Developer Documentation for Mathcad Automation Software

Piero Panariello

July 9, 2021

Contents

1 Technologies Used

1. Python is the programming language of choice

2. PySimpleGUI : used to render the graphical user interface. Documentation

3. MathcadPy : wrapper written in python used to access the Mathcad api. Documentation

4. Openpyxl : used to interface with the excel documents. Documentation

5. PyInstaller: used to freeze the python program into an executable. Documentation

6. Look in ~./dist/requirements.txt to view all the dependencies

7. Look in ~./dist/info.txt for more information on how to package the application and how to re-create the python virutal environment.

2 Mathcad API

Currently, the Mathcad API supports Mathcad Prime 3.0 and above. From my testing it works best with Mathcad Prime 7.0 (the lastest version). The API documentation can be located here. You can purchase the SDK from Mathcad to get more information and examples, but I would recommend against it (it's $9000).

3 Datypes and Storage of Data

3.0.1 Equipment Class: (stores all the equipment from the excel file)

1. Class variables:

    - self.items = list()

    List of all the equipment in the excel file, stored as individual dictionaries.

    - self.cur_index = 0

    The current index of the equipment that the user is viewing in the GUI

- self.length = 0

Holds the length of self.items

- self.fields = list()

A list of all the elements from the header row from the excel document

- self.names = list()

A list of all the equipment names from the excel document

- self.inputs = list()

A list of all the inputs from the header row in the excel document

2. Class methods:

- append(self, to_append:dict) Takes in a dict as an argument

Appends self.items with the new equipment dictionary, appends self.names, appends self.inputs

- next_index(self) No arguments

Increments the value of self.cur_index

- prev_index(self) No arguments

Decriments the value of self.cur_index

3.0.2 Outputs Class: (stores the values of the outputs when the user decides to preview the output variables from the Mathcad file)

1. Class variables:

- self.items = list()

Follows the format: alias, [value, unit, power] ['f_p_max_output', [408.81554560308007, 'kg', 0]], ['f_p_min_output', [76.65291480057748, 'kg', 0]], ['f_p_tot_output', [76.65291480057748, 'kg', 0]], ...

2. Class methods:

- append(self, to_append)Takes tuple or list argument

Converts to_append to array and appends self.items

- clear(self)

Clears self.items

- display(self)->list No arguments

Returns a list of variables and values that is easier to display in the GUI. Rounds decimals to 2 digits. Ex: ['f_p_tot_output = 408.82 kg', 'f_p_min_output = 76.65 kg', ...]

4   Excel and Mathcad

The MathcadPy library is used as a wrapper that allows you to access all of the mathcad api endpoints from the comfort of Python. You can read more about the Mathcad API here. The API allows you to modify and change Mathcad Prime files. Despite PTC's documenation, you cannot print documents.

The function get_eqpt_from_xl(filepath:str)->Equipment takes in the filepath of the input excel file and returns the Equipment object. This function is executed right after the choose files window is closed.

The excel table looks similar to the one below:

eqpt_name	mounting_location	project_number	tags
Anesthesia machine	Wall, Floor	1111	Medical, ICU, something
Warming Cabinet	Floor	1111	Medical
Surgical Scrub Sink	Wall	1111	Medical
Retratable Ceiling Column	Ceiling	1111	Medical

The function pre_generate_report(equipment:Equipment, files, generating_multiple_reports = False) acts as a pre-fight test. It checks if the proper template is given for the equipment and passes the equipment and a uniquely generated filename to the generate_report function. The function generate_report(cur_eqpt, equipment:Equipment, file_name:str, template_file:str, files, debug = False)->bool connects to the Mathcad API, opens the template file specific to the mounting location, updates the input values specific to the equipment, and then saves the document. If generateing multiple reports, multithreading is used to speed up the process. Currently 4 threads are being used, but feel free to increase this number if the workflow demands more throughput. This variable is called num_threads in the event generate_report_for_all.

The function mathcad_calculate(eqpt, files, debug = False) allows the user to preview the Mathcad calculation output. It duplicates the template file into a temp file, takes the inputs and waits for the outputs to generate. It then deletes the temp file when finished. It returns a dictionary with the output values. The debug variable changes if Mathcad will display the windows being edited or not. When debug = False, no window is shown, when debug = True, windows are shown.

5 Rendering to the GUI

Choose equipment: Once the user has input the excel file they want to read from, the program extracts all information in the get_eqpt_from_xl function and places all the equipment names in the Choose Equipment column.

Inputs: Once we get the Equipment from the get_eqpt_from_xl function, we can then render it to the input fields in the GUI.

