Бесплатная электронная книга

УЧУСЬ google-apps-script

Free unaffiliated eBook created from
Stack Overflow contributors.

#google-apps-script

Около

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.

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

глава 1: Начало работы с скриптом google-apps

замечания

Официальный обзор для Google Apps Script опубликован по адресу http://www.google.com/script/start , оттуда

> Скрипт Google Apps - это язык сценариев облачных вычислений JavaScript, который предоставляет простые способы автоматизации задач продуктов Google и сторонних служб и создания веб-приложений.

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

> Скрипт приложений основан на JavaScript 1.6 , плюс несколько функций от 1.7 и 1.8 . Таким образом, многие базовые функции JavaScript доступны в дополнение к встроенным и расширенным службам Google : вы можете использовать общие объекты, такие как Array , Date , RegExp и т. Д. , А также глобальные объекты Math и Object . Однако, поскольку код приложения Script работает на серверах Google (а не на стороне клиента, за исключением страниц HTML-сервиса), функции на основе браузера, такие как DOM-манипуляция или API окна , недоступны.

Examples

Установка или настройка

Скрипт Google Apps не требует установки или установки. Единственное требование - учетная запись Google. Работает учетная запись Gmail, а также учетная запись Google Apps for Work / Education / Government. Вы можете создать новую учетную запись Google, перейдя на accounts.google.com.

Начните свой первый скрипт, перейдя на script.google.com . Вы также можете открыть Google Apps Script под `tools -> Script editor...` из многих Google Apps, то есть *Документов, Таблиц, Форм и т* . Д. Скрипт Google Apps также можно добавить прямо на Google Диск с помощью функции « `Connect more apps...` .

Официальную документацию можно найти на developers.google.com/apps-script/ .

Для запуска сценариев приложений они должны содержать файл code.gs. Файл code.gs должен содержать функцию с именем doGet (автономные скрипты) или функцию onOpen (

аддон-скрипты). Быстрые запуски в документации содержат примеры.

Если api включен в сценарии приложения, он также должен быть включен в консоли разработчика. Однако консоль разработчиков содержит api, которая может быть включена, но не отображается в интерфейсе приложения-скрипта. Например, SDK Marketplace необходимо включить в консоли разработчиков до того, как приложение может быть опубликовано в магазине воспроизведения Google или в широкомасштабном развертывании домена G Suite.

Для приложений Google для обучения / работы / правительства в консоли администратора домена есть настройки, которые можно настроить, чтобы разрешать или запрещать выполнение сценариев приложений.

Типы скриптов

Скрипты Google App имеют три типа.

- Standalone
- Связано с Google Apps
- Веб-приложения

Автономный скрипт

Автономные скрипты не привязаны ни к каким приложениям Google, а именно к *документам, таблицам или формам и т. Д.* Автономный скрипт можно создать, посетив *script.google.com* или подключив сценарий приложения Google с помощью Google диска. Автономный скрипт может использоваться для самостоятельного программирования приложений Google, может использоваться как веб-приложение или может быть настроен для автоматического запуска с установочного триггера. См. Документацию для автономного скрипта.

Связано с Google Apps

Сценарий, связанный с Google Apps, также известен как сценарий, связанный с контейнером; в отличие от автономных скриптов, привязаны к приложениям Google, то есть к *Документам Google или Google Таблицам и т. д.* Скрипт, связанный с контейнером, можно создать, выбрав `tools> Script editor` из Google App. Некоторые функции, такие как диалоги, подсказки, меню и боковая панель, предоставляются только сценариями, связанными с контейнерами. Кроме того, сценарий, связанный с контейнером, используется для создания надстроек Google . См. Документацию для сценариев, связанных с контейнером.

Веб-приложения

Скрипт Google App Script можно использовать в качестве веб-приложения, так как к нему можно получить доступ через браузер. Веб-приложение может предоставлять

пользовательский интерфейс в браузере и может использовать приложения Google, то есть *документы, листы и т. Д.* Оба автономных сценария и скрипты, привязанные к Google Apps, могут быть превращены в веб-приложения. Для того чтобы любой скрипт работал как веб-приложение, скрипт должен отвечать двум требованиям:

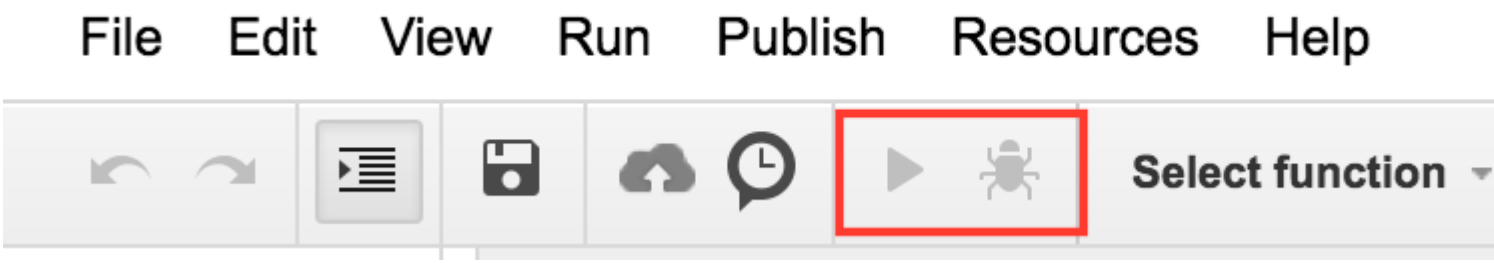
- включают функцию `doGet()` или `doPost()` .
- Функция возвращает объект HTML HtmlOutput HTML или объект TextOutput службы контента.

Функции Inshort, `doGet()` и `doPost()` работают как обработчики http get и post request соответственно.

