

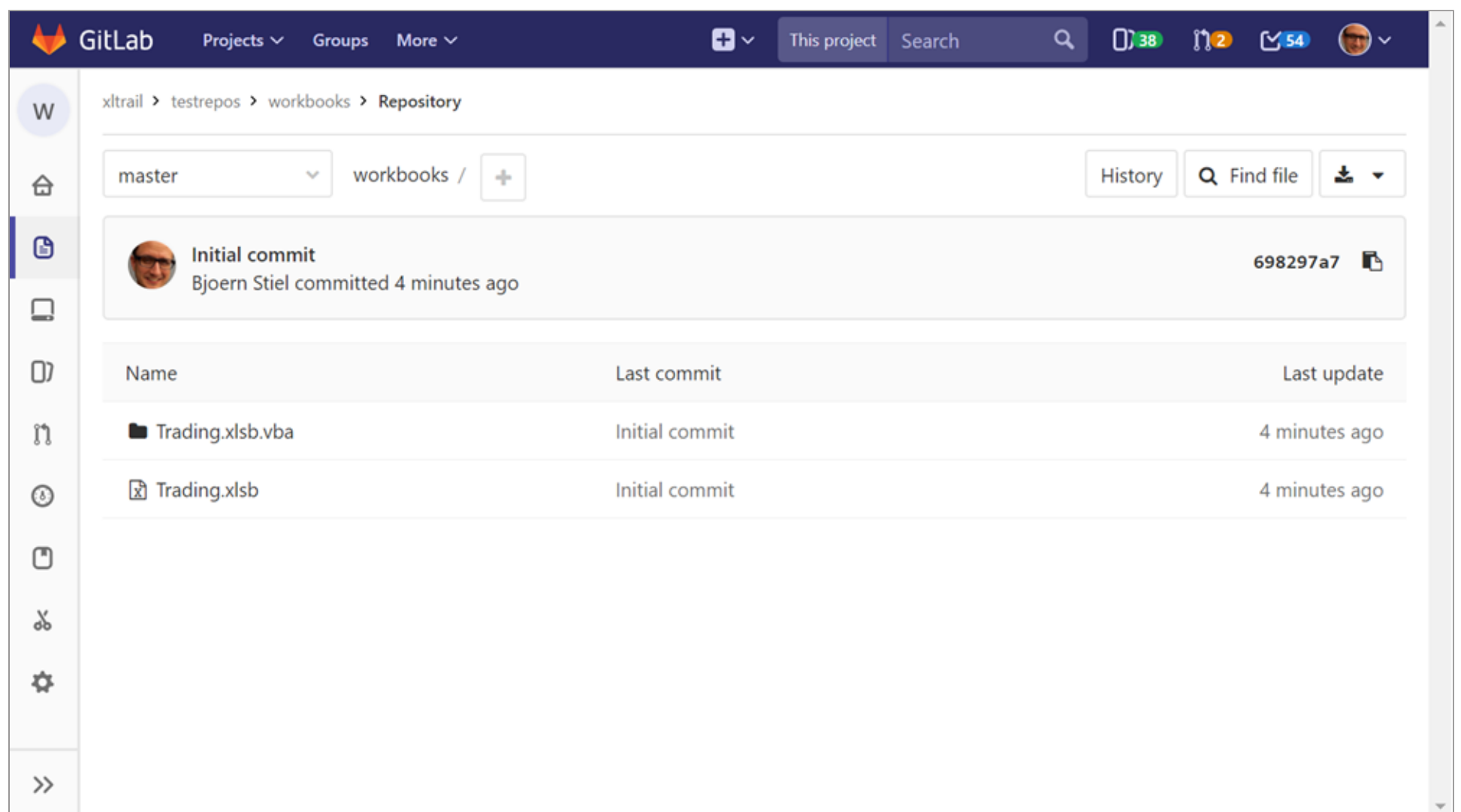
Last updated on August 26, 2021

How to use Git hooks to version-control your Excel VBA code

Posted by [Björn Stiel](#) - [44 Comments](#)

Exporting your VBA modules into stand-alone `.bas/.cls/.frm` files is a simple and effective way to make Git aware of code inside your Excel workbooks. The goal is to end up with a separate files for each of your VBA modules so that you can benefit from the Git functionalities.

