Local installation of the technical manual

Prerequisites

To be able to work with Sphinx and RST files you will need :

- Git
- python 3.10 or higher
- LATEX (needed for pdf generation)

Windows

- GIT
  - install and download : https://git-scm.com/download/win

- Python
  - Installation of packages : https://www.python.org/downloads/windows/
    You can install python 3.x.x (anything from 3.10 should work)
- Latex
  - You can use MiKTeX https://miktex.org/download
    - If asked it will be simpler to configure the option of allow all package to install
  - Install perl.exe : https://www.perl.org/get.html

Debian

- GIT
  - apt-get install git
- Python3 is already available on most distributions of linux
- For PDF generation :
  - apt-get install latexmk
  - apt-get install texlive-fonts-recommended texlive-latex-recommended exlive-latex-extra

- ○ apt-get install dvipng texlive-xetex fonts-freefont-otf
- ○ optional : apt-get install texlive-lang-all

To connect afterwards to Git it is best to use SSH. To find out how to do that : https://docs.github.com/en/authentication/connecting-to-github-with-ssh

Two ways of local installation

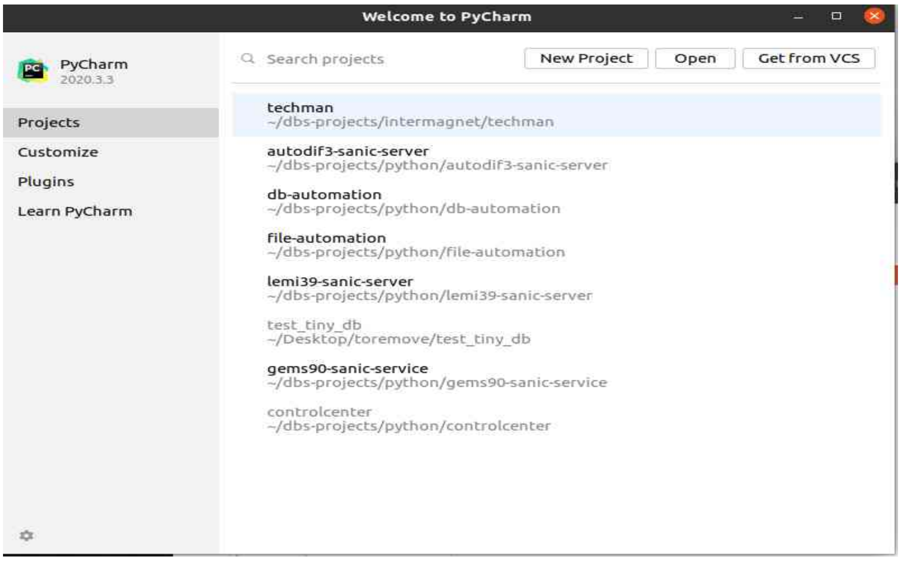
Using Pycharm community edition

To make the installation easier and have a python programming environment in your tooling set you can use : https://www.jetbrains.com/pycharm/download/

Dowload the community edition and install it on your computer.

This tool facilitate the work with git, python and virtual environments for python.

Once you open it, it will ask you to create a project :



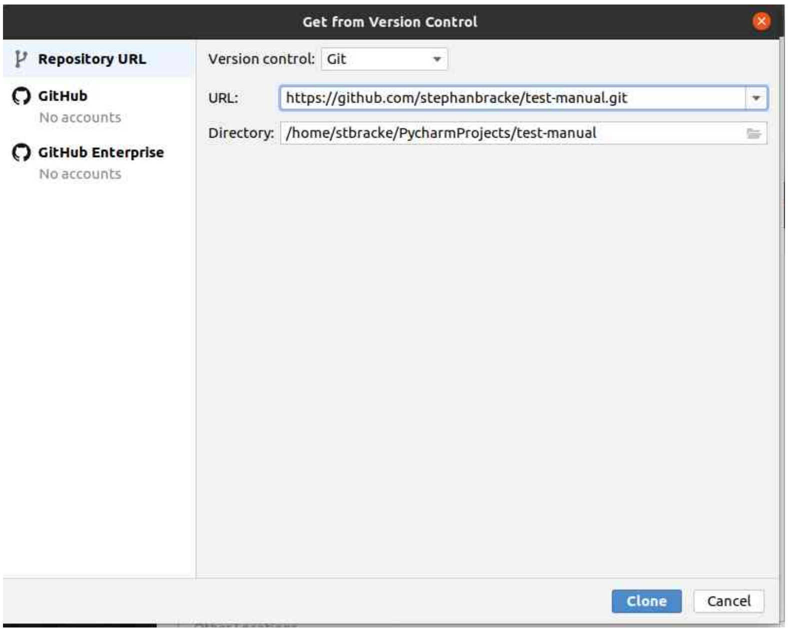
techman
~/dbs-projects/intermagnet/techman
autodif3-sanic-server
~/dbs-projects/python/autodif3-sanic-server
db-automation
~/dbs-projects/python/db-automation
file-automation
~/dbs-projects/python/file-automation
lemi39-sanic-server
~/dbs-projects/python/lemi39-sanic-server
test_tiny_db
~/Desktop/toremove/test_tiny_db
gems90-sanic-service
~/dbs-projects/python/gems90-sanic-service
controlcenter
~/dbs-projects/python/controlcenter

