Owl Demo Project

● ● ●

Revamping Google Sheets with Google App Script

Bringing some context

Where: Marketing organization at a global beauty company

Why: Operations department goal to streamline processes and leverage data

What: Opportunities to revamp existing Google Sheets trackers

When: First semester of 2020

Who/How: Collaboration with team leads and producers

How Much: 20% allocated during 6 months

The starting point

Project expense tracker used prior to 2020

Data sources and reports generated manually

The Problem

Overview

➔ Simple project expense tracker in Google Sheets used by over 10 users
➔ Used to track expenses, support accounting and budget process

Pain Points

➔ Large amounts of time dedicated to data manipulation and reporting
➔ Process prone to errors, lacking data integrity and organized data structure

Project Overview

UI and UX

2 ● User runs functions via menu selector

3
- Create new project tracker
- Consolidate data from selected tabs
- Fix broken formulas
- Protect active sheet
- Create backup copy of application

1 ● User updates information on desired sheets

Intake Process

2
- User runs functions via menu selector

4
- Email notification sent to recipient

1
- User input information on Intake sheet and complete necessary fields

3
- New project sheets are created with data from intake sheet

Plan Start	Project	Key	Go Live	Responsible		dget	GL Code	Cost Center	IO CODE	Expense Period	New Tab Name	Template Type	Created	Notification
1/1/2020	January Initiatives	PRC1	Jan 2020	John Doe	Random notes 1	$20,000	GL CODE 1	CC01	IO CODE 1	Jan 2020	01_Jan	Default - Simple Project	Created	☑
1/1/2020	February Initiatives	PRC2	Feb 2020	Betsy Dean	Random notes 2	$60,000	GL CODE 2	CC02	IO CODE 2	Feb 2020	02_Feb	Full Project	Created	☑
1/1/2020	March Initiatives	PRC3	Mar 2020	Mary Dwight	Random notes 3	$20,000	GL CODE 3	CC03	IO CODE 3	Mar 2020	03_Mar	Default - Simple Project	Created	☑
4/1/2020	April Initiatives	PRC4	Apr 2020	Jimmy Chow	Random notes 4	$20,000	GL CODE 4	CC04	IO CODE 4	Apr 2020	04_Apr	Default - Simple Project	Created	☑
4/1/2020	May Initiatives	PRC5	May 2020	Paul Will	Random notes 5	$20,000	GL CODE 5	CC05	IO CODE 5	May 2020	05_May	Default - Simple Project	Created	☑
4/1/2020	June Initiatives	PRC6	Jun 2020	Steve Stove	Random notes 6	$60,000	GL CODE 6	CC06	IO CODE 6	Jun 2020	06_Jun	Full Project	Created	☑
7/1/2020	July Initiatives	PRC7	Jul 2020	Hillary Biden	Random notes 7	$20,000	GL CODE 7	CC07	IO CODE 7	Jul 2020	07_Jul	Default - Simple Project	Created	☑
7/1/2020	August Initiatives	PRC8	Aug 2020	Joe Harris	Random notes 8	$20,000	GL CODE 8	CC08	IO CODE 8	Aug 2020	08_Aug	Full Project	Created	☑
9/1/2020	September Initiatives	PRC9	Sep 2020	Ricky Johnson	Random notes 9	$60,000	GL CODE 9	CC09	IO CODE 9	Sep 2020	09_Sep	Full Project	Created	☑
9/20/2020	October Initiatives	PRC10	Oct 2020	Ja H			GL CODE 9	CC10	IO CODE 10	Oct 2020	10_Sep	Full Project	Created	☑
	November Initiatives	PRC11	Nov 2020	Jan H								Full Project		☐
	December Initiatives	PRC12	De	Jan H								Default - Simple Project		☐
														☐
														☐
														☐

Data Consolidation

2 • User runs functions via menu selector

1 • User defines sheets and triggers for consolidation

3 • Output sheet updated after data extraction and transformation

Support Tabs Tracker Tabs

Challenges and Solutions

The biggest challenge required some research and creativity to solve

Problem

- Overcome execution time limit of 5 minutes
- Issue started when running consolidation for over 30 trackers
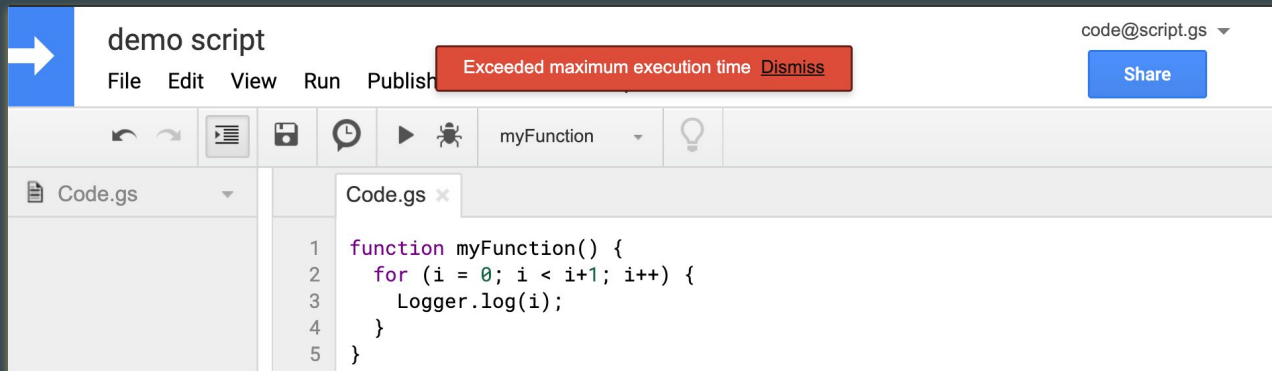
Approach to Solution