One common way to achieve this is via Excel's `Workbook.AfterSave` (or `Workbook.BeforeSave`) event. Every time you hit "save" in Excel, some VBA code is executed and saves a copy of your workbook's VBA content to the filesystem. You end up with your VBA files alongside your workbook which can then be pushed to your Git server.



Git hooks

There are a few downsides to using Excel events. You are dependent on Excel, so if you copy your workbook from an email or another folder into your Git repository folder, your VBA export function will not run. Distributing the export function (either via copy and paste or as an Addin) and ensuring it runs reliably is another pain point.

An alternative approach is to exploit Git's built-in hooks. Hooks are programs you can place in a hooks directory to trigger actions at certain points in Git's execution. You can find a list of available hooks in <https://git-scm.com/docs/githooks>. We will use the pre-commit hook to do the following when you call `git commit`:

- extract the VBA modules from your workbook and write them as `.bas` or `.cls` or `.frm` files to your repository
- add these VBA files to your commit via `git add -- ./src.vba`
- finally execute the `git commit` command

A Python script to extract your VBA code

We use the Python package [oletools](#) to extract the VBA code from the Excel file (in fact, this works for any MS Office file). Thus, we no longer have to resort to Excel itself to get hold of the VBA code.

Our script requires Python 3 and oletools. `oletools` can be installed via pip, Python's package manager: `pip install -U oletools`.

Create a file named `pre-commit.py` in the `.git/hooks` folder inside your repository and add the following code:

NOTE: The following code has been updated multiple times according to feedback, last time: 26-Aug-2021

```
import os
import shutil
from oletools.olevba3 import VBA_Parser

EXCEL_FILE_EXTENSIONS = ('xlsb', 'xls', 'xls', 'xla', 'xlt', 'xlam',)
KEEP_NAME = False # Set this to True if you would like to keep "Attribute VB_Name"

def parse(workbook_path):
    vba_path = 'src.vba'
    vba_parser = VBA_Parser(workbook_path)
    vba_modules = vba_parser.extract_all_macros() if vba_parser.detect_vba_macros() else []

    for _, _, filename, content in vba_modules:
        lines = []
        if '\r\n' in content:
            lines = content.split('\r\n')
        else:
            lines = content.split('\n')
        if lines:
            content = []
            for line in lines:
                if line.startswith('Attribute') and 'VB_' in line:
                    if 'VB_Name' in line and KEEP_NAME:
                        content.append(line)
                else:
                    content.append(line)
            if content and content[-1] == '':
                content.pop(len(content)-1)
            non_empty_lines_of_code = len([c for c in content if c])
            if non_empty_lines_of_code > 0:
                if not os.path.exists(os.path.join(vba_path)):
                    os.makedirs(vba_path)
                with open(os.path.join(vba_path, filename), 'w', encoding='utf-8') as f:
                    f.write('\n'.join(content))

if __name__ == '__main__':
    for root, dirs, files in os.walk('.'):
        for f in dirs:
            if f.endswith('.vba'):
                shutil.rmtree(os.path.join(root, f))

        for f in files:
            if f.endswith(EXCEL_FILE_EXTENSIONS):
                parse(os.path.join(root, f))
```

This Python script finds any Excel files that can contain VBA and dumps the content into a subfolder named `src.vba`. Note that this script works unchanged if you only use 1 workbook. If you have more than one Workbook in one repository, you'll need to replace `vba_path = 'src.vba'` with `vba_path = workbook_path + '.vba'` and edit your `pre-commit` file below accordingly to have a `git add -- ./<workbookname>.vba` line for each of your workbooks.