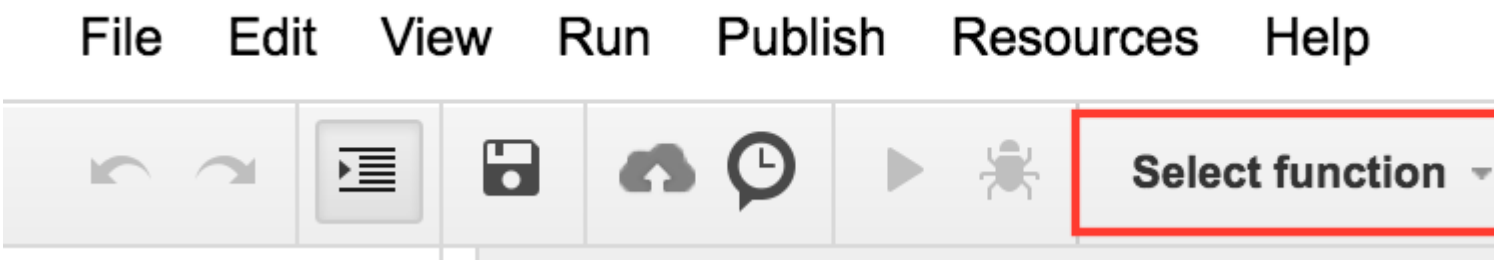
Более подробную информацию о веб-приложениях см. В официальной документации .

Выполнение / отладка вашего скрипта

Попробуйте запустить свой код с панели инструментов, как показано ниже:



В вашем коде, если у вас есть несколько функций, перед запуском следует указать функцию, с которой вы хотите работать. Например :



Кроме того, вы можете нажать **ctrl + r** с клавиатуры, чтобы запустить код. Он сначала сохранит код, если не будет сохранен, а затем запустит его. Но для этого вам необходимо выбрать функцию, как показано на изображении выше.

Кроме того, если ваш сценарий вызывается некоторыми внешними действиями, все же вы сможете просмотреть журналы, щелкнув view-> logs, если вы регистрируете что-либо после выполнения кода.

Привет, мир

Мы собираемся сказать Hello как окно сообщения.

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

Чтобы выполнить скрипт, нажмите ▶ или выберите пункт меню « **Выполнить**» -> **helloWorld**

Более глубокий взгляд на скрипт Google Apps

Скрипт Google Apps - это платформа, основанная на JavaScript, которая в первую очередь используется для автоматизации и расширения Google Apps. Скрипт приложений работает исключительно в инфраструктуре Google, не требующей настройки или конфигурации сервера. Онлайновая среда IDE служит интерфейсом для всей платформы, соединяющей все службы, доступные для скриптов приложений. Пользовательская аутентификация вызывается в платформу через OAuth2 и не требует никакого кода или настройки автором сценария.

Сценарий приложений работает на стороне сервера, но может иметь пользовательские интерфейсы, созданные с помощью HTML, CSS, JavaScript или любых других технологий, поддерживаемых браузером. В отличие от Nodejs, который управляется событиями, скрипты приложений выполняются в поточной модели. Все вызовы сценария генерируют уникальный экземпляр этого скрипта, который выполняется изолированно от всех других экземпляров. Когда экземпляр скрипта завершает выполнение, он уничтожается.

Функции в скрипте приложений блокируются, поэтому обратный вызов и шаблоны асинхронного программирования не нужны. Блокировка используется для предотвращения одновременного выполнения критических разделов кода, таких как файл IO, разными экземплярами.

На практике написания скриптов приложений просты. Ниже приведен простой скрипт, который создает новую таблицу из таблицы шаблонов.

```
// Create a new spreadsheet from a template
function createSpreadsheet(){
   var templateFileId = '1Azcz9GwCeHjGl9TXf4aUh6g20Eqmgd1UMSdNVjzIZPk';
   var sheetName = 'Account Log for:' + new Date();
   SpreadsheetApp.openById(templateFileId).copy(sheetName);
}
```

Прочитайте Начало работы с скриптом google-apps онлайн: https://riptutorial.com/ru/google-apps-script/topic/1154/начало-работы-с-скриптом-google-apps

глава 2: DriveApp

Examples

Создайте новую папку в корне Google Диска

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

Используйте функцию `createNewFolderInGoogleDrive` для создания папки с именем `Test folder` в корне Google Диска:

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

`newFolder` имеет класс Тип папки :

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

Создайте новый файл в Google Диске определенного типа 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
};
```

Создайте новый текстовый файл в корневой папке Google Диска

```
 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
};
```

Создайте новый файл на диске Google из блоба

```
 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
};
```

Получить все папки - поместить папки в токен продолжения - затем извлечь из токена

```
 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
};
```

Получите все файлы - поместите их в токен продолжения - затем извлеките их

```
 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);
};
```

Добавление папки в корневой диск

```
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
};
```

Создайте новый текстовый файл и добавьте его в корневую папку

```
 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
};
```

Получить все файлы в папке накопителя

```
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++;
  }
}
```

Прочитайте DriveApp онлайн: https://riptutorial.com/ru/google-apps-script/topic/5363/driveapp

глава 3: DriveApp - getFileById (id)

замечания

Также можно получить файл по URL-адресу файла. Идентификатор файла находится в URL-адресе, поэтому использование идентификатора вместо всего URL означает, что этот параметр короче. Сохранение URL-адреса, а не идентификатора занимает больше места.

Examples

Получите файл с Google Диска, используя идентификатор файла

