Elevating Your Google Apps Script Skills: 5 Practical Exercises

Exercise 1: Simple Google Sheets Manipulation

Task:

Create a Google Apps Script function that calculates the sum of values in a specified column in Google Sheets.

Code:

```
function calculateSum() {
  // Specify the sheet name and column
  var sheetName = "Sheet1";
  var columnName = "A";

  // Access the active sheet
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName(sheet
Name);
```

```
  // Get the values in the specified column
  var values = sheet.getRange(columnName + "1:" +
columnName + sheet.getLastRow()).getValues();

  // Calculate the sum
  var sum = values.reduce(function(acc, current) {
    return acc + Number(current[0]);
  }, 0);

  // Log the sum
  Logger.log("Sum of values in column " + columnName + ": " +
sum);
}
```

Explanation:

1. Specify the sheet name and column to target.
2. Access the active sheet using SpreadsheetApp.
3. Get the values in the specified column using getRange.
4. Use the reduce function to calculate the sum.
5. Log the sum using Logger.log.

Exercise 2: Send Email from Gmail

Task:

Create a Google Apps Script function that sends an email using Gmail.

Code:

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

  // Send email
  GmailApp.sendEmail(recipient, subject, body);
}
```

Explanation:

1. Specify the recipient, subject, and body of the email.
2. Use GmailApp.sendEmail to send the email.

Exercise 3: Google Calendar Event Creation

Task:

Create a Google Apps Script function that adds an event to Google Calendar.

Code:

```
function createCalendarEvent() {
  var calendarId = "your_calendar_id@example.com";
```

```
  var eventTitle = "Meeting with Client";
  var eventDate = new Date("2024-01-01T10:00:00");
  var eventDuration = 60; // in minutes

  // Create event
  var calendar = CalendarApp.getCalendarById(calendarId);
  var event = calendar.createEvent(eventTitle, eventDate, new
Date(eventDate.getTime() + (eventDuration * 60 * 1000)));

  Logger.log("Event created: " + event.getTitle());
}
```

Explanation:

1. Specify the calendar ID, event title, date, and duration.
2. Access the calendar using CalendarApp.
3. Create an event using createEvent.

Exercise 4: Google Drive File Upload

Task:

Create a Google Apps Script function that uploads a file to Google Drive.

Code:

```
function uploadFileToDrive() {
  var folderId = "your_folder_id";
```

```
  var fileName = "SampleFile.txt";
  var fileContent = "This is a sample file content.";

  // Create file in Drive
  var folder = DriveApp.getFolderById(folderId);
  var file = folder.createFile(fileName, fileContent);

  Logger.log("File uploaded: " + file.getName());
}
```

Explanation:

1. Specify the folder ID, file name, and file content.
2. Access the folder using DriveApp.
3. Create a file in the folder using createFile.

Exercise 5: Google Forms Response Handling

Task:

Create a Google Apps Script function that logs responses from a Google Form.

Code:

```
function onFormSubmit(e) {
  var responses = e.values;
  var timestamp = responses[0];
  var name = responses[1];
```

```
  var email = responses[2];

  // Log the form responses
  Logger.log("Timestamp: " + timestamp);
  Logger.log("Name: " + name);
  Logger.log("Email: " + email);
}
```

Explanation:

1. Use the onFormSubmit trigger function to handle form submissions.
2. Access form responses using the e.values object.
3. Extract specific data from the responses.
4. Log the form responses using Logger.log.
5. These exercises cover various Google Apps Script functionalities. Make sure to replace placeholders like email addresses, folder IDs, and calendar IDs with your actual values when testing these scripts.