How to Create a Web Form that can send emails using your Gmail account
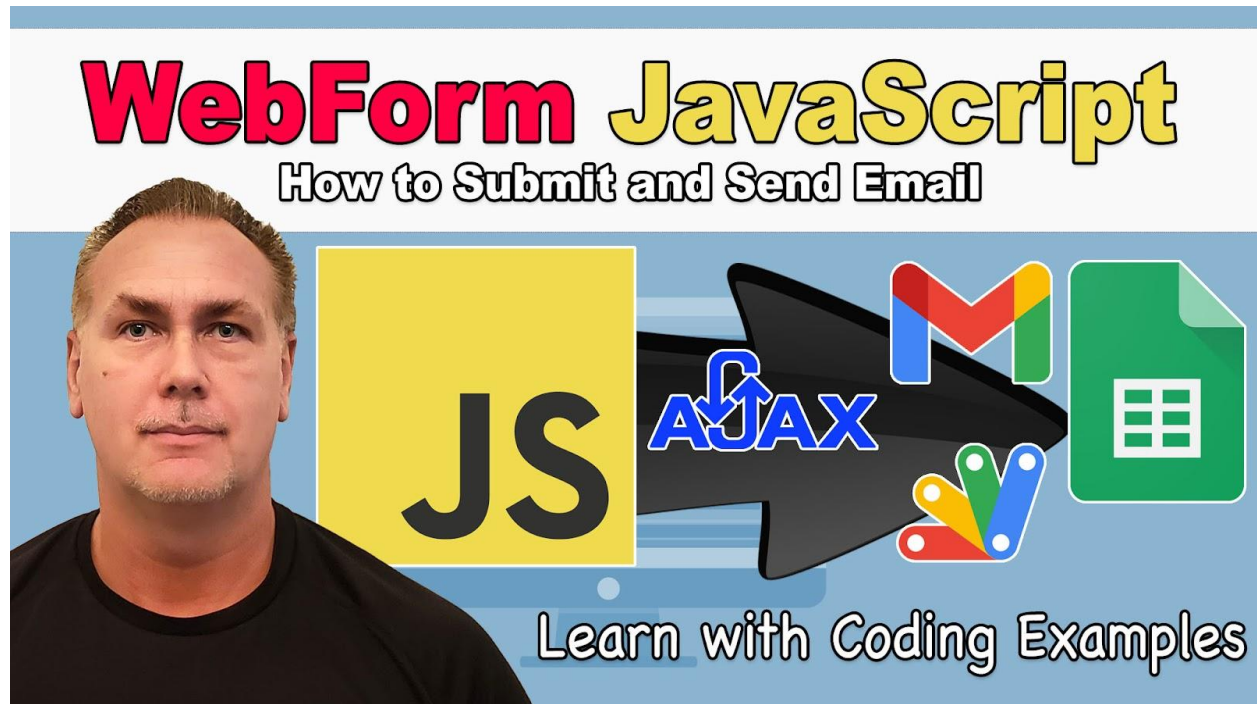
HTML Send Email via Apps Script

How to send an email from a HTML form, using Google Apps Script and JavaScript. Front-end code to send emails from html forms.

Exercise : Create an HTML form with fields for the data that you want to send. Setup the JavaScript to validate the form input values, and create an object of data to send to the Apps Script endpoint.

1. Add HTML input fields for the user to be able to enter information. Apply CSS styling as needed to style your form fields
2. Using JavaScript select the input fields as JavaScript variables
3. Create an event that invokes a function named submitter() when the form submit button is clicked
4. Using e.preventDefault(); to prevent the default action of the form submission to prepare for AJAX.
5. Add conditions on the input field values, set up a variable to add content to if errors are caught.

Laurence Svekis Course Content - https://basescripts.com/

6. If errors are caught in the conditions, output them in a new element that gets created with JavaScript.
7. Set a timeout to remove the error element and reset the input field border colors back to default.
8. Create an object the contains all the form input data with property names that match the data.

