



EBook Gratis

APRENDIZAJE google-apps-script

Free unaffiliated eBook created from
Stack Overflow contributors.

#google-
apps-script

Tabla de contenido

Acerca de.....	1
Capítulo 1: Empezando con google-apps-script.....	2
Observaciones.....	2
Examples.....	2
Instalación o configuración.....	2
Tipos de scripts.....	3
Ejecutando / depurando tu script.....	4
Hola Mundo.....	4
Una mirada más profunda a Google Apps Script.....	4
Capítulo 2: Aplicaciones Script Web Apps.....	6
Observaciones.....	6
Examples.....	6
Formulario de aplicación web.....	6
Capítulo 3: Crear una función personalizada para las hojas de Google.....	12
Introducción.....	12
Examples.....	12
Gravedad estándar personalizada constante.....	12
Ejemplo básico.....	13
Capítulo 4: DriveApp.....	14
Examples.....	14
Crear una nueva carpeta en una raíz de Google Drive.....	14
Crear nuevo archivo en Google Drive de un determinado tipo de Mime.....	14
Crear un nuevo archivo de texto en la carpeta raíz de Google Drive.....	14
Crear un nuevo archivo en Google Drive desde un blob.....	14
Obtenga todas las carpetas: coloque las carpetas en un token de continuación y luego recup.....	15
Obtenga todos los archivos, póngalos en un token de continuación, luego recupérellos.....	15
Agregar una carpeta a la unidad raíz.....	16
Crea un nuevo archivo de texto y agrégalo a la carpeta raíz.....	16
Obtener todos los archivos en una carpeta de la unidad.....	17
Capítulo 5: DriveApp - getFileById (id).....	19

Observaciones.....	19
Examples.....	19
Obtenga un archivo de Google Drive utilizando el ID de archivo.....	19
Capítulo 6: Firebase y AppScript: Introducción.....	20
Introducción.....	20
Examples.....	20
Conexión a un proyecto de Firebase en GAS y transferencia de datos de Google Spreadsheet a.....	20
Instalar el recurso Firebase en el AppScript.....	20
Ahora vamos a tomar un ejemplo para leer y escribir datos de Firebase.....	22
¿Cómo encontrar el firebaseURL y la clave secreta?.....	23
Ahora ha insertado el firebaseURL y la clave secreta. Ahora estás listo para ir. Haga clic.....	24
Algunas funciones más para implementar lectura y escritura.....	24
1. Escribir un dato simple para probar si la conexión está funcionando o no.....	24
2. Leer todos los datos.....	24
3. Leer un registro específico.....	24
4. Actualizar un registro específico.....	25
Capítulo 7: GmailApp.....	26
Observaciones.....	26
Examples.....	26
Obtener archivo CSV adjunto a un correo.....	26
Capítulo 8: Google hojas MailApp.....	27
Introducción.....	27
Examples.....	27
Un ejemplo básico de aplicación de correo.....	27
Acceda a los datos de la hoja.....	27
Usa la hoja de datos para enviar un correo electrónico.....	28
Enviando contenido HTML en correo.....	30
Capítulo 9: Hoja de cálculo Añadir menú.....	33
Sintaxis.....	33
Parámetros.....	33
Observaciones.....	33

Examples.....	33
Crear un nuevo menú.....	33
Crear menú personalizado.....	34
Capítulo 10: Llamadas del cliente a Google apps-script.....	35
Introducción.....	35
Examples.....	35
Este es un ejemplo de una llamada del lado del cliente a un script de aplicación de Google.....	35
Capítulo 11: Script de la aplicación web de Google para descarga automática desde Google D.....	36
Introducción.....	36
Observaciones.....	36
Examples.....	36
forms.html.....	36
código.gs.....	37
Cómo funciona.....	38
Capítulo 12: Servicio de hoja de cálculo.....	41
Observaciones.....	41
Examples.....	41
Hoja.....	41
Copie un valor de una hoja a la hoja actual.....	42
Obtener la última fila en una sola columna.....	42
Inserción de matrices como filas.....	43
Capítulo 13: Servicio DriveApp.....	44
Observaciones.....	44
Examples.....	44
Crear una nueva carpeta en la unidad raíz de Google.....	44
Crear nuevo archivo en Google Drive de un determinado tipo de Mime.....	44
Crear un nuevo archivo de texto en la carpeta de la unidad raíz de Google.....	44
Crear un nuevo archivo en Google Drive desde un blob.....	45
Obtenga todas las carpetas: coloque las carpetas en un token de continuación y luego recup.....	45
Obtenga todos los archivos, póngalos en un token de continuación, luego recupérellos.....	46
Capítulo 14: Servicio DriveApp - Archivos por tipo y cadena de búsqueda.....	47
Parámetros.....	47

Examples.....	47
Obtenga archivos por tipo de archivo con cadena coincidente en nombre de archivo.....	47
Capítulo 15: SpreadsheetApp Active Sheet.....	49
Observaciones.....	49
Examples.....	49
getActive () - Obtener hoja de cálculo activa.....	49
Creditos	50

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [google-apps-script](#)

It is an unofficial and free google-apps-script ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official google-apps-script.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con google-apps-script

Observaciones

La descripción oficial de Google Apps Script se publica en <http://www.google.com/script/start> , desde allí

Google Apps Script es un lenguaje de scripts en la nube de JavaScript que proporciona formas sencillas de automatizar tareas en los productos de Google y servicios de terceros y crear aplicaciones web.

Desde https://developers.google.com/apps-script/guides/services/#basic_javascript_features

Apps Script se basa en [JavaScript 1.6](#) , además de algunas características de [1.7](#) y [1.8](#) . Por lo tanto, muchas funciones básicas de JavaScript están disponibles además de los servicios integrados y [avanzados de Google](#) : puede usar objetos comunes como [Array](#) , [Date](#) , [RegExp](#) , *etc.* , así como los [objetos](#) globales de [Math](#) y [Object](#) . Sin embargo, debido a que el código de Apps Script se ejecuta en los servidores de Google (no en el lado del cliente, excepto en [las páginas del servicio HTML](#)), las funciones basadas en el navegador como la manipulación de DOM o la API de [Windows](#) no están disponibles.

Examples

Instalación o configuración

Google Apps Script no requiere instalación o instalación. El único requisito es una cuenta de Google. Una cuenta de Gmail funciona tan bien como una cuenta de Google Apps for Work / Education / Government. Puede crear una nueva cuenta de Google yendo a <accounts.google.com>

Comience su primer script yendo a <script.google.com> . También puede acceder a Google Apps Script en las `tools -> Script editor...` de muchas aplicaciones de Google, es decir , *documentos, hojas, formularios, etc.* Google Apps Script también se puede agregar directamente a su Google Drive con la función `Connect more apps..`

La documentación oficial se puede encontrar en developers.google.com/apps-script/ .

Para que los scripts de aplicación se ejecuten, deben contener un archivo `code.gs`. El archivo `code.gs` debe contener una función llamada `doGet` (secuencias de comandos independientes) o una función `onOpen` (secuencias de comandos adicionales). Los inicios rápidos en la documentación contienen ejemplos.

Si una api está activada en el script de aplicación, también debe estar activada en la consola de desarrolladores. Sin embargo, la consola de desarrolladores contiene api que se pueden activar

pero no aparecen en la interfaz de la aplicación-script. Por ejemplo, Marketplace SDK debe estar activado en la consola de desarrolladores antes de que la aplicación pueda publicarse en Google Play Store o en una implementación de dominio de G Suite.

Para las aplicaciones de Google para educación / trabajo / gobierno, hay configuraciones en la consola de administración de dominio que se pueden ajustar para permitir o no permitir que se ejecuten los scripts de aplicación.

Tipos de scripts

Los scripts de Google App son de tres tipos.

- Ser único
- Limitado a las aplicaciones de Google
- Aplicaciones web

Escritura independiente

Los scripts independientes no están vinculados a ninguna aplicación de Google, es decir, *documentos, hojas o formularios, etc.* Los scripts independientes se pueden crear visitando script.google.com o conectando el script de Google App con Google Drive. El script independiente se puede usar para programar aplicaciones de Google de forma independiente, se puede usar como una aplicación web o se puede configurar para que se ejecute automáticamente desde un disparador instalable. Consulte la [documentación](#) para el script independiente.

Limitado a las aplicaciones de Google

Script enlazado a Google Apps también conocido como script enlazado a contenedor; a diferencia de los scripts independientes, están vinculados a las aplicaciones de Google, es decir, *Google Docs o Google Sheets, etc.* Los scripts encuadrados en contenedores pueden crearse seleccionando `tools> Script editor` desde Google App. Algunas [funciones](#), como los cuadros de diálogo, las indicaciones, los menús y la barra lateral, solo se proporcionan mediante secuencias de comandos de contenedor. Además, las secuencias de comandos de contenedor se utilizan para crear [complementos de Google](#). Consulte la [documentación](#) para los scripts enlazados al contenedor.

Aplicaciones web

Google App Script se puede utilizar como aplicación web, ya que se puede acceder a él mediante el navegador. La aplicación web puede proporcionar una interfaz de usuario en el navegador y puede hacer uso de las aplicaciones de Google, es decir, *documentos, hojas, etc.* Tanto las secuencias de comandos independientes como las de Google Apps pueden convertirse en aplicaciones web. Para que cualquier script funcione como una aplicación web, el script debe cumplir con dos requisitos:

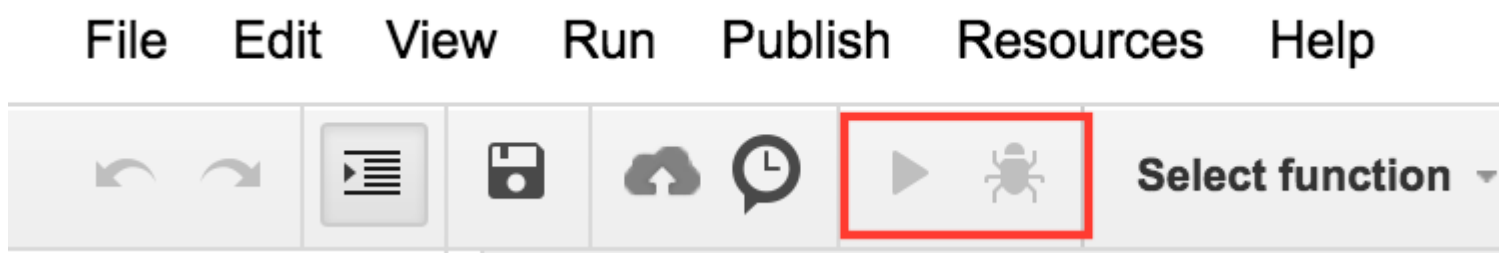
- incluye una función `doGet ()` o `doPost ()`.
- La función devuelve un objeto HTML Servicio `HtmlOutput` o un objeto `Content Service TextOutput`.

