Ultimate Apps Script Questions with Solutions Guide

How can I use Google Apps Script to send emails from a Google Sheet?

```
function sendEmails() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var startRow = 2; // Assumes header row is row 1
  var numRows = sheet.getLastRow() - 1; // Number of
rows to process
  var dataRange = sheet.getRange(startRow, 1, numRows,
4) // Assuming 4 columns of data
  var data = dataRange.getValues();
  for (var i = 0; i < data.length; i++) {
    var row = data[i];
    var emailAddress = row[0];
    var subject = row[1];
    var message = row[2];
    var cc = row[3];
    MailApp.sendEmail(emailAddress, subject, message,
{cc: cc});
  }
}
```

Explanation: This script fetches data from a Google Sheet and sends an email to each recipient. It assumes that the email address is in the first column, the subject is in the second

column, the message is in the third column, and the CC email address is in the fourth column.

How can I add a timestamp to a Google Sheet when a cell is edited?

```
function onEdit(e) {
  var sheet = e.source.getActiveSheet();
  var range = e.range;
  if (sheet.getName() === 'Sheet1' && range.getColumn()
=== 1) {
    range.offset(0, 1).setValue(new Date());
  }
}
```

Explanation: This script adds a timestamp to the adjacent cell in the same row when a cell in column A is edited. It uses the onEdit trigger to detect the edit event and the offset method to write the timestamp to the adjacent cell.

How can I retrieve data from an external API using Google Apps Script?

```
function getWeather() {
  var url =
'https://api.openweathermap.org/data/2.5/weather?q=Lond
on,uk&appid=<your-api-key>';
  var response = UrlFetchApp.fetch(url);
  var json = response.getContentText();
```

```
  var data = JSON.parse(json);
  var temperature = data.main.temp - 273.15; // Convert
from Kelvin to Celsius
  Logger.log('Temperature in London is ' +
temperature.toFixed(1) + '°C');
}
```

Explanation: This script retrieves the current temperature in London from the OpenWeatherMap API and logs it to the execution transcript. It uses the UrlFetchApp.fetch method to send a GET request to the API endpoint, parses the response as JSON, and extracts the temperature value from the data.

How can I delete rows from a Google Sheet based on a condition?

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

Explanation: This script deletes rows from the active sheet if the first cell in the row is blank. It iterates over the rows in reverse order to avoid the issue of row indices changing during deletion.

How can I generate random numbers in Google Apps Script?

```
function generateRandomNumbers() {
  var numbers = [];
  for (var i = 0; i < 10; i++) {
    var random = Math.random() * 100;
    numbers.push(random);
  }
  Logger.log(numbers);
}
```

You can generate random numbers in Google Apps Script using the Math.random() method. This method returns a random number between 0 and 1. You can use this method to generate random integers or floats within a specified range.

Here's an example code snippet that generates a random integer between 1 and 10:

```
function generateRandomInt() {
  var min = 1;
  var max = 10;
  var randomInt = Math.floor(Math.random() * (max - min
+ 1)) + min;
  Logger.log(randomInt);
}
```

In this example, we define the minimum and maximum values of the range we want to generate random integers from. We then use the Math.floor() method to round down the result of Math.random() * (max - min + 1), which will generate a random

float between min and max, inclusive. Finally, we add min to this value to ensure that the result is within the specified range.

If you wanted to generate a random float between 0 and 1, you could simply use the Math.random() method by itself:

```
function generateRandomFloat() {
  var randomFloat = Math.random();
  Logger.log(randomFloat);
}
```

These are just a couple of examples of how you can use the Math.random() method to generate random numbers in Google Apps Script.

How can I send an email from Google Sheets using Google Apps Script?

```
function sendEmail() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var startRow = 2; // First row of data to process
  var numRows = 1; // Number of rows to process
  var dataRange = sheet.getRange(startRow, 1, numRows,
2);
  var data = dataRange.getValues();
  var emailAddress = data[0][0]; // First column
  var message = data[0][1]; // Second column
  var subject = "Sending emails from Google Sheets";
  MailApp.sendEmail(emailAddress, subject, message);
}
```

How can I sort data in Google Sheets using Google Apps Script?

```
function sortData() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getRange("A2:B10"); // Range to
sort
  range.sort({column: 1, ascending: true}); // Sort by
column A, ascending
}
```

How can I add a new sheet to a Google Spreadsheet using Google Apps Script?

```
function addSheet() {
  var spreadsheet =
SpreadsheetApp.getActiveSpreadsheet();
  var newSheet = spreadsheet.insertSheet("New Sheet");
  newSheet.getRange("A1").setValue("New Sheet");
}
```

How can I format cells in a Google Spreadsheet using Google Apps Script?

```
function formatCells() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getRange("A1:B10"); // Range to
format
```

```
range.setFontSize(12).setHorizontalAlignment("center").
setBorder(true, true, true, true, true, true);
}
```

How can I create a custom menu in a Google Spreadsheet using Google Apps Script?

```
function onOpen() {
  var ui = SpreadsheetApp.getUi();
  ui.createMenu("Custom Menu")
    .addItem("Sort Data", "sortData")
    .addItem("Format Cells", "formatCells")
    .addSeparator()
    .addItem("Send Email", "sendEmail")
    .addToUi();
}
```

How can I set a trigger to run a function in Google Apps Script?

```
function setTrigger() {
  ScriptApp.newTrigger("myFunction")
    .timeBased()
    .atHour(12)
    .everyDays(1)
    .create();
}
```

How can I create a new folder in Google Drive using Google Apps Script?

```
function createFolder() {
  var folderName = "New Folder";
  var folder = DriveApp.createFolder(folderName);
  Logger.log("Folder URL: " + folder.getUrl());
}
```

How can I get the last row of data in a Google Spreadsheet using Google Apps Script?

```
function getLastRow() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var lastRow = sheet.getLastRow();
  Logger.log("Last Row: " + lastRow);
}
```

How can I get the current user's email address in Google Apps Script?

```
function getUserEmail() {
  var userEmail = Session.getActiveUser().getEmail();
  Logger.log("User Email: " + userEmail);
}
```

How can I format cells in a Google Sheet using Google Apps Script?

```
function formatCells() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A1:C5');
  range.setBackground('#ffffff');
  range.setFontSize(12);
  range.setHorizontalAlignment('center');
  range.setVerticalAlignment('middle');
}
```

Explanation: This code formats cells in a Google Sheet using various formatting functions of the Range object. The getRange() function is used to get a range object representing cells A1:C5. The setBackground() function is used to set the background color of the cells to white. The setFontSize() function is used to set the font size of the cells to 12. The setHorizontalAlignment() function is used to set the horizontal alignment of the cells to center, and the setVerticalAlignment() function is used to set the vertical alignment of the cells to middle.

