#1

function sendMyEmail(){

const email = 'gappscourses@gmail.com';

const subject = 'Test Email Script';

const body ='Hi This is an email';

GmailApp.sendEmail(email,subject,body);

}

function sendMyEmail2(){

const email = 'gappscourses@gmail.com';

const subject = 'Test Email Script 2';

const body ='Hi This is an email 2';

MailApp.sendEmail(email,subject,body);

}

#2

function mailService(){

const quotaVal = MailApp.getRemainingDailyQuota();

Logger.log(quotaVal);

}

function sendMyEmail1(){

const email = Session.getActiveUser().getEmail();

Logger.log(email);

const subject = 'Test Email';

const html = '<h1>Hello World</h1>'

const obj = {

to:email,

subject:subject,

htmlBody:html};

MailApp.sendEmail(obj);

}

function sendMyEmail3(){

const email = Session.getActiveUser().getEmail();

const subject = 'Test Email Attachment 3';

const html = '<h1>Hello World</h1>'

const id = '1pBcphLBsVcps0nsrAjH6bo8HvV4T1dS1';

const file = DriveApp.getFileById(id);

const opts = {

name:'My File',

cc:email,

noReply: true,

replyTo: 'test@gmail.com',

htmlBody:html,

attachments:[file.getAs(MimeType.PDF)]};

MailApp.sendEmail(email,subject,'',opts);

}

#3

function sendanEmail1(){

const email = Session.getActiveUser().getEmail();

const subject = 'Test Email Attachment 4';

const html = '<h1>Hello World</h1>'

const id = '1pBcphLBsVcps0nsrAjH6bo8HvV4T1dS1';

const file = DriveApp.getFileById(id);

const opts = {

name:'My File',

htmlBody:html,

attachments:[file.getAs(MimeType.PDF)]};

GmailApp.sendEmail(email,subject,'',opts);

}

function makeDraft(){

const email = Session.getActiveUser().getEmail();

const subject = 'Test Email 5';

const html = '<h1>Hello World</h1>'

GmailApp.createDraft(email,subject,'',{htmlBody:html})

}

function getDrafts(){

const drafts = GmailApp.getDrafts();

//Logger.log(drafts[0]);

for(let i=0;i<drafts.length;i++){

//Logger.log(drafts[i]);

}

drafts.forEach((draft)=>{

Logger.log(draft.getMessage());

let message = draft.getMessage();

if(message.getSubject() == 'Test Email 5'){

draft.send();

}

Logger.log(message.getSubject());

//Logger.log(draft.getId());

})

}

#4

function getmyThreads(){

const unread = GmailApp.getInboxUnreadCount();

//Logger.log(unread)

const threads = GmailApp.getInboxThreads();

//Logger.log(threads);

threads.forEach((thread)=>{

//Logger.log(thread.getId());

//Logger.log(thread.getMessageCount());

if(thread.isUnread()){

const messages = thread.getMessages();

thread.reply('Got the message - I read it - Thread');

let val = 0;

messages.forEach((message)=>{

Logger.log(message.getSubject());

val++;

message.reply('Reply on Message '+val,{

htmlBody:'<h1>wow HTML '+val+'</h1>'

});

})

Logger.log(thread.getId());

thread.markRead();

}

})

}

#5

function emailTempSender(){

const html = '<h1>Wow HTML is here</h1>';

let htmlBody = HtmlService.createHtmlOutputFromFile('email').getContent();

htmlBody = htmlBody.replace(/#TITLE/g,'Custom title');

htmlBody = htmlBody.replace(/#MESSAGE/g,'Custom message');

htmlBody = htmlBody.replace(/#NAME/g,'USER name');

const email = Session.getActiveUser().getEmail();

const subject = 'HTML Custom Message';

const prop = {

htmlBody : htmlBody

};

GmailApp.sendEmail(email,subject,'',prop);

}

#6

function sendHTMLtemp(){

//https://docs.google.com/document/d/1\_CzyhBlVdw3rUWyFubnm9zzfIGsniMmDpcT6c4T8OPk/export?format=html

const id = '1\_CzyhBlVdw3rUWyFubnm9zzfIGsniMmDpcT6c4T8OPk';

const url = 'https://docs.google.com/document/d/'+id+'/export?format=html';

const param ={

mehtod : "get",

headers : {

"Authorization":"Bearer "+ScriptApp.getOAuthToken()

},

muteHttpExceptions:true

}

let html = UrlFetchApp.fetch(url,param);

html = html.getContentText();

html = html.replace(/#USER/g, 'Laurence');

html = html.replace(/#MYNAME/g, 'Svekis Teacher');

const email = Session.getActiveUser().getEmail();

const subject = 'HTML from DOC '+id;

const prop = {

htmlBody:html

};

Logger.log(html);