Las funciones `Inshort`, `doGet()` y `doPost()` funcionan como los controladores `http get` y `post request` respectivamente.

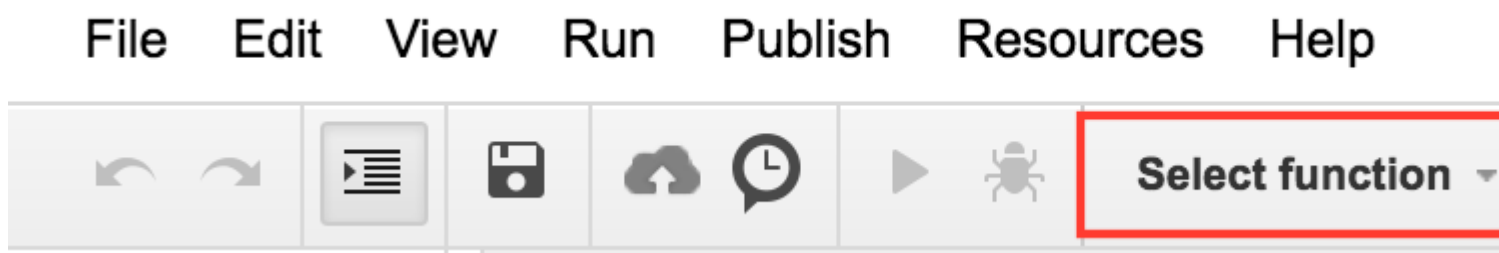
Para más detalles sobre aplicaciones web, consulte la [documentación](#) oficial.

Ejecutando / depurando tu script

Intente ejecutar su código desde la barra de herramientas como se muestra a continuación:



En su código, si tiene más de una función, antes de ejecutarla, debe mencionar la función con la que desea ejecutar. Por ejemplo :



Alternativamente, puede presionar **ctrl + r** desde su teclado para ejecutar el código. Guardará el código primero, si no se guarda, y luego lo ejecutará. Pero, para que esto funcione, debe haber seleccionado la función, como se ve en la imagen de arriba.

Además, si algunas actividades externas invocan su script, aún podrá ver los registros haciendo clic en ver-> registros si está registrando algo después de que se ejecuta el código.

Hola Mundo

Vamos a saludar como un cuadro de mensaje.

```
function helloWorld()
{
  Browser.msgBox("Hello World");
}
```

Para ejecutar el script, haga clic en ► o seleccione el elemento de menú **Ejecutar -> helloWorld**

Una mirada más profunda a Google Apps Script

Google Apps Script es una plataforma como servicio basado en JavaScript que se utiliza

principalmente para automatizar y ampliar Google Apps. Apps Script se ejecuta exclusivamente en la infraestructura de Google y no requiere aprovisionamiento ni configuración del servidor. Un IDE en línea sirve como interfaz para toda la plataforma que conecta todos los servicios que están disponibles para Apps Script. La autenticación del usuario se incluye en la plataforma a través de OAuth2 y no requiere ningún código o configuración por parte del autor del script.

Apps Script se ejecuta en el lado del servidor, pero puede tener interfaces de usuario creadas con HTML, CSS, JavaScript o cualquier otra tecnología compatible con el navegador. A diferencia de Node.js, que está controlado por eventos, los scripts de aplicación se ejecutan en un modelo de subprocesos. Todas las llamadas a un script generan una instancia única de ese script que se ejecuta de forma aislada de todas las demás instancias. Cuando una instancia de un script termina su ejecución se destruye.

Las funciones en Apps Script están bloqueando, por lo que no se necesitan patrones de devolución de llamada y de programación asíncrona. El bloqueo se utiliza para evitar que las secciones críticas de código, como el archivo IO, se ejecuten simultáneamente en diferentes instancias.

En la práctica, escribir scripts de aplicaciones es simple. A continuación se muestra un sencillo script que crea una nueva hoja de cálculo a partir de una plantilla de hoja de cálculo.

```
// Create a new spreadsheet from a template
function createSpreadsheet() {
  var templateFileId = '1Azcz9GwCeHjG19TXf4aUh6g20Eqmgd1UMSdNVjzIZPk';
  var sheetName = 'Account Log for:' + new Date();
  SpreadsheetApp.openById(templateFileId).copy(sheetName);
}
```

Lea Empezando con google-apps-script en línea: <https://riptutorial.com/es/google-apps-script/topic/1154/empezando-con-google-apps-script>

Capítulo 2: Aplicaciones Script Web Apps

Observaciones

Esta es una aplicación web de formulario de ejemplo, el bit del lado del cliente muestra algunos diseños UX básicos, como un botón de envío deshabilitado cuando se envía el formulario, o un mensaje de error si falla ... etc.

El bit de Script de aplicaciones es muy básico. Contiene solo el código necesario para entregar el html y validar el campo.

Aquí hay un enlace a esta aplicación de ejemplo en acción: [Ejemplo de formulario de script de aplicaciones](#)

Nota: Debes iniciar sesión en una cuenta de Google.

La estructura del archivo de Apps Script es así:

- Code.gs
- index.html
- Hoja de estilo.html
- JavaScript.html

Examples

Formulario de aplicación web

Script de aplicaciones:

```
//Triggered when the page is navigated to, serves up HTML
function doGet(){
  var template = HtmlService.createTemplateFromFile('index');
  return template.evaluate()
    .setTitle('Example App')
    .setSandboxMode(HtmlService.SandboxMode.IFRAME);
}

//Called from the client with form data, basic validation for blank values
function formSubmit(formData){
  for(var field in formData){
    if(formData[field] == ''){
      return {success: false, message: field + ' Cannot be blank'}
    }
  }
  return {success: true, message: 'Sucessfully submitted!'};
}
```

HTML

```

<!DOCTYPE html>
<html>

  <head>
    <base target="_top">
    <link href="https://ssl.gstatic.com/docs/script/css/add-ons1.css" rel="stylesheet">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.4/jquery.min.js"
type="text/javascript"></script>
  </head>

  <body>
    <div id="mainForm">
      <h1>Example Form</h1>
      <form>
        <div>
          <div class="inline form-group">
            <label for="name">Name</label>
            <input id="nameInput" style="width: 150px;" type="text">
          </div>
        </div>
        <div>
          <div class="inline form-group">
            <label for="city">City</label>
            <input id="cityInput" style="width: 150px;" type="text">
          </div>
          <div class="inline form-group">
            <label for="state">State</label>
            <input id="stateInput" style="width: 40px;" type="text">
          </div>
          <div class="inline form-group">
            <label for="zip-code">Zip code</label>
            <input id="zip-codeInput" style="width: 65px;" type="number">
          </div>
        </div>
        <div class="block form-group">
          <label for="typeSelect">Type</label>
          <select id="typeSelect">
            <option value="">
              </option>
            <option value="Type 1 ">
              Type 1
            </option>
            <option value="Type 2 ">
              Type 2
            </option>
            <option value="Type 3 ">
              Type 3
            </option>
            <option value="Type 4 ">
              Type 4
            </option>
          </select>
        </div>
        <button class="action" id="submitButton" type="button">Submit</button>
        <button class="clear" id="clearFormButton" type="button">Clear Form</button>
      </form>
      <div class="hidden error message">
        <div class="title">Error:</div>
        <div class="message"></div>
      </div>
      <div class="hidden success message">

```

```

        <div class="title">Message:</div>
        <div class="message">Sucessfully submitted</div>
    </div>
</div>
<?!= HtmlService.createHtmlOutputFromFile('JavaScript').getContent(); ?>
<?!= HtmlService.createHtmlOutputFromFile('Stylesheet').getContent(); ?>
</body>

</html>

```

CSS

```

<style>
.hidden {
    display: none;
}

.form-group {
    margin: 2px 0px;
}

#submitButton {
    margin: 4px 0px;
}

body {
    margin-left: 50px;
}

.message {
    padding: 2px;
    width: 50%;
}

.message > * {
    display: inline-block;
}

.message .title {
    font-weight: 700;
    font-size: 1.1em;
}

.success.message {
    border: 1px solid #5c9a18;
    background: #e4ffe4;
    color: #2a8e2a;
}

.error.message {
    background: #f9cece;
    border: 1px solid #7d2929;
}

.error.message .title {
    color: #863030;
}

button.clear {
    background: -moz-linear-gradient(top, #dd6e39, #d17636);

```

```

background: -ms-linear-gradient(top, #dd6e39, #d17636);
background: -o-linear-gradient(top, #dd6e39, #d17636);
background: -webkit-linear-gradient(top, #dd6e39, #d17636);
background: linear-gradient(top, #dd6e39, #d17636);
border: 1px solid transparent;
color: #fff;
text-shadow: 0 1px rgba(0, 0, 0, .1);
}

button.clear:hover {
background: -moz-linear-gradient(top, #ca602e, #bd6527);
background: -ms-linear-gradient(top, #ca602e, #bd6527);
background: -o-linear-gradient(top, #ca602e, #bd6527);
background: -webkit-linear-gradient(top, #ca602e, #bd6527);
background: linear-gradient(top, #ca602e, #bd6527);
border: 1px solid transparent;
color: #fff;
text-shadow: 0 1px rgba(0, 0, 0, .1);
}
</style>

```

JavaScript

```

<script>
var inputs = [
  'nameInput',
  'cityInput',
  'stateInput',
  'zip-codeInput',
  'typeSelect'
];

$(function(){
  var pageApp = new formApp();
  $('#submitButton').on('click', pageApp.submitForm);
  $('#clearFormButton').on('click', pageApp.clearForm);
});

var formApp = function(){
  var self = this;

  //Clears form input fields, removes message, enables submit
  self.clearForm = function(){
    for(var i = 0; i < inputs.length; i++){
      $('#'+inputs[i]).val('');
    }
    toggleSubmitButton(false);
    setErrorMessage(false);
    setSuccessMessage(false);
  }

  //Submits the form to apps script
  self.submitForm = function(){
    toggleSubmitButton(true);
    setSuccessMessage(false);
    setErrorMessage(false);

    google.script.run
      .withSuccessHandler(self.sucessfullySubmitted)
      .withFailureHandler(self.failedToSubmit)

```

```

        .formSubmit(self.getFormData());
    };

    //Retrieves the form data absed on the input fields
    self.getFormData = function(){
        var output = {};
        for(var i = 0; i < inputs.length; i++){
            output[inputs[i]] = $('#'+inputs[i]).val();
        }
        console.log(output)
        return output;
    }

    //When the apps script sucessfully returns
    self.successfullySubmitted = function(value){
        if(value.success){
            setSuccessMessage(true, value.message);
        } else {
            setErrorMessage(true, value.message);
            toggleSubmitButton(false);
        }
    }

    //When the apps script threw an error
    self.failedToSubmit = function(value){
        toggleSubmitButton(false);
        setErrorMessage(true, value.message);
    }
}

//Disables/enables the submit button
function toggleSubmitButton(disabled){
    $('#submitButton').prop('disabled', disabled);
}

//Sets the general message box's message and enables or disabled the error box
function setSuccessMessage(show, message){
    if(show){
        $('.success.message').removeClass('hidden');
        $('.success.message .message').text(message);
    } else {
        $('.success.message').addClass('hidden');
        $('.success.message .message').text('');
    }
}

//Sets the error message box's message and enables or disabled the error box
function setErrorMessage(show, message){
    if(show){
        $('.error.message').removeClass('hidden');
        $('.error.message .message').text(message);
    } else {
        $('.error.message').addClass('hidden');
        $('.error.message .message').text('');
    }
}

function getFormData(){
    var output = {};
    for(var i = 0; i < inputs.length; i++){
        output[inputs[i]] = $('#'+inputs[i]).val();
    }
}

```

```
}  
  return output;  
}  
</script>
```

Lea Aplicaciones Script Web Apps en línea: <https://riptutorial.com/es/google-apps-script/topic/4874/aplicaciones-script-web-apps>

Capítulo 3: Crear una función personalizada para las hojas de Google

Introducción

Una función personalizada en google docs está vinculada a un documento específico (y, por lo tanto, solo se puede utilizar en ese documento).