Setting up the Git pre-commit hook

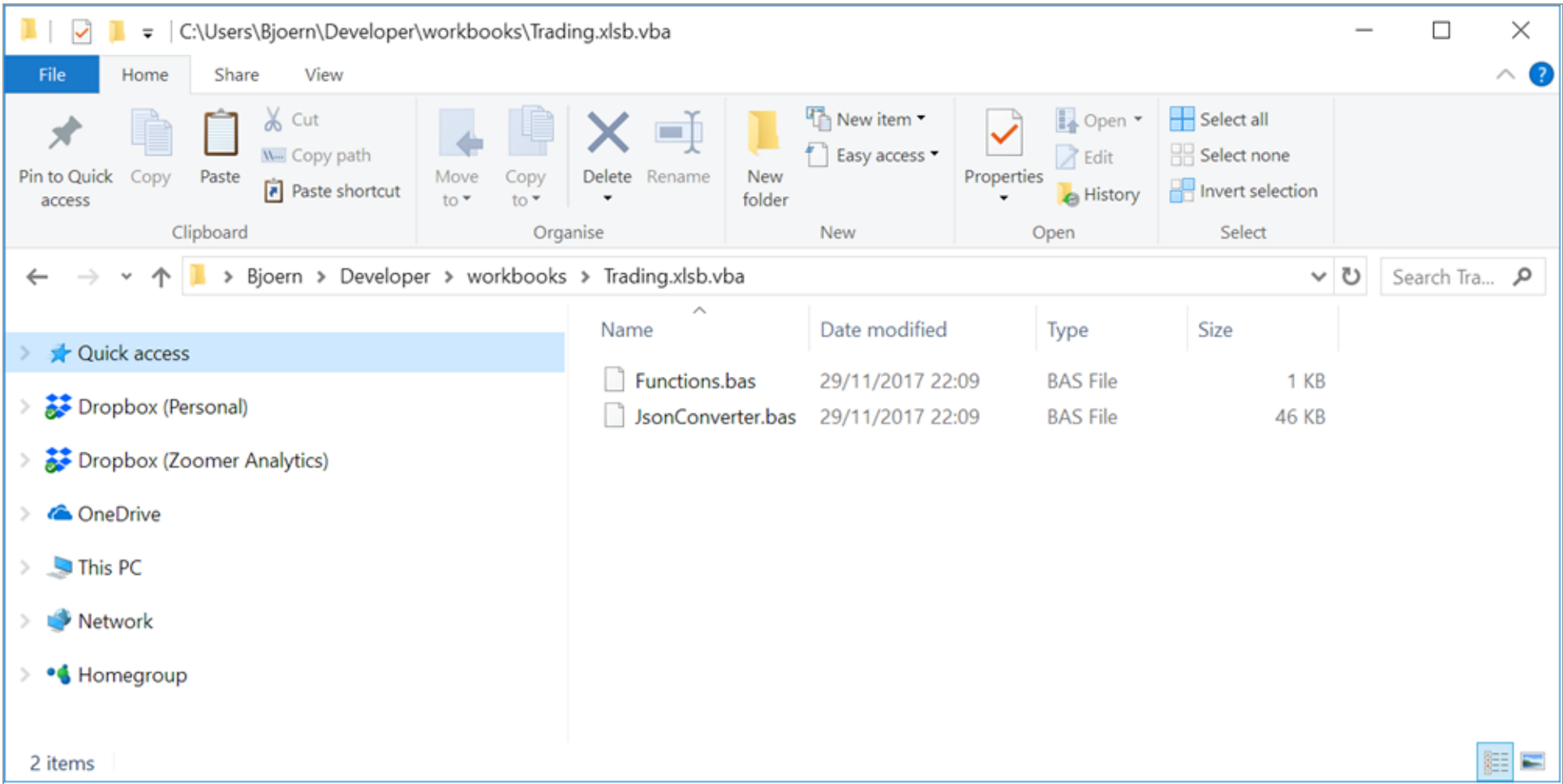
The only missing bit is to set up the Git `pre-commit` now. Create the file `pre-commit` in `.git/hooks` inside your repository and add the following code:

```
#!/bin/sh

python .git/hooks/pre-commit.py
git add -- ./src.vba
```

Note: On Mac you need to run `chmod +x .git/hooks/pre-commit`, otherwise it will not trigger the hook.

From now on, when you execute `git commit`, the hook extracts the VBA code and adds it to your commit. You automatically end up with the `.bas` files without having to rely on Excel.



Conclusion

In order to automagically export your workbook’s VBA modules into stand-alone text files on every `git commit`, you need:

- Python with `oletools` installed and
- the files `pre-commit` and `pre-commit.py` in the `.git/hooks` directory

With that in place, any `git commit` automatically takes care of dumping your workbooks VBA content as text files to your filesystem. Which is something that Git understands well.

Before you go: a zero-setup, open-source alternative

If you are less of a DIY person, I recommend checking out our free, open-source Git extension [Git XL](#). Instead of having to rely on a workaround, xltrail client is a Git extension that integrates directly with Git and makes `git diff` work with Excel workbook files. It also supports `git merge` for those instances where your colleague works on the same workbook and you need to merge their changes in. Plus, xltrail client goes a step beyond the VBA-only approach and also understands sheets. For installation instructions, docs and an example video, check out the [docs](#) and the [GitHub repository](#).