GmailApp.sendEmail(email,subject,'',prop);

}

#7

function sender(){

const id = '14KNTH7qzgQuYRodf\_zvJDThrLkoKW4BgE1jG8Eld8wI';

const ss = SpreadsheetApp.openById(id);

const sheet = ss.getSheetByName('emails');

const data = sheet.getDataRange().getValues();

//Logger.log(data);

const people = data.slice(1);

//Logger.log(people);

people.forEach((person,index)=>{

Logger.log(person);

const start = index + 2;

sendMyEmails(person);

const updateRange = sheet.getRange(start,7,1,1);

updateRange.setValues([['sent']]);

})

}

function sendTester(){

const myArr =['id','Laurence',"svekis",'Teacher','test','gappscourses@gmail.com'];

sendMyEmails(myArr);

}

function sendMyEmails(person){

const email = person[5];

Logger.log(email);

let htmlBody = HtmlService.createHtmlOutputFromFile('email').getContent();

htmlBody = htmlBody.replace(/#TITLE/g,person[3]);

htmlBody = htmlBody.replace(/#FIRST/g,person[1]);

htmlBody = htmlBody.replace(/#LAST/g,person[2]);

htmlBody = htmlBody.replace(/#MESSAGE/g,person[4]);

MailApp.sendEmail(email,'From List','',{

htmlBody : htmlBody

})

}

#8

function messageStars(){

const emails = GmailApp.getInboxThreads();

emails.forEach((thread)=>{

//Logger.log(thread);

const messages = thread.getMessages();

messages.forEach((message)=>{

const finder = 'alert';

const test = message.getSubject().includes(finder);

//Logger.log(test);

if(test && message.isStarred()){

message.unstar();

}else if(test){

Logger.log(message.getSubject());

message.star();

}

//const hasStar = message.isStarred();

//Logger.log(hasStar);

})

})

}

#9

function testLabels(){

const myName = 'NewLabelTester';

const myLabel = GmailApp.getUserLabelByName(myName);

Logger.log(myLabel);

if(myLabel){

const emails = GmailApp.getInboxThreads();

emails.forEach((thread)=>{

Logger.log(thread.getFirstMessageSubject());

const val = thread.getFirstMessageSubject();

if(val.includes('Attachment')){

//thread.addLabel(myLabel);

thread.removeLabel(myLabel);

}

})

}

}

function getallTest(){

const threads = GmailApp.getUserLabelByName('Test').getThreads();

threads.forEach((thread)=>{

Logger.log(thread.getFirstMessageSubject());

})

}

function addTestLabel(){

let val = 'Test';

let myLabel = GmailApp.getUserLabelByName(val);

if(myLabel == null){

myLabel = GmailApp.createLabel(val);

}

const emails = GmailApp.getInboxThreads();

emails.forEach((thread)=>{

const first = thread.getFirstMessageSubject();

if(first.includes(val)){

thread.addLabel(myLabel);

}

})

}

function createmyLabel(){

const myName = 'NewLabelTester';

const myLabel = GmailApp.createLabel(myName);

Logger.log(myLabel);

}

#10

function chatThreads(){

const threads = GmailApp.getChatThreads(0,2);

Logger.log(threads);

threads.forEach((thread)=>{

Logger.log(thread.getMessageCount());

const messages = thread.getMessages();

messages.forEach((message)=>{

})

})

}

#11

function searchInbox(){

const results = GmailApp.search('has:attachment',0,2);

Logger.log(results);

results.forEach((thread)=>{

Logger.log(thread);

const messages = thread.getMessages();

messages.forEach((message)=>{

Logger.log(message.getRawContent());

})

})

}

https://support.google.com/mail/answer/7190?hl=en

#12

function searchInbox(){

const results = GmailApp.search('has:attachment',0,5);

//Logger.log(results);

let counter = 0;

results.forEach((thread)=>{

//Logger.log(thread);

const messages = thread.getMessages();

messages.forEach((message)=>{

const attachments = message.getAttachments();

//Logger.log(attachments);

attachments.forEach((data)=>{

Logger.log(data.getName());

Logger.log(data.getContentType());

if(data.getContentType() == 'application/pdf'){

const pdf = data.getAs('application/pdf');

counter++;

const newFile = DriveApp.createFile(pdf);

newFile.setName('NEW PDF '+counter+'.pdf');

}

})

})

})

}

#13

function tester(){

const val = GmailApp.getStarredThreads();

const myLabels = GmailApp.getUserLabels();

myLabels.forEach((label)=>{

Logger.log(label.getName());

})

const createDraftMessage = GmailApp.createDraft('test@gmail.com','Subjet','Body');

Logger.log(createDraftMessage );

val.forEach((thread)=>{

Logger.log(thread);

thread.addLabel(myLabels[0]);

})

}