Por lo tanto, debe crearse con la edición de secuencias de comandos de ese documento (Herramientas -> Editor de secuencias de comandos). Una vez guardado, se puede usar como cualquier otra fórmula de hoja de cálculo regular.

Examples

Gravedad estándar personalizada constante

Esta función devuelve la constante de gravedad estándar en las unidades de aceleración especificadas (1 para cm / s², 2 para ft / s², 3 para m / s²)

```
/**
 * Returns the standard gravity constant in the specified acceleration units
 * Values taken from https://en.wikipedia.org/wiki/Standard_gravity on July 24, 2016.
 *
 * @param {number} input 1 for cm/s2, 2 for ft/s2, 3 for m/s2
 *
 * @customfunction
 */
function sg(units_key) {
  var value;
  switch(units_key) {
    case 1:
      value = 980.665;
      break;
    case 2:
      value = 32.1740;
      break;
    case 3:
      value = 9.80665;
      break;
    default:
      throw new Error('Must to specify 1, 2 or 3');
  }
  return value;
}
```

Para usar la función, debe estar vinculada a una hoja de cálculo utilizando el editor de secuencias de comandos (Herramientas -> Editor de secuencias de comandos ...). Una vez que se agrega la función, se puede usar como cualquier otra función de las hojas de Google llamando a la función en la fórmula de una celda.

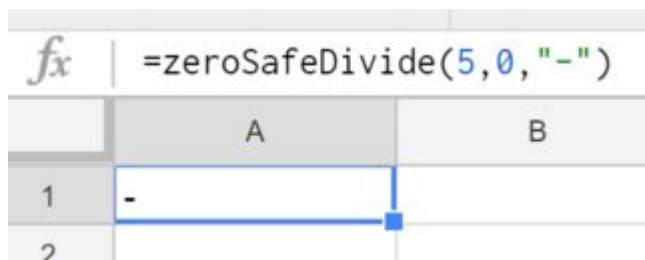
Observe cómo se muestra la función en autocompletar cuando se escribe en una fórmula. Esto se debe al comentario de varias líneas sobre la declaración de la función que se utiliza para describir qué hace la función similar a JSDoc y Javadoc. Para que la fórmula aparezca en autocompletar, la etiqueta `@customfunction` debe especificarse en el comentario.

Ejemplo básico

Para evitar errores antiestéticos `#DIV/0` en una hoja de cálculo, se puede usar una función personalizada.

```
/**
 * Divides n by d unless d is zero, in which case, it returns
 * the given symbol.
 *
 * @param {n} number The numerator
 * @param {d} number The divisor
 * @param {symbol} string The symbol to display if `d == 0`
 * @return {number or string} The result of division or the given symbol
 *
 * @customfunction
 */
function zeroSafeDivide(n, d, symbol) {
  if (d == 0)
    return symbol;
  else
    return n / d;
}
```

Para usar la función, debe estar vinculada a una hoja de cálculo utilizando el editor de secuencias de comandos (**Herramientas -> Editor de secuencias de comandos ...**). Una vez que se agrega la función, se puede usar como cualquier otra función de las hojas de Google llamando a la función en la fórmula de una celda.



Observe cómo se muestra la función en autocompletar cuando se escribe en una fórmula. Esto se debe al comentario de varias líneas sobre la declaración de la función que se utiliza para describir qué hace la función similar a JSDoc y Javadoc. Para que la fórmula aparezca en autocompletar, la etiqueta `@customfunction` debe especificarse en el comentario.

Lea [Crear una función personalizada para las hojas de Google en línea:](https://riptutorial.com/es/google-apps-script/topic/5572/crear-una-funcion-personalizada-para-las-hojas-de-google)

<https://riptutorial.com/es/google-apps-script/topic/5572/crear-una-funcion-personalizada-para-las-hojas-de-google>

Capítulo 4: DriveApp

Examples

Crear una nueva carpeta en una raíz de Google Drive

```
function createNewFolderInGoogleDrive(folderName) {  
  return DriveApp.createFolder(folderName);  
}
```

Use la función `createNewFolderInGoogleDrive` para crear una carpeta llamada `Test folder` en una raíz de Google Drive:

```
var newFolder = createNewFolderInGoogleDrive('Test folder');
```

`newFolder` tiene tipo de [carpeta de clase](#):

```
// output id of new folder to log  
Logger.log(newFolder.getId());
```

Crear nuevo archivo en Google Drive de un determinado tipo de Mime

```
function createGoogleDriveFileOfMimeType() {  
  var content, fileName, newFile; //Declare variable names  
  
  fileName = "Test File " + new Date().toString().slice(0,15); //Create a new file name with  
  date on end  
  content = "This is the file Content";  
  
  newFile = DriveApp.createFile(fileName, content, MimeType.JAVASCRIPT); //Create a new file in  
  the root folder  
};
```

Crear un nuevo archivo de texto en la carpeta raíz de Google Drive

```
function createGoogleDriveTextFile() {  
  var content, fileName, newFile; //Declare variable names  
  
  fileName = "Test Doc " + new Date().toString().slice(0,15); //Create a new file name with  
  date on end  
  content = "This is the file Content";  
  
  newFile = DriveApp.createFile(fileName, content); //Create a new text file in the root folder  
};
```

Crear un nuevo archivo en Google Drive desde un blob

```
function createGoogleDriveFileWithBlob() {
```

```

var blob, character, data, fileName, i, L, max, min, newFile, randomNmbr; //Declare variable names

fileName = "Test Blob " + new Date().toString().slice(0,15); //Create a new file name with
date on end

L = 500; //Define how many times to loop
data = "";
max = 126;
min = 55;

for (i=0; i<L; i+=1) { //Loop to create data
    randomNmbr = Math.floor(Math.random() * (max-min+1) + min); //Create a random number
    //Logger.log('randomNmbr: ' + randomNmbr);
    character = String.fromCharCode(randomNmbr);

    //Logger.log('character: ' + character); //Print the character to the Logs
    data = data + character;
};

blob = Utilities.newBlob(data, MimeType.PLAIN_TEXT, fileName); //Create a blob with random
characters

newFile = DriveApp.createFile(blob); //Create a new file from a blob

newFile.setName(fileName); //Set the file name of the new file
};

```

Obtenga todas las carpetas: coloque las carpetas en un token de continuación y luego recupérelas

```

function processGoogleDriveFolders() {
    var arrayAllFolderNames, continuationToken, folders, foldersFromToken, thisFolder; //Declare
variable names

    arrayAllFolderNames = []; //Create an empty array and assign it to this variable name

    folders = DriveApp.getFolders(); //Get all folders from Google Drive in this account
    continuationToken = folders.getContinuationToken(); //Get the continuation token

    Utilities.sleep(18000); //Pause the code for 3 seconds

    foldersFromToken = DriveApp.continueFolderIterator(continuationToken); //Get the original
folders stored in the token
    folders = null; //Delete the folders that were stored in the original variable, to prove that
the continuation token is working

    while (foldersFromToken.hasNext()) { //If there is a next folder, then continue looping
        thisFolder = foldersFromToken.next(); //Get the next folder
        arrayAllFolderNames.push(thisFolder.getName()); //Get the name of the next folder
    };

    Logger.log(arrayAllFolderNames); //print the folder names to the Logs
};

```

Obtenga todos los archivos, póngalos en un token de continuación, luego recupérelas

```

function processGoogleDriveFiles() {
  var arrayAllFileNames, continuationToken, files, filesFromToken, fileIterator, thisFile; //Declare
variable names

  arrayAllFileNames = []; //Create an empty array and assign it to this variable name

  files = DriveApp.getFiles(); //Get all files from Google Drive in this account
  continuationToken = files.getContinuationToken(); //Get the continuation token

  Utilities.sleep(18000); //Pause the code for 3 seconds

  filesFromToken = DriveApp.continueFileIterator(continuationToken); //Get the original files
stored in the token
  files = null; //Delete the files that were stored in the original variable, to prove that the
continuation token is working

  while (filesFromToken.hasNext()) { //If there is a next file, then continue looping
    thisFile = filesFromToken.next(); //Get the next file
    arrayAllFileNames.push(thisFile.getName()); //Get the name of the next file
  };

  Logger.log(arrayAllFileNames);
};

```

Agregar una carpeta a la unidad raíz

```

function DriveAppAddFolder(child) { //Adds file to the root drive in Google Drive
  var body, returnedFolder; //Declare variable names

  if (!child) {
    body = "There is no folder";
    MailApp.sendEmail(Session.getEffectiveUser().getEmail(), "", "Error Adding Folder!", body)
    return;
  };

  returnedFolder = DriveApp.addFolder(child); //Add a folder to the root drive

  Logger.log('returnedFolder: ' + returnedFolder); //Print the folder results to the Logs
};

function createNewFolderInGoogleDrive() {
  var folder, newFolderName, timeStamp, dateTimeAsString;

  timeStamp = new Date(); //Create a new date
  dateTimeAsString = timeStamp.toString().slice(0,15);

  newFolderName = 'Test Folder Name ' + dateTimeAsString; //Create new folder name with
date/time appended to name

  folder = DriveApp.createFolder(newFolderName); //Create a new folder
  DriveAppAddFolder(folder); //Call a function and pass a folder to the function
};

```

Crea un nuevo archivo de texto y agrégalo a la carpeta raíz.

```

function DriveAppAddFile(child) { //Adds file to the root drive in Google Drive