How can I insert an image into a Google Sheet using Google Apps Script?

```
function insertImage() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
```

```
  var image =
UrlFetchApp.fetch('https://www.example.com/image.jpg');
  sheet.insertImage(image, 1, 1);
}
```

Explanation: This code inserts an image into a Google Sheet using the insertImage() function of the Sheet object. The getSheetByName() function is used to get a reference to the sheet where the image should be inserted. The UrlFetchApp.fetch() function is used to fetch the image from a URL and store it in the image variable. Finally, the insertImage() function is used to insert the image into cell A1 of the sheet.

How can I add a new sheet to a Google Sheet using Google Apps Script?

```
function addSheet() {
  var spreadsheet =
SpreadsheetApp.getActiveSpreadsheet();
  spreadsheet.insertSheet('New Sheet');
}
```

Explanation: This code adds a new sheet to a Google Sheet using the insertSheet() function of the Spreadsheet object. The getActiveSpreadsheet() function is used to get a reference to the currently active spreadsheet. The insertSheet() function is used to insert a new sheet with the name 'New Sheet'.

How can I get the value of a cell in a Google Sheet using Google Apps Script?

```
function getCellValue() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var value = sheet.getRange('A1').getValue();
  Logger.log('Cell value: ' + value);
}
```

Explanation: This code gets the value of a cell in a Google Sheet using the getValue() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the cell. The getRange() function is used to get a range object representing cell A1. Finally, the getValue() function is used to get the value of the cell, which is logged to the execution log using the Logger.log() function.

How can I set the value of a cell in a Google Sheet using Google Apps Script?

```
function setCellValue() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A1');
  range.setValue('New Value');
}
```

Explanation: This code sets the value of a cell in a Google Sheet using the setValue() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the cell. The getRange() function is used to get a range object representing cell A1. Finally, the setValue() function is used to set the value of the cell to 'New Value'.

How can I get the last row in a Google Sheet using Google Apps Script?

```
function getLastRow() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var lastRow = sheet.getLastRow();
  Logger.log('Last row: ' + lastRow);
}
```

Explanation: This code gets the last row in a Google Sheet using the getLastRow() function of the Sheet object. The getSheetByName() function is used to get a reference to the sheet containing the data. Finally, the getLastRow() function is used to get the index of the last row with data in it, which is logged to the execution log using the Logger.log() function.

How can I get the last column in a Google Sheet using Google Apps Script?

```
function getLastColumn() {
```

```
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var lastColumn = sheet.getLastColumn();
  Logger.log('Last column: ' + lastColumn);
}
```

Explanation: This code gets the last column in a Google Sheet using the getLastColumn() function of the Sheet object. The getSheetByName() function is used to get a reference to the sheet containing the data. Finally, the getLastColumn() function is used to get the index of the last column with data in it, which is logged to the execution log using the Logger.log() function.

How can I get all the values in a row of a Google Sheet using Google Apps Script?

```
function getRowValues() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var values = sheet.getRange('A1:D1').getValues()[0];
  Logger.log('Row values: ' + values);
}
```

Explanation: This code gets all the values in a row of a Google Sheet using the getValues() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the row of data (in this case, row 1). The [0] at the end of the getValues() function call is used to extract the values from the array of arrays returned by the

function (since we only want one row of data). Finally, the values are logged to the execution log using the Logger.log() function.

How can I get all the values in a column of a Google Sheet using Google Apps Script?

```
function getColumnValues() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var values =
sheet.getRange('A1:A4').getValues().map(function(row) {
    return row[0];
  });
  Logger.log('Column values: ' + values);
}
```

Explanation: This code gets all the values in a column of a Google Sheet using the getValues() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the column of data (in this case, column A). The map() function is used to extract the values from the array of arrays returned by the getValues() function (since we only want one column of data). Finally, the values are logged to the execution log using the Logger.log() function.

How can I clear the contents of a range in a Google Sheet using Google Apps Script?

```
function clearRange() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A1:D4');
  range.clearContent();
}
```

Explanation: This code clears the contents of a range in a Google Sheet using the clearContent() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the range of data to be cleared. Finally, the clearContent() function is used to clear the contents of the range.

How can I format the background color of a range in a Google Sheet using Google Apps Script?

```
function setRangeBackgroundColor() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A1:D4');
  range.setBackground('#ffffff'); // set to white
```

```
}
```

Explanation: This code formats the background color of a range in a Google Sheet using the setBackground() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the range of data to be formatted. Finally, the setBackground() function is used to set the background color of the range to white.

How can I format the font size of a range in a Google Sheet using Google Apps Script?

```
function setRangeFontSize() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A1:D4');
  range.setFontSize(14); // set to 14 point font
}
```

Explanation: This code formats the font size of a range in a Google Sheet using the setFontSize() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the range of data to be formatted. Finally, the setFontSize() function is used to set the font size of the range to 14 points.

How can I format the boldness of a range in a Google Sheet using Google Apps Script?

```
function setRangeBoldness() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A1:D4');
  range.setFontWeight('bold'); // set to bold
}
```

Explanation: This code formats the boldness of a range in a Google Sheet using the setFontWeight() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the range of data to be formatted. Finally, the setFontWeight() function is used to set the boldness of the range to bold.

How can I format the alignment of text in a range in a Google Sheet using Google Apps Script?

```
function setRangeTextAlignment() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A1:D4');
```

```
  range.setHorizontalAlignment('center'); // center
align
  range.setVerticalAlignment('middle'); // middle align
}
```

Explanation: This code formats the alignment of text in a range in a Google Sheet using the setHorizontalAlignment() and setVerticalAlignment() functions of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the range of data to be formatted. Finally, the setHorizontalAlignment() function is used to set the horizontal alignment of the range to center, and the setVerticalAlignment() function is used to set the vertical alignment of the range to middle.