HTML CODE

```html
<!DOCTYPE html>
<html>

<head>
  <title>JavaScript Course</title>
  <style>
    * {
      box-sizing: border-box;
    }

    label {
      display: block;
      font-family: Arial, Helvetica, sans-serif;
      font-size: 0.8em;
      padding-top: 15px;
    }

    .myForm {
      width: 90%;
      margin: auto;
      border: 1px solid #ddd;
```

```css
      padding: 10px;

      background-color: #eee;

    }


    #myForm input,

    textarea {

      width: 100%;

      padding: 10px;


    }


    input[type="submit"] {

      background-color: black;

      color: white;

      text-transform: capitalize;

      font-size: 1.2em;

      border-radius: 10px;

    }
  </style>
</head>

<body>
  <div class="myForm">
    <form id="myForm">
      <h1>Send Message</h1>
      <div>
        <label for="name">Name:</label>
        <input type="text" id="name">
```

```html
        </div>
        <div>
            <label for="email">Email:</label>
            <input type="email" id="email">


        </div>
        <div>
            <label for="message">Message:</label>
            <textarea id="message"></textarea>


        </div>
        <input type="submit" value="send message">
    </form>
  </div>
  <script src="app3.js"></script>
</body>

</html>
```

JavaScript Code

```javascript
const url = '';
const myForm = document.querySelector('#myForm');
const myName = document.querySelector('#name');
const myEmail = document.querySelector('#email');
const myMessage = document.querySelector('#message');

myName.value = 'Laurence Svekis';
myEmail.value = 'g*****courses+100@gmail.com';
myMessage.value = 'Hello World';
```

```javascript
myForm.addEventListener('submit', submitter);

function submitter(e) {
  console.log('submitted');
  e.preventDefault();
  let message = '';
  if (myName.value.length < 5) {
    myName.style.borderColor = 'red';
    message += `<br>Name needs to be 5 characters`;
  }
  if (myEmail.value.length < 5) {
    myEmail.style.borderColor = 'red';
    message += `<br>Email is missing`;
  }
  if (message) {
    const div = document.createElement('div');
    div.innerHTML = message;
    div.style.color = 'red';
    myForm.append(div);
    setTimeout(() => {
      div.remove();
      myName.style.borderColor = '';
      myEmail.style.borderColor = '';
    }, 5000);

  } else {
    const myObj = {
      name: myName.value,
      email: myEmail.value,
```

```
        message: myMessage.value
    };
    console.log(myObj);
  }
}
```

Send POST request to Google Apps Script

In this lesson the application will be updated to make a fetch request using the POST method to the Google Apps Script endpoint, and insert the form field data into a spreadsheet.

Send Message

Name:

Laurence Svekis3333

Email:

gappscourses+100333@gmail.com

Message:

Hello World NEW

Send Message

name	email	message	status
Laurence Svekis	g******courses+100@gmail.com	Hello World	success
Laurence Svekis 2	g******courses+5550@gmail.com	Hello World Working ????/	success
Laurence Svekis 2	g******courses+5550@gmail.com	Hello World Working ????/	success
Laurence Svekis3333	g******courses+100333@gmail.com	Hello World NEW	success

Exercise : Update the fetch method to POST, include the form field data as an object in the POST request body contents. Create the Google Apps Script endpoint using a webapp to receive the GET and POST request data from the AJAX request from JavaScript.
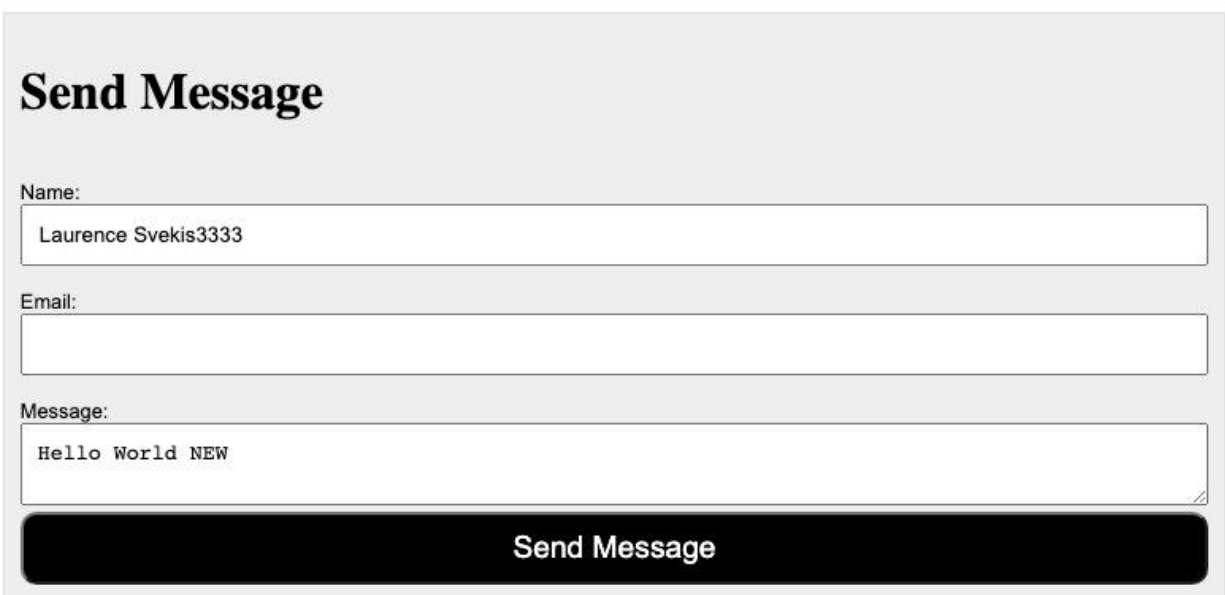
Create a web app with Google Apps Script GET method.
1. Go to https://script.google.com/ create a new project