```

```

var body,returnedFolder;//Declare variable names

if (!child) {
  body = "There is no file";
  MailApp.sendEmail(Session.getEffectiveUser().getEmail(), "", "Error Adding File!", body)
  return;
};

returnedFolder = DriveApp.addFile(child);

Logger.log('returnedFolder: ' + returnedFolder);
};

function createNewFileInGoogleDrive() {
  var content,file,newFileName,timeStamp,dateTimeAsString;

  timeStamp = new Date();//Create a new date
  dateTimeAsString = timeStamp.toString().slice(0,15);

  content = "This is test file content, created at: " + dateTimeAsString;//Create content for
new file
  newFileName = 'Test File ' + dateTimeAsString;//Create new file name with date/time appended
to name

  file = DriveApp.createFile(newFileName, content);//Create a new file
  DriveAppAddFile(file);//Call a function and pass a file to the function
};

```

Obtener todos los archivos en una carpeta de la unidad

```

function onOpen() {

  // Add a custom menu to run the script
  var ss = SpreadsheetApp.getActiveSpreadsheet();
  var searchMenuEntries = [ {name: "Run", functionName: "search"}];
  ss.addMenu("Get Files", searchMenuEntries);
}

function getFiles() {

  // Get the active spreadsheet and the active sheet
  var ss = SpreadsheetApp.getActiveSpreadsheet();
  var ssid = ss.getId();

  // Look in the same folder the sheet exists in. For example, if this template is in
  // My Drive, it will return all of the files in My Drive.
  var ssparents = DriveApp.getFileById(ssid).getParents();
  var sheet = ss.getActiveSheet();

  // Set up the spreadsheet to display the results
  var headers = [["Last Updated", "File Owner", "File Name", "File URL"]];
  sheet.getRange("A1:D").clear();
  sheet.getRange("A1:D1").setValues(headers);

  // Loop through all the files and add the values to the spreadsheet.
  var folder = ssparents.next();
  var files = folder.getFiles();
  var i=1;

```

```
while(files.hasNext()) {
    var file = files.next();
    if(ss.getId() == file.getId()){
        continue;
    }
    sheet.getRange(i+1, 1, 1,
4) .setValues([[file.getLastUpdated(),file.getOwner().getName(),file.getName(),
file.getUrl()]]);
    i++;
}
}
```

Lea DriveApp en línea: <https://riptutorial.com/es/google-apps-script/topic/5363/driveapp>

Capítulo 5: DriveApp - getFileById (id)

Observaciones

También es posible obtener un archivo por la URL del archivo. El ID de un archivo está en la url, por lo que usar el ID en lugar de la URL completa significa que el parámetro es más corto. Almacenar la URL en lugar de la ID ocupa más espacio.

Examples

Obtenga un archivo de Google Drive utilizando el ID de archivo

```
function getGoogleDriveFileById(id) {
  var file;

  file = DriveApp.getFileById(id); //Returns a file - The "id" must be a string

  //One way to manually get a file ID
  // - Open the file from Google Drive
  // - The file ID is in the URL in the browsers address bar
  //https://docs.google.com/spreadsheets/d/File_ID_is_here/edit#gid=0
};
```

Lea DriveApp - getFileById (id) en línea: <https://riptutorial.com/es/google-apps-script/topic/6087/driveapp---getfilebyid--id->

Capítulo 6: Firebase y AppScript: Introducción

Introducción

Integre Firebase con Google AppScript para leer y escribir datos en la base de datos de Firebase.

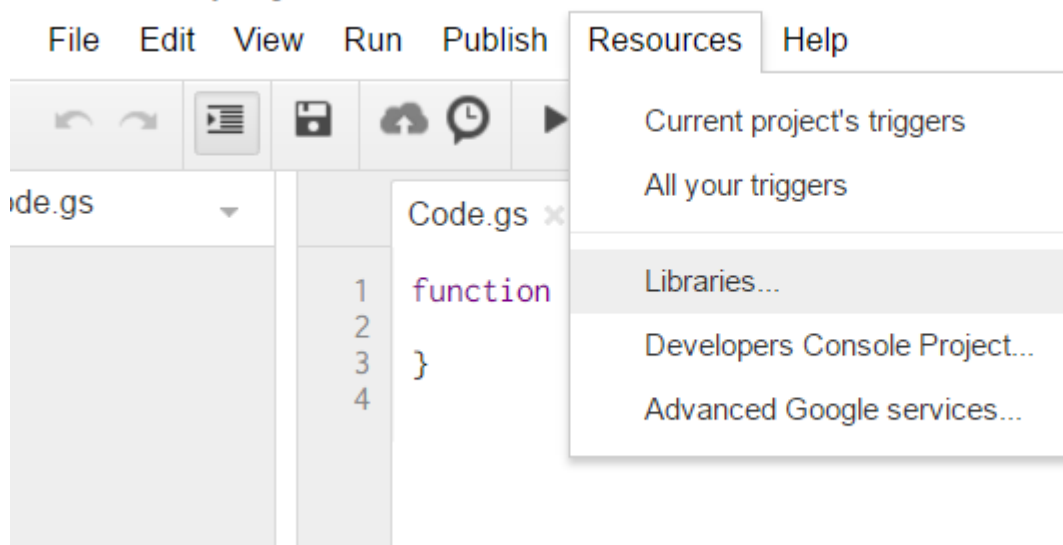
Firebase es un sistema de base de datos NoSQL de Google que utiliza una base de datos en tiempo real para ayudar a crear y alojar aplicaciones en dispositivos móviles, de escritorio y tabletas. Las bases de datos NoSQL utilizan los objetos JSON para almacenar los datos en formato estructurado.

Examples

Conexión a un proyecto de Firebase en GAS y transferencia de datos de Google Spreadsheet a Firebase

Instalar el recurso Firebase en el AppScript

- Para hacer eso, haga clic en Recursos y luego en Bibliotecas.
- Firebase tiene una clave de biblioteca de proyecto única que debe instalarse en AppScript.



- Haga clic en Bibliotecas. Aparecerá la siguiente ventana emergente. Ingrese la siguiente clave de proyecto en el cuadro de texto. **MYeP8ZEEt1yIVDxS7uyg9pIDOcoke7-2I** Esta es la clave de la biblioteca del proyecto para Firebase.

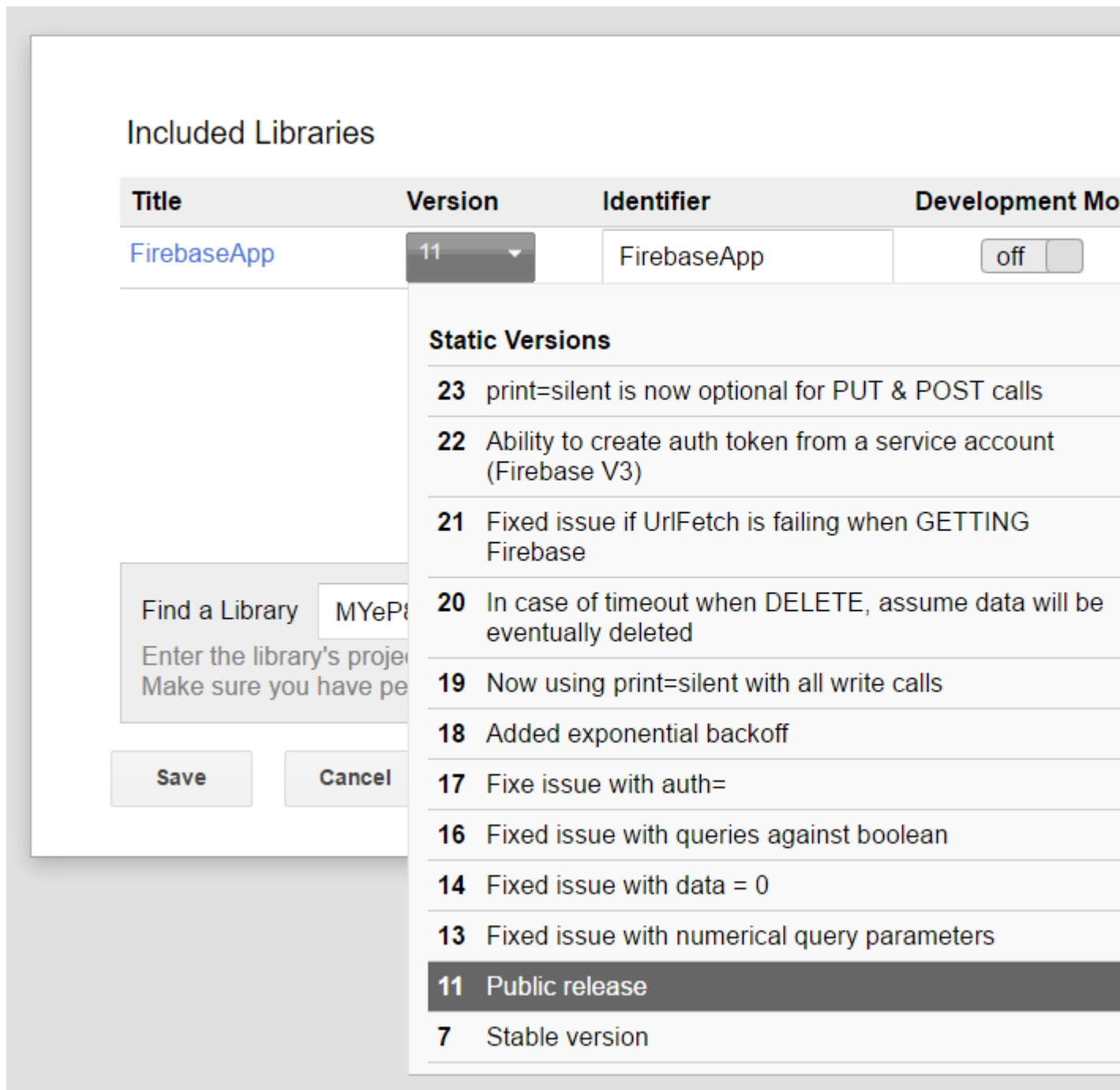
Included Libraries

Title	Version	Identifier	Development Mode
FirebaseApp	...	FirebaseApp	off

Find a Library

Enter the library's project key (found under File > Project Properties).
Make sure you have permissions to access the library or its containing spreadsheet.

- Ahora en la versión elegir la versión de lanzamiento público estable.



- Haga clic en Guardar. Ahora Firebase se instaló correctamente en su AppScript para que pueda trabajar.

Ahora vamos a tomar un ejemplo para leer y escribir datos de Firebase.

- Ahora tomamos una tabla de muestra diseñada en Google Sheets.