How can I freeze rows or columns in a Google Sheet using Google Apps Script?

```
function freezeRowsAndColumns() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  sheet.setFrozenRows(1); // freeze the first row
  sheet.setFrozenColumns(2); // freeze the first two
columns
}
```

Explanation: This code freezes rows or columns in a Google Sheet using the setFrozenRows() and setFrozenColumns() functions of the Sheet object. The getSheetByName() function is used to get a reference to the sheet containing the data. The setFrozenRows()

function is used to freeze the first row of the sheet, and the setFrozenColumns() function is used to freeze the first two columns of the sheet.

How can I sort a range in a Google Sheet using Google Apps Script?

```
function sortRange() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A2:C10');
  range.sort(1); // sort by the first column in
ascending order
}
```

Explanation: This code sorts a range in a Google Sheet using the sort() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the range of data to be sorted. Finally, the sort() function is used to sort the range by the first column in ascending order.

How can I filter a range in a Google Sheet using Google Apps Script?

```
function filterRange() {
```

```
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A1:C10');
  var criteria =
SpreadsheetApp.newFilterCriteria().whenTextStartsWith('
A').build();
  var filter =
range.createFilter().setColumnFilterCriteria(1,
criteria); // filter by the first column starting with
"A"
}
```

Explanation: This code filters a range in a Google Sheet using the createFilter() and setColumnFilterCriteria() functions of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the range of data to be filtered. The newFilterCriteria() function is used to create a filter criteria object that specifies the text that the first column should start with. Finally, the createFilter() function is used to create a filter based on the filter criteria object, and the setColumnFilterCriteria() function is used to set the filter for the first column of the range.

How can I copy data from one sheet to another in a Google Sheet using Google Apps Script?

To copy data from one sheet to another in a Google Sheet using Google Apps Script, you can use the copyTo() function of the Range object. Here is an example code:

```
function copyData() {
  var sourceSheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('S
ource Sheet');
  var sourceRange = sourceSheet.getRange('A1:B10');
  var targetSheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('T
arget Sheet');
  var targetRange = targetSheet.getRange('C1');
  sourceRange.copyTo(targetRange);
}
```

Explanation:

1. The getSheetByName() function is used to get a reference to the source sheet containing the data that you want to copy.
2. The getRange() function is used to get a reference to the range of cells that you want to copy from the source sheet.
3. The getSheetByName() function is used to get a reference to the target sheet where you want to paste the data.
4. The getRange() function is used to get a reference to the cell in the target sheet where you want to paste the data.
5. The copyTo() function is used to copy the data from the source range to the target range.

Note: The copyTo() function can also be used with other options like formatOnly, contentsOnly, and destinationSpreadsheetId.

How can I send an email from a Google Sheet using Google Apps Script?

```
function sendEmail() {
```

```
  var recipient = 'recipient@example.com';
  var subject = 'Subject of the email';
  var body = 'Body of the email';
  MailApp.sendEmail(recipient, subject, body);
}
```

Explanation: This code sends an email from a Google Sheet using the sendEmail() function of the MailApp object. The recipient, subject, and body variables are used to specify the recipient email address, subject, and body of the email, respectively. The sendEmail() function is then used to send the email.

How can I send an email with an attachment from a Google Sheet using Google Apps Script?

```
function sendEmailWithAttachment() {
  var recipient = 'recipient@example.com';
  var subject = 'Subject of the email';
  var body = 'Body of the email';
  var file = DriveApp.getFileById('fileId'); // replace
fileId with the ID of the file to be attached
  MailApp.sendEmail({
    to: recipient,
    subject: subject,
    body: body,
    attachments: [file.getAs(MimeType.PDF)]
  });
}
```

Explanation: This code sends an email with an attachment from a Google Sheet using the sendEmail() function of the MailApp

object. The recipient, subject, and body variables are used to specify the recipient email address, subject, and body of the email, respectively. The DriveApp.getFileById() function is used to get a reference to the file to be attached, and the getAs() function is used to specify the MIME type of the file. Finally, the sendEmail() function is used to send the email with the attachment.

How can I create a new Google Doc from a Google Sheet using Google Apps Script?

```
function createNewDoc() {
  var docName = 'Name of the new Google Doc';
  var doc = DocumentApp.create(docName);
  var body = doc.getBody();
  body.appendParagraph('This is the first paragraph of
the new Google Doc.');
}
```

Explanation: This code creates a new Google Doc from a Google Sheet using the create() function of the DocumentApp object. The docName variable is used to specify the name of the new Google Doc. The create() function is then used to create the new Google Doc, and the getBody() function is used to get a reference to the body of the document. Finally, the appendParagraph() function is used to add a new paragraph to the body of the document.

How can I add a new row to a Google Sheet using Google Apps Script?

```
function addNewRow() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var values = ['Value 1', 'Value 2', 'Value 3'];
  sheet.appendRow(values);
}
```

Explanation: This code adds a new row to a Google Sheet using the appendRow() function of the Sheet object. The getSheetByName() function is used to get a reference to the sheet where the new row should be added. The values variable is used to specify the values to be added to the new row. Finally, the appendRow() function is used to add the new row to the sheet.

How can I send an email with Google Apps Script?

```
function sendEmail() {
  var recipient = "example@gmail.com";
  var subject = "Test Email";
  var body = "This is a test email sent with Google
Apps Script!";
  MailApp.sendEmail(recipient, subject, body);
}
```

Explanation: The sendEmail() function uses the MailApp class to send an email. The recipient's email address, subject, and body of

the email are defined as variables and passed as parameters to the sendEmail() method.

How can I set a cell value in Google Sheets using Google Apps Script?

```
function setCellValue() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('S
heet1');
  sheet.getRange('A1').setValue('Hello, world!');
}
```

Explanation: The setCellValue() function uses the getActiveSpreadsheet() method to get a reference to the active spreadsheet, and getSheetByName() to get a reference to a specific sheet. Then, the setValue() method is used to set the value of the cell A1 to "Hello, world!".