```
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
};
```

Прочитайте DriveApp - getFileById (id) онлайн: https://riptutorial.com/ru/google-apps-script/topic/6087/driveapp---getfilebyid--id-

глава 4: Firebase и AppScript: Введение

Вступление

Интегрируйте Firebase с Google AppScript для чтения и записи данных в базе данных Firebase.

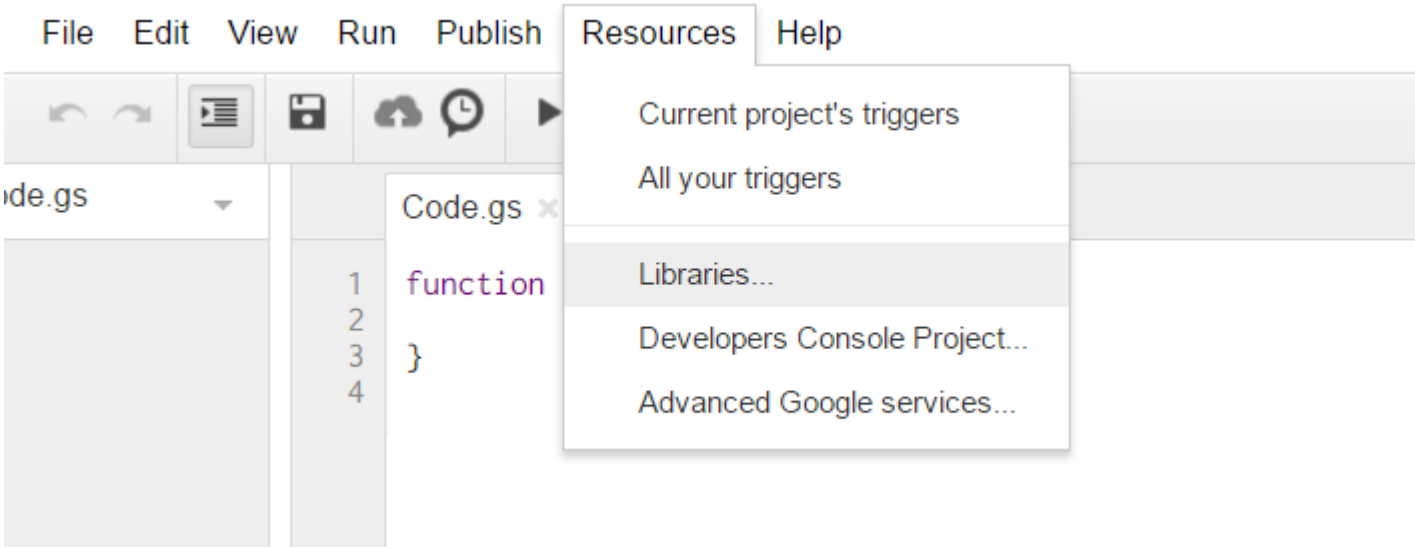
Firebase - это система баз данных NoSQL от Google, которая использует базу данных в реальном времени для создания и размещения приложений на мобильных, настольных и планшетных устройствах. Базы данных NoSQL используют объекты JSON для хранения данных в структурированном формате.

Examples

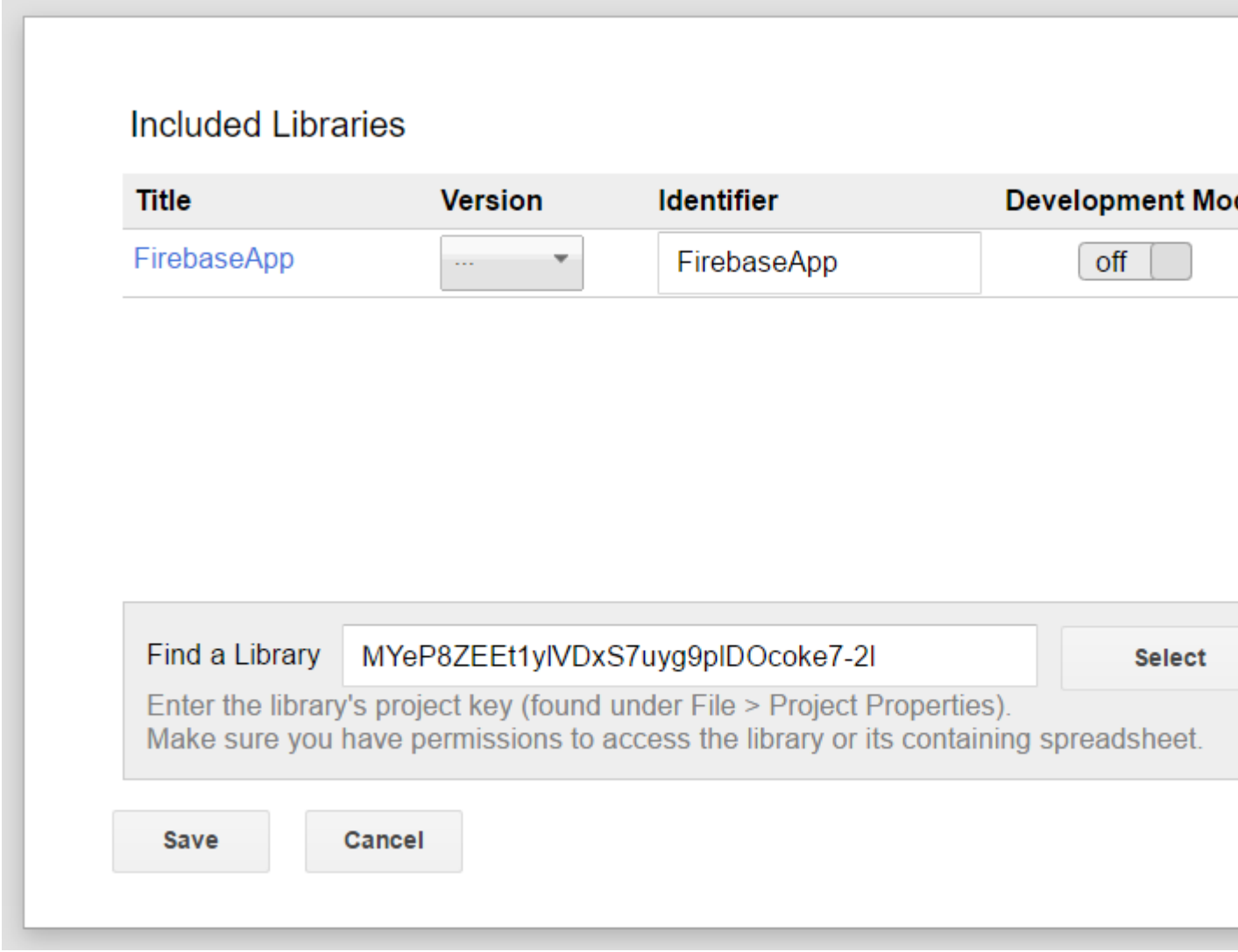
Подключение к проекту Firebase в GAS и передача данных из Google Spreadsheet в Firebase

Установите ресурс Firebase в AppScript

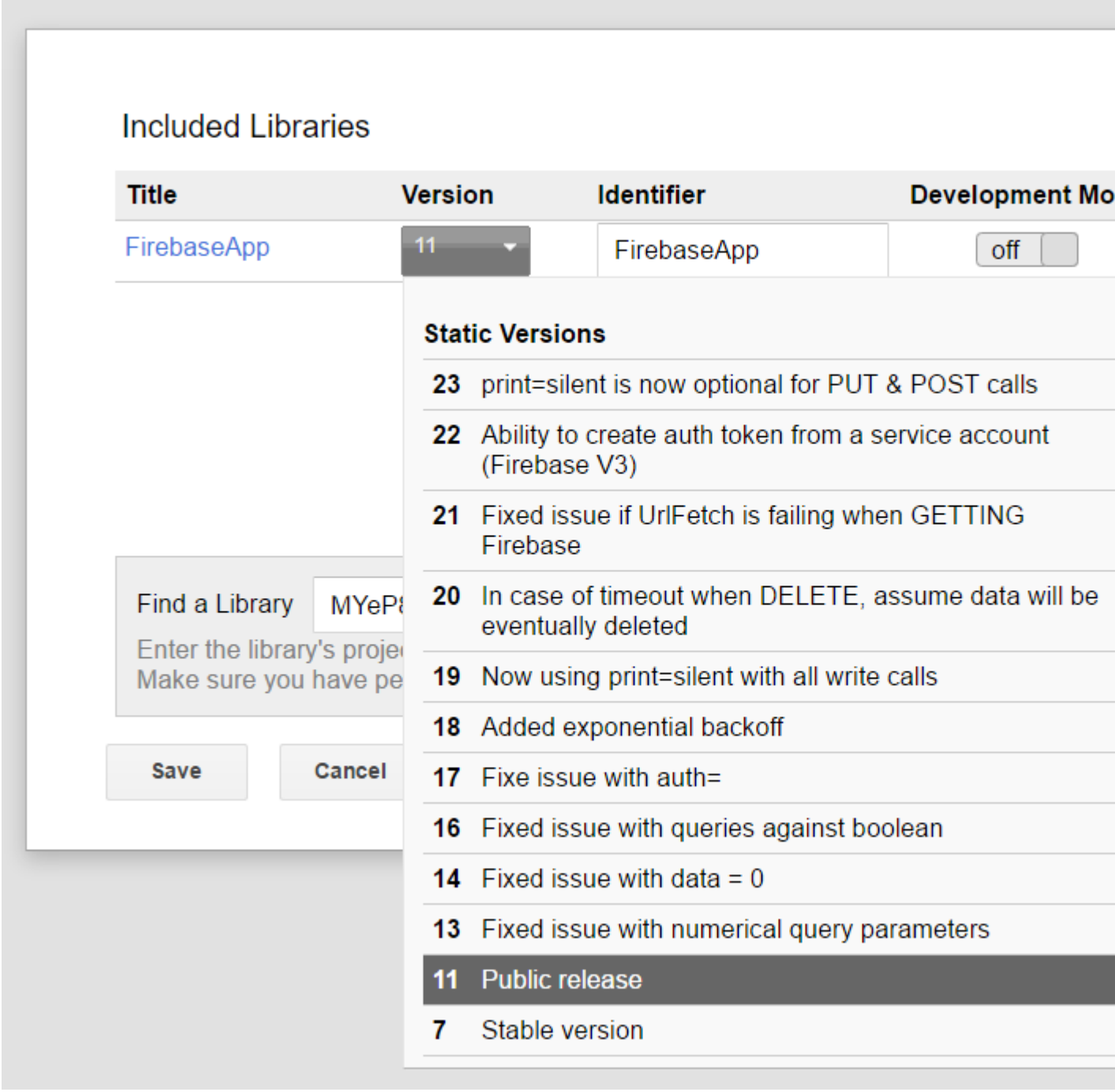