[← Previous Post](#)

[Back to Blog](#)

G

Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name



Share

BestNewestOldest

F

Falko

5 years ago

Hi, I used this tool the last time around one year ago and everything just worked fine for me. When I tried out again today with latest Python 3.7.4 and oletools 0.54.2 it didn't work anymore. After trying around a little bit I could figure out, that I had to downgrade to oletools 0.53.1 and everythings works again like expected. I'm not a Python specialist so can't help you to fix the problem. Only wanted to let you and other users know about it. Maybe someone else can provide a fix here.

20Reply • Share ›



Felix ZumsteinMod → Falko

5 years ago

Hi Falko, thanks for letting us know about this issue and also your workaround! We'll be looking into this and update accordingly.

00Reply • Share ›



Doug → Felix Zumstein

3 years ago

Looks like this is still an issue. Falko's workaround worked for me.

00Reply • Share ›



Felix ZumsteinMod

5 years ago edited

We have updated the code to work correctly with Python 3.7, to optionally include/exclude the header lines in the exported modules and to fix an issue which wasn't properly committing new or deleted modules. Also, it keeps now the proper file extension (bas/cls/frm).

20Reply • Share ›



schifazl

6 years ago

This is great, but what about reimporting the files? E.g. when multiple people work on the same repo and need to also pull changes and merge other perople's work with theirs, reimporting would help a lot

20Reply • Share ›



Bjoern StielMod → schifazl

6 years ago

yes, correct, that way round is a lot tickier as you need to be able to write back into your workbook (which, as far as I know, no open-source Python package supports). So, either you have to rely on VBA itself (not so nice) or you might want to check out our open-source xltrail client: <https://www.xltrail.com/client> - which supports merging (it is built on top of a commercial C# package which supports writing). You can use the hooks and the extension side-by-side. Feel free to comment here (or drop me an email bjoern.stiel@zoomeranalytics.com) if you need any help!

00Reply • Share ›



Simon

4 years ago

Is there a way to get this working with MS Access as well? That would solve a whole bunch of issues for me at work

10Reply • Share ›



Felix ZumsteinMod → Simon

4 years ago

You should be able to just replace the EXCEL_FILE_EXTENSIONS with the extensions of MS access which should make it work.

10Reply • Share ›



Simon → Felix Zumstein



Simon

→ Felix Zumstein

4 years ago edited

I did try adding to that line:

```
EXCEL_FILE_EXTENSIONS = ('xlsb', 'xls', 'xlsm', 'xla', 'xlt', 'xlam', 'accdb')
```

But it gave me the following error:

```
Traceback (most recent call last):
File ".git/hooks/pre-commit.py", line 48, in <module>
parse(os.path.join(root, f))
File ".git/hooks/pre-commit.py", line 12, in parse
vba_parser = VBA_Parser(workbook_path)
File "C:\Program Files\Python37\lib\site-packages\oletools\olevba3.py", line 2407, in __init__

raise FileOpenError(msg)
oletools.olevba3.FileOpenError: Failed to open file .\Very Noter.accdb is not a supported file type, cannot
extract VBA Macros.

fatal: pathspec './src.vba' did not match any files
Completed with errors, see above.
```

I'll try removing all the excel extensions and just leaving accdb.

EDIT: Same error...

0 0 Reply • Share ›



Felix Zumstein Mod → Simon

4 years ago

Sorry, just checked and olevba only supports Word, Excel and PowerPoint but not Access. So no, it won't work.

0 0 Reply • Share ›



Simon → Felix Zumstein

4 years ago

Rats. I wonder if there's a module that does support it...

0 0 Reply • Share ›

D

Daniel Gesua → Simon

4 years ago

Hi Simon and Felix,

I spent the last two days working on some python code that does something similar for MS Access modules (class, standard and form modules), and queries and I think I have something going. Thing is I'm new to the open source community and don't quite know how to get started.

It uses win32com to do all the heavy lifting. It creates a new folder in the directory of the .accdb file of interest called "git_exports" which itself contains two directories ("modules" and "queries") where it then opens the database of interest, loops through the relevant objects, reads their code and saves them in the correct folder with the correct extension.

Maybe some day I'll be able to say I took part in creating a cool project/website like this one hahaha.