2. Using the method doGet(e) create a new function that will return the e parameters from the URL request object.
3. Create and return the e parameters as a string value. return ContentService.createTextOutput(JSON.stringify(e))
4. Set the MimeType to JSON setMimeType(ContentService.MimeType.JSON)
5. Deploy a new webapp set the configuration to execute as your account, and who has access to Anyone
6. Copy the web app URL to the JavaScript application as the GET fetch endpoint url.
7. Try sending the fetch request to the Google Apps Script web app and check the response data in the console.

Apps Script Code

```
function doGet(e) {
 return
ContentService.createTextOutput(JSON.stringify(e)).setMimeType(ContentService.MimeType.JSON);
}
```

Send Message

Name:

Laurence Svekis3333

Email:

Message:

Hello World NEW

Send Message

Google Apps Script Testing adding to sheet data

1. Create a new spreadsheet add in the first row the names of the input fields from the HTML form
2. Get the spreadsheet ID from its URL and copy it as the id into the Google Apps Script.
3. Create a function named tester() to be used to test the input of values into the sheet. The doPost() is much harder to troubleshoot so best practice is to get the functionality working first then add it to the doPost() method.
4. Within the testing function open the sheet, SpreadsheetApp.openById(id).getSheetByName('emails') set as a variable
5. Get the data from the spreadsheet, selecting the first row of heading info as an array. ss.getDataRange().getValues()[0]
6. Create an object that can be used for testing with the same property names as the headings in the sheet, and the names from the input fields from the JavaScript object.
7. Loop through the heading names, if there is a value within the data object add that heading using the index value to a temporary holding array.
8. Using appendRow add the holding data to the sheet.
9. Try and run the test application to add to the spreadsheet columns a new row of data.

```javascript
function tester() {
 const id = '1csURUCONXy*****@M-c0';
 const ss = SpreadsheetApp.openById(id).getSheetByName('emails');
 const sheetdata = ss.getDataRange().getValues();
 const str = '{"name":"Laurence
Svekis","email":"gapps*****@.com","message":"Hello
World","status":"success"}';
 const json = JSON.parse(str);
 Logger.log(json);
 Logger.log(sheetdata[0]);
```

```
const holder = [];
sheetdata[0].forEach((heading, index) => {
  if (json[heading]) holder[index] = (json[heading]);
})
Logger.log(holder);
ss.appendRow(holder);
}
```

```
19    const sheetdata = ss.getDataRange().getValues();
20    const str = '{"name":"Laurence Svekis","email":"gappscourses
      +100@gmail.com","message":"Hello World","status":"success"}';
21    const json = JSON.parse(str);
22    Logger.log(json);
```

Execution log

4:01:27 PM	Notice	Execution started
4:01:28 PM	Info	Hello World
4:01:28 PM	Info	[name, email, message, status]
4:01:28 PM	Info	[Laurence Svekis, gappscourses+100@gmail.com, Hello World, success]
4:01:28 PM	Notice	Execution completed

Create a web app with Google Apps Script POST method.
1. Create a new function with the name of doPost(e)
2. Add the testing function code into the doPost()
3. Using the JSON.parse convert the submitted e parameter contents data as a usable JavaScript object. JSON.parse(e.postData.contents)
4. Get the last row of data using the getLastRow() method and add that into the post data contents object.
5. Return the response back to the data.

