Google Apps Script Examples 8

Example 1: Create a new Google Form

This code will create a new Google Form.

```
function createForm() {
  // Create a new Google Form.
  var form = FormApp.create('My Form');

  // Add a question to the form.
  var question = form.addQuestion('What is your
name?');
  question.setRequired(true);

  // Add another question to the form.
  var question = form.addQuestion('What is your email
address?');
  question.setRequired(true);

  // Save the form.
  form.save();
}
```

This code works by first creating a new Google Form object. Then, it adds two questions to the form, one for the user's name and one for the user's email address. Finally, it saves the form.

This code could be used to create a form that users can fill out to provide their contact information. For example, you could use this code to create a form that users can fill out to sign up for your newsletter or to request a consultation.

Example 2: Send an email notification to the user

This code will send an email notification to the user.

```
function sendEmail() {
  // Get the user's email address.
  var userEmail = Utilities.getUserEmail();

  // Create an email message.
  var message = GmailApp.createEmail();
  message.setSubject('Your email notification');
  message.setBody('This is the body of your email
notification.');

  // Send the email notification.
  message.sendTo(userEmail);
}
```

This code works by first getting the user's email address. Then, it creates an email message and sets the subject line and body of the message. Finally, it sends the email notification.

This code could be used to send email notifications to users about important events or updates. For example, you could use this code to send email notifications to users when their account is created or when their password is changed.

Example 3: Create a Google Sheet

This code will create a new Google Sheet.

```
function createSheet() {
  // Create a new Google Sheet.
```

```
  var sheet = SpreadsheetApp.create('My Sheet');

  // Add a row to the sheet.
  var row = sheet.appendRow([1, 2, 3]);

  // Add another row to the sheet.
  var row = sheet.appendRow([4, 5, 6]);

  // Save the sheet.
  sheet.save();
}
```

This code works by first creating a new Google Sheet object. Then, it adds two rows to the sheet, one with the values 1, 2, and 3 and one with the values 4, 5, and 6. Finally, it saves the sheet.

This code could be used to create a Google Sheet that stores data. For example, you could use this code to create a Google Sheet that stores the names and contact information of your customers.

Example 4: Get the current time

This code will get the current time.

```
function getCurrentTime() {
  // Get the current time.
  var currentTime = Utilities.now();

  // Display the current time.
  Logger.log('The current time is: ' + currentTime);
```

```
}
```

This code works by first getting the current time using the Utilities.now() method. Then, it displays the current time using the Logger.log() method.

This code could be used to get the current time for any purpose. For example, you could use this code to get the current time to display on a website or to log the current time in a database.

Example 5: Get the user's location

This code will get the user's location.

```
function getUserLocation() {
  // Get the user's location.
  var userLocation = Geolocation.getGeolocation();

  // Display the user's location.
  Logger.log('The user's location is: ' +
userLocation);
}
```

This code works by first getting the user's location using the Geolocation.getGeolocation() method. Then, it displays the user's location using the Logger.log() method.

Example 6: Update Google Sheets from Forms

Automatically Update Google Sheets from Google Forms Responses: This code automatically updates a Google Sheet with responses from a specified Google Form. It first sets up a trigger to run the updateSheet function whenever a new response is

submitted to the form. When the function is run, it retrieves the response data and adds it to the specified sheet.

```
function updateSheet(e) {
  var ss = SpreadsheetApp.openById('spreadsheet_id');
// specify spreadsheet ID
  var sheet = ss.getSheetByName('Sheet1'); // specify
sheet name
  var data = e.response.getItemResponses();
  var rowData = [];

  for (var i = 0; i < data.length; i++) {
    rowData.push(data[i].getResponse());
  }

  sheet.appendRow(rowData);
}

function createTrigger() {
  var form = FormApp.openById('form_id'); // specify
form ID

ScriptApp.newTrigger('updateSheet').forForm(form).onFor
mSubmit().create();
}
```

The FormApp service is used to access the specified form, and the SpreadsheetApp service is used to access the specified spreadsheet and sheet. The updateSheet function is set up to run whenever a new response is submitted to the form. It retrieves the response data using the getItemResponses() method and adds it to the specified sheet using the appendRow() method. The

createTrigger function sets up a trigger to run the updateSheet function whenever a new response is submitted to the form.

