

GitHub Documentation

Setting up & version control and sharing

Author: Behdad Shaarbaf Ebrahimi

Setting up

Note: This documentation is mostly compatible with Linux OS.

There are two major ways setting up a repository can happen:

- i. Setting it up from scratch using terminal
- ii. Cloning a repository
- iii. A brief look at documentation by Sphinx

1) Setting up a repository from scratch

The first step is to make sure you have git installed on your system.

In order to check you can open a terminal and type:

`git --version`

This command will show you the version of the git installed. As a result, if you do not have git on your system it will complain about the input "git" command.

Installing Git on linux Debian/Ubuntu :

- Open terminal
- `Sudo apt-get install git` ----- Enter university credentials (password)

If on any other OS you can use this link:

<https://www.linode.com/docs/development/version-control/how-to-install-git-on-linux-mac-and-windows/>

1) Setting up a repository from scratch

After finishing with Git installation, defining a directory as Github repository use the steps below:

- `cd` to directory you want to make it a repository (`cd /hpc/<upi>/My_repo`)
- `git init`.

You will see the message:

Initialized empty Git repository in `~dir/.git/`

NOTE: In order to check and see if you git initialization files are created, you can do:

- Move to you people drive (`~/people/<upi ID>`)
- `ls -A`
- You may find all the folders including git folders created

1) Setting up a repository from scratch

A professional repository should consist of the following parts:

- Source directory (For you scripts and functions and classes)
- Test folder (For test driven scripting)
- Documentation folder (For documenting your scripts functions and classes)
- Examples folder (This is my personal preference and I have seen the group's favor towards having working examples 😊)
- A "README" file as a brief explanation on your GitHub page.
- Optional but useful: a .gitignore file

1) Setting up a repository from scratch

The last step is to put the repository made on the cloud. Steps are:

- `git remote add origin git@github.com:<your_username>/<repository_name>.git`

Note: The repository needs to be created on your profile first. Then you can add the remote address!

- `git pull origin <branch_name>` : the first branch default name is "master"
- `git remote -v` : after doing this command you should be able to see the address to your repository

You make changes to README.rst file and setup the description to the repository. Write some scripts and functions the same as the structure discussed in the previous slide.

After making all the changes:

- `cd to the directory of your repository`
- `git add .`
- `git commit -m "A brief explanation of the changes"`
- `git push origin master`

Now your repository is set up.

For better understanding of reStructuredText (*.rst) format you can refer to the links below:

- <https://github.com/behdadebsh/LungPhD> (fork, clone and see the source files as an example)
- <http://docutils.sourceforge.net/rst.html#user-documentation>

2) Something recommended – Git Prompt

This is a feature that makes your terminal look different when in GitHub repositories. You may find different types and themes for this feature on the internet but, my personal preferred one and easy to setup is provided here:

The easiest way is using git clone.

- `cd ~` (To make sure you are on your home directory)
- `git clone https://github.com/magicmonty/bash-git-prompt.git .bash-git-prompt --depth=1`
- `gedit .bashrc`
- Copy this text into your bashrc file:

```
GIT_PROMPT_ONLY_IN_REPO=1  
source ~/.bash-git-prompt/gitprompt.sh
```

Note: There are other setup methods for git prompt available on:
<https://github.com/magicmonty/bash-git-prompt>

