# Docs Notes

Caffe2 docs are split up into three areas:

1) Caffe2.ai website: this comes from the gh-pages branch via the markdown files found in `gh-pages:caffe2/_docs`; you can run this locally with jekyll and push changes directly to github if you're a contributor without the need for a PR
2) Operators catalogue: this comes from scripts in the master branch found in `master:caffe2/python/docs`; running `github_generator > operators-catalogue.md` and moving this md file to `gh-pages:caffe2/_docs` manually is currently the process
3) C++ API and Python API Doxygen docs: these are generated by Doxygen manually or by using `master:caffe2/docs/process.py` and copied to the `gh-pages:caffe2/doxygen-c` and `gh-pages:caffe2/doxygen-python` folders

## Docs Maintenance Setup

You need to have access to the master branch to generate the docs from source and you need access to the gh-pages branch to publish them. Since you often want the files simultaneously and since the folder structures between master and gh-pages differ so greatly, it is advised to clone each branch to different folders. Otherwise if you switch back and forth between gh-pages and master within the same local repo, you'll end up seeing folders in branches that shouldn't be there, and it can get really confusing once you start running Doxygen and generating thousands of new files.

```
cd ~
git clone https://github.com/caffe2/caffe2.git c2master && cd c2master && git checkout maste
cd ~
git clone https://github.com/caffe2/caffe2.git c2docs && cd c2docs && git checkout gh-pages
```

## Generating Operator Catalog

To update the operator catalog, a script must be run on the master branch and copied to your docs checkout. To trigger this use:

1. `cd ~/c2master`
2. `python caffe2/python/docs/github.py ~/c2docs/_docs/operators-catalogue.md`

## Generating API Docs with Doxygen

Support for generating the docs has been included in the CMake build process, but it is turned off by default. To trigger building of the docs use:

1. `cd ~/c2master`
2. `mkdir build && cd build`

3. `cmake -DBUILD_DOCS=ON .. && make`

This will create a docs subfolder in the build folder. You can launch either API's web page docs by opening the index.html file found at `build/docs/doxygen-c/html/index.html` or `build/docs/doxygen-python/html/index.html`.

To push to caffe2.ai, copy these from the build directory to your docs checkout: 1. `cd ~/c2docs` 2. `rm -rf doxygen-c` 3. `rm -rf doxygen-python` 4. `cp -r ~/c2master/build/docs/doxygen-c .` 5. `cp -r ~/c2master/build/docs/doxygen-python .` 6. `git add -A .`

### Install Doxygen

You will need to install Doxygen to build Caffe2 with the API docs.

### MacOS X

1. Press Command+Space and type Terminal and press enter/return key.
2. Run in Terminal app: `ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/inst` `< /dev/null 2> /dev/null` and press enter/return key. Wait for the command to finish.
3. Run: `brew install doxygen`

Other operating systems' installation instructions can be found on the Doxygen website.

### Doxygen Notes

Doxygen seems to behave better if the processing between C++ and Python was split up. This is why there are two different links to cover each API.

C++ API docs work out of the box with the Caffe2 source code. Python docs require "python blocks" which are (often) currently missing in the Python code.

The Doxygen customization includes these files in the doxygen folder:

- header.html - logo links back to the main docs page
- footer.html - includes the Facebook OSS footer
- stylesheet.css - Doxygen's default CSS; tweaked to fix formatting problems with the custom logo, header, and footer
- main.css - copied from the caffe2ai CSS, so this should be refreshed after the design changes (this overrides/extends stylesheet.css)

It also extracts info from markdown files found in the source tree. A legacy installation file was in the /docs folder and this was removed. These file show up in the top navigation under "Related Pages".

### Scripts for Adding the Doxygen Preamble and Publishing the Docs

Some of the Python files may be missing the preamble Doxygen needs to generate docs, so the Python script called "process.py" that resides in the /docs folder is used to prepare the docs by looking for the block and if it doesn't exist prepend and customize the python blocks section with the module's path (e.g. Module caffe2.python.examples.char_rnn). It was noted that you need to delete the previous version of docs when you regenerate the docs or else things get messy, so the script deals with that as well.

To manually create API documents, go to your master checkout:

1. `cd c2master`
2. `cd docs && python process.py`

If you want to push these to caffe2.ai, go to your docs checkout: 1. `cd c2docs` 2. Copy the files generated in build/docs to your gh-pages branch, commit, and push. 3. `doxygen-c` and `doxygen-python` both go in the root folder of `gh-pages` 4. `operators-catalogue.md` goes in `_docs`

### Running Doxygen Manually

It may be beneficial to run Doxygen on your own and not try to inject the preamble. Just note that any files without a preamble will not show up in the docs.