6. Deploy a web app, you can use the managed deployments to use the existing URL or create a new one and update the URL in your JavaScript code.
7. In the JavaScript code update the AJAX request to the new URL endpoint of the webapp.
8. Send the fetch request as POST with the body of the input field data as a stringified value.
9. Try the code and update the input field values to check the results in the Google Sheet rows.

```javascript
function doPost(e) {
 const id = '1csUR***c0';
 const ss = SpreadsheetApp.openById(id).getSheetByName('emails');
 const sheetdata = ss.getDataRange().getValues();
 const json = JSON.parse(e.postData.contents);
 json.status = 'success';
 const holder = [];
 sheetdata[0].forEach((heading, index) => {
   if (json[heading]) holder[index] = (json[heading]);
 })
 ss.appendRow(holder);
 json.rowval = ss.getLastRow();
 return
ContentService.createTextOutput(JSON.stringify(json)).setMimeType(ContentService.MimeType.JSON);
}
```

```
fetch(url,{
    method:'POST',
    body:JSON.stringify(data)
})
.then(res => res.json())
.then(json =>{
    console.log(json);
})
```

	A	B	C	D
1	name	email	message	status
2	Laurence Svekis	gappscourses+100@gmail.co	Hello World	success
3	Laurence Svekis 2	gappscourses+5550@gmail.c	Hello World Working ??	success
4	Laurence Svekis 2	gappscourses+5550@gmail.c	Hello World Working ??	success
5	**Laurence Svekis3333**	gappscourses+100333@gma	Hello World NEW	success
6	Laurence Svekis	gappscourses+100@gmail.co	Hello World	success
7				
8				
9				

JavaScript Code

```
const url = 'https://script.google.com/macros/s/AKf***C/exec';

const myForm = document.querySelector('#myForm');

const myName = document.querySelector('#name');

const myEmail = document.querySelector('#email');

const myMessage = document.querySelector('#message');


myName.value = 'Laurence Svekis';

myEmail.value = 'gapps*****@@gmail.com';

myMessage.value = 'Hello World';

myForm.addEventListener('submit', submitter);


function submitter(e) {
  console.log('submitted');
  e.preventDefault();
  let message = '';
  if (myName.value.length < 5) {
    myName.style.borderColor = 'red';
    message += `<br>Name needs to be 5 characters`;
  }
```

```javascript
    if (myEmail.value.length < 5) {
        myEmail.style.borderColor = 'red';
        message += `<br>Email is missing`;
    }
    if (message) {
        const div = document.createElement('div');
        div.innerHTML = message;
        div.style.color = 'red';
        myForm.append(div);
        setTimeout(() => {
            div.remove();
            myName.style.borderColor = '';
            myEmail.style.borderColor = '';
        }, 5000);

    } else {
        const myObj = {
            name: myName.value,
            email: myEmail.value,
            message: myMessage.value
        };
        addSendMail(myObj);
    }
}

function addSendMail(data){
    console.log(data);
    fetch(url,{
        method:'POST',
```

```
      body:JSON.stringify(data)
  })
  .then(res => res.json())
  .then(json =>{
    console.log(json);
  })
}


function addSendMailGET(data){
  const url1 = url + '?id=100';
  fetch(url1)
  .then(res => res.json())
  .then(json =>{
    console.log(json);
  })
}
```

Google Apps Script Source Code Complete

```
function doPost(e) {
 const id = '1csURUCONX****cR7gM-c0';
 const ss = SpreadsheetApp.openById(id).getSheetByName('emails');
 const sheetdata = ss.getDataRange().getValues();
 const json = JSON.parse(e.postData.contents);
 json.status = 'success';
 const holder = [];
 sheetdata[0].forEach((heading, index) => {
   if (json[heading]) holder[index] = (json[heading]);
 })
```

```
ss.appendRow(holder);
json.rowval = ss.getLastRow();
return
ContentService.createTextOutput(JSON.stringify(json)).setMimeType(Content
Service.MimeType.JSON);
}

function tester() {
 const id = '1csURUC*****@gM-c0';
 const ss = SpreadsheetApp.openById(id).getSheetByName('emails');
 const sheetdata = ss.getDataRange().getValues();
 const str = '{"name":"Laurence
Svekis","email":"gap*****@@gmail.com","message":"Hello
World","status":"success"}';
 const json = JSON.parse(str);
 Logger.log(json);
 Logger.log(sheetdata[0]);
 const holder = [];
 sheetdata[0].forEach((heading, index) => {
   if (json[heading]) holder[index] = (json[heading]);
 })
 Logger.log(holder);
 ss.appendRow(holder);
}

function doGet(e) {
 return
ContentService.createTextOutput(JSON.stringify(e)).setMimeType(ContentSe
rvice.MimeType.JSON);
```

```
}
```

