Hosting Documentation on GitHub

Create an empty gh-pages branch

To create a branch of your github software repository which contains nothing but documentation, first create a new, empty branch on your local copy and push it up.

```
cd /path/to/repoName
git symbolic-ref HEAD refs/heads/gh-pages
rm .git/index
git clean -fdx
echo "My GitHub Page" > index.md
git add .
git commit -a -m "First pages commit"
git push origin gh-pages
```

Setup your github repository to serve a webpage with your docs

Login to your account on github.com and navigate to your repository
Click the “Settings” tab on the main repository page
Click the “Pages” tab on the left

While on that page:

- Select the source to be the ***gh-pages*** branch
- Set the source to point to ***root***.
- Select the ***theme*** of interest
- Hit the ***Save*** button
- The webpage to host your document page will be shown there:
  https://jeffersonlab.github.io/<repoName>/

One more thing is needed for the above page to be hosted. In the gh-pages repo, add or edit any existing ***index.md*** file. This is what the server will display. This is written in markdown language. The following index.md file shows the doxygen, javadoc, and pdf files:

```
---------------------------
# **Documentation**
---------------------------

Here are links to the documentation contained in the github repository

_____

## **Version X.Y**
```

* [User's Guide PDF](*https://jeffersonlab.github.io/repoName/docDir/users_guide/Users_Guide.pdf*)
* [Javadoc for Java Library](*https://jeffersonlab.github.io/repoName/docDir/javadoc/index.html*)
* [Doxygen for C Library](*https://jeffersonlab.github.io/repoName/docDir/doxygen/C/html/index.html*)
* [Doxygen for C++ Library](*https://jeffersonlab.github.io/repoName/docDir/doxygen/CC/html/index.html*)

Create file to do the github action of generating javadoc and doxygen files

In your checked-out directory on your machine, on the main (not gh-pages) branch, add a file which defines a github action. This action will automatically re-generate doxygen and javadoc code upon the pushing of changes to the repository.

The file will look something like:

```
# Automatic generation of doxygen and javadoc files for C, C++, and Java code
# in myBranch which will be copied and checked into the gh-pages branch.
name: Documentation generation CI

on:
  push:
    branches: [ myBranch ]

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
      # checkout myBranch
      - uses: actions/checkout@v2

      # generate the C doxygen files
      - name: Doxygen Action C
        uses: mattnotmitt/doxygen-action@v1.3.1
        with:
          working-directory: '.'
          doxyfile-path: 'doc/doxygen/DoxyfileC'

      # generate the C++ doxygen files
      - name: Doxygen Action C++
        uses: mattnotmitt/doxygen-action@v1.3.1
        with:
          working-directory: '.'
          doxyfile-path: 'doc/doxygen/DoxyfileCC'

      # generate the javadoc files
      - name: Set up JDK 8
        uses: actions/setup-java@v2
        with:
          java-version: '8'
          distribution: 'adopt'
```

```
    - name: Javadoc Action
      run: ant -noinput -buildfile build.xml javadoc

    # clean up the javadoc files including removing timestamps. OPTIONAL.
    - name: Tidy up the javadocs
      id: tidy
      uses: cicirello/javadoc-cleanup@v1
      with:
        path-to-root: doc/javadoc

    # store the doc files
    - name: Upload Output Directory
      uses: actions/upload-artifact@v2
      with:
        name: doc-files
        path: doc
        retention-days: 1

  copy:

    runs-on: ubuntu-latest
    needs: build

    steps:
      # checkout the gh-pages branch
      - uses: actions/checkout@v2
        with:
          ref: gh-pages

      # download the doc files, most of which are generated above
      - name: Download Output Directory
        uses: actions/download-artifact@v2
        with:
          name: doc-files
          path: docDir

      # add, commit and push to gh-pages
      - name: Commit changes
        uses: EndBug/add-and-commit@v7
        with:
          author_name: Carl Timmer
          author_email: timmer@jlab.org
          message: 'Update docs'
          branch: gh-pages
          add: '["docDir/doxygen/C/html/*", "docDir/doxygen/CC/html/*", "docDir/javadoc/",
"docDir/users_guide/Users_Guide.pdf"]'
```

This action has 2 jobs defined. The first, ***build***, will copy your branch, generate the docs from it, and upload the ***doc*** directory to storage. The second, ***copy***, copies the gh-pages branch, downloads the stored directory, then adds, commits, & pushes the changes to the gh_pages branch. The second job does not start until the first has been completed. Notice that not only

the generated docs, but also existing pdf or other non-generated files can be added to the gh-pages branch – as long as they're among the files copied along with the generated files.

Name this file something like myDocGenerator.yml and place this into your repository's

.github/workflows

directory of the main branch. Of course, you'll need to modify it to suit you such as the paths to the Doxyfiles & the java code, the java version to use, and the name of the directory to store things on the gh-pages branch.

This file was setup to use ***ant*** to generate the javadoc. If you're using another method, replace the "Javadoc Action" with one that works for you. There are many available to do the job without having to create one from scratch.

Note that in the case of evio, there is an additional file in doc/users_guide/evio_Users_Guide.pdf which automatically gets uploaded, then subsequently downloaded in the ***copy*** job without having to explicitly mention it.

You'll also, most likely, need to modify your .gitignore file by adding the line:

!.github/

Watch the Action

When you've pushed this file to the repository, it will start the github action. On the repository's github web site,

- Click on the Actions tab

It will show the progress and success (or failure) of your action. Enjoy the iterative process in getting it to work.

If the action is successful, look at the gh-pages branch and see if the doc files are listed there. Modify your index.md file to point to all the proper doc files. Finally, point your browser to:

https://jeffersonlab.github.io/<repoName>

to see if it's working. Consider modifying your main branch's README.md file to contain a link to this.