- Для этого нажмите «Ресурсы», а затем «Библиотеки».
- Firebase имеет уникальный ключ библиотеки проекта, который необходимо установить в AppScript.



- Нажмите «Библиотеки». Появится следующее всплывающее окно. Введите следующий код проекта в текстовое поле. **MYeP8ZEEt1ylVDxS7uyg9plDOcoke7-2l** Это ключ библиотеки проекта для Firebase.



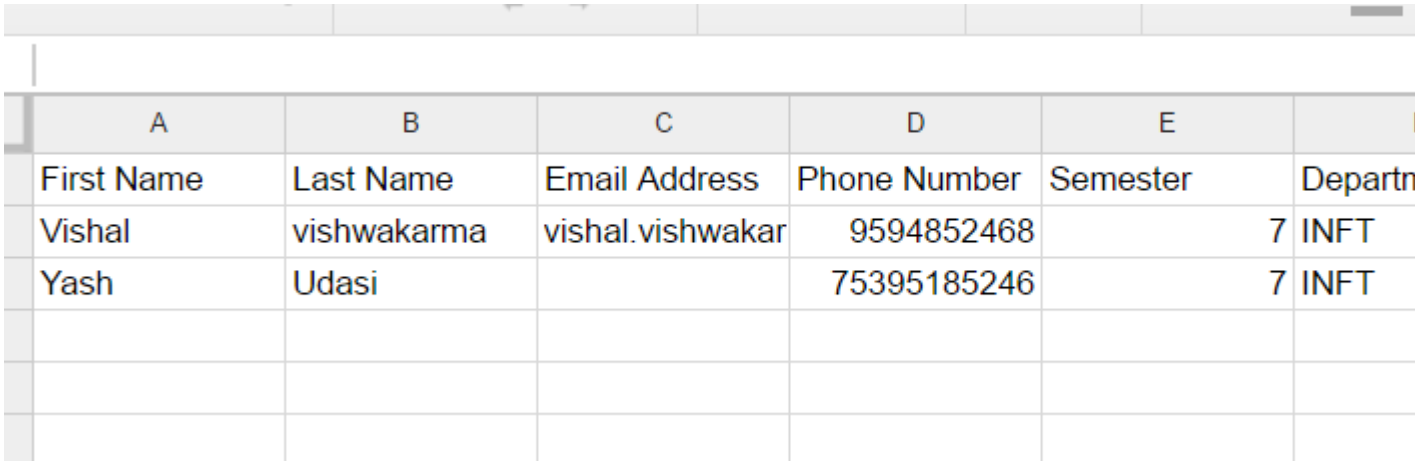
- Теперь в версии выберите стабильную версию общедоступного выпуска.



- Нажмите «Сохранить». Теперь Firebase успешно установлена в вашем приложении, чтобы вы могли работать.

Теперь давайте возьмем пример для чтения и записи данных из Firebase.

- Теперь мы возьмем примерную таблицу, разработанную в Google Таблицах.



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

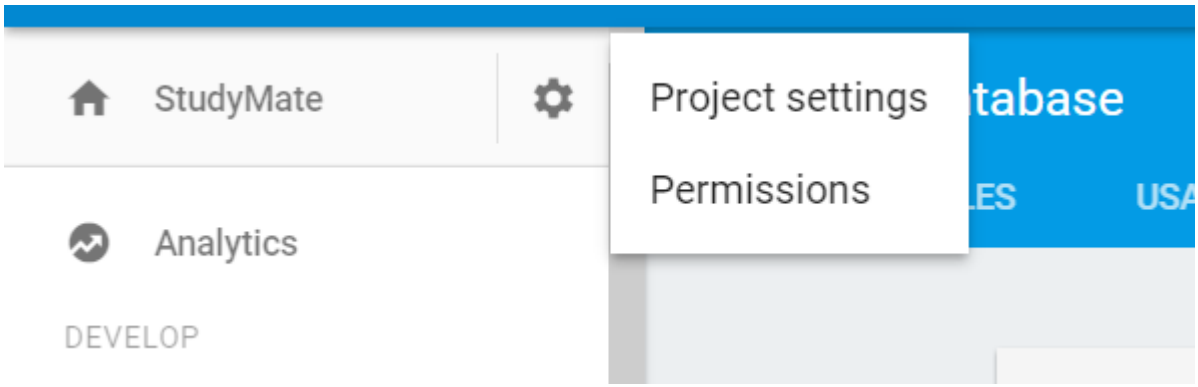
- Теперь создадим базу данных в Firebase, используя эту таблицу в листах. Добавьте следующий код в AppScript.