A	B	C	D	E	
First Name	Last Name	Email Address	Phone Number	Semester	Departm
Vishal	vishwakarma	vishal.vishwakar	9594852468		7 INFT
Yash	Udasi		75395185246		7 INFT

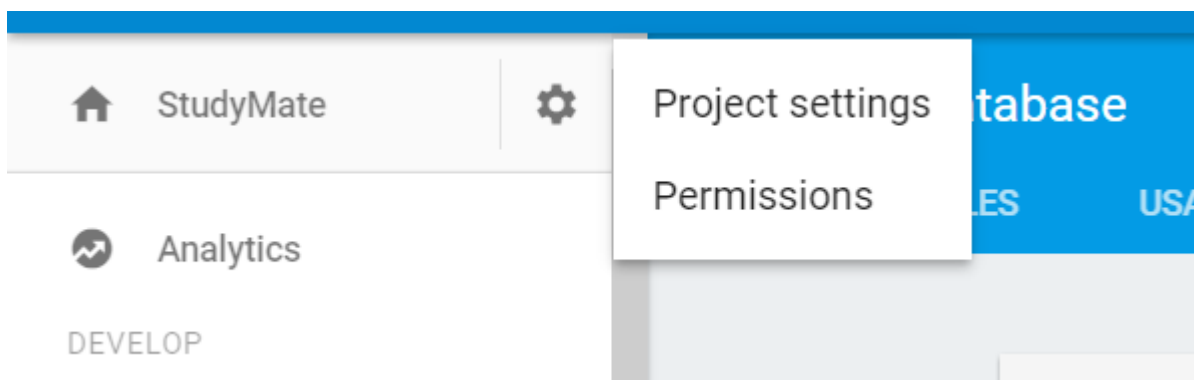
- Ahora para construir la base de datos en Firebase usando esta tabla en las hojas. Agregue el siguiente código en el AppScript.

```
function writeDataToFirebase() {
  var ss = SpreadsheetApp.openById("1LACsj0s3syAa9gvORdRWBhJ_YcXHybjQfHPgw3TLQ6g");
  var sheet = ss.getSheets()[0];
  var data = sheet.getDataRange().getValues();
  var dataToImport = {};
  for(var i = 1; i < data.length; i++) {
    var firstName = data[i][0];
    var lastName = data[i][1];
    dataToImport[firstName + '-' + lastName] = {
      firstName:firstName,
      lastName:lastName,
      emailAddress:data[i][2],
      semester:data[i][4],
      department:data[i][5],
    };
  }
  var firebaseUrl = "https://example-app.firebaseio.com/";
  var secret = "secret-key";
  var base = FirebaseApp.getDatabaseByUrl(firebaseUrl, secret);
  base.setData("", dataToImport);
}
```

Reemplace la ID de la hoja de cálculo y el firebaseURL y la clave secreta.

¿Cómo encontrar el firebaseURL y la clave secreta?

- Ve a tu Firebase Dashboard y haz clic en el engranaje de configuración en la esquina superior izquierda. Haga clic en Configuración del proyecto.



- Vaya a la sección Cuentas de servicio donde puede encontrar el databaseURL. Esto sirve como el firebaseURL.
- Ahora haga clic en la pestaña Secretos de la base de datos y podrá encontrar la clave secreta.

Ahora ha insertado el firebaseURL y la clave secreta. Ahora estás listo para ir. Haga clic en el código de ejecución en el motor de AppScript.

- Le pedirá que revise los permisos por primera vez cuando ejecute.
- Haga clic en Revisar permisos y Permitir.
- Ahora ejecuta su función y puede ver la tabla creada en Firebase Database.

Para ver la base de datos, vaya al panel de Firebase y haga clic en la base de datos para ver la base de datos.

Algunas funciones más para implementar lectura y escritura.

1. Escribir un dato simple para probar si la conexión está funcionando o no.

```
function myFunction(){
  var firebaseUrl = "https://example-app.firebaseio.com/";
  var secret = "secret-key";
  var base = FirebaseApp.getDatabaseByUrl(firebaseUrl, secret);
  base.setData("test", "Hello Firebase");
}
```

2. Leer todos los datos.

```
function getAllData() {
  var firebaseUrl = "https://example-app.firebaseio.com/";
  var secret = "secret-key";
  var base = FirebaseApp.getDatabaseByUrl(firebaseUrl, secret);
  var data = base.getData();
  for(var i in data) {
    Logger.log(data[i].firstName + ' ' + data[i].lastName);
  }
}
```

Los datos leídos se muestran en los registros. Para verificar los registros, haga clic en Ver → Registros o simplemente use Control + Intro.

3. Leer un registro específico.

```
function getContact() {
```

```
var firebaseUrl = "https://example-app.firebaseio.com/";
var secret = "secret-key";
var base = FirebaseApp.getDatabaseByUrl(firebaseUrl, secret);
var contact = base.getData("Yash-Udasi");
Logger.log(contact);
}
```

Los datos leídos se muestran en los registros. Para verificar los registros, haga clic en haga clic en Ver → Registros o simplemente use Control + Intro.

4. Actualizar un registro específico.

```
function updateData() {
  var firebaseUrl = "https://example-app.firebaseio.com/";
  var secret = "secret-key";
  var base = FirebaseApp.getDatabaseByUrl(firebaseUrl, secret);
  base.updateData("Yash-Udasi/emailAddress", "yash.udasi@fyuff.com");
}
```

Lea **Firestore y AppScript: Introducción en línea**: <https://riptutorial.com/es/google-apps-script/topic/9417/firebase-y-appscript--introduccion>

Capítulo 7: GmailApp

Observaciones

Consulte también la [referencia de API](#) oficial de GmailApp para obtener más detalles sobre los métodos disponibles.

Examples

Obtener archivo CSV adjunto a un correo

Supongamos que tenemos un sistema que envía informes diarios por correo electrónico en forma de archivos CSV adjuntos y que queremos acceder a ellos.

```
function getCsvFromGmail() {
  // Get the newest Gmail thread based on sender and subject
  var gmailThread = GmailApp.search("from:noreply@example.com subject:\"My daily report\"", 0,
  1)[0];

  // Get the attachments of the latest mail in the thread.
  var attachments = gmailThread.getMessages()[gmailThread.getMessageCount() -
  1].getAttachments();

  // Get and parse the CSV from the first attachment
  var csv = Utilities.parseCsv(attachments[0].getDataAsString());
  return csv;
}
```

Lea GmailApp en línea: <https://riptutorial.com/es/google-apps-script/topic/5899/gmailapp>

Capítulo 8: Google hojas MailApp

Introducción

Este servicio permite a los usuarios enviar correos electrónicos con control completo sobre el contenido del correo electrónico. A diferencia de GmailApp, el único propósito de MailApp es enviar un correo electrónico. MailApp no puede acceder a la bandeja de entrada de Gmail de un usuario.

Es más probable que los cambios en los scripts escritos con GmailApp desencadenen una solicitud de nueva autorización de un usuario que los scripts de MailApp.

Examples

Un ejemplo básico de aplicación de correo

MailApp es la api de Google App Script que puede usarse para enviar correo

```
function sendEmails() {  
  
    var subject = "A subject for your new app!";  
    var message = "And this is the very first message";  
    var recipientEmail = "abc@example.com";  
  
    MailApp.sendEmail(recipientEmail, subject, message);  
}
```

La clase MailApp está limitada a [cuotas](#) basadas en su cuenta de Google:

- Usuario consumidor (es decir, cuenta personal de Gmail): 100 destinatarios / día
- Cliente de Google Apps (legado): 100 destinatarios / día
- GSuite (básico / Gov / Edu / Business): 1500 destinatarios / día

Puedes consultar tu cuota de correo electrónico dentro de `MailApp`

```
function checkQuota() {  
    Logger.log(MailApp.getRemainingDailyQuota());  
}
```

Acceda a los datos de la hoja

```
function getSheetData() {  
  
    var sheet = SpreadsheetApp.getActiveSheet();  
  
    var startRow = 2; // First row of data to process  
    var numRows = 100; // Number of rows to process  
    var startCol = 1; // First column of data to process
```



```

var numCols = 15;    // Number of columns to process

var dataRange = sheet.getRange(startRow, startCol, numRows, numCols);

// Fetch values for each row in the Range.
var data = dataRange.getValues();

return data;
}

```

También puede modificar la función anterior de la siguiente manera para obtener un rango dinámico de datos del contenido presente en la hoja:

```

function getDataSheet() {

    sheet = SpreadsheetApp.getActiveSheet();

    //Get data range based on content
    var dataRange = sheet.getDataRange();

    // Fetch values for each row in the Range.
    var data = dataRange.getValues();

    return data;
}

```

Usa la hoja de datos para enviar un correo electrónico

Dado: Una hoja de empleados que han solicitado un reembolso.

Requisito: debemos enviar un correo electrónico al empleado cuando se procese su reembolso

Por lo tanto, la hoja es la siguiente:

A	B	C
Name	Email Address	Reimberseme amount
Ramesh	ramesh@sample.com	200
Vidhita	vidhita@sample.com	50
Jhanvi	jhanvi@sample.com	30

La función para enviar un correo electrónico es la siguiente:

```

function getDataSheet() {

    sheet = SpreadsheetApp.getActiveSheet();

