Apps Script Coding Examples

Sending an email using Gmail service

```
function sendEmail() {
  const rep = "example@example.com";
  const sub = 'Hello World';
  const message = 'Hi, Laurence';
  GmailApp.sendEmail(rec, sub, message);
}

function mySender2() {
  const rep = "example@example.com";
  const sub = 'Hello World';
  const message = 'Hi, Laurence';
  MailApp.sendEmail(rep,sub,message);
```

```
}
```

This script uses the Gmail service to send an email to a specified recipient with a given subject and message. You can call this function from a Google Sheets, Forms, or Docs script. Using the MailApp service is a lighter way to send emails if all you are doing is sending emails using apps script.

Accessing and modifying data in a Google Sheet

```
function updater1(){
  const sheet = SpreadsheetApp.getActiveSheet();
  const val = 'Laurence Svekis';
  const range = sheet.getRange('A1');
  range.setValue(val);
  Logger.log(sheet);
}
```

Bound script example script updates the value in cell A1 of the active sheet in the current Google Sheet. You can use this script to update values in specific cells or ranges in a Google Sheet.

```
function updater1(){
  const id = '132trziyDvLsk8qIk';
  const sheet =
SpreadsheetApp.openById(id).getSheetByName('Sheet12');
  const val = 'Laurence Svekis 2';
  const range = sheet.getRange(1,4,2,2);
  range.setValue(val);
  const range2 = sheet.getRange(4,1,2,2);
  range2.setValues([[val,val],[val,val]]);
  Logger.log(sheet);
}
```

Standalone script examples, selecting the sheet, and then selecting the desired range to use. getRange(row, column, numRows, numColumns) If you use the setValue all the cells within the range will receive the same string value, if you want specific values for the cells within the range, use the setValues() method as it will set different values within the cells, using an array with nested arrays for each row.

Creating a new Google Calendar event

```
function createEvent() {
  var calendar =
CalendarApp.getCalendarById("calendar-id");
  var title = "Meeting";
  var start = new Date("February 19, 2023 9:00:00
GMT-05:00");
  var end = new Date("February 19, 2023 10:00:00
GMT-05:00");
  var event = calendar.createEvent(title, start, end);
}
```

This script creates a new event in a specified Google Calendar with a given title, start time, and end time. You can modify the values to create events at different times and with different titles.

Copying a Google Sheet to a new spreadsheet

```
function copySheet() {
  var sheet = SpreadsheetApp.getActiveSpreadsheet();
```

```
  var newSheet = sheet.copy("Copy of My Sheet");
  var newUrl = newSheet.getUrl();
}
```

This script copies the current Google Sheet to a new spreadsheet with a specified name. The new spreadsheet's URL is returned, so you can use it to access the new sheet.

Adding a custom menu to a Google Sheet

```
function onOpen() {
  var ui = SpreadsheetApp.getUi();
  ui.createMenu("Custom Menu")
      .addItem("Update Sheet", "updateSheet")
      .addItem("Send Email", "sendEmail")
      .addToUi();
}
```

This script adds a custom menu to the current Google Sheet. When the user clicks on the "Custom Menu" option, a dropdown menu appears with options to run the "updateSheet" and "sendEmail" functions.

Generating a PDF from a Google Doc

```
function createPDF() {
  var doc = DocumentApp.getActiveDocument();
  var pdf = DriveApp.createFile(doc.getAs('application/pdf'));
  var pdfUrl = pdf.getUrl();
}
```

This script generates a PDF version of the current Google Doc and creates a new file in Google Drive with the PDF content. The URL of the new file is returned, so you can use it to access the PDF.

Creating a new folder in Google Drive

```
function createFolder() {
  var folderName = "My Folder";
  var folder = DriveApp.createFolder(folderName);
  var folderUrl = folder.getUrl();
}
```

This script creates a new folder in Google Drive with a specified name. The URL of the new folder is returned, so you can use it to access the folder.

Creating a new Google Form

```
function createForm() {
  var form = FormApp.create('My Form');
  var formUrl = form.getPublishedUrl();
}
```

This script creates a new Google Form with the specified name and returns its published URL.