2) Something recommended - Git Prompt

```
✓ /hpc/bsha219/Python/Behdad/lung_segmentation [master L|...33]
15:11 $
✓ /hpc/bsha219/Python/Behdad/lung_segmentation [master L|...33]
15:11 $
✓ /hpc/bsha219/Python/Behdad/lung_segmentation [master L|...33]
15:11 $ cd ..
bsha219@bn371671:/hpc/bsha219/Python/Behdad$ cd lung_segmentation/
✓ /hpc/bsha219/Python/Behdad/lung_segmentation [master L|...33]
15:12 $ git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        README.rst
        docs/

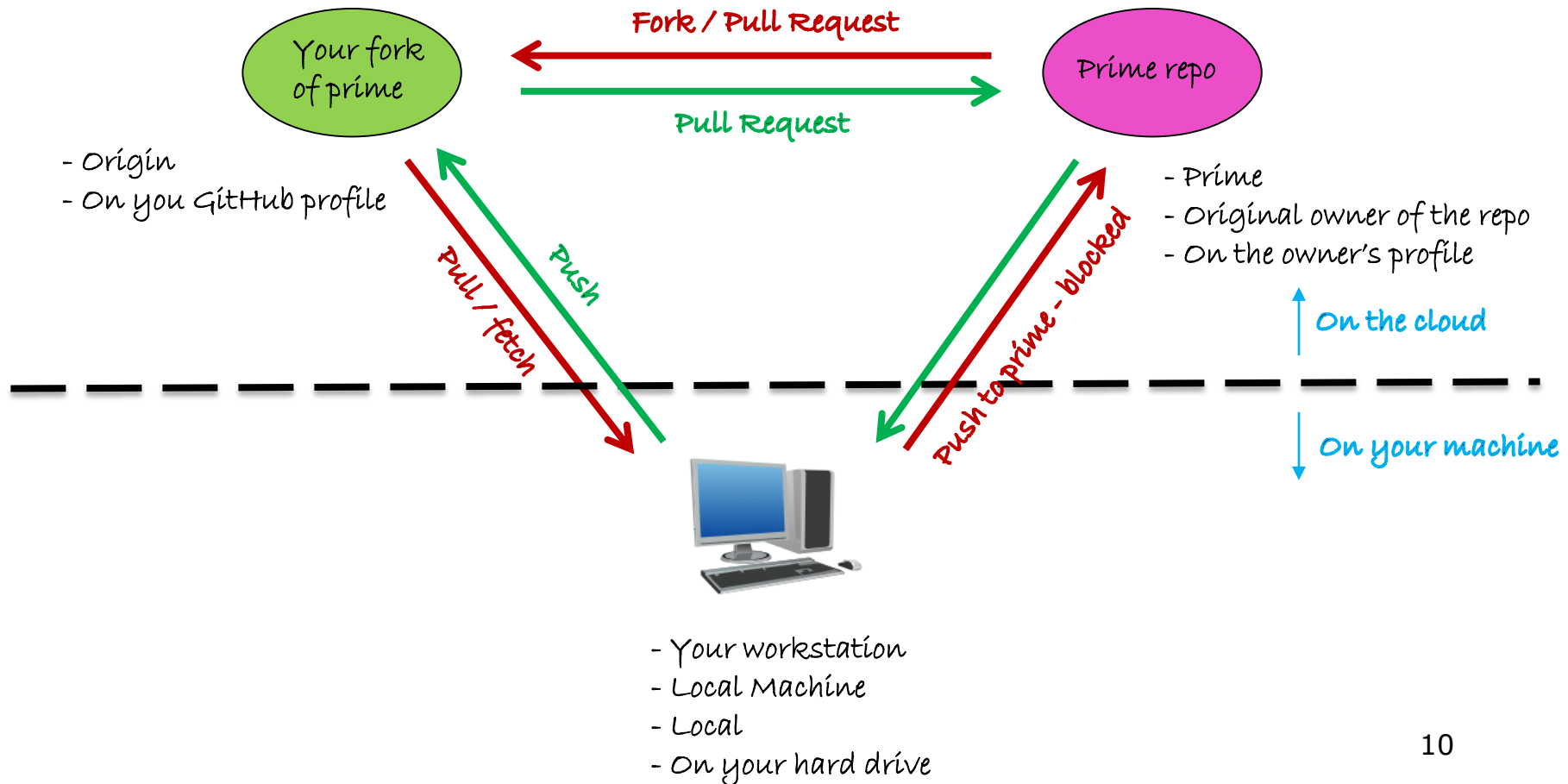
nothing added to commit but untracked files present (use "git add" to track)
✓ /hpc/bsha219/Python/Behdad/lung_segmentation [master L|...33]
15:15 $ ls docs/
_build conf.py index.rst make.bat Makefile _static _templates
✓ /hpc/bsha219/Python/Behdad/lung_segmentation [master L|...33]
15:16 $ vi .gitignore
✓ /hpc/bsha219/Python/Behdad/lung_segmentation [master L|...6]
```