```

```

startRow = 2; // First row of data to process
numRows = 100; // Number of rows to process
startCol = 1; //First column of data to process
numCols = 15; // Number of columns to process

var dataRange = sheet.getRange(startRow, startCol, numRows, numCols);

// Fetch values for each row in the Range.
var data = dataRange.getValues();

return data;
}

function getMessage(name, amount) {
    return "Hello " + name + ", Your reimbursement for amount " + amount + " is processed
successfully";
}

function sendEmail() {

    var emailSent = "Yes";
    var reimbursed = "Yes";
    var emailCol = 5;

    var data = getDataSheet();

    for (var i = 0; i < data.length; i++) {

        var row = data[i];

        var isReimbursed = row[3];
        var isEmailSent = row[4];
        var name = row[0];
        var amount = row[2];

        if(isReimbursed == reimbursed && isEmailSent != emailSent) {

            var subject = "Reimbursement details";
            var message = getMessage(name, amount);

            var recipientEmail = row[1];

            MailApp.sendEmail(recipientEmail, subject, message);

            //Sheet range starts from index 1 and data range starts from index 0
            sheet.getRange(1 + i, emailCol).setValue(emailSent);
        }
    }
}
}

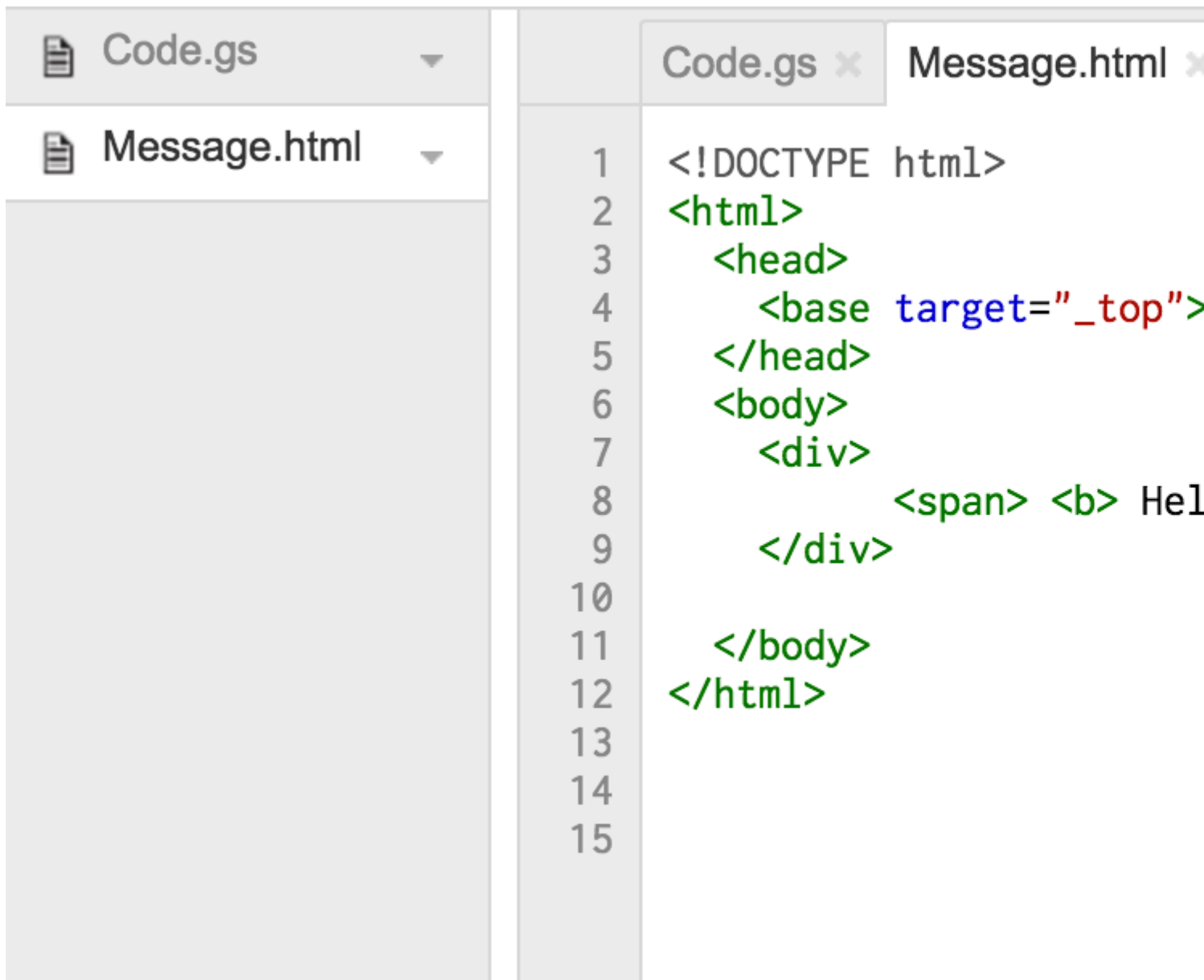
```

A	B	C
Name	Email Address	Reimberseme amount
Ramesh	ramesh@sample.com	20
Vidhita	vidhita@sample.com	5
Jhanvi	jhanvi@sample.com	3

Enviando contenido HTML en correo

En el ejemplo anterior, si queremos enviar contenido HTML como mensaje en el correo electrónico, cree un archivo HTML yendo a **Archivo -> Nuevo -> archivo HTML**

Ahora puede ver un archivo HTML además de su archivo gs de la siguiente manera:



The screenshot shows a code editor with two tabs: 'Code.gs' and 'Message.html'. The 'Message.html' tab is active and displays the following HTML code:

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <base target="_top">
5   </head>
6   <body>
7     <div>
8       <span> <b> He1
9     </div>
10
11   </body>
12 </html>
13
14
15
```

Ahora, actualice el método `getMessage ()` del ejemplo anterior de la siguiente manera:

```
function getMessage(name, amount) {
  var htmlOutput = HtmlService.createHtmlOutputFromFile('Message'); // Message is the name of
the HTML file

  var message = htmlOutput.getContent()
  message = message.replace("%name", name);
  message = message.replace("%amount", amount);

  return message;
}
```

La llamada a la api de *MailApp* también debe cambiarse

```
MailApp.sendEmail(recipientEmail, subject, message, {htmlBody : message});
```

Entonces todo el código será el siguiente:

```

function getDataSheet() {

    sheet = SpreadsheetApp.getActiveSheet();

    startRow = 2; // First row of data to process
    numRows = 100; // Number of rows to process
    startCol = 1; //First column of data to process
    numCols = 15; // Number of columns to process

    var dataRange = sheet.getRange(startRow, startCol, numRows, numCols);

    // Fetch values for each row in the Range.
    var data = dataRange.getValues();

    return data;
}

function getMessage(name, amount) {
    var htmlOutput = HtmlService.createHtmlOutputFromFile('Message');

    var message = htmlOutput.getContent()
    message = message.replace("%name", name);
    message = message.replace("%amount", amount);

    return message;
}

function sendEmail() {

    var emailSent = "Yes";
    var reimbursed = "Yes";
    var emailCol = 5;

    var data = getDataSheet();

    for (var i = 0; i < data.length; i++) {

        var row = data[i];

        var isReimbursed = row[3];
        var isEmailSent = row[4];
        var name = row[0];
        var amount = row[2];

        if(isReimbursed == reimbursed && isEmailSent != emailSent) {

            var subject = "Reimbursement details";
            var message = getMessage(name, amount);

            var recipientEmail = row[1];

            MailApp.sendEmail(recipientEmail, subject, message, {htmlBody : message});

            sheet.getRange(startRow + i, emailCol).setValue(emailSent);
        }
    }
}

```

Lea Google hojas MailApp en línea: <https://riptutorial.com/es/google-apps-script/topic/5298/google-hojas-mailapp>

Capítulo 9: Hoja de cálculo Añadir menú

Sintaxis

1. addMenu (nombre, subMenus)

Parámetros

Nombre	Descripción
nombre	El nombre del menú a crear.
subMenus	una variedad de mapas de JavaScript

Observaciones

Por lo general, deseará llamar a addMenu desde la función onOpen para que el menú se cree automáticamente cuando se carga la hoja de cálculo.

```
// The onOpen function is executed automatically every time a Spreadsheet is loaded
function onOpen() {
  var activeSheet = SpreadsheetApp.getActiveSpreadsheet();
  var menuItems = [];
  // When the user clicks on "addMenuExample" then "Menu 1", the function Myfunction1 is
  executed.
  menuItems.push({name: "Menu 1", functionName: "Myfunction1"});
  menuItems.push(null); // adding line separator
  menuItems.push({name: "Menu 2", functionName: "Myfunction2"});

  activeSheet.addMenu("addMenuExample", menuEntries);
}
```

Examples

Crear un nuevo menú

Creas un nuevo menú en la interfaz de usuario de la hoja de cálculo. Cada entrada del menú ejecuta una función definida por el usuario.

```
var activeSheet = SpreadsheetApp.getActiveSpreadsheet();
var menuItems = [];
// When the user clicks on "addMenuExample" then "Menu 1", the function Myfunction1 is
executed.
menuItems.push({name: "Menu 1", functionName: "Myfunction1"});
menuItems.push(null); // adding line separator
menuItems.push({name: "Menu 2", functionName: "Myfunction2"});
```

```
activeSheet.addMenu("addMenuExample", menuEntries);
```

Crear menú personalizado

```
/*
```

Método: para crear un menú personalizado Esta es la primera función que se debe llamar cuando se carga la aplicación

```
*/
```

```
function onOpen() {  
  var ui = SpreadsheetApp.getUi();  
  // Or DocumentApp or FormApp.  
  ui.createMenu('My HR')  
    .addItem('Send Form to All', 'sendIDPForm_All')  
    .addItem('Trigger IDP System', 'applyCategory')  
    .addToUi();  
}
```

Lea Hoja de cálculo Añadir menú en línea: <https://riptutorial.com/es/google-apps-script/topic/4253/hoja-de-calculo-anadir-menu>

Capítulo 10: Llamadas del cliente a Google apps-script

Introducción

Google Appscript funciona bien como una plataforma independiente y en el formato de complemento para documentos, hojas y formularios de Google. Sin embargo, hay ocasiones en que un navegador cliente puede necesitar llamar a una aplicación de Google para realizar alguna acción.

Por lo tanto, Google introdujo las solicitudes del lado del cliente a los scripts de aplicaciones de Google. Para resolver este problema, Google introdujo las [bibliotecas del lado del cliente](#).

Examples

Este es un ejemplo de una llamada del lado del cliente a un script de aplicación de Google

```
<script src="https://apis.google.com/js/api.js"></script>
<script>
function start() {
  // 2. Initialize the JavaScript client library.
  gapi.client.init({
    'apiKey': 'YOUR_API_KEY',
    // clientId and scope are optional if auth is not required.
    'clientId': 'YOUR_WEB_CLIENT_ID.apps.googleusercontent.com',
    'scope': 'profile',
  }).then(function() {
    // 3. Initialize and make the API request.
    return gapi.client.request({
      'path': 'https://people.googleapis.com/v1/people/me',
    })
  }).then(function(response) {
    console.log(response.result);
  }, function(reason) {
    console.log('Error: ' + reason.result.error.message);
  });
};
// 1. Load the JavaScript client library.
gapi.load('client', start);
</script>
```

Lea Llamadas del cliente a Google apps-script en línea: <https://riptutorial.com/es/google-apps-script/topic/8875/llamadas-del-cliente-a-google-apps-script>

Capítulo 11: Script de la aplicación web de Google para descarga automática desde Google Drive

Introducción

Esta sencilla aplicación web de Google Script (independiente) permite que Google Drive se vuelva a sondear para que los archivos se descarguen en la PC local del usuario. Muestra cómo usar un script de aplicación para proporcionar la función de: 1. La interfaz de usuario (una simple en este ejemplo) 2. La página de descarga de archivos. Para una explicación más completa de cómo funciona, lea el Ejemplo "Cómo funciona".

Observaciones

El Web Script debe ser publicado para que funcione.

Los complementos deben estar habilitados para <https://script.google.com>

Examples

forms.html

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

    setInterval(
    function ()
    {
      document.getElementById('messages').innerHTML = 'Event Timer Fetching';
      google.script.run
        .withSuccessHandler(onSuccess)
        .withFailureHandler(onFailure)
        .fetchFromGoogleDrive();
    }, 60000);

    function callFetchGoogleDrive() {
      document.getElementById('messages').innerHTML = 'Fetching';
      google.script.run
        .withSuccessHandler(onSuccess)
        .withFailureHandler(onFailure)
        .fetchFromGoogleDrive();
    }

    function onSuccess(sHref)
    {
```

```

    if(new String(sHref).valueOf() == new String("").valueOf())
    {
        document.getElementById('messages').innerHTML = 'Nothing to download';
    }
    else
    {
        document.getElementById('messages').innerHTML = 'Success';
        document.getElementById('HiddenClick').href = sHref;
        document.getElementById('HiddenClick').click(); // Must enable Pop Ups for
https://script.google.com
    }
}

function onFailure(error)
{
    document.getElementById('messages').innerHTML = error.message;
}

</script>
</head>
<body>
<div id="messages">Waiting to DownLoad!</div>
<div>
    <a id="HiddenClick" href="" target="_blank" onclick="google.script.host.close"
style="visibility: hidden;">Hidden Click</a>
</div>
<div>
    <button type="button" onclick='callFetchGoogleDrive();' id="Fetch">Fetch Now!</button>
</div>
</body>
</html>