Accessing a Google Sheet by ID

```
function getSheetById() {
  var sheetId = "your-sheet-id";
  var sheet = SpreadsheetApp.openById(sheetId);
  var sheetName = sheet.getName();
}
```

This script opens a Google Sheet with the specified ID and returns its name.

Reading data from a Google Sheet

```
function readSheet() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('S
heet1');
  var data = sheet.getDataRange().getValues();
  var firstRow = data[0];
}
```

This script retrieves data from the specified sheet in the current Google Sheet and returns the values of the data range as a two-dimensional array. In this example, the first row of the data range is assigned to the firstRow variable.

Creating a new Google Slides presentation

```
function createSlides() {
  var presentation = SlidesApp.create('My
Presentation');
  var presentationUrl = presentation.getUrl();
}
```

This script creates a new Google Slides presentation with the specified name and returns its URL.

Adding a slide to a Google Slides presentation

```
function addSlide() {
  var presentation = SlidesApp.getActivePresentation();
  var slide = presentation.appendSlide();
  var slideIndex = slide.getIndex();
}
```

This script adds a new slide to the end of the current Google Slides presentation and returns its index.

Adding an image to a Google Slides slide

```
function addImage() {
  var presentation = SlidesApp.getActivePresentation();
  var slide = presentation.getSlides()[0];
  var imageUrl = "https://example.com/image.jpg";
  var image = slide.insertImage(imageUrl);
}
```

This script adds an image from the specified URL to the first slide of the current Google Slides presentation.

Reading data from an external API and writing it to a Google Sheet

```
function writeData() {
  var apiKey = "your-api-key";
  var url = "https://example.com/api?key=" + apiKey;
  var response = UrlFetchApp.fetch(url);
  var data = JSON.parse(response.getContentText());
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('S
heet1');
  sheet.getRange(1, 1, data.length,
data[0].length).setValues(data);
}
```

This script retrieves data from an external API, parses the JSON response, and writes the data to the specified sheet in the current Google Sheet.

Copying a file to a specific folder in Google Drive

```
function copyFile() {
  var fileId = "your-file-id";
  var folderId = "your-folder-id";
  var file = DriveApp.getFileById(fileId);
  var folder = DriveApp.getFolderById(folderId);
  var copy = file.makeCopy(file.getName(), folder);
}
```

This script copies a file with the specified ID to the specified folder in Google Drive.

Sending an email from a Google Sheet

```
function sendEmail() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName("S
heet1");
  var data = sheet.getDataRange().getValues();
  var emailAddress = "recipient@example.com";
  var message = "Hello, " + data[1][0] + "! This is an
automated message.";
  var subject = "Automated email";
  MailApp.sendEmail(emailAddress, subject, message);
```

```
}
```

This script retrieves data from a specified sheet in the current Google Sheet and sends an email to the specified recipient with the message and subject line.

Creating a new Google Document

```
function createDoc() {
  var doc = DocumentApp.create("My Document");
  var docUrl = doc.getUrl();
}
```

This script creates a new Google Document with the specified name and returns its URL.

Appending text to a Google Document

```
function appendText() {
  var doc = DocumentApp.getActiveDocument();
  var body = doc.getBody();
  body.appendParagraph("This is a new paragraph.");
}
```

This script appends a new paragraph to the end of the current Google Document.

Creating a new Google Form with multiple choice questions

```
function createForm() {
  var form = FormApp.create('My Form');
  var item = form.addMultipleChoiceItem();
  item.setTitle('What is your favorite color?')
      .setChoices([
        item.createChoice('Red'),
        item.createChoice('Blue'),
        item.createChoice('Green')
      ]);
  var formUrl = form.getPublishedUrl();
}
```

This script creates a new Google Form with a multiple choice question asking about a favorite color.