-DG

1 0 Reply • Share ›



Simon → Daniel Gesua

4 years ago

Oh nice work Daniel! Do you fancy publishing your git repository, and maybe we can iterate on it a bit (if necessary)?

0 0 Reply • Share ›

D

Daniel Gesua → Simon

4 years ago

So? Did it work?

0 0 Reply • Share ›



Simon → Daniel Gesua

4 years ago

I'm overseas at the mo but I'll have a look when I get back to work. Might be worth hitting up Felix to see if he wants to incorporate it into his project

0 0 Reply • Share ›

D

Daniel Gesua → Simon

— 🚩

4 years ago

Safe travels.

0 0 Reply • Share ›

D

Daniel Gesua → Simon

— 🚩

4 years ago

Hi Simon,

Does the free version of GitHub allow me to publish and collaborate on free open source projects such as this one?

Thanks,

Daniel Gesua

0 0 Reply • Share ›

D

Daniel Gesua → Daniel Gesua

— 🚩

4 years ago

Hi Simon,

Never mind. I answered my own question.

I just made it and published it. Please don't judge my work too harshly hahaha. I'm kinda new to python and I am a *total* novice to git.

Here's the link:
<https://github.com/danielge...>

I'd love to hear your thoughts, be they guidance, ideas, or constructive criticism.

Thanks,

Daniel Gesua

0 0 Reply • Share ›

R

Rado

— 🚩

5 years ago

Hi Bjoern,

Nice tool. I am still getting this error:
Traceback (most recent call last):
File ".git/hooks/[pre-commit.py](#)", line 42, in <module>
parse(os.path.join(root, f))
File ".git/hooks/[pre-commit.py](#)", line 30, in parse
with open(os.path.join(vba_path, filename), 'w', encoding='utf-8') as f:
FileNotFoundError: [Errno 2] No such file or directory: 'src.vba\\Tento_zo\\x9ait.bin'
I have a xlsx file. It seems that the code can't access the vbaProject.bin file.

Any clue? Thanks a lot

1 0 Reply • Share ›



Felix Zumstein Mod → Rado

— 🚩

5 years ago

Hey Rado, would you be able to send me a sample workbook in order to reproduce the issue? You'll find my email on my GH profile: <https://github.com/fzumstein>

0 0 Reply • Share ›

R

Rado → Felix Zumstein

— 🚩

5 years ago edited

Hey Felix, I found a problem. It seems that is a problem with encoding. I have slovak version of Excel, that is encoded in cp1250. Unfortunately, the names of vba objects are also translated to slovak. Also part of the code itself like MsgBoxes are written in slovak. I fixed the content encoding by changing latin-1 to cp1250 in decode function content.decode('cp1250'). But I am not able to fix in standard way the filename variable - the product of vba_parser that contains escape character \\x9a <http://www.codetable.net/he...> - the real cause of my error. It seems to me that vba parser is producing a wrong file name. To be precise it is a vba workbook name.

3 0 Reply • Share ›



LeB   Rado

5 years ago

I had somehow similar error but caused by national character in VBA source code comment. I fixed it by encoding content written to file as unicode. This should also work in your case if you encode file name.

1 0 Reply • Share ›



Rado  LeB 

5 years ago

Hi LeB, no that's a different case. The problem is, if you have a different language version of Excel then English, all the names of the vba objects like workbooks, sheets, forms, etc... are also translated to that language by default. I do not have workbook vba object named "This_workbook" but is named "Tento_zošit" in Slovak. And letter "š" used in the name is that replaced by escape character mentioned above. In Excel graphical user interface you can rename workbook to any name you like but in the vba code behind there is still its default name affected by your language version of your Excel. The only way how to fix it is rename of vba objects in Excel vba editor. But in my opinion, this is not a standard way how to fix it. The problem is that vba_parser that is used by code displayed here is wrongly expecting that workbook's name will be in an exact encoding (or an exact language/languages) and that is a wrong premise.

1 0 Reply • Share ›



Joe Albert

6 years ago

Heyyy! This is super useful [@Bjoern Stiel](#)

One quick question-- this auto appends .bas to every file in the .xism file. I have a macro that utilizes classes (.cls) and forms (.frm) . Is there a quick workaround to have these exported with the correct extensions?

1 0 Reply • Share ›



Joe Albert  Joe Albert

6 years ago

Actually-- figured this out myself!