3) Clone a repository

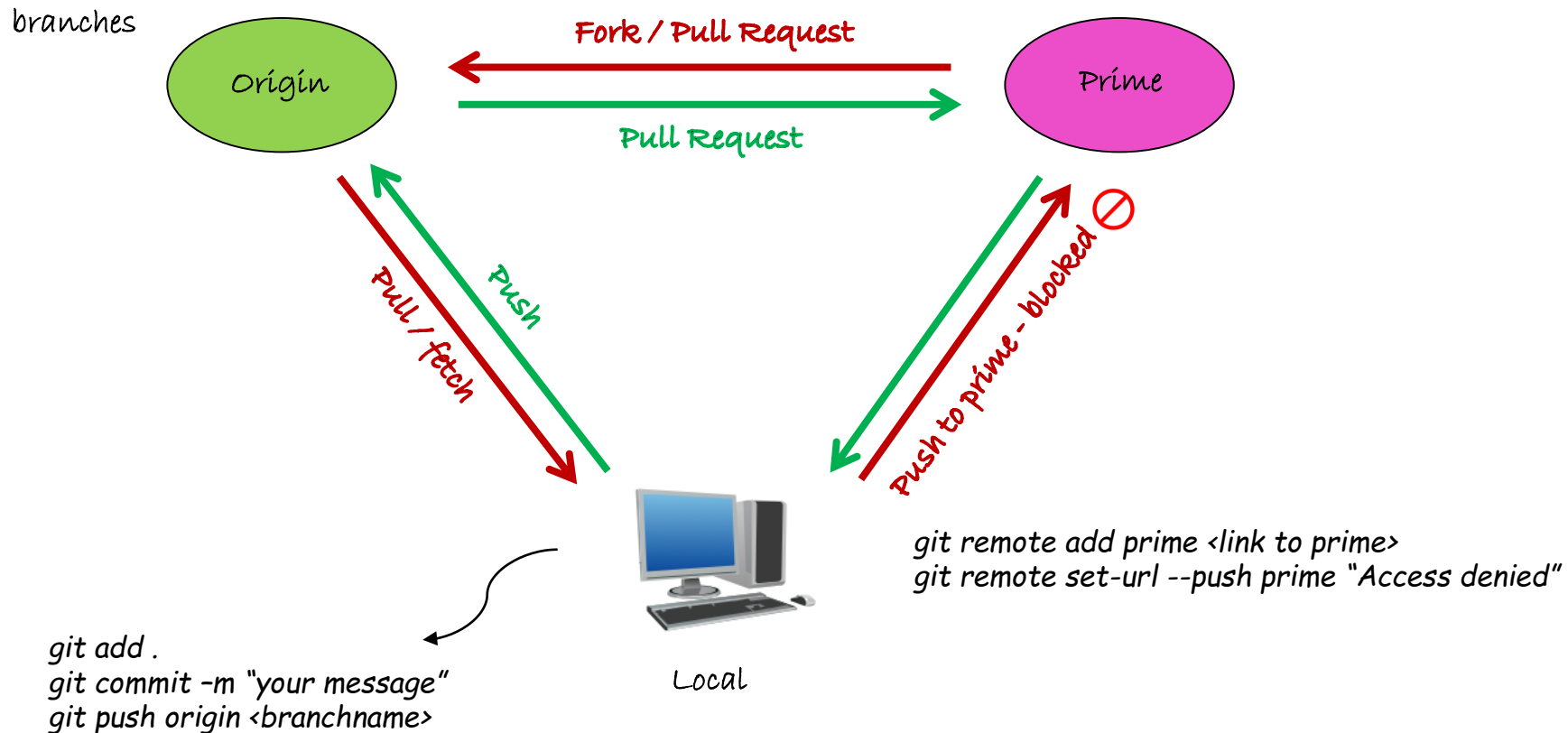
The easiest way to acquire a repository is clone it on your local machine. Some terms that are needed to be defined for consistency between the group:

- Origin
- Prime
- Fork
- Pull
- Push
- Local
- Branch
- Add
- Commit

3) Clone a repository



3) Clone a repository



4) Useful commands

Now that we have familiarized ourselves with useful phrases some git commands are helpful to know:

git remote -v : shows the list of remotes to the repository i.e. origin, prime and any other remotes that defined. Remember origin is known word to GitHub but prime is defined by us.

git branch <branch_name> : creates a new branch with the name given

git checkout <branch_name> : switch between branches

git status : shows the status of the files added and the commits. This step is recommended before doing a push command to check for the last time what is going on the cloud. You do not want to add something that creates a bug in your script.

git fetch : practically only updates the history from the remote since your last pull, therefore, you can do a diff command and see the differences between your local working version and the version on remote (origin, prime, or...)

git merge : merges the changes on you local repository (after fetch) and the changes on your local working

5) Documentation by Sphinx

How to install sphinx:

- Different methods for different OS available:
- `pip install -U sphinx`
- `git clone` (via source)
- `apt-get install python(2/3)-sphinx --` For Debian/Ubuntu
- `conda install`
- MacOS with Homebrew (`brew install ...`) & MacPorts (`sudo port install ...`)

As discussed in GitHub setting up, a "docs" directory is designated for documentation!

For start after installing sphinx, change directory to your docs folder in your repository on your local machine. The first step would be:

sphinx-quickstart

This command will immediately direct us to setting up a documentation. Basically the settings will be set through questions (This will edit the `conf.py` made after getting finished with this). In the next slides, a brief description is provided by screenshots of the terminal.