How can I get the last row of a sheet in Google Sheets using Google Apps Script?

```
function getLastRow() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('S
heet1');
  var lastRow = sheet.getLastRow();
  Logger.log(lastRow);
}
```

Explanation: The getLastRow() function uses the getActiveSpreadsheet() method to get a reference to the active spreadsheet, and getSheetByName() to get a reference to a specific sheet. The getLastRow() method is used to get the index of the last row in the sheet.

How can I get the current date and time using Google Apps Script?

```
function getCurrentDateTime() {
  var now = new Date();
  Logger.log(now);
}
```

Explanation: The getCurrentDateTime() function creates a new Date object, which represents the current date and time. The Logger.log() method is used to print the date and time to the log.

How can I create a new sheet in a Google Sheet using Google Apps Script?

```
function createNewSheet() {
  var spreadsheet =
SpreadsheetApp.getActiveSpreadsheet();
  spreadsheet.insertSheet('New Sheet');
}
```

Explanation: The createNewSheet() function uses the getActiveSpreadsheet() method to get a reference to the active

spreadsheet, and insertSheet() to create a new sheet with the specified name.

How can I delete a sheet in a Google Sheet using Google Apps Script?

```
function deleteSheet() {
  var spreadsheet =
SpreadsheetApp.getActiveSpreadsheet();
  var sheet = spreadsheet.getSheetByName('Sheet1');
  spreadsheet.deleteSheet(sheet);
}
```

Explanation: The deleteSheet() function uses the getActiveSpreadsheet() method to get a reference to the active spreadsheet, and getSheetByName() to get a reference to a specific sheet. The deleteSheet() method is used to delete the sheet.

How can I get the values of a range of cells in Google Sheets using Google Apps Script?

```
function getValues() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('S
heet1');
  var range = sheet.getRange('A1:B10');
  var values = range.getValues();
```

```
  Logger.log(values);
}
```

Explanation: The getValues() function uses the getActiveSpreadsheet() method to get a reference to the active spreadsheet

How can I sort data in a Google Sheet using Google Apps Script?

```
function sortData() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('S
heet1');
  var range = sheet.getRange('A2:D11');
  range.sort([{column: 1, ascending: true}, {column: 2,
ascending: true}]);
}
```

Explanation: The sortData() function uses the getActiveSpreadsheet() method to get a reference to the active spreadsheet, and getSheetByName() to get a reference to a specific sheet. The getRange() method is used to get a range of cells, and the sort() method is used to sort the data in ascending order based on two columns: column 1 and column 2.

How can I format cells in a Google Sheet using Google Apps Script?

```
function formatCells() {
```

```
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('S
heet1');
  var range = sheet.getRange('A2:D11');
  range.setBackground('#FFDAB9');
  range.setFontSize(14);
  range.setFontWeight('bold');
}
```

Explanation: The formatCells() function uses the getActiveSpreadsheet() method to get a reference to the active spreadsheet, and getSheetByName() to get a reference to a specific sheet. The getRange() method is used to get a range of cells, and the setBackground(), setFontSize(), and setFontWeight() methods are used to format the cells with a background color, font size, and font weight.

How can I use a custom function in Google Sheets using Google Apps Script?

```
function myFunction(input) {
  return input.toUpperCase();
}
```

Explanation: The myFunction() function is a custom function that can be used in Google Sheets. The function takes an input parameter and returns the uppercase version of the input. To use the function in a Google Sheet, simply enter "=myFunction(A1)" in a cell, where A1 is the cell with the input value.

How can I format the font color of a range in a Google Sheet using Google Apps Script?

```
function setRangeFontColor() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getRange('A1:D4');
  range.setFontColor('#ff0000'); // set to red
}
```

Explanation: This code formats the font color of a range in a Google Sheet using the setFontColor() function of the Range object. The getSheetByName() function is used to get a reference to the sheet containing the data. The getRange() function is used to get a range object representing the range of data to be formatted. Finally, the setFontColor() function is used to set the font color of the range to red.

How can I copy a sheet from one Google Sheet to another using Google Apps Script?

```
function copySheet() {
  var ss = SpreadsheetApp.getActiveSpreadsheet();
  var sourceSheet = ss.getSheetByName('Source Sheet');
  var targetSpreadsheet =
SpreadsheetApp.openById('Target Spreadsheet ID');
  sourceSheet.copyTo(targetSpreadsheet);
}
```

Explanation: This code copies a sheet named 'Source Sheet' from the active Google Sheet to a target Google Sheet with the

specified ID ('Target Spreadsheet ID') using the copyTo() function of the sheet object.

How can I protect a range of cells in a Google Sheet using Google Apps Script?

```
function protectRange() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getRange('A1:B5');
  var protection =
range.protect().setDescription('Protected Range');
  protection.addEditor('user@example.com');
  protection.removeEditors(protection.getEditors());
  if (protection.canDomainEdit()) {
    protection.setDomainEdit(false);
  }
}
```

Explanation: This code protects a range of cells (A1:B5) in the active sheet of a Google Sheet using the protect() function of the range object. The setDescription() function is used to set a description for the protection. The addEditor() function is used to add a user (user@example.com) to the list of editors who can edit the protected range. The removeEditors() function is used to remove all editors except the specified user. The canDomainEdit() function is used to check if the protection allows editing by users in the domain. If so, the setDomainEdit() function is used to disable domain editing.

How can I get the current date and time in a Google Sheet using Google Apps Script?

```
function getCurrentDateTime() {
  var date = new Date();
  var sheet = SpreadsheetApp.getActiveSheet();
  sheet.getRange('A1').setValue(date);
}
```

Explanation: This code gets the current date and time using the Date() function, and then sets the value of cell A1 in the active sheet of a Google Sheet to the current date and time using the setValue() function of the range object.

How can I sort a range of cells in a Google Sheet using Google Apps Script?

```
function sortRange() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getRange('A2:C6');
  range.sort(1); // Sort by column 1 (ascending)
}
```

Explanation: This code sorts a range of cells (A2:C6) in the active sheet of a Google Sheet in ascending order based on the values in column 1 (the first column) using the sort() function of the range object.