```

código.gs

```

function doGet(e) {
    var serveFile = e.parameter.servefile;
    var id = e.parameter.id;

    if(serveFile)
    {
        return downloadFile(id); // and Hyde
    }

    return HtmlService.createHtmlOutputFromFile('form.html'); // Jekyll
}

function fetchFromGoogleDrive() { // Jekyll
    var fileslist = DriveApp.searchFiles("your search criteria goes here + and trashed =
false"); // the 'and trashed = false' prevents the same file being download more than once

    if (fileslist.hasNext()) {
        var afile = fileslist.next();
        var html = ScriptApp.getService().getUrl()+"?servefile=true&id="+afile.getId();
        return html;
    }
    else
    {
        return '';
    }
}

```

```

}

function downloadFile(id){ // and Hyde
  try
  {
    var afile = DriveApp.getFileById(id);

    var aname = afile.getName();
    var acontent = afile.getAs('text/plain').getDataAsString();

    var output = ContentService.createTextOutput();
    output.setMimeType(ContentService.MimeType.CSV);
    output.setContent(acontent);
    output.downloadAsFile(aname);
    afile.setTrashed(true);
    return output;
  }
  catch (e) {
    return ContentService.createTextOutput('Nothing To Download')
  }
}

```

Cómo funciona

Google Drive (Standalone) Web App para descargar automáticamente (Sondeo) archivos desde Drive a la PC local del usuario (Carpeta de descarga).

DriveApp proporciona mecanismos para buscar y descargar archivos. Sin embargo, el mecanismo de descarga tiene algunas limitaciones importantes debido a la arquitectura cliente / servidor que heredó Google Apps. (No es culpa de Google)

El lado del servidor DriveApp no proporciona una función directa para descargar a la PC local porque el servidor no tiene idea de dónde está el cliente y la descarga del archivo al servidor en sí no tendría sentido.

El código del lado del servidor necesita un mecanismo para proporcionar al código del lado del cliente los datos del archivo o un enlace al archivo. Se proporcionan estos dos mecanismos, pero los datos del primero se limitan a ser utilizados directamente por el código del lado del cliente. El cliente no tiene ningún mecanismo para guardar los datos, una vez obtenidos, en la PC local. Por lo tanto, se puede utilizar para mostrar los datos en la propia página web.

El segundo mecanismo permite que la url de la secuencia de comandos (en sí) o la url del archivo de Drive se devuelva. La URL del archivo de Drive no es muy útil ya que no se puede usar directamente en el navegador del cliente para descargar el archivo. Colocar esta URL en el ancla (y hacer clic en ella) solo da como resultado una página web que se abre pero que en realidad no hace nada (excepto posiblemente ver el archivo en línea).

Eso deja el script url. Sin embargo, la URL del script solo proporciona el script y no el archivo.

Para iniciar una descarga, el archivo del servicio Drive debe devolverse desde la función doGet / doPost de la secuencia de comandos del lado del servidor utilizando ContentService createTextOutput exactamente como se muestra en las guías en línea de Google. Sin embargo,

esto implica que no puede haber ningún otro elemento de UI en la página web generado por los resultados devueltos por doGet / doPost.

Esto nos deja con una solución muy poco atractiva. Una página web en blanco sin elementos de usuario de la interfaz de usuario que descarga una página, se cierra y requiere que se abra manualmente cada vez que se requiera otra descarga.

Obviamente, otra página web de alojamiento podría proporcionar la IU y el enlace al script de descarga de la aplicación web para resolver este problema.

Este script utiliza un enfoque del Dr. Jekyll y Mr Hyde para resolver este problema.

Si el script se abre sin parámetros para GET (doGet), entonces se muestra de forma predeterminada un formulario. Esta será la condición cuando la aplicación publicada se abra por primera vez por un usuario. La forma proporcionada en este ejemplo es extremadamente simple.

Si el script se abre con el parámetro `servefile = true`, entonces el script se comporta como una descarga de archivos de Drive.

El javascript del lado del cliente contiene un mecanismo de sondeo (temporizador de eventos `setInterval`) que periódicamente llama al script del lado del servidor para verificar la disponibilidad de otro archivo para descargar.

Cuando se ejecuta el script del lado del servidor si encuentra algún archivo de Drive que coincida con los criterios de búsqueda *, devuelve la url del script adjunto con los parámetros:

```
? servefile = true & id = the_id_of_the_google_drive_file
```

(* Los criterios de búsqueda en este ejemplo simple están codificados en la secuencia de comandos del lado del servidor. Si es necesario, podrían pasar fácilmente del cliente al servidor).

Esta información se devuelve como una cadena al cliente a través del mecanismo reconocido con `SucursHandler`.

La secuencia de comandos java del cliente luego actualiza el HREF de un ancla oculta con esta información devuelta y luego hace clic en el ancla automáticamente.

Esto hace que se inicie otra invocación de la aplicación / script. Cuando se inicie la nueva invocación de la aplicación, doGet detectará el parámetro de archivo de servicio y, en lugar de devolver la interfaz de usuario, devolverá el archivo al navegador. El archivo devuelto será el identificado por el parámetro de ID proporcionado que fue previamente devuelto por la búsqueda descrita anteriormente.

Dado que el archivo con el ID proporcionado todavía existe, se descargará y la nueva invocación de la aplicación se cerrará, quedando la primera invocación para repetir este proceso.

Se proporciona un botón en la interfaz simple si el usuario / probador se muestra impaciente por esperar el temporizador, pero no es necesario y, de lo contrario, puede eliminarse.

Por supuesto, la forma simple puede extenderse para proporcionar una interfaz de usuario más

rica si es necesario. Tales como proporcionar los criterios de búsqueda de archivos.

Lea Script de la aplicación web de Google para descarga automática desde Google Drive en línea: <https://riptutorial.com/es/google-apps-script/topic/8212/script-de-la-aplicacion-web-de-google-para-descarga-automatica-desde-google-drive>

Capítulo 12: Servicio de hoja de cálculo

Observaciones

La referencia de API oficial para el servicio de hoja de cálculo se puede encontrar en <https://developers.google.com/apps-script/reference/spreadsheets/>.

Examples

Hoja

Obtener una referencia a una pestaña de hoja con nombre

```
var spread_sheet = SpreadsheetApp.getActiveSpreadsheet();//Get active spreadsheet
var sheet_with_name_a = spread_sheet.getSheetByName("sheet_tab_name");
```

Obtención de la pestaña de hoja activa

```
var spread_sheet = SpreadsheetApp.getActiveSpreadsheet();
var active_sheet = spread_sheet.getActiveSheet();
```

Insertar columna

```
var spread_sheet = SpreadsheetApp.getActiveSpreadsheet();
var active_sheet = spread_sheet.getActiveSheet();
active_sheet.insertColumnAfter(1); // This inserts a column after the first column position
active_sheet.insertColumnBefore(1); // This inserts a column in the first column position
active_sheet.insertColumns(1); // Shifts all columns by one
active_sheet.insertColumns(1, 3); // Shifts all columns by three
active_sheet.insertColumnsAfter(1); // This inserts a column in the second column position
active_sheet.insertColumnsBefore(1, 5); // This inserts five columns before the first column
```

Insertar fila

```
var spread_sheet = SpreadsheetApp.getActiveSpreadsheet();
var active_sheet = spread_sheet.getActiveSheet();
active_sheet.insertRowAfter(1); // This inserts a row after the first row position
active_sheet.insertRowBefore(1); // This inserts a row in the first row position
active_sheet.insertRows(1); // Shifts all rows by one
active_sheet.insertRows(1, 3); // Shifts all rows by three
active_sheet.insertRowsAfter(1); // This inserts a row in the second row position
active_sheet.insertRowsBefore(1, 5); // This inserts five rows before the first row
```

Valor de celda

```
var spread_sheet = SpreadsheetApp.getActiveSpreadsheet();
var active_sheet = spread_sheet.getActiveSheet();
var cell = range.getCell(1, 1);
var cell_value = cell.getValue();
```

```
cell.setValue(100);
```

Copiar celdas

```
var spread_sheet = SpreadsheetApp.getActiveSpreadsheet();
var active_sheet = spread_sheet.getActiveSheet();
var rangeToCopy = active_sheet.getRange(1, 1, sheet.getMaxRows(), 5);
rangeToCopy.copyTo(sheet.getRange(1, 6));
```

Fórmula

```
var spread_sheet = SpreadsheetApp.getActiveSpreadsheet();
var active_sheet = spread_sheet.getActiveSheet();
var range = active_sheet.getRange("B5");
var formula = range.getFormula();
range.setFormula("=SUM(B3:B4)");
```

Copie un valor de una hoja a la hoja actual

Imagine que tenemos una hoja de cálculo de Google por separado y necesitamos obtener el valor de la celda B2 a la celda D5 en su hoja actual.

```
function copyValueandPaste()
{
    var source = SpreadsheetApp.openById('spread sheet id is here'); //Separate spreadsheet
    book
    var sourcesheet = source.getSheetByName('Sheet1'); //Sheet tab with source data
    var sourceCellValue = sourcesheet.getRange('B2').getValue(); // get B2 cell value

    var thisBook = SpreadsheetApp.getActive(); // Active spreadsheet book
    var thisSheet = thisBook.getSheetByName('Sheet1'); // Target sheet
    thisSheet.getRange('D5').setValue(sourceCellValue); //Set value to target sheet D5 cell
}
```

Puede encontrar el ID de la hoja de cálculo desde su URL.

Obtener la última fila en una sola columna

```
function lastRowForColumn(sheet, column){
    // Get the last row with data for the whole sheet.
    var numRows = sheet.getLastRow();

    // Get all data for the given column
    var data = sheet.getRange(1, column, numRows).getValues();

    // Iterate backwards and find first non empty cell
    for(var i = data.length - 1 ; i >= 0 ; i--){
        if (data[i][0] != null && data[i][0] != ""){
            return i + 1;
        }
    }
}
```

Inserción de matrices como filas

Insertar una fila en la parte inferior de una hoja de cálculo es fácil:

```
var someSheet = SpreadsheetApp.getActiveSpreadsheet().getSheets()[0];
someSheet.appendRow(["Frodo", "Baggins", "Hobbit", "The Shire", 33]);
```

Tenga en cuenta que esto agregará la fila después de la última fila *no vacía* .