Click on get from VCS (version control system)

We choose to use git and type in the url needed :

git@github.com:INTERMAGNET/imag-tech-man.git

From 2021 HTTPS username/password is not supported anymore so you will need to use SSH



Get from Version Control

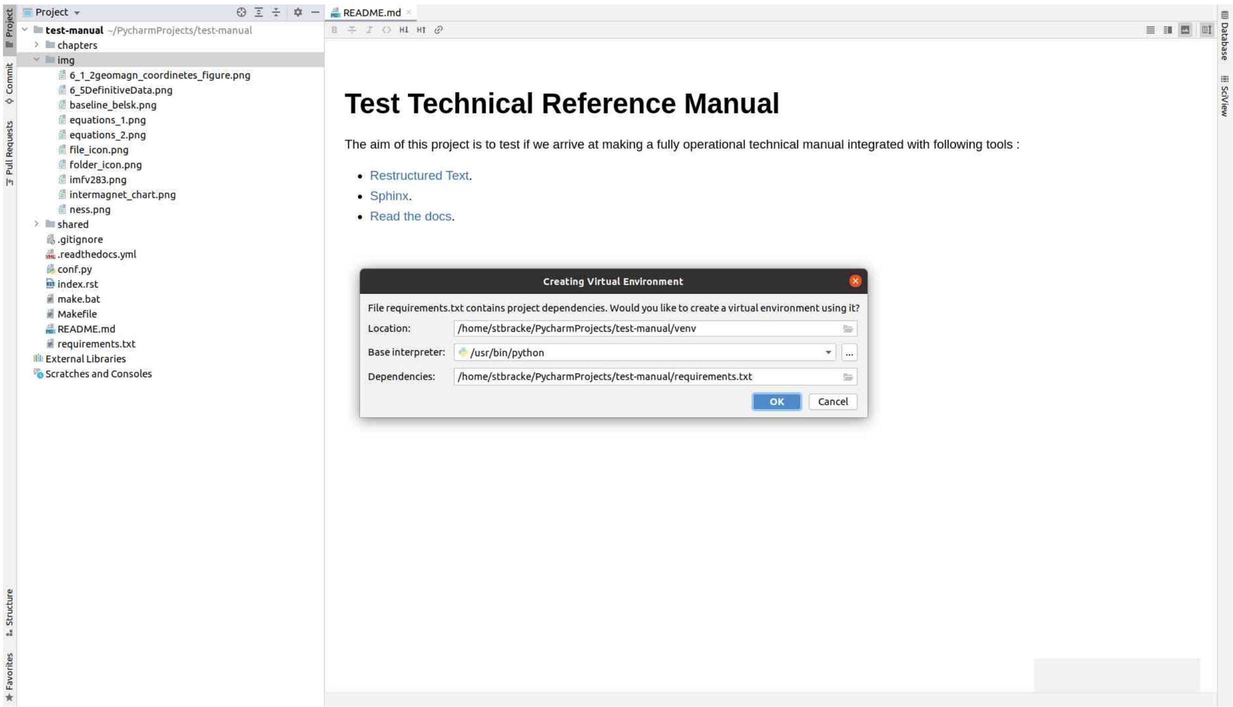
Version control: Git

URL: https://github.com/stephanbracke/test-manual.git

Directory: /home/stbracke/PycharmProjects/test-manual

You click on clone and the git project will be cloned locally on the chosen directory.