Example 7: Generate Docs from Sheets Data

Automatically Generate Google Docs from Google Sheets Data: This code automatically generates Google Docs from data in a specified sheet. It first retrieves the data from the sheet and loops through it. For each row of data, a new document is created with the specified title and content.

```
function generateDocsFromSheet() {
  var ss = SpreadsheetApp.openById('spreadsheet_id');
// specify spreadsheet ID
  var sheet = ss.getSheetByName('Sheet1'); // specify
sheet name
  var data = sheet.getDataRange().getValues();
  var templateId = 'template_id'; // specify document
template ID
  var template = DocumentApp.openById(templateId);
  var destinationId = 'destination_id'; // specify
destination folder ID

  for (var i = 1; i < data.length; i++) {
    var row = data[i];
    var title = row[0];
    var content = row[1];
    var doc = template.makeCopy(title);
    var body = doc.getBody();
    body.replaceText('{{content}}', content);
    var docBlob = doc.getAs('application/pdf');
```

```
    var folder = DriveApp.getFolderById(destinationId);
    folder.createFile(docBlob);
  }
}
```

The SpreadsheetApp service is used to access the specified spreadsheet and sheet. The DocumentApp service is used to access the specified document template and create new documents. The generateDocsFromSheet function retrieves the data from the sheet and loops through it. For each row of data, a new document is created with the specified title and content. The makeCopy() method is used to create a copy of the template, and the replaceText() method is used to replace the placeholder text with the actual content. The getAs() method is used to convert the document to PDF format, and the createFile() method is used to save the PDF file to the specified folder.

Example 8: Generate Calendar Events from Sheets

Automatically Generate Google Calendar Events from Google Sheets Data: This code automatically generates Google Calendar events from data in a specified sheet. It first retrieves the data from the sheet and

```
function generateEventsFromSheet() {
  var ss = SpreadsheetApp.openById('spreadsheet_id');
// specify spreadsheet ID
  var sheet = ss.getSheetByName('Sheet1'); // specify
sheet name
  var data = sheet.getDataRange().getValues();
  var calendarId = 'calendar_id'; // specify calendar
ID
```

```
  var calendar =
CalendarApp.getCalendarById(calendarId);

  for (var i = 1; i < data.length; i++) {
    var row = data[i];
    var title = row[0];
    var startTime = new Date(row[1]);
    var endTime = new Date(row[2]);
    var description = row[3];
    calendar.createEvent(title, startTime, endTime,
{description: description});
  }
}
```

The SpreadsheetApp service is used to access the specified spreadsheet and sheet. The CalendarApp service is used to access the specified calendar and create new events. The generateEventsFromSheet function retrieves the data from the sheet and loops through it. For each row of data, a new event is created with the specified title, start time, end time, and description.

Example 9: Send email notifications from Sheets

Automatically Send Email Notifications from Google Sheets Data: This code automatically sends email notifications to specified recipients from data in a specified sheet. It first retrieves the data from the sheet and loops through it. For each row of data, an email is sent to the specified recipients with the specified subject and message.

```
function sendEmailsFromSheet() {
  var ss = SpreadsheetApp.openById('spreadsheet_id');
// specify spreadsheet ID
  var sheet = ss.getSheetByName('Sheet1'); // specify
sheet name
  var data = sheet.getDataRange().getValues();

  for (var i = 1; i < data.length; i++) {
    var row = data[i];
    var recipients = row[0];
    var subject = row[1];
    var message = row[2];
    MailApp.sendEmail(recipients, subject, message);
  }
}
```

The SpreadsheetApp service is used to access the specified spreadsheet and sheet. The MailApp service is used to send emails. The sendEmailsFromSheet function retrieves the data from the sheet and loops through it. For each row of data, an email is sent to the specified recipients with the specified subject and message.

Example 10: Generate Slides from Sheets

Automatically Generate Google Slides from Google Sheets Data: This code automatically generates Google Slides from data in a specified sheet. It first retrieves the data from the sheet and loops through it. For each row of data, a new slide is created with the specified title and content.

```
function generateSlidesFromSheet() {
```

```
  var ss = SpreadsheetApp.openById('spreadsheet_id');
// specify spreadsheet ID
  var sheet = ss.getSheetByName('Sheet1'); // specify
sheet name
  var data = sheet.getDataRange().getValues();
  var templateId = 'template_id'; // specify slide
template ID
  var template = SlidesApp.openById(templateId);
  var destinationId = 'destination_id'; // specify
destination folder ID

  for (var i = 1; i < data.length; i++) {
    var row = data[i];
    var title = row[0];
    var content = row[1];
    var slide = template.duplicate();
    slide.rename(title);
    var slideBlob = slide.getBlob();
    var folder = DriveApp.getFolderById(destinationId);
    folder.createFile(slideBlob);
  }
}
```

The SpreadsheetApp service is used to access the specified spreadsheet