```
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);
}
```

Замените идентификатор электронной таблицы и firebaseURL и секретный ключ.

Как найти firebaseURL и секретный ключ?

- Перейдите в панель Firebase Dashboard и нажмите на настройку в левом верхнем углу. Нажмите «Настройки проекта».



- Перейдите в раздел «Учетные записи службы», где вы можете найти базу данных URL. Это служит firebaseURL.
- Теперь перейдите на вкладку «Секреты базы данных», и вы можете найти секретный ключ.

Теперь вы вставили firebaseURL и секретный ключ. Теперь все готово. Нажмите на код запуска в движке AppScript.

- Он попросит сначала просмотреть разрешения в первый раз при запуске.
- Нажмите «Разрешения» и «Разрешить».
- Теперь вы запускаете свою функцию, и вы можете увидеть таблицу, созданную в базе данных Firebase.

Чтобы увидеть базу данных, перейдите на панель мониторинга Firebase и щелкните по базе данных, которую вы можете просмотреть в базе данных.

Еще несколько функций для реализации чтения и записи.

1. Чтобы написать простые данные, чтобы проверить, работает ли соединение, нет.

```
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. Чтобы прочитать все данные

```
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);
          }
}
```

Чтение данных отображается в журналах. Чтобы проверить журналы, нажмите на кнопку «Просмотр» → «Журналы» или просто нажмите «Управление» + «Ввод».

3. Чтобы прочитать конкретную запись

```
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);
}
```

Чтение данных отображается в журналах. Чтобы проверить журналы, нажмите на кнопку «Просмотр» → «Журналы» или просто нажмите «Управление» + «Ввод».

4. Чтобы обновить определенную запись.

```
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");
}
```

Прочитайте Firebase и AppScript: Введение онлайн: https://riptutorial.com/ru/google-apps-script/topic/9417/firebase-и-appscript--введение

глава 5: GmailApp

замечания

См. Также официальную ссылку API для GmailApp для получения дополнительной информации о доступных методах.

Examples

Получить файл CSV, прикрепленный к почте

Предположим, что у нас есть система, которая отправляет ежедневные отчеты по электронной почте в виде прикрепленных файлов CSV и что мы хотим получить к ним доступ.

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

Прочитайте GmailApp онлайн: https://riptutorial.com/ru/google-apps-script/topic/5899/gmailapp

глава 6: Активный лист таблицы SpreadsheetApp

замечания

Метод: getActive ()

Тип возврата: электронная таблица

Examples

getActive () - Получить активную электронную таблицу

Это возвращает текущую активную электронную таблицу или null, если ее нет.

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

Прочитайте Активный лист таблицы SpreadsheetApp онлайн: https://riptutorial.com/ru/google-apps-script/topic/5861/активный-лист-таблицы-spreadsheetapp

глава 7: Клиент звонит в Google apps-script

Вступление

Google appscript хорошо работает как отдельная платформа и в формате аддона для документов, листов и форм Google. Тем не менее, есть моменты, когда браузеру браузера может потребоваться позвонить в приложение Google, чтобы выполнить какое-либо действие.

Таким образом, Google представила клиентские запросы на приложения-приложения Google. Чтобы решить эту проблему, Google представил библиотеки на стороне клиента

Examples

Это пример вызова клиентской стороны в приложение-приложение 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>
```

Прочитайте Клиент звонит в Google apps-script онлайн: https://riptutorial.com/ru/google-apps-script/topic/8875/клиент-звонит-в-google-apps-script

глава 8: Листы Google MailApp

Вступление

Эта служба позволяет пользователям отправлять электронные письма с полным контролем над содержимым электронной почты. В отличие от GmailApp, единственная цель MailApp - отправлять электронную почту. MailApp не может получить доступ к почтовому ящику Gmail пользователя.

Изменения в сценариях, написанных с помощью GmailApp, скорее всего, вызовут запрос повторной авторизации от пользователя, чем скрипты MailApp.

Examples

Основной пример MailApp

MailApp - это api из Google App Script, который можно использовать для отправки почты

```
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);
}
```

Класс MailApp ограничен квотами на основе вашего аккаунта Google:

- Пользователь потребителя (например, личная учетная запись Gmail): 100 получателей / день
- Клиент Google Apps (устаревший): 100 получателей / день
- GSuite (основной / Gov / Edu / Business): 1500 получателей / день

Вы можете проверить свою квоту электронной почты в `MailApp`

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

Доступ к данным из листа

```
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;
}
```

Вы также можете изменить приведенную выше функцию следующим образом, чтобы получить динамический диапазон данных из содержимого, содержащегося в листе:

```
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;
}
```

Использовать данные Листа для отправки электронной почты

Дано - У вас есть лист сотрудников, которые запросили компенсацию.

Требование. Мы должны отправить электронное письмо сотруднику при обработке их возмещения

Таким образом, лист выглядит следующим образом:

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

Функция отправки сообщения электронной почты выглядит следующим образом:

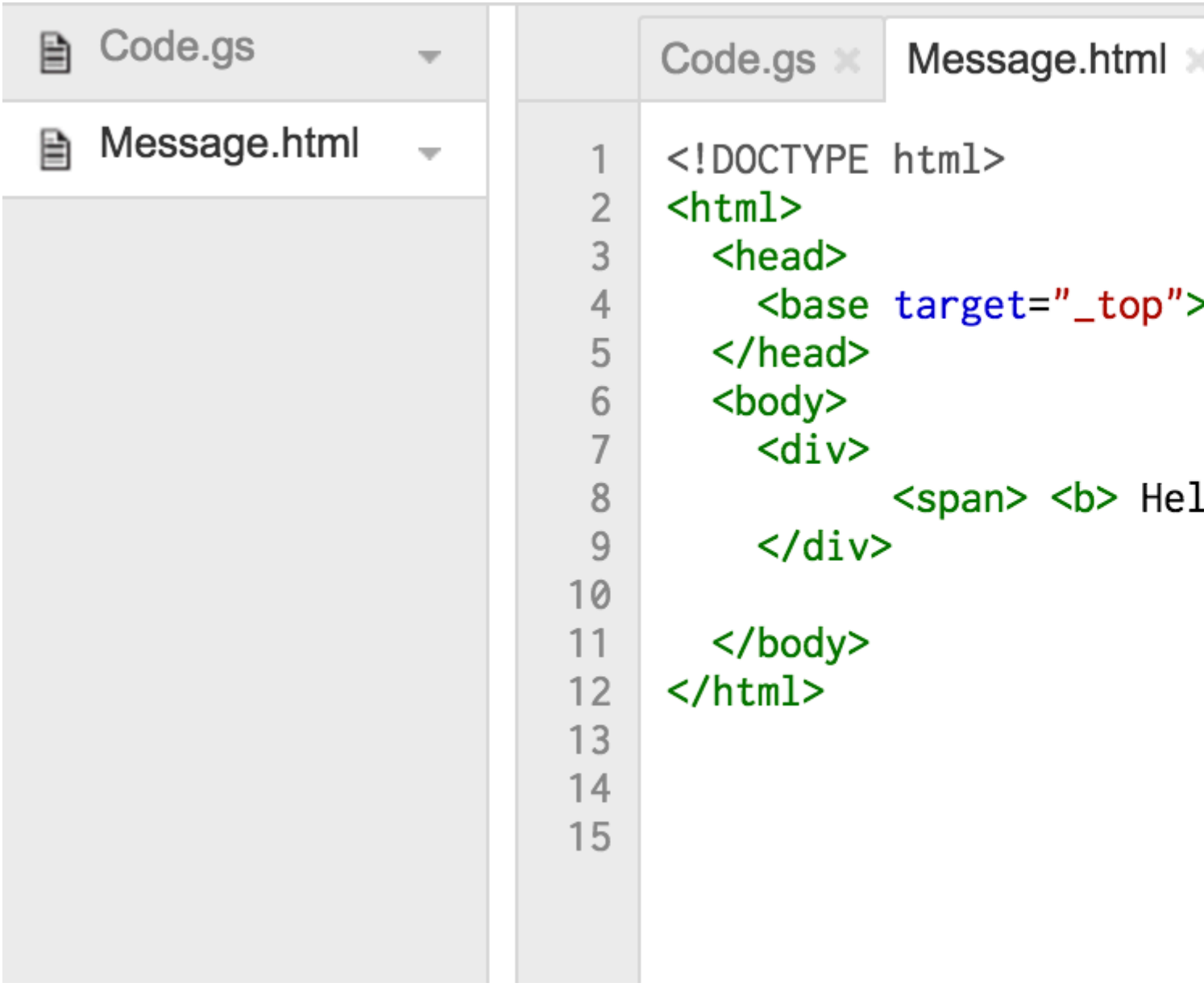
```
function getDataSheet() {

  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

Отправка HTML-содержимого по почте

В приведенном выше примере, если мы хотим отправить HTML-содержимое в виде сообщения в электронном письме, создайте файл HTML, перейдя в **Файл -> Создать -> HTML-файл**

Теперь вы можете увидеть HTML-файл, помимо вашего gs-файла, следующим образом:



Теперь обновите метод *getMessage ()* из приведенного выше примера следующим образом:

```
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;
}
```

Также необходимо *изменить* обращение к *MailApp* api

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

Таким образом, весь код будет выглядеть следующим образом:

```
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);
    }
  }
}
```

Прочитайте Листы Google MailApp онлайн: https://riptutorial.com/ru/google-apps-script/topic/5298/листы-google-mailapp

глава 9: Меню добавления таблиц

Синтаксис

1. addMenu (имя, subMenus)

параметры

название	Описание
название	имя создаваемого меню
подменю	массив карт JavaScript

замечания

Обычно вам нужно вызвать addMenu из функции onOpen, чтобы меню автоматически создавалось при загрузке электронной таблицы.

```
 // 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

Создать новое меню

Создает новое меню в пользовательском интерфейсе электронных таблиц. Каждая запись в меню запускает пользовательскую функцию.

```
   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);
```

Создание пользовательского меню

/ *

Метод: создание пользовательского меню. Это первая функция, которая будет вызываться при загрузке приложений

---

* /

```
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();
}
```

Прочитайте Меню добавления таблиц онлайн: https://riptutorial.com/ru/google-apps-script/topic/4253/меню-добавления-таблиц

глава 10: Обслуживание электронных таблиц

замечания

Официальную ссылку API для службы электронных таблиц можно найти на странице https://developers.google.com/apps-script/reference/spreadsheet/ .

Examples

Простынь

Получение ссылки на вкладку именованного листа

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

Получение активной вкладки листа

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

Вставить колонку

```
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
```

Вставить строку

```
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
```

Значение ячейки

```
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);
```

Копировать ячейки

```
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));
```

формула

```
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)");
```

Скопировать значение с одного листа на текущий лист

Представьте, что у нас есть отдельная электронная таблица Google, и нам нужно получить значение ячейки B2 в ячейке D5 на вашем текущем листе.