After that you will see :



```
test-manual ~/PycharmProjects/test-manual
  chapters
  img
    6_1_2geomagn_coordinetes_figure.png
    6_5DefinitiveData.png
    baseline_belsk.png
    equations_1.png
    equations_2.png
    file_icon.png
    folder_icon.png
    imfv283.png
    intermagnet_chart.png
    ness.png
  shared
  .gitignore
  .readthedocs.yml
  conf.py
  index.rst
  make.bat
  Makefile
  README.md
  requirements.txt
External Libraries
Scratches and Consoles
```

README.md

Test Technical Reference Manual

The aim of this project is to test if we arrive at making a fully operational technical manual integrated with following tools :

- Restructured Text.
- Sphinx.
- Read the docs.

Creating Virtual Environment

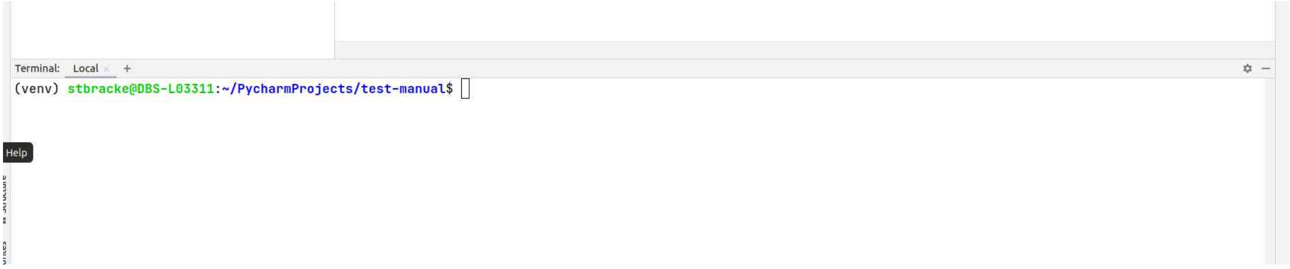
File requirements.txt contains project dependencies. Would you like to create a virtual environment using it?

Location: /home/stbracke/PycharmProjects/test-manual/venv

Base interpreter: /usr/bin/python

Dependencies: /home/stbracke/PycharmProjects/test-manual/requirements.txt

Pycharm will detect the requirements.txt and ask if he has to create the virtual environment for you. You click on ok. This will create and download the dependencies. Once it is finished you open on the bottom view a terminal window.



```
(venv) stbracke@DBS-L03311:~/PycharmProjects/test-manual$
```