How can I get the values of all cells in a range in a Google Sheet using Google Apps Script?

```
function getRangeValues() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getRange('A1:C3');
  var values = range.getValues();
  Logger.log(values);
}
```

Explanation: This code gets the values of all cells in a range (A1:C3) in the active sheet of a Google Sheet using the getValues() function of the range object. The Logger.log() function is used to log the values to the execution log.

How can I format a range of cells in a Google Sheet using Google Apps Script?

```
function formatRange() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getRange('A1:C3');
  range.setBackground('red');
  range.setFontColor('white');
  range.setFontWeight('bold');
  range.setHorizontalAlignment('center');
  range.setVerticalAlignment('middle');
}
```

Explanation: This code formats a range of cells (A1:C3) in the active sheet of a Google Sheet using various formatting functions of the range object. The setBackground() function is used to set

the background color of the range to red. The setFontColor() function is used to set the font color of the range to white. The setFontWeight() function is used to set the font weight of the range to bold. The setHorizontalAlignment() function is used to set the horizontal alignment of the range to center. The setVerticalAlignment() function is used to set the vertical alignment of the range to middle.

How can I get the email addresses of all members in a Google Group using Google Apps Script?

```
function getGroupMembers() {
  var group =
GroupsApp.getGroupByEmail('example@googlegroups.com');
  var members = group.getUsers();
  for (var i = 0; i < members.length; i++) {
    Logger.log(members[i].getEmail());
  }
}
```

Explanation: This code gets the email addresses of all members in a Google Group (example@googlegroups.com) using the getGroupByEmail() function of the Groups service. The getUsers() function is used to get an array of user objects, and then a for loop is used to iterate over the array and log the email addresses of the users to the execution log using the getEmail() function of the user object.

How can I create a new calendar event in Google Calendar using Google Apps Script?

```
function createEvent() {
  var calendar = CalendarApp.getDefaultCalendar();
  var event = calendar.createEvent('New Event',
    new Date('2023-04-12T10:00:00Z'),
    new Date('2023-04-12T11:00:00Z'),
    {description: 'Event Description', location: 'Event
Location'});
  Logger.log('Event ID: ' + event.getId());
}
```

Explanation: This code creates a new calendar event in the default Google Calendar using the createEvent() function of the Calendar service. The createEvent() function takes four arguments: the event title ('New Event'), the start time of the event (new Date('2023-04-12T10:00:00Z')), the end time of the event (new Date('2023-04-12T11:00:00Z')), and an object containing additional parameters for the event ({description: 'Event Description', location: 'Event Location'}). The getId() function is used to get the ID of the created event, which is then logged to the execution log.

How can I send an email using Gmail in Google Apps Script?

```
function sendEmail() {
  var recipient = 'recipient@example.com';
  var subject = 'Email Subject';
```

```
  var body = 'Email Body';
  GmailApp.sendEmail(recipient, subject, body);
}
```

Explanation: This code sends an email to a recipient ('recipient@example.com') using the sendEmail() function of the Gmail service. The sendEmail() function takes three arguments: the email recipient, the email subject, and the email body.

How can I create a new Google Document using Google Apps Script?

```
function createDocument() {
  var document = DocumentApp.create('New Document');
  var body = document.getBody();
  var paragraph = body.appendParagraph('Hello,
world!');

paragraph.setHeading(DocumentApp.ParagraphHeading.HEADI
NG1);
  Logger.log('Document URL: ' + document.getUrl());
}
```

Explanation: This code creates a new Google Document using the create() function of the Document service. The create() function takes a single argument, which is the title of the new document ('New Document'). The getBody() function is used to get the body of the new document, and the appendParagraph() function is used to add a new paragraph to the body with the text 'Hello, world!'. The setHeading() function is used to set the heading of the paragraph to HEADING1. Finally, the getUrl() function is used

to get the URL of the new document, which is then logged to the execution log.

How can I create a new Google Slide using Google Apps Script?

```
function createSlide() {
  var presentation = SlidesApp.create('New
Presentation');
  var slide = presentation.getSlides()[0];
  var shape =
slide.insertShape(SlidesApp.ShapeType.RECTANGLE);

shape.setWidth(200).setHeight(200).setTop(100).setLeft(
100);
  Logger.log('Presentation URL: ' +
presentation.getUrl());
}
```

Explanation: This code creates a new Google Slide using the create() function of the Slides service. The create() function takes a single argument, which is the title of the new presentation ('New Presentation'). The getSlides() function is used to get an array of slide objects, and the [0] index is used to select the first slide in the array. The insertShape() function is used to insert a rectangle shape onto the slide, and the setWidth(), setHeight(), setTop(), and setLeft() functions are used to position and size the shape. Finally, the getUrl() function is used to get the URL of the new presentation, which is then logged to the execution log.

How can I get the current date and time using Google Apps Script?

```
function getCurrentDateTime() {
  var now = new Date();
  Logger.log('Current Date and Time: ' +
now.toLocaleString());
}
```

Explanation: This code gets the current date and time using the Date() function, which creates a new Date object with the current date and time. The toLocaleString() function is used to convert the date object to a string representation of the date and time in the local timezone, which is then logged to the execution log.

How can I add a menu to a Google Sheet using Google Apps Script?

```
function onOpen() {
  var ui = SpreadsheetApp.getUi();
  ui.createMenu('Custom Menu')
    .addItem('Menu Item 1', 'menuItem1')
    .addItem('Menu Item 2', 'menuItem2')
    .addToUi();
}

function menuItem1() {
  Logger.log('Menu Item 1 selected');
}
```

```
function menuItem2() {
  Logger.log('Menu Item 2 selected');
}
```

Explanation: This code adds a custom menu to a Google Sheet using the createMenu() function of the Spreadsheet service. The createMenu() function takes a single argument, which is the name of the new menu ('Custom Menu'). The addItem() function is used to add two menu items to the new menu, with the labels 'Menu Item 1'

How can I send an email from a Google Sheet using Google Apps Script?

```
function sendEmail() {
  var email = 'recipient@example.com';
  var subject = 'Test Email';
  var body = 'This is a test email sent from Google
Apps Script.';
  MailApp.sendEmail(email, subject, body);
  Logger.log('Email sent to ' + email);
}
```

Explanation: This code sends an email using the sendEmail() function of the Mail service. The sendEmail() function takes three arguments: the recipient email address ('recipient@example.com'), the subject of the email ('Test Email'), and the body of the email ('This is a test email sent from Google Apps Script.'). Finally, the Logger.log() function is used to log a message to the execution log indicating that the email was sent.

