How to create a list of files, folders, and subfolders and then export as Excel

A productivity tool to Marie Kondo your folders

Ji Wei Liew Oct 26, 2020 · 4 min read ★

Photo by Pranav Madhu on Unsplash

> *Can you create a list of files, folders, and subfolders for our project folder?*

This request looks deceptively simple, but you can already imagine the *n* other modifications that may come once the first version of this list is done.

> *Can you also list the files types and add hyperlinks to the file? And maybe add the file size while you're at it?*

Depending on the size of the folder, number of files and how nested it might be, you could have chosen to complete this task manually as adding new fields might be simple. Now, let's consider another request:

> *I've re-arranged some folders and renamed some files, can you update that list?*

Fret not, Python to the rescue!

`os.walk()` is the main function that we'll be using to create a script to loop through the

```
df = pd.DataFrame(columns=['File','File Type',
                           'Folder Location','Link', 'Path'])

for root, dir, files in os.walk(path):
    files = [f for f in files if not f.startswith('~') and
f!='Thumbs.db']
    paths = [os.path.join(root, f) for f in files]
    exts = [f.rsplit('.',1)[-1].lower() for f in files]
    filetypes = [ext_desc(ext) for ext in exts]
    file_links = ['=HYPERLINK("{}","link")'.format(p) if len(p) < 256
else '' for p in paths]
    folders = [p.rsplit('\\',1)[0] for p in paths]
    df1 = pd.DataFrame({'File': files,
                        'File Type': filetypes,
                        'Folder Location': folders,
                        'Link': file_links,
                        'Path': paths})
    df = df.append(df1)
```

Full code available at the end of story.

Code explained:

1. Create an empty DataFrame with the required columns in the required order:

```
df = pd.DataFrame(columns=['File','File Type',
                           'Folder Location','Link', 'Path'])
```

2. Initiate the `for` loop:

```
for root, dir, files in os.walk(path):
```

`path` refers to the main folder path which we are interested. `os.walk` returns a tuple `root`, `dir` and `files`. We'll need only `root` and `files` for this.

```
# 'Files' columns
files = [f for f in files if not f.startswith('~') and
f!='Thumbs.db']
```

Temporary files start with a tilde `~` and it might not be meaningful to include these, hence the list comprehension has an `if` condition to exclude such files. Similarly, `Thumbs.db` is just a Thumbnail image file and explicitly excluded.

```
# 'Path' column
paths = [os.path.join(root, f) for f in files]
```

`os.walk` documentation recommends to use `os.path.join` as opposed to string concatenation and here's a stackoverflow answer why that is recommended.

```
# 'File Type' column
exts = [f.rsplit('.', 1)[-1].lower() for f in files]
filetypes = [ext_desc(ext) for ext in exts]
```

`exts` is a list of extensions formed by right-splitting `rsplit` the filenames. `filetypes` is a list of file types and `ext_desc` is a function that maps each extension to to an appropriate name.

```
def ext_desc(ext):

    d_ext_desc = {'xlsx':'Microsoft Excel File',
                  'docx':'Microsoft Word Doc'}
    # additional extensions and descriptions can be added to
d_ext_desc

    try:
        desc = d_ext_desc[ext]
    except KeyError:
        desc = ''     # Any file extensions not mapped will be empty.
    else:
```

The links to the files are added in a slightly different way.

```
file_links = ['=HYPERLINK("{}","link")'.format(p) if len(p) < 256
else '' for p in paths]
```

`=HYPERLINK()` is an Excel formula which creates a hyperlink when given a link location (can be webpage or file location) The second argument, `link` , is optional and serves to shorten the displayed text. The `if` condition `len(p)<256` is a limitation in Excel, where links longer than 255 characters do not work.

```
folders = [p.rsplit('\\',1)[0] for p in paths]
```

`folders` should be self-explanatory by now. :)

4. Appending to the original dataframe

```
df1 = pd.DataFrame({'File': files,
                    'File Type': filetypes,
                    'Folder Location': folders,
                    'Link': file_links,
                    'Path': paths})
df = df.append(df1)
```

As each list comprehension is a column, all that is left is to build a dataframe and append to the original dataframe. The final dataframe can be written to Excel by using `df.to_excel('some_filename.xlsx')`

5. Generalizing this for future use

- Generating the full file index takes time if the folder contains a lot of files, hence I've added a keyword argument to `break` the `for` loop when the number of records exceeds 500. This is especially useful when adding new columns or modifying how

- Modifying the path in the script might be troublesome, hence I've added a `tkinter.filedialog` method to prompt the user to select the folder if the path is not provided.