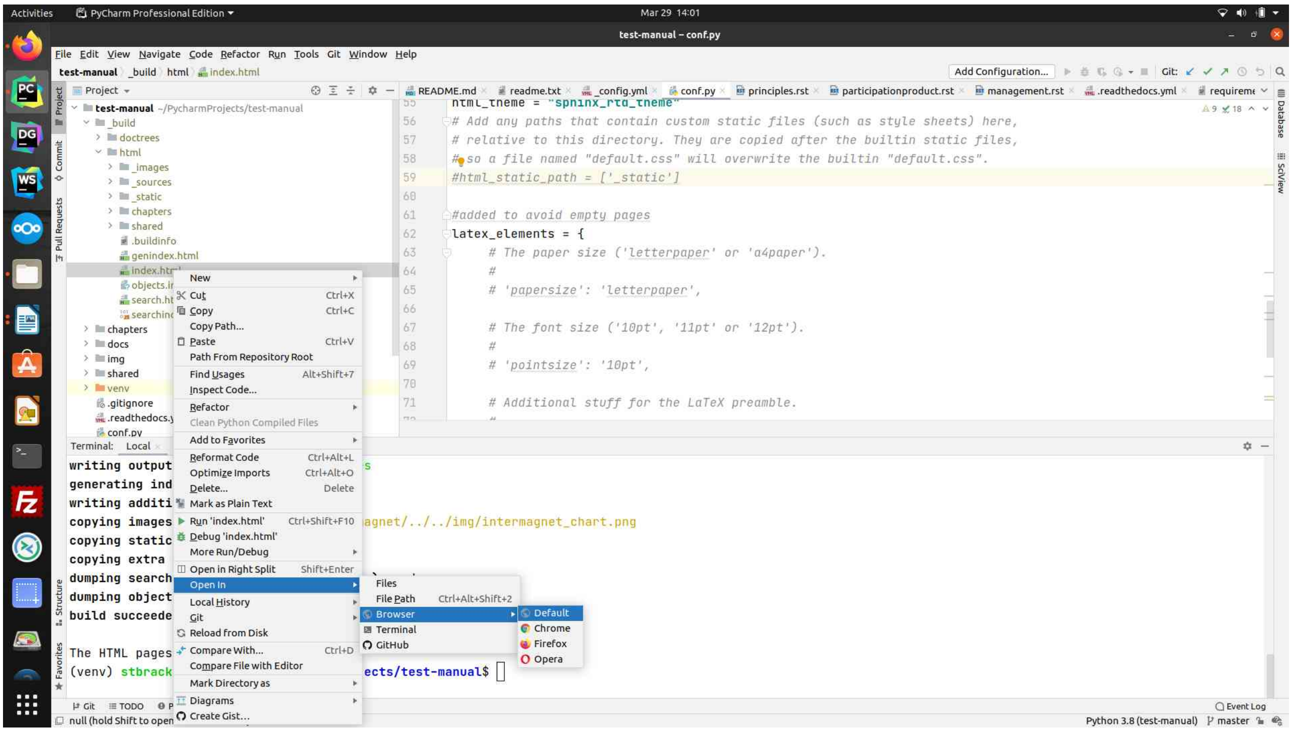
If all was successful You notice (venv) before the prompt,if not you can open a new terminal by clicking on the plus sign or restart your environment.

If you want to learn more on the use of python virtual environments you can checkout : https://docs.python.org/3.10/library/venv.html

Once you have it up and running you can start test installation by typing in the terminal window :

- make html
- make latexpdf
- make epub

The result is build in the _build directory which you can now explore and test.



```
html_theme = "sphinx_rtd_theme"
# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named "default.css" will overwrite the builtin "default.css".
#html_static_path = ['_static']

#added to avoid empty pages
latex_elements = {
    # The paper size ('letterpaper' or 'a4paper').
    #
    # 'papersize': 'letterpaper',

    # The font size ('10pt', '11pt' or '12pt').
    #
    # 'pointsize': '10pt',

    # Additional stuff for the LaTeX preamble.
```

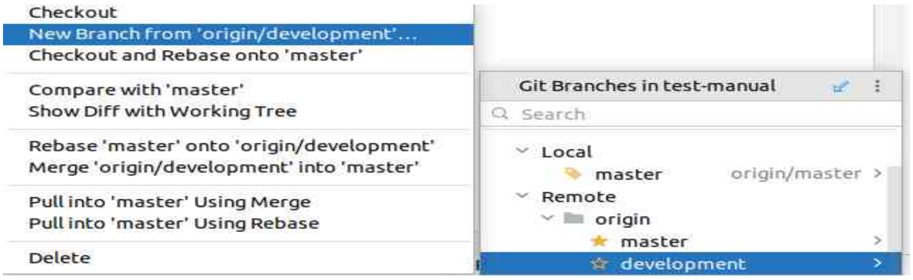
```
writing output                                   s
generating ind
writing additi
copying images                                   agnet/../../img/intermagnet_chart.png
copying static
copying extra
dumping search
dumping object
build succeede

The HTML pages
(venv) stbrack                                   ects/test-manual$
```

You can open it up from within PyCharm by right click and select open by browser.

If you look at the left bottom corner you will notice the following info :

- The python version used
- The branch you are currently on : master

If we want to make changes and afterwards commit them on the shared repository we will need to switch to the development branch. We just click on the current branch (left hand corner). Then use a go to the directory Remote/origin/development.


Checkout
New Branch from 'origin/development'...
Checkout and Rebase onto 'master'
Compare with 'master'
Show Diff with Working Tree
Rebase 'master' onto 'origin/development'
Merge 'origin/development' into 'master'
Pull into 'master' Using Merge
Pull into 'master' Using Rebase
Delete
Git Branches in test-manual
Search
Local
master
origin/master
Remote
origin
master
development

There we can choose for Checkout. This will create a local branch development that will be compared with the remote branch development

Now we are ready to make our first changes and commit it to Git from within PyCharm on the local branch development.

