GitExcely: Version control for Excel via Git

GitHub: <https://github.com/hellofromtheothersky/Excel-data-version-control-via-Git>

In this demo, we will walk through these points:

* Create a new project
* Edit Excel and discover its text version after being parsed
* How to deal with conflict situation

Demo images:

1. **Init project**

Create a repo on Git to manage an Excel file named **Department\_Excel\_Docs\_Sample.xlsx**

Here are command lines to install gitexcely and create the project:

pip install gitexcely

gitexcely init –-path Department\_Excel\_Docs\_Sample

Then, we will get a local repo with a pre-set layout to work with

A screenshot of a computer

Description automatically generated

Below are meanings of some important file/folder of the tool

**A screenshot of a computer program

Description automatically generated**

1. **First commit**

Create a new simple excel file named peoples.xlsx file in the EXCEL/ folder

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

I want to specify some features (the header row, index/key columns) of my sheet, the **EXCEL\_METADATA.json** is used for this purpose

A computer screen shot of a program

Description automatically generatedNote: even if I don’t specify any features in the files, the Excel file will still be parse in its natural way *(A, B, C, D… for the column name instead of the header row, and number line count instead of the key columns)*

**And after create commit action**, a pre-commit action from git will check to find the changed Excel files and parse it into text. Then, we get this after creating peoples.xlsx.

A screenshot of a computer

Description automatically generated

Here is the structure of path to store data in text:

EXCEL\_TEXT/{EXCEL\_FILE\_NAME}/{SHEET\_NAME}/{ROW\_NAME}/

ROW\_NAME is default count by line (“L1”, “L2”, “L3” …), but if the key columns of data are declared, it will replace by the very data of key columns in this row (“James”, “TrungHieu”, “GiaTien”), if duplicate occurs, the row name will come with a prefix is the line count (“L6\_James”, “L9\_James”).

***values.csv*, and *styles.json* will store data of only one row.**

***“I choose this approach because, for most of the Excel use cases, we store row-based data. And splitting a whole table into rows and store it many files help us control version better, easily track, change focally for a single row”***

After committing, we also have a change log file to view changes summary before pushing:

A screen shot of a computer

Description automatically generated

1. **Make some changes**

Now, I will make some changes on “James” row,

* Change “London” to “New York”
* Make the font size of column “name” bigger:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Commit again, and we get:

Changes summary:

A screen shot of a computer

Description automatically generated

Changes detail in files:

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

Parse Excel elements into text and utilize current git diff, we can now check the changes more easily

1. **Solve conflict**

The big difference between current Excel different tracking system (MS Version History or others) and GitExcely is the ability to work and collaborate in the real Git style including create branch, merge and solve conflict.

And beside the parsing function from Excel into text for us to check diff of a binary file through text, GitExcely also have generating function Excel from text which help user who found conflict from text, and solve directly on text to utilize current git support (like what do we with code version control)

A diagram of a software

Description automatically generated

For example, I will create a feature branch basing on main branch. Make a change on the same position to create conflict when merging and then trying to solve the conflict

A blue line on a black background

Description automatically generated

**On the feature branch:**

* Add a new row with key “Siam”
* On “TrungHieu” row, change “Quy Nhon” to “Ha Noi”

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generatedCommit data

A screenshot of a computer

Description automatically generated

Two new files (Siam\styles.json and Siam\values.csv) for the new row, and a modified file for the updated row (TrungHieu\values.csv)

**On the main branch:**

Still on “TrungHieu” row, change “Quy Nhon” to “Ho Chi Minh city”

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Commit data:

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

**Merge feature into main branch:**

In this case, conflict occur in two files:

* .xlsx file, which is a binary file, git cannot help us find the conflict parts
* EXCEL\_TEXT/peoples/Sheet1/TrungHieu/values.csv file

To solve the conflict, we will manipulate on the text version instead, and then when commit, the Excel will be generated again to mimic its text version

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Solve conflict on the file – which is row data that have confict

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Merge

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generatedChecking the changes

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

And finally, the conflict was solved, excel file was updated right the same as text version