For anyone who is interested in having file specific export names I've edited the for loop to pull the filename from the VBA_parser and then use that to write the files! :3

hope this can be helpful! <33

```
def parse(workbook_path):
    vba_path = workbook_path + '.vba'
    vba_parser = VBA_Parser(workbook_path)
    vba_modules = vba_parser.extract_all_macros() if vba_parser.detect_vba_macros() else []

    for _, _, filename, content in vba_modules:
        decoded_content = content.decode('latin-1')
        lines = []
        if '\r\n' in decoded_content:
            lines = decoded_content.split('\r\n')
        else:
            lines = decoded_content.split('\n')
        if lines:
            content = [line for line in lines[1:] if not (
                line.startswith('Attribute') and 'VB_' in line)]
            if content and content[-1] == ":":
                content.pop(len(content)-1)
            non_empty_lines_of_code = len([c for c in content if c])
            if non_empty_lines_of_code > 0:
                if not os.path.exists(os.path.join(vba_path)):
                    os.makedirs(vba_path)
                with open(os.path.join(vba_path, filename), 'w') as f:
                    f.write('\n'.join(content))
```

1 0 Reply • Share ›



schifazi  Joe Albert

6 years ago

Nice, but the indentation is gone. Could you please paste your code in pastebin or some similar site?

2 0 Reply • Share ›



Felix Zumstein Mod  schifazi

5 years ago


We have updated the code in the article to reflect that change now.

0 0 Reply • Share ›

A

Aadu rocks

4 years ago

 [View](#) – uploads.disquscdn.com Hi Guys, I really need some help here. SO when I run this commit through VS 2019 I get this error in yellow below. Also, I tried launching the powershell window in the hooks path and tried running the scrip manually but nothing happens. I ran another test scrip to print hello worls and that worked, so I'm not sure what I'm missing here

1 1 Reply • Share ›



Felix Zumstein

Mod

 Aadu rocks

4 years ago

Do you have python on your PATH? i.e. can you type "python" on a command prompt so it starts a Python interpreter? Maybe also have a look at Git XL which might be an easier zero-config option.

0 0 Reply • Share ›



Nik x.1 Foundation

2 years ago

Can someone point me to a similar solution for PowerPoint?

0 0 Reply • Share ›

L

Luc Paoli

3 years ago

heya, just thought I'd let you know the .decode method broke for me when it was already encoded in latin-1. Fixed by adding:
try:

```
decoded_content = content.decode('latin-1')
except AttributeError:
    decoded_content = content
```

0 0 Reply • Share ›

J

Jon Tankersley

3 years ago

I've stumbled into a couple of issues with the basic script.

- 1) decode fails
- 2) I've got some customUI stuff that should also be 'extracted' as part of the source tree

Here's what I did to get around the decode:

```
try:
    decoded_content = content.decode('latin-1')
except:
    decoded_content = content
```

For the RibbonX customUI stuff (custom ribbon and button images, etc.)

from zipfile import ZipFile # up near the top, it could probably be anywhere.

```
for f in files:
    if f.endswith(EXCEL_FILE_EXTENSIONS):
        parse(os.path.join(root, f))
```

```
with ZipFile(os.path.join(root, f), 'r') as zipObj: # modern Excel is a zip file....
    namelist = zipObj.namelist() # get the list of files in the zipfile
    # we only want customUI/customUI.xml or customUI14.xml and any customUI/images
    namelist = [x for x in namelist if (x.startswith('customUI/customUI') or x.startswith('customUI/images'))]
    if f.find("special_case1") > 0:
        zipObj.extractall("special_case1_VBA",namelist)
    elif f.find("special_case2")> 0:
        zipObj.extractall("special_case2_VBA",namelist)
    else:
        zipObj.extractall(".",namelist)
```

This extracts either (or both) of the customUI xml files and any of the custom images included in the customUI folder.

0 0 Reply • Share ›

S

sirtheta

3 years ago

It seems there has been an update in python. The code above is not working anymore. content doesnt need to decode. Just change
decoded_content = content.decode('latin-1') to decoded_content = content

0 0 Reply • Share ›



Kerry Choy

3 years ago

Was getting:
AttributeError: 'str' object has no attribute 'decode'

Seems that something in the underlying products have changed so the code object does not need to be unencoded now. This hack works for me:

```
# decoded_content = content.decode('latin-1')
decoded_content = content
```

00 Reply • Share ›



Felix Zumstein Mod → Kerry Choy

2 years ago

Thanks, this has now been fixed.