Insertar una fila en algún lugar en el medio es un poco más de trabajo:

```
var someSheet = SpreadsheetApp.getActiveSpreadsheet().getSheets()[0];

var newRowIndex = 2;
var row = ["Gandalf", "?", "Wizard", "?", 2019];
someSheet.insertRowBefore(newRowIndex);
// getRange(row, col, numRows, numCols)
someSheet.getRange(newRowIndex, 1, 1, row.length).setValues([row]); // Note 2D array!
```

Gran parte de este código inútil se puede abstraer en una función auxiliar:

```
function insertRowBefore(sheet, rowIndex, rowData) {
  sheet.insertRowBefore(rowIndex);
  sheet.getRange(rowIndex, 1, 1, rowData.length).setValues([rowData]);
}
```

Lo que reduce nuestro ejemplo a solo:

```
var someSheet = SpreadsheetApp.getActiveSpreadsheet().getSheets()[0];
insertRowBefore(someSheet, 2, ["Gandalf", "?", "Wizard", "?", 2019]);
```

Lea Servicio de hoja de cálculo en línea: <https://riptutorial.com/es/google-apps-script/topic/2688/servicio-de-hoja-de-calculo>

Capítulo 13: Servicio DriveApp

Observaciones

Los tipos Mime de Google no se pueden usar para el tercer parámetro de Tipos Mime. El uso de un tipo Mime de Google generará un error que indica:

No se puede usar "DriveApp.createFile ()" para crear tipos MIME de Google. Por favor, utilice el servicio de unidad avanzada

MimeType.GOOGLE_APPS_SCRIPT

MimeType.GOOGLE_DOCS

MimeType.GOOGLE_DRAWINGS

MimeType.GOOGLE_FORMS

MimeType.GOOGLE_SHEETS

MimeType.GOOGLE_SLIDES

Examples

Crear una nueva carpeta en la unidad raíz de Google

```
function createNewFolderInGoogleDrive() {
  var folderName,newFolder;//Declare variable names

  folderName = "Test Folder " + new Date().toString().slice(0,15);//Create a new folder name
with date on end
  newFolder = DriveApp.createFolder(folderName);//Create a new folder in the root drive
};
```

Crear nuevo archivo en Google Drive de un determinado tipo de Mime

```
function createGoogleDriveFileOfMimeType() {
  var content,fileName,newFile;//Declare variable names

  fileName = "Test File " + new Date().toString().slice(0,15);//Create a new file name with
date on end
  content = "This is the file Content";

  newFile = DriveApp.createFile(fileName,content,MimeType.JAVASCRIPT);//Create a new file in
the root folder
};
```

Crear un nuevo archivo de texto en la carpeta de la unidad raíz de Google

```
function createGoogleDriveTextFile() {
  var content,fileName,newFile;//Declare variable names

  fileName = "Test Doc " + new Date().toString().slice(0,15);//Create a new file name with
date on end
  content = "This is the file Content";

  newFile = DriveApp.createFile(fileName,content);//Create a new text file in the root folder
};
```

Crear un nuevo archivo en Google Drive desde un blob

```
function createGoogleDriveFileWithBlob() {
  var blob,character,data,fileName,i,L,max,min,newFile,randomNmbr;//Declare variable names

  fileName = "Test Blob " + new Date().toString().slice(0,15);//Create a new file name with
date on end

  L = 500;//Define how many times to loop
  data = "";
  max = 126;
  min = 55;

  for (i=0;i<L;i+=1) { //Loop to create data
    randomNmbr = Math.floor(Math.random()*(max-min+1)+min);//Create a random number
    //Logger.log('randomNmbr: ' + randomNmbr);
    character = String.fromCharCode(randomNmbr);

    //Logger.log('character: ' + character);//Print the character to the Logs
    data = data + character;
  };

  blob = Utilities.newBlob(data, MimeType.PLAIN_TEXT, fileName);//Create a blob with random
characters

  newFile = DriveApp.createFile(blob);//Create a new file from a blob

  newFile.setName(fileName);//Set the file name of the new file
};
```

Obtenga todas las carpetas: coloque las carpetas en un token de continuación y luego recupérelas

```
function processGoogleDriveFolders() {
  var arrayAllFolderNames,continuationToken,folders,foldersFromToken,thisFolder;//Declare
variable names

  arrayAllFolderNames = [];//Create an empty array and assign it to this variable name

  folders = DriveApp.getFolders();//Get all folders from Google Drive in this account
  continuationToken = folders.getContinuationToken();//Get the continuation token

  Utilities.sleep(18000);//Pause the code for 3 seconds

  foldersFromToken = DriveApp.continueFolderIterator(continuationToken);//Get the original
folders stored in the token
  folders = null;//Delete the folders that were stored in the original variable, to prove that
```

```

the continuation token is working

while (foldersFromToken.hasNext()) { //If there is a next folder, then continue looping
    thisFolder = foldersFromToken.next(); //Get the next folder
    arrayAllFolderNames.push(thisFolder.getName()); //Get the name of the next folder
};

Logger.log(arrayAllFolderNames); //print the folder names to the Logs
};

```

Obtenga todos los archivos, póngalos en un token de continuación, luego recupérellos

```

function processGoogleDriveFiles() {
    var arrayAllFileNames, continuationToken, files, filesFromToken, fileIterator, thisFile; //Declare
    variable names

    arrayAllFileNames = []; //Create an empty array and assign it to this variable name

    files = DriveApp.getFiles(); //Get all files from Google Drive in this account
    continuationToken = files.getContinuationToken(); //Get the continuation token

    Utilities.sleep(18000); //Pause the code for 3 seconds

    filesFromToken = DriveApp.continueFileIterator(continuationToken); //Get the original files
    stored in the token
    files = null; //Delete the files that were stored in the original variable, to prove that the
    continuation token is working

    while (filesFromToken.hasNext()) { //If there is a next file, then continue looping
        thisFile = filesFromToken.next(); //Get the next file
        arrayAllFileNames.push(thisFile.getName()); //Get the name of the next file
    };

    Logger.log(arrayAllFileNames);
};

```

Lea Servicio DriveApp en línea: <https://riptutorial.com/es/google-apps-script/topic/6395/servicio-driveapp>

Capítulo 14: Servicio DriveApp - Archivos por tipo y cadena de búsqueda

Parámetros

Nombre del parámetro	Usar para
cadena de búsqueda	la cadena que se encuentra en el nombre del archivo

Examples

Obtenga archivos por tipo de archivo con cadena coincidente en nombre de archivo

Obtenga todos los formularios de Google con la palabra "Sin título" en el nombre del archivo.

```
function mainSearchFunction(searchStr) {
  var fileInfo,arrayFileIDs,arrayFileNames,arrayOfIndexNumbers,
      allFileIDsWithStringInName,i,searchStr,thisID;//Declare variables

  if (!searchStr) {
    searchStr = "Untitled";//Assign a string value to the variable
  };

  fileInfo = getFilesOfType();//Run a function that returns files information
  arrayFileNames = fileInfo[1];//Get the array of file names
  arrayOfIndexNumbers = searchFileNamesForString(arrayFileNames,searchStr);

  //Logger.log('searchStr: ' + searchStr)
  //Logger.log(arrayOfIndexNumbers)

  allFileIDsWithStringInName = [];
  arrayFileIDs = fileInfo[0];

  for (i=0;i<arrayOfIndexNumbers.length;i+=1) {
    thisID = arrayFileIDs[arrayOfIndexNumbers[i]];
    allFileIDsWithStringInName.push(thisID);
  };

  Logger.log(allFileIDsWithStringInName)
};

function getFilesOfType() {
  var allFormFiles,arrFileName,arrFileID,arrFileUrls,thisFile;

  allFormFiles = DriveApp.getFilesByType(MimeType.GOOGLE_FORMS);
  arrFileName = [];
  arrFileID = [];
  arrFileUrls = [];

  while (allFormFiles.hasNext()) {
```

```

    thisFile=allFormFiles.next();
    arrFileName.push(thisFile.getName());
    arrFileID.push(thisFile.getId());
    arrFileUrls.push(thisFile.getUrl());
};

//Logger.log(arrFileName)
return [arrFileID,arrFileName];
};

function searchFileNamesForString(arrayFileNames,searchStr) {
    var arrayIndexNumbers,i,L,thisName;

    arrayIndexNumbers = [];

    L = arrayFileNames.length;

    for (i=0;i<L;i+=1){
        thisName = arrayFileNames[i];
        Logger.log(thisName);
        Logger.log('thisName.indexOf(searchStr): ' + thisName.indexOf(searchStr));

        if (thisName.indexOf(searchStr) !== -1) {
            arrayIndexNumbers.push(i);
        };
    };

    return arrayIndexNumbers;
};

```

Lea Servicio DriveApp - Archivos por tipo y cadena de búsqueda en línea:

<https://riptutorial.com/es/google-apps-script/topic/4049/servicio-driveapp---archivos-por-tipo-y-cadena-de-busqueda>

Capítulo 15: SpreadsheetApp Active Sheet

Observaciones

Método: `getActive ()`

Tipo de devolución: [Hoja de cálculo](#)

Examples

`getActive ()` - Obtener hoja de cálculo activa

Esto devuelve la hoja de cálculo activa actualmente, o nula si no hay ninguna.

```
var currentSheet = SpreadsheetApp.getActive();  
var url = currentSheet.getUrl();  
Logger.log( url );
```

Lea [SpreadsheetApp Active Sheet en línea](https://riptutorial.com/es/google-apps-script/topic/5861/spreadsheetapp-active-sheet): <https://riptutorial.com/es/google-apps-script/topic/5861/spreadsheetapp-active-sheet>

Creditos

S. No	Capítulos	Contributors
1	Empezando con google-apps-script	Albert Portnoy , Community , Douglas Gaskell , iJay , MShoaib91 , Rubén , Saloni Vithalani , Shyam Kansagra , Spencer Easton , sudo bangbang , Supertopoz
2	Aplicaciones Script Web Apps	Douglas Gaskell
3	Crear una función personalizada para las hojas de Google	Francky_V , Joshua Dawson , Pierre-Marie Richard , Rubén
4	DriveApp	Brian , Kos , nibarius , Sandy Good , Wolfgang
5	DriveApp - getFileById (id)	Sandy Good
6	Firebase y AppScript: Introducción	Joseba , Vishal Vishwakarma
7	GmailApp	nibarius
8	Google hojas MailApp	Bhupendra Piprava , Brian , Jordan Rhea , Kos , nibarius , Saloni Vithalani
9	Hoja de cálculo Añadir menú	Bishal , iJay , nibarius
10	Llamadas del cliente a Google apps-script	Supertopoz
11	Script de la aplicación web de Google para descarga automática desde Google Drive	Walter
12	Servicio de hoja de cálculo	cdrini , iJay , nibarius , Sandy Good , sudo bangbang
13	Servicio DriveApp	Sandy Good
14	Servicio DriveApp -	nibarius , Sandy Good

Archivos por tipo y cadena de búsqueda		
15	SpreadsheetApp Active Sheet	iJay