How can I copy a Google Sheet using Google Apps Script?

```
function copySheet() {
  var originalSheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('O
riginal Sheet');
  var destinationSheet =
originalSheet.copyTo(SpreadsheetApp.getActiveSpreadshee
t());
  destinationSheet.setName('Copied Sheet');
  Logger.log('Sheet copied to ' +
destinationSheet.getName());
}
```

Explanation: This code copies a Google Sheet using the copyTo() function of the Sheet object. The copyTo() function takes a single argument, which is the destination spreadsheet object (SpreadsheetApp.getActiveSpreadsheet() in this case). The setName() function is used to set the name of the copied sheet to 'Copied Sheet'. Finally, the Logger.log() function is used to log a message to the execution log indicating that the sheet was copied.

How can I get data from a web API using Google Apps Script?

```
function getApiData() {
  var url = 'https://api.example.com/data';
  var response = UrlFetchApp.fetch(url);
```

```
  var data = JSON.parse(response.getContentText());
  Logger.log('Data: ' + JSON.stringify(data));
}
```

Explanation: This code retrieves data from a web API using the fetch() function of the UrlFetch service. The fetch() function takes a single argument, which is the URL of the API endpoint ('https://api.example.com/data' in this case). The getContentText() function is used to get the content of the response as a string, which is then parsed as JSON using the JSON.parse() function. Finally, the Logger.log() function is used to log a message to the execution log containing the data as a JSON string.

How can I read data from a Google Sheet using Google Apps Script?

```
function readData() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getDataRange();
  var values = range.getValues();
  for (var i = 0; i < values.length; i++) {
    Logger.log('Row ' + (i+1) + ': ' +
values[i].join(', '));
  }
}
```

Explanation: This code reads data from a Google Sheet using the getDataRange() and getValues() functions of the Sheet object. The getDataRange() function is used to get a range object

representing all the data in the sheet, and the getValues() function is used to get a 2D array of the values in the range. The for loop is used to iterate over each row of the array and log the values to the execution log.

How can I delete rows from a Google Sheet using Google Apps Script?

```
function deleteRows() {
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('D
ata Sheet');
  var range = sheet.getDataRange();
  var values = range.getValues();
  for (var i = values.length - 1; i >= 0; i--) {
    if (values[i][0] == 'delete') {
      sheet.deleteRow(i+1);
    }
  }
}
```

Explanation: This code deletes rows from a Google Sheet based on a condition using the deleteRow() function of the Sheet object. The getDataRange() and getValues() functions are used to get a 2D array of the values in the sheet. The for loop is used to iterate over each row of the array, starting from the last row, and delete the row if the first column contains the string 'delete'.

How can I get the URL of the current Google Sheet using Google Apps Script?

```
function getSheetUrl() {
  var sheet = SpreadsheetApp.getActiveSpreadsheet();
  var url = sheet.getUrl();
  Logger.log('Sheet URL: ' + url);
}
```

Explanation: This code gets the URL of the current Google Sheet using the getUrl() function of the Spreadsheet object. The getActiveSpreadsheet() function is used to get the currently active spreadsheet object. Finally, the Logger.log() function is used to log a message to the execution log containing the sheet URL.

How can I add a menu to a Google Sheet using Google Apps Script?

```
function onOpen() {
  var ui = SpreadsheetApp.getUi();
  ui.createMenu('Custom Menu')
      .addItem('Menu Item 1', 'menuItem1')
      .addItem('Menu Item 2', 'menuItem2')
      .addToUi();
}

function menuItem1() {
  // Do something
}
```

```
function menuItem2() {
  // Do something else
}
```

Explanation: This code adds a custom menu to a Google Sheet using the createMenu() and addItem() functions of the Ui service. The onOpen() function is a reserved function that runs automatically when the sheet is opened. The createMenu() function is used to create a new menu with the name 'Custom Menu'. The addItem() function is used to add two menu items to the menu with the labels 'Menu Item 1' and 'Menu Item 2', and the functions 'menuItem1' and 'menuItem2' as their respective actions. Finally, the addToUi() function is used to add the menu to the UI of the sheet.

How can I create a menu in Google Sheets using Google Apps Script?

```
function onOpen() {
  var ui = SpreadsheetApp.getUi();
  ui.createMenu('Custom Menu')
      .addItem('Menu Item 1', 'menuItem1')
      .addItem('Menu Item 2', 'menuItem2')
      .addToUi();
}

function menuItem1() {
  // Code for menu item 1
}

function menuItem2() {
```

```
  // Code for menu item 2
}
```

Explanation: This code creates a custom menu in the Google Sheet using the onOpen() function. The getUi() function returns the user interface for the spreadsheet, and the createMenu() function creates a new menu with the specified name (Custom Menu). The addItem() function adds menu items to the menu, with the first parameter being the name of the menu item and the second parameter being the function to be executed when the menu item is clicked. The addToUi() function adds the menu to the user interface.

How can I protect a range in a Google Sheet using Google Apps Script?

```
function protectRange() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getRange('A1:B10');
  var protection =
range.protect().setDescription('Protected Range');
  protection.removeEditors(protection.getEditors());
  if (protection.canDomainEdit()) {
    protection.setDomainEdit(false);
  }
}
```

Explanation: This code protects a range of cells (A1:B10) in the active sheet of a Google Sheet. The getActiveSheet() function returns the active sheet, and the getRange() function returns the range of cells to be protected. The protect() function protects the range and returns a protection object, which is stored in the

protection variable. The setDescription() function sets a description for the protection. The removeEditors() function removes all editors from the protection, and the canDomainEdit() function checks if the domain is allowed to edit the protection. If the domain is allowed to edit the protection, the setDomainEdit() function sets the domain to not be able to edit the protection.