Send Email when the form is submitted

Send an email to your email address with the form content when the web form is submitted. Send a response confirmation email to the user's email address from the submitted form content.

Thank you for your email Inbox ×

gappscourses@gmail.... 4:31 PM (16 minutes ago)
to gappscourses+777

Thank you Laurence Svekis 7777

We will respond shortly. Message recevied ID 6

Exercise : Update the Google Apps Script to send emails to the user's email address in response to the web form submission, send a second email to your email when the form data is submitted with the form field information.

Create a test function to send emails using data from an object
1. Create a function named sendMyEmail that will handle the sending of the emails using an object as the source for the data.
2. Create a function to validate an email address before trying to send an email to that user. This should be included to avoid errors in the Google Apps Script which would result in a CORS issue on the web form.
3. Create a testing function with simulated object data that would be coming from the form. Include the rowval that is set from the sheet row that was inserted.
4. Using the MailApp service, use the sendEmail method to send an email, with an object of parameters for the method.   Set the to, subject and htmlBody to the desired values for the email. You should

use the form data structure for the object, to simulate the form submission.
5. Check if the email is valid that the user provided, if it is then using the sendMail send a custom response to the user.
6. Using the test data ensures that you are able to send the email, and accept permissions for the app to use the mailApp service. This is required otherwise the app will not be able to send emails.
7. Move the sendEmail function to the doPost() method, using the real submitted data.
8. Deploy the webapp for the endpoint, if you create a new webapp then ensure you copy the new URL to the web application.

You should be able to send emails to the user, to yourself and also the data should still be added into the spreadsheet whenever the form is submitted.

```
function sendMyEmail(data) {
  let emailBody = `<div>Name ${data.name}</div>`;
  emailBody += `<div>Email ${data.email}</div>`;
  emailBody += `<div>Message ${data.message}</div>`;
  MailApp.sendEmail({
    to: 'g*****@@gmail.com',
    subject: 'NEW Web Form Email',
    htmlBody: emailBody
  });
  if (validateEmail(data.email)) {
    let repHTML = `<h2>Thank you ${data.name}</h2>`;
    repHTML += `<div>We will respond shortly. Message received ID ${data.rowval}</div>`;
    MailApp.sendEmail({
      to: data.email,
      subject: 'Thank you for your email',
      htmlBody: repHTML
    });
```

```
      return true;
  } else {
      return false;
  }
}


function validateEmail(email) {
  const re = /\S+@\S+\.\S+/;
  return re.test(email);
}


function testEmail() {
  const val = {
      email: 'gapps*****@gmail.com',
      name: 'tester',
      message: 'Hello World',
      rowval: 50
  }
  Logger.log(sendMyEmail(val));
}
```


Update JavaScript to manage the submission of the data and provide user feedback

1. When the form is submitted, disable the submit button to avoid a second submission. If there are an errors in the form fields enable the button
2. If the form fields are successfully filled out, hide the form element.
3. In the addSendMail() function create a new element, add it to the main content area. Add text for the user to see that their form is submitted.
4. Once a successful response is returned then update the text in the new field with the ID or row value of the imputed content in the sheet. If there was not a success response then show the form for a second submission of data.
5. Make adjustments to the management of the submission process to keep the user informed of the AJAX that the information is sent and the stage of the results from the submission of the data.

