Google Apps Script Examples 6

Example 1: Add a new column to a Google Sheet

This code will add a new column to a Google Sheet.

```
function addColumn() {
  // Get the sheet object.
  var sheet = SpreadsheetApp.getActiveSpreadsheet();
```

```
  // Get the current sheet name.
  var sheetName = sheet.getName();

  // Get the column index of the last column.
  var lastColumnIndex = sheet.getMaxColumns();

  // Add a new column at the specified index.
  sheet.insertColumn(lastColumnIndex + 1);

  // Set the column name.
  sheet.getRange(lastColumnIndex + 1, 1).setValue('New
Column');
}
```

This code works by first getting the sheet object. Then, it gets the current sheet name and the column index of the last column. Finally, it adds a new column at the specified index and sets the column name.

This code could be used to add a new column to a Google Sheet for any purpose. For example, you could use this code to add a new column to a Google Sheet that stores the names and contact information of your customers.

Example 2: Sort the rows in a Google Sheet

This code will sort the rows in a Google Sheet.

```
function sortRows() {
  // Get the sheet object.
```

```
  var sheet = SpreadsheetApp.getActiveSpreadsheet();

  // Get the column index of the column to sort by.
  var columnIndex = 2;

  // Sort the rows in ascending order.
  sheet.sortRange(1, 1, sheet.getMaxRows(),
columnIndex, true);
}
```

This code works by first getting the sheet object. Then, it gets the column index of the column to sort by and sorts the rows in ascending order.

This code could be used to sort the rows in a Google Sheet for any purpose. For example, you could use this code to sort the rows in a Google Sheet that stores the names and contact information of your customers by their last name.

Example 3: Filter the rows in a Google Sheet

This code will filter the rows in a Google Sheet.

```
function filterRows() {
  // Get the sheet object.
  var sheet = SpreadsheetApp.getActiveSpreadsheet();

  // Get the column index of the column to filter by.
  var columnIndex = 2;

  // Get the value to filter by.
```

```
  var value = 'John Doe';

  // Filter the rows that match the specified value.
  sheet.filterRange(1, 1, sheet.getMaxRows(),
columnIndex, value);
}
```

This code works by first getting the sheet object. Then, it gets the column index of the column to filter by and the value to filter by and filters the rows that match the specified value.

This code could be used to filter the rows in a Google Sheet for any purpose. For example, you could use this code to filter the rows in a Google Sheet that stores the names and contact information of your customers by their last name.

Example 4: Create a pivot table in a Google Sheet

This code will create a pivot table in a Google Sheet.

```
function createPivotTable() {
  // Get the sheet object.
  var sheet = SpreadsheetApp.getActiveSpreadsheet();

  // Get the range of data to include in the pivot
table.
  var range = sheet.getRange('A1:C10');

  // Create a new pivot table.
  var pivotTable = sheet.newPivotTable(range);
```

```
  // Set the column headers for the pivot table.
  pivotTable.addColumnHeader('Name');
  pivotTable.addColumnHeader('Email');
  pivotTable.addColumnHeader('Phone');

  // Set the row headers for the pivot table.
  pivotTable.addRowHeader('Country');

  // Set the values for the pivot table.
  pivotTable.setValue('Count', 'Count');

  // Display the pivot table.
  pivotTable.show();
}
```

This code works by first getting the sheet object. Then, it gets the range of data to include in the pivot table and creates a new pivot table. Finally, it sets the column headers, the row headers, the values, and displays the pivot table.

Example 5: Sending Emails with Attachments

Automatically Send Emails with Attachments: This code automatically sends emails with attachments to a list of recipients. It first retrieves the files to be attached from a specified folder in Google Drive. It then loops through the list of recipients and sends them an email with the attachment(s).

```
function sendEmailWithAttachments() {
  var folder = DriveApp.getFolderById("folder_id"); //
specify folder ID
```

```
  var files = folder.getFiles();

  while (files.hasNext()) {
    var file = files.next();
    var blob = file.getBlob();

    var recipients = ['email1@example.com',
'email2@example.com']; // specify recipient email
addresses
    var subject = 'Email Subject';
    var body = 'Email Body';

    for (var i = 0; i < recipients.length; i++) {
      GmailApp.sendEmail(recipients[i], subject, body,
{attachments: [blob]});
    }
  }
}
```

The DriveApp service is used to access the specified folder and retrieve the files within it. The while loop iterates through each file and retrieves its Blob object. The for loop then sends an email to each recipient, attaching the Blob object as an attachment.

Example 6: Merge Multiple sheets into one

Merge Multiple Sheets into One: This code merges multiple sheets into one sheet. It first retrieves all the sheets in the specified spreadsheet and loops through them. For each sheet, it copies its data and pastes it into a new sheet.

```
function mergeSheets() {
```

```
  var ss = SpreadsheetApp.openById('spreadsheet_id');
// specify spreadsheet ID
  var sheets = ss.getSheets();
  var masterSheet = ss.insertSheet('Master Sheet');

  for (var i = 0; i < sheets.length; i++) {
    var sheet = sheets[i];
    var range = sheet.getDataRange();
    var values = range.getValues();

    if (i == 0) {

masterSheet.getRange(range.getA1Notation()).setValues(v
alues);
    } else {
      var lastRow = masterSheet.getLastRow();
      masterSheet.getRange(lastRow + 1, 1,
values.length, values[0].length).setValues(values);
    }
  }
}
```

The SpreadsheetApp service is used to access the specified spreadsheet and retrieve all its sheets. The for loop iterates through each sheet and retrieves its data. If it is the first sheet, its data is directly copied to the new sheet. If it is not the first sheet, its data is appended to the last row of the new sheet.