Outputs: If the user clicks the Preview Calculation Outputs button, the inputs from the current equipment being used is sent to the template corresponding to the correct mounting location and the output fields are gathered via the mathcad_calculate function. The outputs are saved in the Outputs class and the information is displayed in the GUI.

Preview Images: The user has the option to include preview images that correspond to the mounting locations. The images must be included in the excel document. Use the example_sheet.xlsx as a template. Images muse be .png or .jpg or .jpeg. The images are gathered from the excel sheet using the get_images_from_xl(self, num_images:int) function. Images are stored as binaries. When the user views a different equipment, the image corresponding to the mounting location is loaded into the Image Preview section of the GUI. Use ./dist/img_to_b64.py to convert images from .png/.jpg to binaries. It provides a python file called output.py with the binaries stored in variables. Preview images are automatically converted to binaries in get_images_from_xl.

6 Saving to the Database

Saving to Database: The database is a .csv file which holds some important information about generated repors. The function save_to_csv is used. When the user decides to save the report to the database, they are saving the equipment name, mounting location, tags, and the generated report's unique filename. The user can choose a specific database to save to, or it will save to the default database located in the mathcad_automation_output folder.

The table looks like the one below:

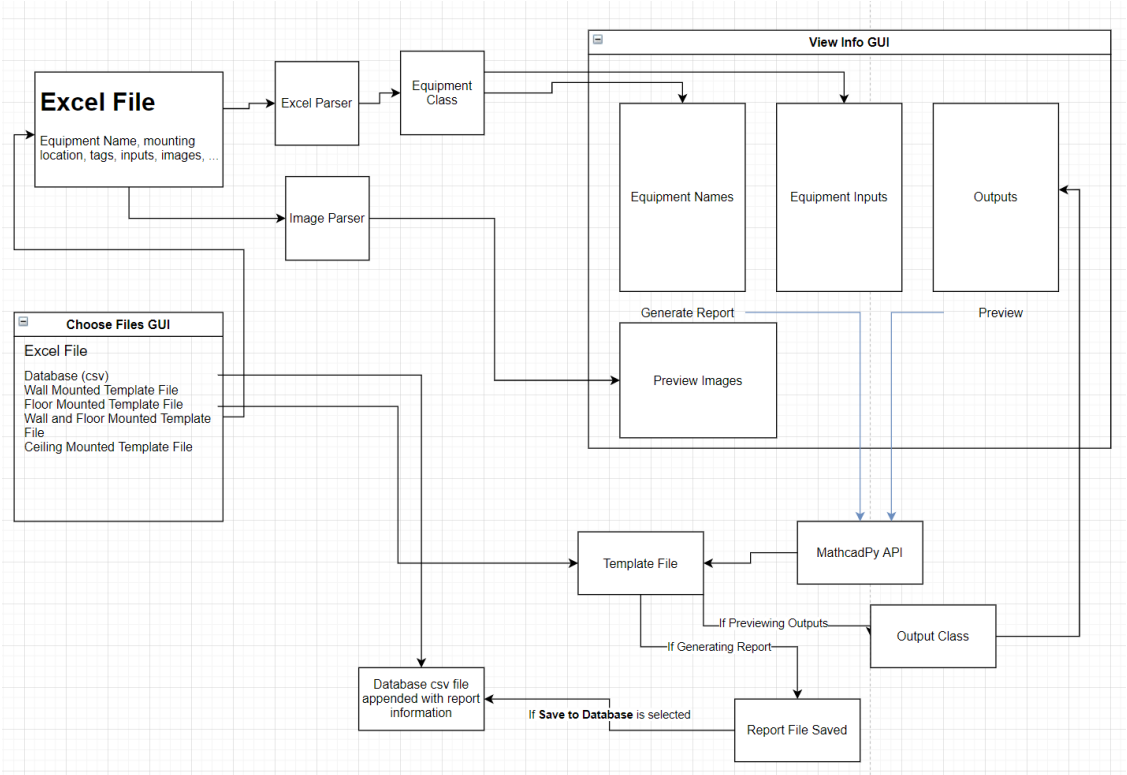| Date	Tags	Name	Mounting Location	File Name
2021-06-30	MEDICAL	RETRACTABLE CEILING COLUMN	CEILING	test.mcdx
2021-06-30	MEDICAL	RETRACTABLE CEILING COLUMN	CEILING	Retractable_….mcdx
2021-06-30	MEDICAL	SURGICAL SCRUB SINK	WALL	Surgical_….mcdx

7 Flowchart

Figure 1: Flow chart of program functions