Creating a new Google Sheet and populating it with data

```
function createSheet() {
  var sheet = SpreadsheetApp.create("My Sheet");
  var data = [
    ["Name", "Age", "Location"],
    ["John", 32, "New York"],
    ["Mary", 25, "Chicago"],
    ["David", 43, "Los Angeles"]
  ];
  sheet.getRange(1, 1, data.length,
data[0].length).setValues(data);
  var sheetUrl = sheet.getUrl();
}
```

This script creates a new Google Sheet, populates it with the specified data, and returns its URL.

Sorting data in a Google Sheet

```
function sortData() {
```

```
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName("S
heet1");
  sheet.getRange("A2:C").sort(1);
}
```

This script sorts the data in columns A to C of the specified sheet in ascending order.

Creating a new Google Drive folder

```
function createFolder() {
  var folder = DriveApp.createFolder("My Folder");
  var folderUrl = folder.getUrl();
}
```

This script creates a new folder in Google Drive with the specified name and returns its URL.

Adding a file to a Google Drive folder

```
function addFileToFolder() {
  var fileId = "your-file-id";
  var folderId = "your-folder-id";
```

```
  var file = DriveApp.getFileById(fileId);
  var folder = DriveApp.getFolderById(folderId);
  folder.addFile(file);
}
```

This script adds a file with the specified ID to the specified folder in Google Drive.

Adding a trigger to run a function at a specific time

```
function addTrigger() {
  ScriptApp.newTrigger("myFunction")
    .timeBased()
    .atDate(2023, 3, 1)
    .create();
}
```

This script creates a new trigger that will run the myFunction function on March 1, 2023.

Copying data from one Google Sheet to another

```
function copyData() {
  var sourceSheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName("S
heet1");
  var targetSheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName("S
heet2");
  var sourceData =
sourceSheet.getDataRange().getValues();
  targetSheet.getRange(1, 1, sourceData.length,
sourceData[0].length).setValues(sourceData);
}
```

This script copies the data from Sheet1 to Sheet2 in the same Google Sheet.

Getting data from a Google Form response

```
function getFormResponse() {
  var form = FormApp.openByUrl("form-url");
```

```
  var responses = form.getResponses();
  var lastResponse = responses[responses.length - 1];
  var itemResponses = lastResponse.getItemResponses();
  var name = itemResponses[0].getResponse();
  var email = itemResponses[1].getResponse();
}
```

This script retrieves the name and email address from the most recent response to a Google Form.

Creating a new Google Slides presentation

```
function createSlides() {
  var presentation = SlidesApp.create("My
Presentation");
  var slide = presentation.appendSlide();
  slide.insertTextBox("Hello, world!", 100, 100);
  var presentationUrl = presentation.getUrl();
}
```

This script creates a new Google Slides presentation with a single slide that contains a text box.

Copying a Google Document to another user's Drive

```
function copyDocToDrive() {
  var docId = "document-id";
  var targetEmail = "target-email@example.com";
  var file = DriveApp.getFileById(docId);
  var targetFolder =
DriveApp.getFolderById(targetFolderId);
  var targetFile = file.makeCopy("Copy of My Document",
targetFolder);
  var targetUser =
UserManager.getUserByEmail(targetEmail);

DriveApp.getFolderById(targetFolderId).addViewer(target
User);
  MailApp.sendEmail(targetEmail, "Document copied",
"Here is a copy of My Document: " +
targetFile.getUrl());
}
```

This script makes a copy of a Google Document and shares it with the specified user via email.

Creating a new Google Forms quiz

```
function createQuiz() {
  var form = FormApp.create('My Quiz');
  var item = form.addCheckboxItem();
  item.setTitle('What is the capital of France?')
      .setPoints(1)
      .setChoices([
        item.createChoice('Paris', true),
        item.createChoice('London'),
        item.createChoice('Berlin')
      ]);
  form.setIsQuiz(true);
  form.setConfirmationMessage("Thanks for taking the quiz!");
  var formUrl = form.getPublishedUrl();
}
```

This script creates a new Google Forms quiz with a multiple choice question about the capital of France and sets the correct answer to Paris.

