
https://basescripts.com/

Connect with Laurence

BaseScript https://basescripts.com/

GitHub https://github.com/lsvekis

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



Questions?

Get more Apps Script Content at:

https://basescripts.com/

Google Developer Experts