To keep good practice we need to :

- create a local new branch to make the changes into
- commit the changes
- push changes/local branch to git
- Ask a pull request

A small explanation how you can do that from within Pycharm is included at the end of the document.

Raw Installation (in command/terminal window)

- Clone project locally
  - Open command prompt/terminal (cmd on windows)
  - go into installation directory and type
    - git clone git@github.com:INTERMAGNET/imag-tech-man.git

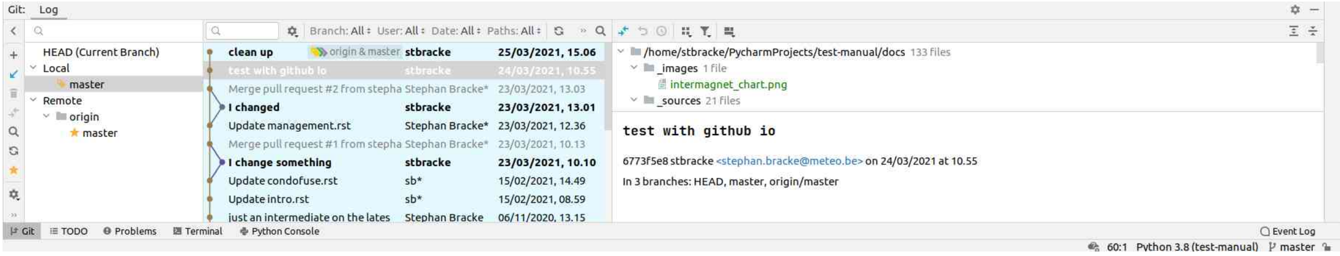
- If you want to work clean you work with a python virtual environment
  This step can be skipped if you want to install the needed packages in the global python distribution
  - create virtual environment for python
    - `python3 -m venv techman-venv`
    - `this creates a directory techman-venv with all necessary binaries of python`
    - `you will now have two directories`
      - `test-manual containing your local github repo`
      - `techman-venv your python virtual environment`
  - activate virtual environment
    - windows : techman-venv\Scripts\activate.bat
    - linux : source techman-venv/bin/activate
  - if activation worked you will see (techman-venv) before your prompt
- Now go into your github directory test-manual
  - here you can use git commands to view commit etc
    - git status : gives overview of the current git repository
    - git branch -a : all your branches
    - git remote -v : all remote repositories normally original pointing to the remote
- Install now required packages (into your virtual environment if activate)
  - pip3 install -r requirements.txt
- from now you should be able to generate the html and pdf by typing
  - make html
  - make latexpdf
- This will create a directory _build with all necessary files

If you are a fan of command line working you can do nearly everything with git from command line. Here a small introduction : https://git-scm.com/docs/user-manual but there is a lot of documentation available on the internet.

Working with git from within PyCharm

Working with git locally can be difficult if you don’t do it regularly. Therefore plenty tools exist to make it easier. One we can use is PyCharm which comes with git integration.

It has a visual view on Git the bottom left.



clean up	origin & master	stbracke	25/03/2021, 15.06
test with github io		stbracke	24/03/2021, 10.55
Merge pull request #2 from stepha		Stephan Bracke*	23/03/2021, 13.03
I changed		stbracke	23/03/2021, 13.01
Update management.rst		Stephan Bracke*	23/03/2021, 12.36
Merge pull request #1 from stepha		Stephan Bracke*	23/03/2021, 10.13
I change something		stbracke	23/03/2021, 10.10
Update condofuse.rst		sb*	15/02/2021, 14.49
Update intro.rst		sb*	15/02/2021, 08.59
just an intermediate on the lates		Stephan Bracke	06/11/2020, 13.15

/home/stbracke/PycharmProjects/test-manual/docs 133 files
_images 1 file
intermagnet_chart.png
_sources 21 files

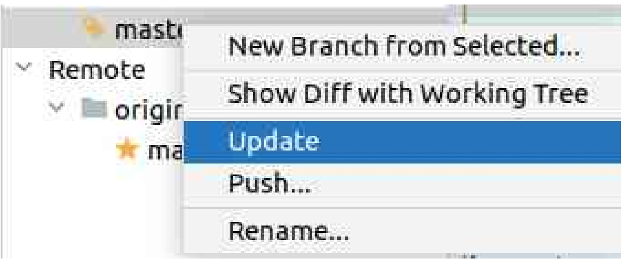
test with github io

6773f5e8 stbracke <stephan.bracke@meteo.be> on 24/03/2021 at 10.55

In 3 branches: HEAD, master, origin/master

This view gives us the possibility to see all that is currently available. One thing to understand is that git always shows the situation with local copies (even of the remote repository). To be sure that we really have the latest view. It is always nice to do an update of the local repository view and contact the remote server before we start our work.