How can I get the current date and time using Google Apps Script?

```
function getCurrentDateTime() {
  var now = new Date();
  Logger.log(now);
}
```

Explanation: This code gets the current date and time using the Date() constructor and logs it to the execution log using the Logger.log() function.

How can I copy a sheet within a Google Sheet using Google Apps Script?

```
function copySheet() {
  var ss = SpreadsheetApp.getActiveSpreadsheet();
  var sheetToCopy = ss.getSheetByName('Sheet1');
  var newSheet = sheetToCopy.copyTo(ss).setName('Copy
of Sheet1');
}
```

Explanation: This code copies a sheet (Sheet1) in the active Google Sheet to a new sheet with the name Copy of Sheet1. The

getActiveSpreadsheet() function returns the active spreadsheet, the getSheetByName() function returns the sheet to be copied, and the copyTo() function copies the sheet to the active spreadsheet and returns the new sheet. The setName() function sets the name of the new sheet.

How can I get the last row of data in a Google Sheet using Google Apps Script?

```
function getLastRow() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getDataRange();
  var lastRow = range.getLastRow();
  Logger.log(lastRow);
}
```

Explanation: This code gets the last row of data in the active sheet of a Google Sheet. The getActiveSheet() function returns the active sheet, and the getDataRange() function returns the range of cells that contain data in the sheet. The getLastRow() function returns the last row of the range, which is the last row of data in the sheet. The Logger.log() function logs the last row to the execution log.

How can I send an email from a Google Sheet using Google Apps Script?

```
function sendEmail() {
  var recipient = 'recipient@example.com';
```

```
  var subject = 'Email Subject';
  var body = 'Email Body';
  MailApp.sendEmail(recipient, subject, body);
}
```

Explanation: This code sends an email to a recipient with the specified email address (recipient@example.com), subject (Email Subject), and body (Email Body) using the sendEmail() function of the MailApp class.

How can I add a new row of data to a Google Sheet using Google Apps Script?

```
function addRow() {
  var sheet = SpreadsheetApp.getActiveSheet();
  sheet.appendRow(['New Value 1', 'New Value 2']);
}
```

Explanation: This code adds a new row to the active sheet of a Google Sheet with the values New Value 1 in column A and New Value 2 in column B using the appendRow() function of the sheet object.

How can I get the value of a cell in a Google Sheet using Google Apps Script?

```
function getCellValue() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var cell = sheet.getRange('A1');
```

```
  var value = cell.getValue();
  Logger.log(value);
}
```

Explanation: This code gets the value of cell A1 in the active sheet of a Google Sheet. The getActiveSheet() function returns the active sheet, and the getRange() function returns the cell at the specified coordinates. The getValue() function returns the value of the cell, which is stored in the value variable. The Logger.log() function logs the value to the execution log.

How can I delete a row of data from a Google Sheet using Google Apps Script?

```
function deleteRow() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var rowToDelete = 2;
  sheet.deleteRow(rowToDelete);
}
```

Explanation: This code deletes the second row of data from the active sheet of a Google Sheet using the deleteRow() function of the sheet object.

How can I add a custom function to a Google Sheet using Google Apps Script?

```
function myFunction(input) {
  return input * 2;
}
```

Explanation: This code defines a custom function named myFunction that takes one input parameter and returns the input value multiplied by 2. This function can be used in a Google Sheet by typing =myFunction(A1) in a cell, where A1 is the cell containing the input value.

How can I create a custom menu in a Google Sheet using Google Apps Script?

```
function onOpen() {
  var ui = SpreadsheetApp.getUi();
  ui.createMenu('Custom Menu')
      .addItem('Menu Item 1', 'menuItem1')
      .addSeparator()
      .addSubMenu(ui.createMenu('Submenu')
          .addItem('Submenu Item 1', 'submenuItem1'))
      .addToUi();
}

function menuItem1() {
  Browser.msgBox('You clicked Menu Item 1!');
}

function submenuItem1() {
  Browser.msgBox('You clicked Submenu Item 1!');
}
```

Explanation: This code adds a custom menu to a Google Sheet when it is opened. The onOpen() function creates the custom menu, which has one menu item (Menu Item 1) and one submenu item (Submenu Item 1). When the user clicks on Menu Item 1, the menuItem1() function is called, which displays a

message box. When the user clicks on Submenu Item 1, the submenuItem1() function is called, which also displays a message box.

How can I get the value of a cell in a Google Sheet using Google Apps Script?

```
function getCellValue() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var cell = sheet.getRange("A1");
  var value = cell.getValue();
  Logger.log(value);
}
```

Explanation: This code gets the value of cell A1 in the active sheet of a Google Sheet. The getActiveSheet() function returns the active sheet, the getRange() function returns a range of cells (in this case, cell A1), and the getValue() function returns the value of the cell. The value of the cell is logged to the execution log using the Logger.log() function.

How can I set the value of a cell in a Google Sheet using Google Apps Script?

```
function setCellValue() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var cell = sheet.getRange("A1");
  cell.setValue("Hello, World!");
}
```

Explanation: This code sets the value of cell A1 in the active sheet of a Google Sheet to "Hello, World!". The getActiveSheet() function returns the active sheet, the getRange() function returns a range of cells (in this case, cell A1), and the setValue() function sets the value of the cell.

How can I get the values of multiple cells in a Google Sheet using Google Apps Script?

```
function getCellValues() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getRange("A1:B2");
  var values = range.getValues();
  Logger.log(values);
}
```

Explanation: This code gets the values of cells A1 to B2 in the active sheet of a Google Sheet. The getActiveSheet() function returns the active sheet, the getRange() function returns a range of cells (in this case, cells A1 to B2), and the getValues() function returns a two-dimensional array of values.

How can I write data to a Google Sheet using Google Apps Script?

```
function writeToSheet() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var values = [
    ["John", "Doe", "johndoe@example.com"],
```

```
    ["Jane", "Doe", "janedoe@example.com"]
  ];
  sheet.getRange(sheet.getLastRow() + 1, 1,
values.length, values[0].length).setValues(values);
}
```

Explanation This code writes data to the active sheet of a Google Sheet. The getActiveSheet() function returns the active sheet, and the setValues() function sets the values of a range of cells. In this example, the values are stored in a two-dimensional array and the setValues() function sets the values of a range starting from the last row of the sheet (using getLastRow() function) to the length of the values array in both rows and columns.