Adding a custom menu to a Google Sheets

```
function onOpen() {
  var ui = SpreadsheetApp.getUi();
  ui.createMenu("My Menu")
    .addItem("Item 1", "myFunction1")
    .addItem("Item 2", "myFunction2")
    .addSeparator()
    .addSubMenu(ui.createMenu("Submenu")
      .addItem("Subitem 1", "mySubFunction1")
      .addItem("Subitem 2", "mySubFunction2"))
    .addToUi();
}
```

This script adds a custom menu called "My Menu" to a Google Sheets document. The menu includes two top-level items, "Item 1" and "Item 2", and a submenu called "Submenu" with two sub-items.

Setting the font family of a range of cells in a Google Sheets document

```
function setFontFamily() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  var range = sheet.getRange(1, 1, 5, 5);
```

```
  range.setFontFamily("Arial");
}
```

This script sets the font family of a range of cells in a Google Sheets document to "Arial".

Creating a new Google Calendar event

```
function createEvent() {
  var calendar = CalendarApp.getDefaultCalendar();
  var title = "My Event";
  var startTime = new Date("2023-03-01T09:00:00Z");
  var endTime = new Date("2023-03-01T10:00:00Z");
  var event = calendar.createEvent(title, startTime,
endTime);
}
```

This script creates a new event in the user's default Google Calendar with the title "My Event" that starts at 9:00 AM and ends at 10:00 AM on March 1, 2023.

Deleting all rows in a Google Sheets document that match a certain condition

```
function deleteRows() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  var data = sheet.getDataRange().getValues();
  for (var i = data.length - 1; i >= 0; i--) {
    if (data[i][0] == "Delete") {
      sheet.deleteRow(i + 1);
    }
  }
}
```

This script deletes all rows in a Google Sheets document that have the value "Delete" in the first column.

Sending an email from a Google Sheets document

javascript

Copy code

```
function sendEmail() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  var data = sheet.getDataRange().getValues();
  var emailAddress = "example@gmail.com";
  var subject = "My Subject";
  var message = "My Message";
  MailApp.sendEmail(emailAddress, subject, message);
```

```
}
```

This script sends an email with the subject "My Subject" and message "My Message" to the email address "example@gmail.com" using the MailApp service.

Creating a new Google Forms response

```
function createFormResponse() {
  var form = FormApp.openById("FORM_ID");
  var response = form.createResponse();
  var item = form.getItemById(ITEM_ID);
  var answer = item.asTextItem().createResponse("My
Answer");
  response.withItemResponse(answer);
  response.submit();
}
```

This script creates a new response to a Google Form with the ID "FORM_ID", sets the answer to a text item with the ID "ITEM_ID" to "My Answer", and submits the response using the FormApp service.

Importing data from a CSV file to a Google Sheets document

```
function importCSV() {
  var file = DriveApp.getFileById("FILE_ID");
  var csvData =
Utilities.parseCsv(file.getBlob().getDataAsString());
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  sheet.getRange(1, 1, csvData.length,
csvData[0].length).setValues(csvData);
}
```

This script imports data from a CSV file with the ID "FILE_ID" in Google Drive, parses the data using the Utilities service, and sets the values in the active sheet of the current Google Sheets document using the SpreadsheetApp service.

Copying a file in Google Drive to a different folder

```
function copyFile() {
  var file = DriveApp.getFileById("FILE_ID");
  var folder = DriveApp.getFolderById("FOLDER_ID");
  var copy = file.makeCopy("Copy of " + file.getName(),
folder);
}
```

This script makes a copy of a file with the ID "FILE_ID" in Google Drive, renames the copy to "Copy of [original file name]", and moves the copy to a different folder with the ID "FOLDER_ID" using the DriveApp service.

Creating a new Google Document

```
function createDocument() {
  var title = "My Document";
  var body = "My Document Content";
  var doc = DocumentApp.create(title);
  doc.getBody().setText(body);
}
```

This script creates a new Google Document with the title "My Document" and body content "My Document Content" using the DocumentApp service.