Distribute consolidation in parts

```
function CallGetDataFromTab_Trigger_1() {
  GetDataFromTab_Trigger(1);
}

function CallGetDataFromTab_Trigger_2() {
  GetDataFromTab_Trigger(2);
}

function CallGetDataFromTab_Trigger_3() {
  GetDataFromTab_Trigger(3);
}
```

```
function CallCopyPaste_Trigger() {
//cleaning previous range
var DestinationSheet =
  SpreadsheetApp.getActiveSpreadsheet().getSheetB
  yName('Data');

DestinationSheet.getRange('A1:AE10000').activate
  ();
DestinationSheet.getActiveRangeList().clear({con
  tentsOnly: true, skipFilteredRows: true});

var spreadsheet = SpreadsheetApp.getActive();
//tab with all temp data
spreadsheet.setActiveSheet(spreadsheet.getSheetB
  yName('Temp Data'), true);

var sheet = SpreadsheetApp.getActiveSheet();
var data = sheet.getDataRange().getValues();

// write data to sheet
DestinationSheet.getRange(1, 1, data.length,
  data[0].length).setValues(data);

spreadsheet.setActiveSheet(spreadsheet.getSheetB
  yName('Data'), true);

}
```

```
function GetDataFromTab_Trigger(runNumber) {

var CallnumRun = runNumber
var PasteSheet = SpreadsheetApp.getActiveSpreadsheet().getSheetByName('Temp Data');
var rowPaste;
//call function to get last row to paste
var lastRow = goToFirstRowAfterLastRowWithData(PasteSheet,1)

  if(CallnumRun === 1){
    //cleaning previous range on first run
    rowPaste = 2;
    PasteSheet.getRange('A2:AE10000').activate();
    PasteSheet.getActiveRangeList().clear({contentsOnly: true, skipFilteredRows: true});

  }else{
    rowPaste = lastRow;
  }
  var spreadsheet = SpreadsheetApp.getActive();
  //tab with all sheet names, sheet names generate by custom cell function
  spreadsheet.setActiveSheet(spreadsheet.getSheetByName('ADMIN'), true)
  var sheet = SpreadsheetApp.getActiveSheet();
  var data = sheet.getDataRange().getValues();
  ...
```

```
...

//getting tab names to pass next
  for (var i = 0; i < data.length; i++) {

    var StrSheetName = data[i][0];

    //check if tab is needed, function argument used here to get Sheet Name
    if(data[i][2] === runNumber && data[i][0] != ''){

      GetDataFromTab(StrSheetName);

    }
  }

// write data to sheet
PasteSheet.getRange(rowPaste, 1, values.length, values[0].length).setValues(values);

//activate sheet
spreadsheet.setActiveSheet(spreadsheet.getSheetByName('Temp Data'), true);
}
```

Approach to Solution

```
function GetDataFromTab(ShName) {
//return updates on global array

var spreadsheet = SpreadsheetApp.getActive();
spreadsheet.setActiveSheet(spreadsheet.getSheetByName(ShName), true);
var sheet = SpreadsheetApp.getActiveSheet();
var data = sheet.getDataRange().getValues();
//define variables
var strType;
var strCategory;
...
//loop to get data from tab
  for (var i = 9; i < data.length; i++) {
      if(data[i][1]==='Project'){
          strProject=data[i][3];
      }
      if(data[i][1]==='Line Item' || data[i][1]==='Project Budget'){
        strType = data[i][1];
        strCategory = data[i][2];
        ...
        };
        // push a row of data as 2d array
```

Approach to Solution

Extract and Transform data from Sheet (2/2)

```
...
        // push a row of data as 2d array, values[] is the global array

values.push([strType,ShName,strInitiative,'PRC',strProject,strCategory,strExpenseType,strName,strVendor,str
InvoiceNum,strBudget,strUnit,strRate,strQuantity,strEstimate,strActuals,strAssetCount,strNotes,strGLCode,st
rCostCenter,strCode,StrExpenseMonth,strBillTo,strForecast,strFUP,strSavingsCheck,strDateNum,i]);

    }

  }

}
```

Approach to Solution

Opportunities

What could be better

- Improve error and exception handling
- Check for mandatory fields on intake process
- Overcome slow down using big spreadsheets
- Try alternatives to increase speed in data consolidation

New Features

- Develop functionality to systematically use sheet templates
- Develop functionality to check and report data inconsistencies
- Develop functionality to manage lists and enable update on dropdowns
- Develop functionality to manage email notification

Questions?

Stay for quick demo video next