#1

function makeSheet(){

const ss = SpreadsheetApp.create('testSheet1');

const sheet = ss.insertSheet('chart');

sheet.appendRow(['Date','Team A','Team B']);

const today = new Date();

const day = 1000\*60\*60\*24;

for(let i=0;i<20;i++){

const datePlayed = new Date(today.getTime() + (100\*day)+(i\*3\*day));

const score1 = Math.floor(Math.random()\*20);

const score2 = Math.floor(Math.random()\*20);

sheet.appendRow([datePlayed,score1,score2]);

}

const remover = ss.getSheetByName('sheet1');

ss.deleteSheet(remover);

Logger.log(ss.getId());

}

function makeChart(){

const id = '1y1CLO0EqDirKpWBxzqDhAoL-GhudGp0M73Lr9ollGFU';

const ss = SpreadsheetApp.openById(id);

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

Logger.log(ss);

const range = sheet.getDataRange();

const hAxis = {

gridlines: {count:10}

}

const temp = sheet.newChart().asLineChart();

const chart = temp

.addRange(range)

.setPosition(3,5,0,0)

.setNumHeaders(1)

.setOption('hAxis',hAxis)

.build();

sheet.insertChart(chart);

}

#2

function slideMakers(){

const id = '1y1CLO0EqDirKpWBxzqDhAoL-GhudGp0M73Lr9ollGFU';

const ss = SpreadsheetApp.openById(id);

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

const charts = sheet.getCharts();

Logger.log(charts);

const slides = SlidesApp.create(sheet.getName());

const els = slides.getSlides()[0].getPageElements();

els.forEach((el)=>{

el.remove();

})

slides.getSlides()[0].insertTextBox('Welcome to the Charts');

charts.forEach((chart)=>{

const newSlide = slides.appendSlide();

newSlide.insertSheetsChart(chart,20,10,600,400);

})

}

#3

function makeaPDF(){

const id = '1y1CLO0EqDirKpWBxzqDhAoL-GhudGp0M73Lr9ollGFU';

const oldFile = DriveApp.getFileById(id);

Logger.log(oldFile.getName());

const blob = oldFile.getBlob();

const file = DriveApp.createFile(blob);//create from blob

file.setName('Slides Tester');

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

const file2 = DriveApp.createFile(pdf);

file2.setName('As PDF');

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

MailApp.sendEmail(email,'PDF attached','',{

attachments: [file.getAs('application/pdf')],

name:'Attach PDF'

})

}

#4

function onOpen(e){

const ui = DocumentApp.getUi();

ui.createAddonMenu()

.addItem('fun 1','fun1')

.addItem('fun 2','fun2')

.addToUi();

ui.createMenu('Custom Menu')

.addItem('fun 1','fun1')

.addItem('fun 2','fun2')

.addToUi();

//const doc = DocumentApp.getActiveDocument();

//const newContent = String(new Date());

//Logger.log(e);

//const eventDetails = JSON.stringify(e);

//doc.getBody().appendParagraph(eventDetails);

}

function fun1(){

const doc = DocumentApp.getActiveDocument().getBody();

const txt = doc.editAsText();

txt.insertText(10,'New Text \n New Line \n');

}

function fun2(){

const doc = DocumentApp.getActiveDocument().getBody();

const para = doc.appendParagraph('New Paragraph');

}

#5

function onEdit(e) {

logger(JSON.stringify(e));

const range = e.range;

range.setNote('Updated '+new Date());

logger(range.getLastRow());

range.setBackground('#00ff00');

}

function onSelectionChange(e) {

const range = e.range;

logger2(['Selection',JSON.stringify(e),range.getNumRows(),range.getNumColumns(),range.getFontSize(),JSON.stringify(range.getBackgrounds())]);

let total = range.getNumRows() + range.getNumColumns();

if(total > 8){

range.setBackground('#ee00ff');

}else{

range.setBackground('#00cc22');

}

}

function onOpen(e) {

//const ss = SpreadsheetApp.getActiveSpreadsheet();

//const menuItems = [{name:'Fun 1',functionName:'fun1'}];

///ss.addMenu('adv',menuItems);

//ui.createAddonMenu()

logger(JSON.stringify(e));

const ui = SpreadsheetApp.getUi();

ui.createMenu('NEW')

.addItem('fun 1','fun1')

.addToUi();

}

function logger2(val){

const ss = SpreadsheetApp.getActiveSpreadsheet().getSheetByName('logger');

ss.appendRow(val);

}

function logger(val){

const ss = SpreadsheetApp.getActiveSpreadsheet().getSheetByName('logger');

ss.appendRow([val]);

}

function fun1(){

const ss = SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();

ss.appendRow(['NEW','Cols']);

}

#6

function doGet(e) {

//const html = JSON.stringify(e);

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

return HtmlService.createHtmlOutput(html);

}

#7

function fun1(){

const id = '1DbghUBYtBoXeR3B7i3iFVUjkpmEIKWUJnHk7LG6Wi0o';

const doc = DocumentApp.openById(id).getBody();

const val = String(new Date());

Logger.log(doc);

doc.appendParagraph(val);

}

function fun2(){

const id = '1DbghUBYtBoXeR3B7i3iFVUjkpmEIKWUJnHk7LG6Wi0o';

const doc = DocumentApp.openById(id).getBody();

const val = 'Hello World';

doc.appendParagraph(val);

doc.appendHorizontalRule();

}

function addTrigger(){

ScriptApp.newTrigger('fun2')

.timeBased()

.everyMinutes(1)

.create();

}

function updateTriggers(){

const allTriggers = ScriptApp.getProjectTriggers();

Logger.log(allTriggers);

allTriggers.forEach((tri)=>{

Logger.log(tri.getUniqueId());

Logger.log(tri.getEventType());

Logger.log(tri.getHandlerFunction());

if(tri.getHandlerFunction() == 'fun2'){

ScriptApp.deleteTrigger(tri);

}

})

}

https://developers.google.com/apps-script/guides/services/quotas - get quotas