Setting the background color of a range of cells in a Google Sheets document

```
function setBackgroundColor() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  var range = sheet.getRange(1, 1, 5, 5);
  range.setBackground("#FF0000");
}
```

This script sets the background color of a range of cells in a Google Sheets document to red using the setBackgroundColor method of the Range class.

Exporting a Google Sheets document to a PDF file

```
function exportToPDF() {
  var spreadsheet =
SpreadsheetApp.getActiveSpreadsheet();
  var sheet = spreadsheet.getActiveSheet();
  var url = spreadsheet.getUrl();
```

```
  url = url.replace(/edit$/, '') +
'export?exportFormat=pdf&format=pdf&gid=' +
sheet.getSheetId();
  var token = ScriptApp.getOAuthToken();
  var response = UrlFetchApp.fetch(url, {
    headers: {
      'Authorization': 'Bearer ' + token
    }
  });
  var pdfBlob = response.getBlob();
  var folder = DriveApp.getFolderById("FOLDER_ID");
  var file =
folder.createFile(pdfBlob).setName(spreadsheet.getName(
) + ".pdf");
}
```

This script exports the active sheet of the current Google Sheets document to a PDF file, saves the file to a folder with the ID "FOLDER_ID" in Google Drive, and names the file after the name of the Google Sheets document.

Creating a new Google Calendar event

```
function createEvent() {
```

```
  var title = "My Event";
  var description = "My Event Description";
  var startTime = new
Date("2023-02-20T09:00:00-08:00");
  var endTime = new Date("2023-02-20T10:00:00-08:00");
  var event =
CalendarApp.getDefaultCalendar().createEvent(title,
startTime, endTime, {
    description: description
  });
}
```

This script creates a new event on the default Google Calendar with the title "My Event", description "My Event Description", start time of 9:00 AM and end time of 10:00 AM on February 20, 2023, using the CalendarApp service.

Updating a row of data in a Google Sheets document

```
function updateRow() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  var data = sheet.getDataRange().getValues();
```

```
  for (var i = 0; i < data.length; i++) {
    if (data[i][0] == "John Doe") {
      sheet.getRange(i+1, 2).setValue("Updated Value");
    }
  }
}
```

This script updates a row of data in a Google Sheets document by searching for the name "John Doe" in the first column, and setting the value of the corresponding cell in the second column to "Updated Value", using the getDataRange and getRange methods of the Sheet class.

Sending an email with a PDF attachment

```
function sendEmailWithPDF() {
  var emailAddress = "recipient@example.com";
  var subject = "PDF Attachment";
  var body = "Please see the attached PDF file.";
  var pdfFile = DriveApp.getFileById("PDF_FILE_ID");
  var pdfBlob = pdfFile.getBlob();
  MailApp.sendEmail(emailAddress, subject, body,
{attachments: [pdfBlob]});
}
```

This script sends an email to the specified recipient with the subject "PDF Attachment" and body "Please see the attached PDF file". It attaches a PDF file with the ID "PDF_FILE_ID" from Google Drive to the email, using the MailApp and DriveApp services.

Creating a new Google Slides presentation

```
function createPresentation() {
  var title = "My Presentation";
  var slides = SlidesApp.create(title);
  var slide1 = slides.getSlides()[0];
  var shape1 =
slide1.insertShape(SlidesApp.ShapeType.RECTANGLE, 50,
50, 100, 100);
  shape1.getFill().setSolidFill("#4285F4");
  var shape2 =
slide1.insertShape(SlidesApp.ShapeType.ELLIPSE, 200,
50, 100, 100);
  shape2.getFill().setSolidFill("#EA4335");
  var shape3 =
slide1.insertShape(SlidesApp.ShapeType.CHEVRON, 350,
50, 100, 100);
```

```
  shape3.getFill().setSolidFill("#FBBC05");
}
```

This script creates a new Google Slides presentation with the title "My Presentation", using the SlidesApp service. It then inserts three shapes onto the first slide of the presentation, sets the size and position of each shape, and sets the fill color of each shape using the setSolidFill method of the Fill class.