```
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
}
```

Вы можете найти идентификатор электронной таблицы из своего URL-адреса.

Получить последнюю строку в одном столбце

```
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;
```

```
    }
  }
}
```

Вставка массивов в виде строк

Вставка строки в нижней части таблицы легко:

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

Обратите внимание, что это добавит строку после последней *непустой* строки.

Вставка строки где-то посередине - это немного больше работы:

```
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!
```

Многое из этого бесполезного кода можно абстрагировать в вспомогательную функцию:

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

Это сводит наш пример к простому:

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

Прочитайте Обслуживание электронных таблиц онлайн: https://riptutorial.com/ru/google-apps-script/topic/2688/обслуживание-электронных-таблиц

глава 11: Приложения для скриптов приложений

замечания

Это примерное веб-приложение формы, бит на стороне клиента показывает некоторый базовый дизайн UX, такой как отключенная кнопка отправки при отправке формы или сообщение об ошибке, если она не работает ... и т. Д.

Битовый скрипт приложений очень прост. Он содержит только код, необходимый для обслуживания html, и для проверки поля.

Вот ссылка на это приложение в действии: Пример скрипта приложения

Примечание. Вы должны войти в учетную запись Google.

Структура файлов сценариев приложений выглядит так:

- Code.gs
- index.html
- Stylesheet.html
- JavaScript.html

Examples

Форма веб-приложения

Скрипт приложений:

```
//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.sucessfullySubmitted = 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>
```

Прочитайте Приложения для скриптов приложений онлайн: https://riptutorial.com/ru/google-apps-script/topic/4874/приложения-для-скриптов-приложений

глава 12: Скрипт Google Web App для автоматической загрузки с Google Диска

Вступление

Этот простой веб-скрипт Google App (автономный) позволяет Google Диску повторять опрос для файлов, которые будут загружены на локальный ПК пользователя. Показывает, как использовать один скрипт приложения для предоставления функции как: 1. Пользовательского интерфейса (простой в этом примере). 2. Страница загрузки файла. Для более полного объяснения того, как это работает, прочитайте пример «Как это работает».

замечания

Веб-скрипт должен быть опубликован для работы.

Для https://script.google.com необходимо включить Pops ups

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>
```

code.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')
  }
}
```

Как это устроено

Веб-приложение Google Drive (автономное) для автоматической загрузки (опроса) файлов с диска на локальный ПК пользователя (папка с загрузкой).

DriveApp предоставляет механизмы для поиска и загрузки файлов. Однако механизм загрузки имеет некоторые серьезные ограничения из-за архитектуры клиент-сервер, унаследованной Google Apps. (Не ошибка Google)

На стороне сервера DriveApp не предоставляется прямая функция для загрузки на локальный ПК, поскольку сервер не имеет понятия о том, где находится клиент, и загрузка файла на сервер будет бессмысленной.

На стороне сервера необходим механизм для предоставления клиентского кода файла данных или ссылки на файл. Оба этих механизма предоставляются, но данные из первого ограничены использованием кода клиента непосредственно. Клиент не имеет механизма сохранения данных после их получения на локальном ПК. Таким образом, его можно использовать для отображения данных на самой веб-странице.

Второй механизм позволяет вернуть URL-адрес скрипта (самого себя) или URL-адрес файла Drive. URL-адрес файла накопителя не очень полезен, поскольку его нельзя напрямую использовать в клиентском браузере для загрузки файла. Размещение этого url в привязке (и нажатии на него) приводит к открытию веб-страницы, но фактически ничего не делает (кроме, возможно, просмотра файла в Интернете).

Это оставляет URL-адрес скрипта. Однако скрипт url предоставляет только скрипт, а не файл.

Чтобы инициировать загрузку, файл из службы Диска должен быть возвращен из функции

doGet / doPost сценария на стороне сервера, используя ContentService createTextOutput точно так, как показано в онлайн-руководствах Google. Однако это означает, что на веб-странице не может быть другого элемента пользовательского интерфейса, сгенерированного результатами, возвращаемыми doGet / doPost.

Это оставляет нам очень непривлекательное решение. Пустая веб-страница без пользовательских элементов пользовательского интерфейса, которая загружает страницу, закрывается и требует ручного открытия, когда требуется другая загрузка.

Очевидно, что другая веб-страница для хостинга может предоставить пользовательский интерфейс и ссылку на сценарий загрузки веб-приложения для решения этой проблемы.

Этот сценарий использует подход д-ра Джекила и г-на Хайд для решения этой проблемы.

Если скрипт открывается без параметров в GET (doGet), он по умолчанию отображает форму. Это будет условие, когда опубликованное приложение сначала открывается пользователем. Форма, представленная в этом примере, чрезвычайно проста.

Если скрипт открывается с параметром servefile = true, скрипт ведет себя как загрузка файла диска.

Javascript на стороне клиента содержит механизм опроса (таймер события SetInterval), который периодически вызывает скрипт на стороне сервера, чтобы проверить наличие другого загружаемого файла.

Когда скрипт на стороне сервера выполняется, если он находит файл диска, соответствующий критериям поиска *, он возвращает URL-адрес самого скрипта с параметрами:

? Servefile = истина & ID = the_id_of_the_google_drive_file

(* Критерии поиска в этом простом примере жестко закодированы в сценарии на стороне сервера. При необходимости он может быть легко передан от клиента на сервер.)

Эта информация возвращается в виде строки клиенту через распознанный механизм SuccessHandler.

Клиентский Java-скрипт затем обновляет HREF скрытого якоря с помощью этой возвращенной информации, а затем автоматически нажимает на якорь.

Это приводит к запуску другого вызова приложения / скрипта. Когда новый вызов приложения запускает doGet, он обнаружит параметр servefile и вместо того, чтобы возвращать пользовательский интерфейс, он вернет файл в браузер. Возвращенный файл будет идентифицирован параметром ID, который был ранее возвращен описанным выше поиском.

Учитывая, что файл с предоставленным ID все еще существует, он будет загружен, и

новый вызов приложения закроется, оставив первый вызов, чтобы повторить этот процесс.

Кнопка предоставляется на простом интерфейсе, если пользователь / тестер становится нетерпеливым с ожиданием таймера, но он не требуется и в противном случае может быть удален.

Разумеется, простую форму можно расширить, чтобы обеспечить более удобный интерфейс пользователя, если это необходимо. Такие, как предоставление критериев поиска файлов.

Прочитайте Скрипт Google Web App для автоматической загрузки с Google Диска онлайн: https://riptutorial.com/ru/google-apps-script/topic/8212/скрипт-google-web-app-для-автоматической-загрузки-с-google-диска

глава 13: Служба DriveApp

замечания

Типы Mime Google не могут использоваться для третьего параметра Mime Types. Использование типа Mime Google приведет к ошибке:

Невозможно использовать «DriveApp.createFile ()» для создания типов MIME Google. Пожалуйста, используйте Advanced Drive Service

MimeType.GOOGLE_APPS_SCRIPT

MimeType.GOOGLE_DOCS

MimeType.GOOGLE_DRAWINGS

MimeType.GOOGLE_FORMS

MimeType.GOOGLE_SHEETS

MimeType.GOOGLE_SLIDES

Examples

Создайте новую папку в корневом каталоге 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
};
```

