






 *Supercharge Your Google Workspace:*

Google Apps Script Projects!

Create a web app that fetches file details from a Google Drive folder and displays them as clickable URLs

Step 1: Set Up Google Apps Script	3
Step 2: Write the Apps Script Code	4
Code.gs	4
Index.html	4
Step 3: Deploy as Web App	6
How it Works	6

 Exciting News for #GoogleAppsScript Enthusiasts and #WebDevelopers!  

I've just explored an incredible capability of Google Apps Script - creating a web app that fetches file details from a Google Drive folder and displays them as clickable URLs. This simple yet powerful tool is perfect for anyone looking to streamline their workflow or share resources efficiently.

 Here's a quick breakdown:

- #WebApp Development: Using Google Apps Script to create an interactive web interface.

Learn more about JavaScript with Examples and Source Code Laurence Svekis Courses <https://basescripts.com/>


- #GoogleDrive Integration: Seamlessly connect to a Drive folder and retrieve file information.
- #JavaScript & #HTML: A perfect blend to create a user-friendly UI.
- #Automation: Effortlessly list all files in a folder with their direct URLs.

Why it's Useful:

- #ResourceSharing: Ideal for educators and teams who frequently share documents.
- #WorkflowOptimization: Simplify how you access and distribute files.
- #Collaboration: Enhances team collaboration and file management.

Deploying as a Web App:

- Simple deployment process within the Google Apps Script environment.
- Accessible by anyone with the link, based on the permissions set.

 This project highlights the versatility of Google Apps Script and its potential to automate and simplify tasks. Whether you're a seasoned developer or a newbie in the world of #Coding, this is a great way to enhance your skills and productivity.

- [How-to-Build-a-website.pdf](#)
- [exercise-Objects-JavaScript-.pdf](#)
- [Quiz-JavaScript-Objects.pdf](#)
- [Exercise-JavaScript-Arrays.pdf](#)
- [JavaScript-Arrays-quiz.pdf](#)
- [Exercise-JavaScript-Loops.pdf](#)
- [Quiz-JavaScript-Loops.pdf](#)
- [Control-Structures-Exercises.pdf](#)
- [Control-Structures.pdf](#)
- [Google-Apps-Script-Exercise-Youtube-TOC-Generator-1.pdf](#)
- [Exercises-Functions.pdf](#)
- [Functions.pdf](#)
- [Operators.pdf](#)
- [Exercise-Variables-and-Data-Types.pdf](#)
- [Variables-and-Data-Types.pdf](#)
- [JavaScript-Quiz.pdf](#)
- [Manipulating-an-array-in-JavaScript.pdf](#)
- [JavaScript-DOMContentLoaded-Event.pdf](#)
- [Enhancing-Web-Interactivity-with-AJAX-and-JSON.pdf](#)
- [JavaScript- Breakpoints.pdf](#)
- [JavaScript- Callbacks.pdf](#)
- [JavaScript-Parsing-JSON.pdf](#)
- [Searching-and-replacing-strings.pdf](#)
- [JavaScript-hoisting.pdf](#)
- [HTML5-Features.pdf](#)
- [JavaScript-Exercises-Practical-regex-.pdf](#)
- [JavaScript-Exercises-Key-event-handler.pdf](#)
- [JavaScript-Exercises-Promises.pdf](#)
- [JavaScript-Exercises-Read-file.pdf](#)
- [JavaScript-Exercises-HTML5-Canvas.pdf](#)

Creating a web app in Google Apps Script that retrieves files from a specific folder and lists their URLs involves a few steps. Below is a basic outline of how to create such a web app. Note that this is a simple example and might need to be adjusted based on specific requirements and security considerations.

Step 1: Set Up Google Apps Script

Open Google Drive: Go to your Google Drive and create a new Google Apps Script project. You can do this by going to New > More > Google Apps Script.

Write the Script: In the script editor, you'll write the code to fetch files from a specified folder and generate URLs.

Step 2: Write the Apps Script Code

Here's a basic script to get you started:

Code.gs

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

```
function getFilesInFolder(folderId) {  
  var folder = DriveApp.getFolderById(folderId);  
  var files = folder.getFiles();  
  var fileList = [];  
  
  while (files.hasNext()) {  
    var file = files.next();  
    fileList.push({name: file.getName(), url: file.getUrl()});  
  }  
  
  return fileList;  
}
```

Index.html

```
<!DOCTYPE html>  
<html>
```

Learn more about JavaScript with Examples and Source Code Laurence Svekis
Courses <https://basescripts.com/>

```
<head>
  <title>Folder Files List</title>
</head>
<body>
  <input type="text" id="folderId" placeholder="Enter Folder ID">
  <button onclick="listFiles()">Get Files</button>
  <ul id="fileList"></ul>

  <script>
    function listFiles() {
      var folderId = document.getElementById('folderId').value;
      google.script.run.withSuccessHandler(showFiles).getFilesInFolder(folderId);
    }

    function showFiles(files) {
      var list = document.getElementById('fileList');
      list.innerHTML = "";
      files.forEach(function(file) {
        var li = document.createElement('li');
        var link = document.createElement('a');
        link.textContent = file.name;
        link.href = file.url;
        link.target = "_blank";
        li.appendChild(link);
      });
    }
  </script>

```

Learn more about JavaScript with Examples and Source Code Laurence Svekis
Courses <https://basescripts.com/>

```
        list.appendChild(li);
    });
}
</script>
</body>
</html>
```

Step 3: Deploy as Web App

1. Deploy the App: Click on Deploy > New deployment.
2. Choose Type: Select Web app as the type.
3. Set Permissions: Choose who can access your app (usually "Anyone").
4. Deploy: Click Deploy.

How it Works

- HTML Page: Presents a UI where the user can input a folder ID and a button to trigger the file listing.
- Apps Script: The `getFilesInFolder` function fetches files from the specified folder and returns their names and URLs.
- Client-Side JavaScript: Communicates with the Apps Script function and updates the HTML page with the file list.
- Important Notes
- Folder ID: Users need to know the folder ID, which can be found in the URL when they open the folder in Google Drive.

Learn more about JavaScript with Examples and Source Code Laurence Svekis
Courses <https://basescripts.com/>

- Permissions: Ensure the Google Apps Script project has permissions to access Google Drive.
- Sharing Settings: The files in the folder should be appropriately shared, or users might not be able to access them via the URLs.

This basic example can be expanded and customized to suit more specific needs, such as handling different types of files, adding error handling, or customizing the user interface.