00 Reply • Share ›



Ben Jendrick

4 years ago

Unfortunately, after much trial & error, this doesn't seem to be working for me. It creates the directory just fine, but never puts any files into it. And if I have more than 1 file, it entirely kills the commit from the repo. I love the idea, and would wholeheartedly use this if it was reliable (but I'm not expecting miracles in programming). It's probably a configuration issue on my side, or possibly the oletools version as one person mentioned. Awesome work, and thank you for contributing to the community in this way.

P.S. If I could get this to work with Access, my life of programming there would be 1000 times better. Again, thank you!

00 Reply • Share ›



Scott Jackson

4 years ago

I found myself here with basically the same question: "How do I work with git and my Excel VBA projects?". I'm fairly experienced as a VBA developer, but just starting to learn git. The basic outline here occurred to me -- work with VBA Editor and then export the various standard and class modules. I'm wondering, however, if there's any reason not to try flipping this process around, and work with modules outside of Excel and use .git/hooks to import the modified .bas and .cls files back into the workbook which I can then run/test within the VBA Editor.

I'm thinking this through as I type this, but the primary reason I think I'd like to do this is to be able to switch away from the VBA Editor, which I often wish had some features I see in other text editors like code-folding or block editing or more complex screen splits when I need to see multiple subs from with the same module (best I can do in the VBA Editor is horizontal split of a module and then see two subs in the same module). I'd still have to test within the VBA Editor, but editing in vim or Notepad++ might be more enjoyable. Would I lose Intellisense? Or could I somehow get the Microsoft libraries to show code completion in another editor? (That would be a dealbreaker -- but maybe Visual Studio Code or full-blown Visual Studio would solve that?). As for the importation of the modules back into the Excel workbook, I'm sure Python will handle that, but I think Powershell might do that even better as it can work directly with the MS-Office Object Model. I'll have to play around with it and post back.

00 Reply • Share ›



Fil Schoumi

5 years ago

Hello,
This is very useful!! Thanks a lot for your hard work.
I just have a quick question because for me I cannot have the .bas files or others when i make git commit and the <workbookname>.vba does not get created as well.
I installed Python with oletools and create the files pre-commit and [pre-commit.py](#). So I don't understand why it does not work. Anyhelp would be much appreciated. Thanks in advance.

00 Reply • Share ›



Bjoern Stiel Mod → Fil Schoumi

5 years ago edited

Does this command work at all when you execute it from within your repository's root folder? `python .git/hooks/pre-commit.py`

00 Reply • Share ›

F

FlameOfWrath

6 years ago

This is cool.

Your code needs a small update for Python 2.7. [I know, I'm a dinosaur.]

the line:
from oletools.olevba3 import VBA_Parser

needs to drop the '3' to
from oletools.olevba import VBA_Parser

0

0

Reply • Share ›



Jonathan Hill

6 years ago

Thank you Bjorn for an awesome summary of this workflow! I will definitely be trying it out ;)

0

0

Reply • Share ›



Bjoern Stiel

Mod

➔ Jonathan Hill

6 years ago

Thanks! Let me know if you run into any issues.

0

0

Reply • Share ›



Jonathan Hill

➔ Bjoern Stiel

6 years ago

We just customized the [pre-commit.py](#) and pre-commit hook for our specific use case. For example, I wanted to add the vba files in smaller batches, so I removed ``git add *.bas`` from the pre-commit and we wanted to avoid trying to process hidden/temporary copies of open excel files starting with `~.` Awesome content!

0

0

Reply • Share ›



Bjoern Stiel

Mod

➔ Jonathan Hill

6 years ago

Great! Re the temporary excel files, you can also add `~$*` to the ``.gitignore`` file so that git ignores it altogether (in case you haven't done that yet).

8

0

Reply • Share ›

Subscribe

Privacy

Do Not Sell My Data

About Us

- Team
- Privacy Policy & ToS
- Legal Notice
- Contact Us

Resources

- Blog
- xltrail Docs
- xltrail Status
- White Papers

Products

- xltrail
- Git XL
- xlwings (Open Source)
- xlwings PRO

Follow Us

-  YouTube
-  LinkedIn
-  Newsletter
-  Twitter
-  GitHub
-  Meetup London
-  Meetup NYC

© 2024 Zoomer Analytics GmbH. All rights reserved.