JAVASCRIPT

```javascript
const url = 'https://script.google.com/macros/s/AKfyc******XfK2iR/exec';

const myForm = document.querySelector('#myForm');

const myName = document.querySelector('#name');

const myEmail = document.querySelector('#email');

const myMessage = document.querySelector('#message');

const subBtn = document.querySelector('input[type="submit"]');

const main = document.querySelector('.myForm');

myName.value = 'Laurence Svekis';

myEmail.value = 'gapp******@gmail.com';

myMessage.value = 'Hello World';

myForm.addEventListener('submit', submitter);


function submitter(e) {
  console.log('submitted');
  e.preventDefault();
```

```javascript
subBtn.disabled = true;
let message = '';
if (myName.value.length < 5) {
    myName.style.borderColor = 'red';
    message += `<br>Name needs to be 5 characters`;
}
if (myEmail.value.length < 5) {
    myEmail.style.borderColor = 'red';
    message += `<br>Email is missing`;
}
if (message) {
    const div = document.createElement('div');
    div.innerHTML = message;
    div.style.color = 'red';
    myForm.append(div);
    setTimeout(() => {
        div.remove();
        myName.style.borderColor = '';
        myEmail.style.borderColor = '';
    }, 5000);
    subBtn.disabled = false;
} else {
    const myObj = {
        name: myName.value,
        email: myEmail.value,
        message: myMessage.value
    };
    myForm.style.display = 'none';
    addSendMail(myObj);
```

```javascript
    }
}

function addSendMail(data){
    console.log(data);
    const repDiv = document.createElement('div');
    repDiv.textContent = 'Waiting.....';
    main.append(repDiv);
    fetch(url,{
        method:'POST',
        body:JSON.stringify(data)
    })
    .then(res => res.json())
    .then(json =>{
        console.log(json);
        if(json.rowval){
            repDiv.textContent = `Message Sent Your ID is ${json.rowval}`;
        }else{
            repDiv.remove();
            subBtn.disabled = false;
            myForm.style.display = 'block';
        }

    })
}

function addSendMailGET(data){
    const url1 = url + '?id=100';
    fetch(url1)
```

```
    .then(res => res.json())
    .then(json =>{
       console.log(json);
    })
}
```

Google Apps Script

```
function doPost(e) {
 const id = '1csURUCO*******';
 const ss = SpreadsheetApp.openById(id).getSheetByName('emails');
 const sheetdata = ss.getDataRange().getValues();
 const json = JSON.parse(e.postData.contents);
 json.status = 'success';
 const holder = [];
 sheetdata[0].forEach((heading, index) => {
   if (json[heading]) holder[index] = (json[heading]);
 })
 ss.appendRow(holder);
 json.rowval = ss.getLastRow();
 json.result = sendMyEmail(json);
 return
ContentService.createTextOutput(JSON.stringify(json)).setMimeType(Content
Service.MimeType.JSON);
}


function sendMyEmail(data) {
 let emailBody = `<div>Name ${data.name}</div>`;
```

```javascript
emailBody += `<div>Email ${data.email}</div>`;
emailBody += `<div>Message ${data.message}</div>`;
MailApp.sendEmail({
  to: 'gap*****gmail.com',
  subject: 'NEW Web Form Email',
  htmlBody: emailBody
});
if (validateEmail(data.email)) {
  let repHTML = `<h2>Thank you ${data.name}</h2>`;
  repHTML += `<div>We will respond shortly. Message received ID ${data.rowval}</div>`;
  MailApp.sendEmail({
    to: data.email,
    subject: 'Thank you for your email',
    htmlBody: repHTML
  });
  return true;
} else {
  return false;
}
}

function validateEmail(email) {
const re = /\S+@\S+\.\S+/;
return re.test(email);
}

function testEmail() {
const val = {
```

```
    email: 'gappsc*****@gmail.com',
    name: 'tester',
    message: 'Hello World',
    rowval: 50
 }
 Logger.log(sendMyEmail(val));
}

function tester() {
 const id = '1csURU*******gM-c0';
 const ss = SpreadsheetApp.openById(id).getSheetByName('emails');
 const sheetdata = ss.getDataRange().getValues();
 const str = '{"name":"Laurence
Svekis","email":"gapp*******@gmail.com","message":"Hello
World","status":"success"}';
 const json = JSON.parse(str);
 Logger.log(json);
 Logger.log(sheetdata[0]);
 const holder = [];
 sheetdata[0].forEach((heading, index) => {
   if (json[heading]) holder[index] = (json[heading]);
 })
 Logger.log(holder);
 ss.appendRow(holder);
}

function doGet(e) {
```

```
 return
ContentService.createTextOutput(JSON.stringify(e)).setMimeType(ContentSe
rvice.MimeType.JSON);}
```