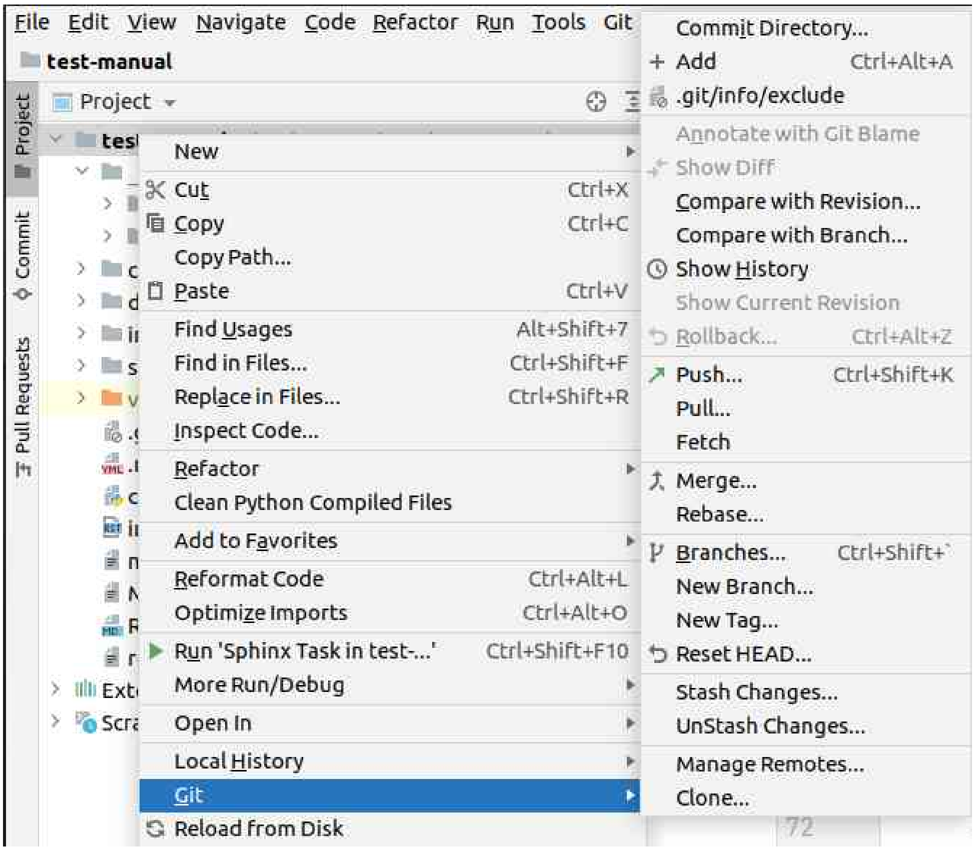
This can be done by right click on the local master and select update from the menu.



This is the e can be done by right click on the local master an quivalent of the fetch command and a fast forward pull to the master branch making the local master in sync with the remote master. It also fetches all remote branches if new ones are available.

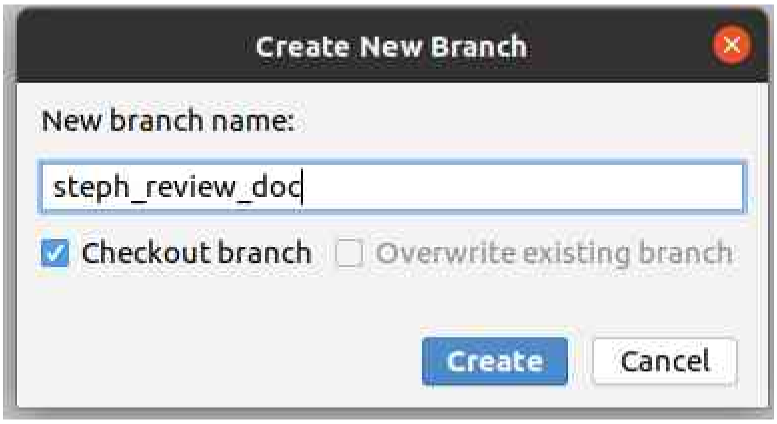
We are now ready to do some changes to the documentation. For this we will create a new branch and checkout this new branch.