5) Documentation by Sphinx

```
Terminal
File Edit View Search Terminal Tabs Help

bsha219@bn371671:~$
bsha219@bn371671:~$
bsha219@bn371671:~$ git --version
git version 2.7.4
bsha219@bn371671:~$
bsha219@bn371671:~$ pwd
/people/bsha219
bsha219@bn371671:~$ ls
anaconda2          java.log.15264      Software.tar.gz
cleanup-fluent-bn371671-7410.sh Lung_Behdad         TDPulmonaryToolkit
cleanup-fluent-hpc6-178078.sh matlab_crash_dump.15264-1 __temp_file.8029
cleanup-fluent-hpc6-185125.sh matlab_crash_dump.16549-1 Templates
Desktop            Music               tetra_cmd.log
Documents          perl5              tetra_mesh.uns
Downloads          Pictures           Videos
Fluenterror.log   Public
hs_error_pid15264.log Software
bsha219@bn371671:~$ cd /hpc/bsha219/Python/
bsha219@bn371671:/hpc/bsha219/Python$ ls
Behdad Kaggle.py Mahyar Merryr temp.py
bsha219@bn371671:/hpc/bsha219/Python$ cd Behdad/
bsha219@bn371671:/hpc/bsha219/Python/Behdad$ ls
Lung_Behdad          Lung_Segmentation.py script_seg.py
Lung_HOWARD_Behdad_V1.py Lung_Segmentation.pyc test.png
bsha219@bn371671:/hpc/bsha219/Python/Behdad$ mkdir lung_segmentation
bsha219@bn371671:/hpc/bsha219/Python/Behdad$ cd lung_segmentation/
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation$ git init
Initialized empty Git repository in /hpc_atog/bsha219/Python/Behdad/lung_segmentation/.git/
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation$ ls -A
.git
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation$ mkdir src
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation$ mkdir tests
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation$ mkdir docs
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation$ vi README.rst
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation$ sphinx-
sphinx-apidoc sphinx-autogen sphinx-build sphinx-quickstart
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation$ cd docs/
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ sphinx-quickstart
Welcome to the Sphinx 1.3.6 quickstart utility.

Please enter values for the following settings (just press Enter to
accept a default value, if one is given in brackets).

Enter the root path for documentation.
> Root path for the documentation [.]:
```



← Since in ~/docs just press Enter

5) Documentation by Sphinx

```

Terminal
File Edit View Search Terminal Tabs Help

Terminal
Enter the root path for documentation.
> Root path for the documentation [.]:

You have two options for placing the build directory for Sphinx output.
Either, you use a directory "build" within the root path, or you separate
"source" and "build" directories within the root path.
> Separate source and build directories (y/n) [n]:

Inside the root directory, two more directories will be created; "_templates"
for custom HTML templates and "_static" for custom stylesheets and other static
files. You can enter another prefix (such as ".") to replace the underscore.
> Name prefix for templates and static dir [_]:

The project name will occur in several places in the built documentation.
> Project name: Lung Segmentation
> Author name(s): Behdad Ebrahimi

Sphinx has the notion of a "version" and a "release" for the
software. Each version can have multiple releases. For example, for
Python the version is something like 2.5 or 3.0, while the release is
something like 2.5.1 or 3.0a1. If you don't need this dual structure,
just set both to the same value.
> Project version: 0.1.0
> Project release [0.1.0]:

If the documents are to be written in a language other than English,
you can select a language here by its language code. Sphinx will then
translate text that it generates into that language.

For a list of supported codes, see
http://sphinx-doc.org/config.html#confval-language.
> Project language [en]:

The file name suffix for source files. Commonly, this is either ".txt"
or ".rst". Only files with this suffix are considered documents.
> Source file suffix [.rst]:

One document is special in that it is considered the top node of the
"contents tree", that is, it is the root of the hierarchical structure
of the documents. Normally, this is "index", but if your "index"
document is a custom template, you can also set this to another filename.
> Name of your master document (without suffix) [index]:

Sphinx can also add configuration for epub output:
> Do you want to use the epub builder (y/n) [n]:
  
```

Handwritten notes:

- ← Your preference to have to separate folders
- ← "y" recommended
- ← Project name and Author
- ← First version of your documentation
- ← Maintain .rst
- ← Name of tree of contents. Default value is fine