How can I search for a specific value in a Google Sheet using Google Apps Script?

```
function searchSheet() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getDataRange();
  var values = range.getValues();
  var searchValue = "John";
  for (var i = 0; i < values.length; i++) {
    for (var j = 0; j < values[i].length; j++) {
      if (values[i][j] == searchValue) {
        var cell = sheet.getRange(i+1, j+1);
        Logger.log(cell.getA1Notation());
      }
    }
  }
}
```

Explanation: This code searches for a specific value (searchValue) in the active sheet of a Google Sheet. The getActiveSheet() function returns the active sheet, the getDataRange() function returns the range of all data in the sheet, and the getValues() function returns a two-dimensional array of values. The code then loops through each value in the array and checks if it matches the search value. If a match is found, the code gets the cell of the matching value using the getRange() function and logs the cell's A1 notation to the execution log using the Logger.log() function.

How can I create a new sheet in a Google Sheet using Google Apps Script?

```
function createSheet() {
  var ss = SpreadsheetApp.getActiveSpreadsheet();
  var sheetName = "New Sheet";
  var sheet = ss.insertSheet(sheetName);
}
```

Explanation: This code creates a new sheet in the active Google Sheet. The getActiveSpreadsheet() function returns the active spreadsheet, and the insertSheet() function creates a new sheet with the specified name (sheetName) and returns the sheet object.

How can I delete a sheet in a Google Sheet using Google Apps Script?

```
function deleteSheet() {
```

```
  var ss = SpreadsheetApp.getActiveSpreadsheet();
  var sheetName = "Sheet to delete";
  var sheet = ss.getSheetByName(sheetName);
  ss.deleteSheet(sheet);
}
```

Explanation: This code deletes a sheet with a specific name (sheetName) from the active Google Sheet. The getActiveSpreadsheet() function returns the active spreadsheet, the getSheetByName() function returns the sheet with the specified name, and the deleteSheet() function deletes the sheet.

How can I send an email using Google Apps Script?

```
function sendEmail() {
  var recipient = "johndoe@example.com";
  var subject = "Test email";
  var body = "This is a test email sent using Google
Apps Script!";
  MailApp.sendEmail(recipient, subject, body);
}
```

Explanation: This code sends an email using the MailApp service in Google Apps Script. The sendEmail() function takes three parameters: the email recipient, the email subject, and the email body.

How can I use a Google Sheets formula in Google Apps Script?

You can use Google Sheets formulas in Google Apps Script by using the getRange() function to get a range of cells, and then using the setFormula() function to set the formula for the range. Here's an example:

Suppose you have a Google Sheet with a table of data that includes a column of numbers in column B. You want to add a new column C that calculates the square of each number in column B using the POWER() function.

Here's how you can use a Google Sheets formula in Google Apps Script to accomplish this:

```
function addColumn() {
  var sheet = SpreadsheetApp.getActiveSheet();
  var range = sheet.getRange('C2:C');
  range.setFormula('=POWER(B2,2)');
}
```

Explanation:

The getActiveSheet() function returns the active sheet of the Google Sheet.

The getRange() function is used to select the range of cells where the formula should be applied. In this case, we want to apply the formula to column C starting at row 2 ('C2:C').

The setFormula() function is used to set the formula for the selected range. In this case, we're using the POWER() function to square the value in each cell in column B. The formula is written as a string in the function argument, and it must start with an equal sign (=) to indicate that it's a formula.

When you run this script, it will add a new column C to the sheet with the square of each value in column B. If you add new values to column B, the formula will automatically update to calculate the square of the new values.

How can I send an email using Google Apps Script?

You can send an email using Google Apps Script by calling the MailApp.sendEmail() method. Here's an example code snippet:

```
function sendEmail() {
  var recipient = "example@email.com";
  var subject = "Test Email";
  var body = "This is a test email.";
  MailApp.sendEmail(recipient, subject, body);
}
```

In this example, we define the recipient's email address, the email subject, and the email body. We then call the MailApp.sendEmail() method with these parameters to send the email.

How can I create a new folder in Google Drive using Google Apps Script?

You can create a new folder in Google Drive using Google Apps Script by calling the DriveApp.createFolder() method. Here's an example code snippet:

```
function createFolder() {
  var folderName = "New Folder";
  var folder = DriveApp.createFolder(folderName);
  Logger.log("Created folder with ID: " +
folder.getId());
}
```

In this example, we define the name of the new folder and call the DriveApp.createFolder() method with this parameter to create the folder. We then log the ID of the new folder using the Logger.log() method.

How can I create a new file in Google Drive using Google Apps Script?

You can create a new file in Google Drive using Google Apps Script by calling the DriveApp.createFile() method. Here's an example code snippet:

```
function createFile() {
  var fileName = "New File";
  var fileContent = "This is a test file.";
  var file = DriveApp.createFile(fileName,
fileContent);
  Logger.log("Created file with ID: " + file.getId());
```

```
}
```

In this example, we define the name and content of the new file and call the DriveApp.createFile() method with these parameters to create the file. We then log the ID of the new file using the Logger.log() method.

How can I get the current date and time in Google Apps Script?

You can get the current date and time in Google Apps Script using the new Date() method. Here's an example code snippet:

```
function getCurrentDateTime() {
  var currentDate = new Date();
  Logger.log(currentDate);
}
```

In this example, we call the new Date() method to create a new Date object with the current date and time. We then log this object using the Logger.log() method.

How can I set a timer in Google Apps Script?

You can set a timer in Google Apps Script using the Utilities.sleep() method. Here's an example code snippet:

```
function setTimer() {
  var timeInMilliseconds = 5000; // 5 seconds
  Utilities.sleep(timeInMilliseconds);
  Logger.log("Timer finished.");
}
```

In this example, we define the duration of the timer in milliseconds and call the Utilities.sleep() method with this parameter to pause the script for the specified amount of time. We then log a message to indicate that the timer has finished.