Создайте новый файл в Google Диске определенного типа 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
};
```

Создайте новый текстовый файл в папке с корневым диском 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
};
```

Создайте новый файл на Google Диске с помощью 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
};
```

Получить все папки - поместить папки в токен продолжения - затем извлечь из токена

```
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
};
```

Получите все файлы - поместите их в токен продолжения - затем извлеките их

```
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);
};
```

Прочитайте Служба DriveApp онлайн: https://riptutorial.com/ru/google-apps-script/topic/6395/служба-driveapp

глава 14: Служба DriveApp - файлы по типу и строке поиска

параметры

Имя параметра	Использовать для
SearchString	строка, которая будет найдена в имени файла

Examples

Получить файлы по типу файла с совпадающей строкой в имени файла

Получите все формы Google со словом «Без названия» в имени файла.

```
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;
};
```

Прочитайте Служба DriveApp - файлы по типу и строке поиска онлайн: https://riptutorial.com/ru/google-apps-script/topic/4049/служба-driveapp---файлы-по-типу-и-строке-поиска

глава 15: Создание пользовательской функции для Google Таблиц

Вступление

Пользовательская функция в документах google привязана к определенному документу (и, следовательно, может использоваться только в этом документе).

Поэтому он должен быть создан с помощью редактирования этого документа (Инструменты -> Редактор сценариев). После сохранения его можно использовать как любую другую формулу регулярных таблиц.

Examples

Стандартная гравитационная настраиваемая постоянная

Эта функция возвращает стандартную константу силы тяжести в указанных единицах ускорения (1 для см / с², 2 для ft / s², 3 для м / с²)

```
/**
 * 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;
}
```

Чтобы использовать эту функцию, ее необходимо привязать к электронной таблице с помощью редактора сценариев (Tools -> Script editor ...). Как только функция добавлена, ее можно использовать как любую другую функцию профайла google, вызывая функцию в

формуле ячейки.

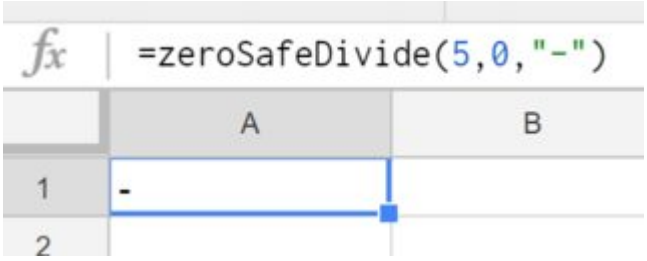
Обратите внимание, как функция отображается в автозаполнении при вводе в формулу. Это связано с многострочным комментарием над объявлением функции, которое используется для описания функции, аналогичной JSDoc и Javadoc. Чтобы формула отображалась в автозаполнении, тег @customfunction должен быть указан в комментарии.

Основной пример

Чтобы избежать неприглядных ошибок `#DIV/0` в электронной таблице, можно использовать настраиваемую функцию.

```
/**
 * 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;
}
```

Чтобы использовать эту функцию, ее необходимо привязать к электронной таблице с помощью редактора сценариев (**Tools -> Script editor ...**). Как только функция добавлена, ее можно использовать как любую другую функцию профайла google, вызывая функцию в формуле ячейки.



Обратите внимание, как функция отображается в автозаполнении при вводе в формулу. Это связано с многострочным комментарием над объявлением функции, которое используется для описания функции, аналогичной JSDoc и Javadoc. Чтобы формула отображалась в автозаполнении, тег `@customfunction` должен быть указан в комментарии.

Прочитайте Создание пользовательской функции для Google Таблиц онлайн: https://riptutorial.com/ru/google-apps-script/topic/5572/создание-пользовательской-функции-для-google-таблиц

кредиты

S. No	Главы	Contributors
1	Начало работы с скриптом google-apps	Albert Portnoy, Community, Douglas Gaskell, iJay, MShoaib91, Rubén, Saloni Vithalani, Shyam Kansagra, Spencer Easton, sudo bangbang, Supertopoz
2	DriveApp	Brian, Kos, nibarius, Sandy Good, Wolfgang
3	DriveApp - getFileById (id)	Sandy Good
4	Firebase и AppScript: Введение	Joseba, Vishal Vishwakarma
5	GmailApp	nibarius
6	Активный лист таблицы SpreadsheetApp	iJay
7	Клиент звонит в Google apps-script	Supertopoz
8	Листы Google MailApp	Bhupendra Piprava, Brian, Jordan Rhea, Kos, nibarius, Saloni Vithalani
9	Меню добавления таблиц	Bishal, iJay, nibarius
10	Обслуживание электронных таблиц	cdrini, iJay, nibarius, Sandy Good, sudo bangbang
11	Приложения для скриптов приложений	Douglas Gaskell
12	Скрипт Google Web App для автоматической загрузки с Google Диска	Walter

13	Служба DriveApp	Sandy Good
14	Служба DriveApp - файлы по типу и строке поиска	nibarius, Sandy Good
15	Создание пользовательской функции для Google Таблиц	Francky_V, Joshua Dawson, Pierre-Marie Richard, Rubén