5) Documentation by Sphinx

```
Terminal
File Edit View Search Terminal Tabs Help

Please indicate if you want to use one of the following Sphinx extensions:
> autodoc: automatically insert docstrings from modules (y/n) [n]: y
> doctest: automatically test code snippets in doctest blocks (y/n) [n]:
> intersphinx: link between Sphinx documentation of different projects (y/n) [n]:
> todo: write "todo" entries that can be shown or hidden on build (y/n) [n]: y
> coverage: checks for documentation coverage (y/n) [n]: n
> pngmath: include math, rendered as PNG images (y/n) [n]:
> mathjax: include math, rendered in the browser by MathJax (y/n) [n]: y
> ifconfig: conditional inclusion of content based on config values (y/n) [n]: n
> viewcode: include links to the source code of documented Python objects (y/n) [n]: y

A Makefile and a Windows command file can be generated for you so that you
only have to run e.g. 'make html' instead of invoking sphinx-build
directly.
> Create Makefile? (y/n) [y]:
> Create Windows command file? (y/n) [y]:

Creating file ./conf.py.
Creating file ./index.rst.
Creating file ./Makefile.
Creating file ./make.bat.

Finished: An initial directory structure has been created.

You should now populate your master file ./index.rst and create other documentation
source files. Use the Makefile to build the docs, like so:
    make builder
where "builder" is one of the supported builders, e.g. html, latex or linkcheck.

bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ ls
_build conf.py index.rst make.bat Makefile _static templates
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ vi conf.py
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ ls
_build conf.py index.rst make.bat Makefile _static templates
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ make
Please use 'make <target>' where <target> is one of
html          to make standalone HTML files
dirhtml       to make HTML files named index.html in directories
singlehtml   to make a single large HTML file
pickle       to make pickle files
json         to make JSON files
htmlhelp     to make HTML files and a HTML help project
qthelp       to make HTML files and a qthelp project
applehelp    to make an Apple Help Book
devhelp      to make HTML files and a Devhelp project
```

Very very useful for large documentations

"y" recommended

These files are made in ./docs folder

Make options

5) Documentation by Sphinx

```

Terminal
File Edit View Search Terminal Tabs Help

Terminal
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ ls
_build conf.py index.rst make.bat Makefile _static _templates
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ vi conf.py
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ ls
_build conf.py index.rst make.bat Makefile _static _templates
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ make
Please use 'make <target>' where <target> is one of
html          to make standalone HTML files
dirhtml       to make HTML files named index.html in directories
singlehtml    to make a single large HTML file
pickle        to make pickle files
json          to make JSON files
htmlhelp      to make HTML files and a HTML help project
qthelp        to make HTML files and a qthelp project
applehelp     to make an Apple Help Book
devhelp       to make HTML files and a Devhelp project
epub          to make an epub
latex         to make LaTeX files, you can set PAPER=a4 or PAPER=letter
latexpdf      to make LaTeX files and run them through pdflatex
latexpdfja    to make LaTeX files and run them through platex/dvipdfmx
text          to make text files
man           to make manual pages
texinfo       to make Texinfo files
info          to make Texinfo files and run them through makeinfo
gettext       to make PO message catalogs
changes       to make an overview of all changed/added/deprecated items
xml           to make Docutils-native XML files
pseudoxml     to make pseudoxml-XML files for display purposes
linkcheck     to check all external links for integrity
doctest       to run all doctests embedded in the documentation (if enabled)
coverage      to run coverage check of the documentation (if enabled)
bsha219@bn371671:/hpc/bsha219/Python/Behdad/lung_segmentation/docs$ make html
sphinx-build -b html -d _build/doctrees . _build/html
Running Sphinx v1.3.6
making output directory...
loading pickled environment... not yet created
building [mo]: targets for 0 po files that are out of date
building [html]: targets for 1 source files that are out of date
updating environment: 1 added, 0 changed, 0 removed
reading sources... [100%] index
looking for now-outdated files... none found
pickling environment... done
checking consistency... done
preparing documents... done
writing output... [100%] index
generating indices... genindex
  
```

Configuration file which can be edited

All the options that you can build through make

- make html documentation
- Shown in your browser
- Can be used by readthedocs

6) Readthedocs

This is a platform made for automatic documentation based on sphinx files.

First you need to make an account in the website if you already do not have one.

Start a project and link it to your GitHub repository. Then you need to build your documentation which is exactly the same version as your html build by sphinx.

As a good example you can refer to documentations below:

- <https://github.com/behdadebsh/LungPhD>
- <https://github.com/LungNoodle/lungsim>
- <https://github.com/duanemalcolm/morphic>

The above are simple examples of documentations which you can make yourself familiar with the basic .rst format 😊