To do this, make sure you have doxygen installed and from the master branch root run:

```
doxygen .Doxyfile-python
doxygen .Doxyfile-c
```

`.Doxyfile-python` and `.Doxyfile-c` are config files that are optimized for each code format. They each use Doxygen's config for `OUTPUT_DIRECTORY` which is set to docs/doxygen-python or docs/doxygen-c respectively.

### Publishing the New Doxygen Files

You will not be able to `git push` to `gh-pages` until you have switched auth methods to SSH:

```
git remote set-url origin git@github.com:caffe2/caffe2.git
```

Doxygen will tend to change and remove a lot of file and a simple copy can leave files behind that shouldn't be there anymore, so it is advised to delete the old doxygen-c and doxygen-python folders first. Copy the new files to the root folder of the gh-pages branch, making sure before you commit, you `git add` the deleted files, the new files, and the modified files. Then `git push` will push these changes without the need for a PR.

### Maintaining the Doxygen Configs

Each of the config files is customized differently and these changes are mentioned further below. A more common update though is the `EXCLUDE` config which is used to exclude whole directories and individual files. As new folders and special files are added it may make sense to update this so Python's API isn't including a bunch of new C++ info. You also end up with some third party files that you probably don't want to include in the docs. `FILE_PATTERNS` is supposed to prevent the Python & C++ docs from getting mixed up, but a combination of `FILE_PATTERNS` and `EXCLUDE` seems have worked best.

Settings that were customized:

OPTIMIZE_OUTPUT_JAVA - turned on for Python config, off for C++ config USE_MDFILE_AS_MAINPAGE - use to flag a markdown file for the mainpage EXTRACT_ALL QUIET WARN_IF_UNDOCUMENTED FILE_PATTERNS DOT_MULTI_TARGETS = YES JAVADOC_AUTOBRIEF = YES QUIET = YES SOURCE_BROWSER = YES VERBATIM_HEADERS = NO SHOW_NAMESPACES = NO for C++ config

Not using this (was in old config file, but seems to be for Latex): EXTRA_PACKAGES = amsmath
amsfonts
xr

## Caffe2.ai

For more info on contributing to Caffe2.ai, take a look at CONTRIBUTING.md found in the root of the gh-pages branch. This will help you setup jekyll locally so you can make your edits and test your changes. The most commonly updated areas are the markdown files in the `_docs` folder as well as the navigation structure (to add things to the sidebar) which is found at `_data/nav_docs.yml`. Each markdown file is required to have a header and if you wish for the markdown doc to appear as an item in the navigation the `id` in the yml file is equivalent to the `docid` in the markdown file.

For example in the Applications of Deep Learning markdown file we have:

```
---
docid: applications-of-deep-learning
title: Applications of Deep Learning
layout: docs
permalink: /docs/applications-of-deep-learning.html
---
```

In the `_data/nav_docs.yml` file we have:

```
- title: Quick Start
  items:
  - id: getting-started
  - id: learn-more
  - id: caffe-migration
- title: Learn
```

```
  items:
  - id: applications-of-deep-learning
  - id: operators
  - id: mobile-integration
...(more navigation here)...
```

To link up the `docid: applications-of-deep-learning` markdown file to the navigation we have it listed under `- title: Learn` then under `items:` and then `- id: applications-of-deep-learning` using `id` instead of `docid`. The order in the yml file is how it is displayed in the browser, and it will pull the `title` from the markdown file's header and display that as the H1 for the page. In this example, `title: Applications of Deep Learning` shows "Applications of Deep Learning" at the top in H1 format, so in your markdown file there's no need to add your own H1 or single hash title for the page.

### TODO:

In the future it would be ideal to expand these Operator Catalogue scripts to provide documentation for all C++ operators and Python modules. Instead it focuses on a subset of operators only, so in the interim Doxygen is used to backfill this gap.

Additionally, the operators-catalogue.md file that is generated is quite large and difficult to search, so breaking it up and adding categories to the Schema of the operators would be very helpful.

To achieve better output in the Doxygen Python docs: http://stackoverflow.com/questions/7690220/how-to-document-python-function-parameter-types

Swap this kind of formatting into py files:

```
def my_method(x, y):"""!
    my_method description

    @type x: int
    @param x: An integer

    @type y: int|string
    @param y: An integer or string

    @rtype: string
    @return: Returns a sentence with your variables in it
    """return "Hello World! %s, %s" % (x,y)
```

Note that the bang (!) is added after the opening comment """! - this seems to do the trick and the remaining comments will be nicely parsed by Doxygen.

**Other Notes**

Useful for Xcode, currently off GENERATE_DOCSET = NO

Look at search engine integration, xml output, etc EXTERNAL_SEARCH = YES