If we click right on the top level directory we will have the possibility to do various git operations on it :



This main menu can be used for many actions. Here we will select New Branch

Following popup appears

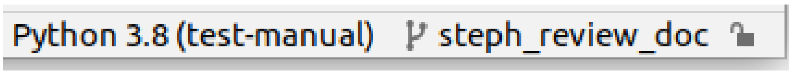


Create New Branch

New branch name: steph_review_doc

☑ Checkout branch □ Overwrite existing branch

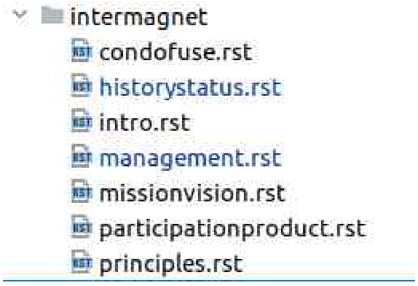
You choose a logical name for the branch that identifies the work you will do.
The Checkout is needed to switch from the master to the new branch.



Python 3.8 (test-manual) steph_review_doc

In the bottom right corner we can verify that we are on the correct branch.

If this is ok we can now safely change stuff in our own private playground branch.
PyCharm will indicate changed files in another color.



```
intermagnet
    condofuse.rst
    historystatus.rst
    intro.rst
    management.rst
    missionvision.rst
    participationproduct.rst
    principles.rst
```

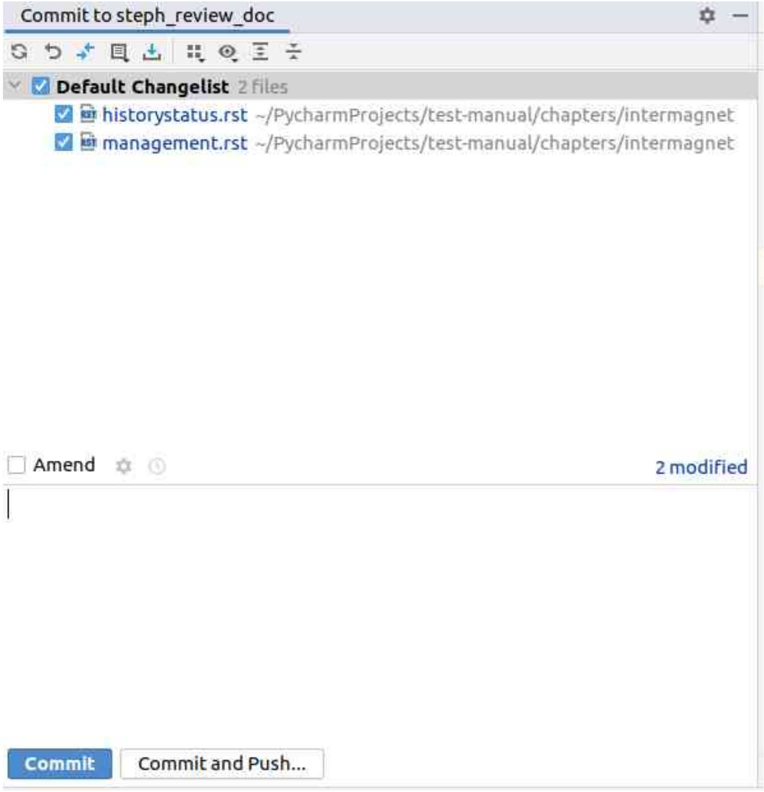
You can type in make html to build the website with current changes included.
Sometimes always rebuilding the html can leave traces of the previous build to make sure that everything is cleaned up before building we can type :

- make clean html

After you have changed sufficient work you can decide to commit and/or push the changes.

- A commit is grouping changes together and make them available as tracing on your local branch
- the push will create a remote branch on the server with the changes.

If we want to do that we just click right on the top directory and select the commit directory entry. This will popup :



Commit to steph_review_doc

☑ Default Changelist 2 files

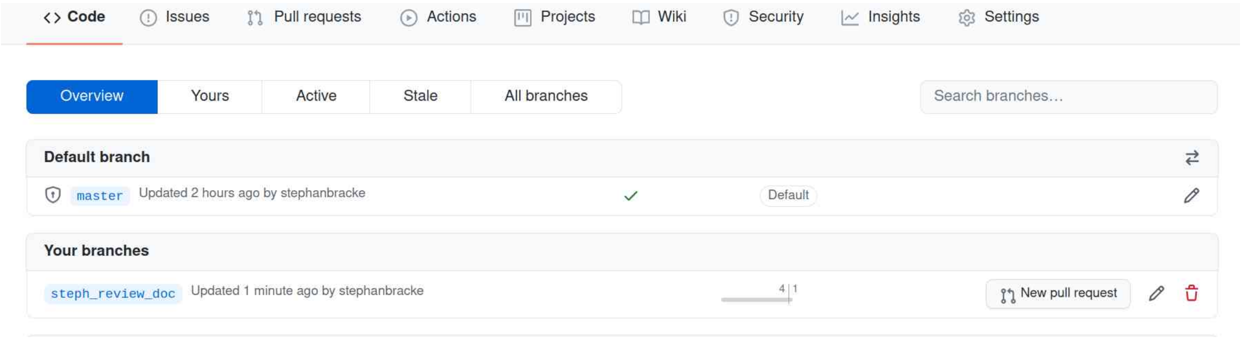
☑ historystatus.rst ~/PycharmProjects/test-manual/chapters/intermagnet

☑ management.rst ~/PycharmProjects/test-manual/chapters/intermagnet

□ Amend

2 modified

Now you select/deselect the files you want to include in your commit and give a meaningful reason
If you choose to commit it will only be local and you need to do a push afterwards.
When you push it now a remote branch will be created where you can create a pull request.
On the remote git you will see this branch beside the master on the remote repository.



Default branch

master Updated 2 hours ago by stephanbracke ✓ Default

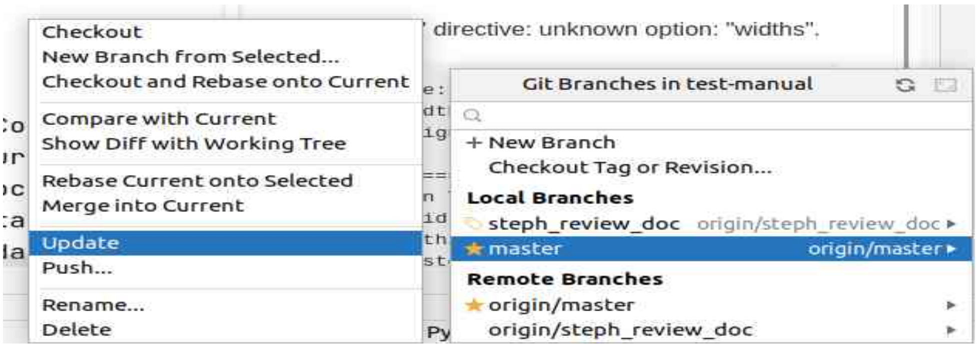
Your branches

steph_review_doc Updated 1 minute ago by stephanbracke 4 | 1

Now you can generate a pull request to include your work.

The pull request will be a task to be reviewed. If accepted you can now checkout your local master again and safely delete your local work branch.(left hand corner)

To switch back to the master you can click on the right hand corner indication of the current active branch.


Checkout
New Branch from Selected...
Checkout and Rebase onto Current
Compare with Current
Show Diff with Working Tree
Rebase Current onto Selected
Merge into Current
Update
Push...
Rename...
Delete
directive: unknown option: "widths".
Git Branches in test-manual
+ New Branch
Checkout Tag or Revision...
Local Branches
steph_review_doc origin/steph_review_doc ▸
★ master origin/master ▸
Remote Branches
★ origin/master ▸
origin/steph_review_doc ▸

By clicking checkout on the local master you will locally switch back to the master. It is also now that you can delete the steph_review_doc (by clicking right on it) and eventually update your master again to have the changes be pulled into the master branch.