How to work on the Technical Manual Online

In this tutorial we will explain how we can work to easily review or change the online technical manual without going through the complexity of a local installation. It is done to simplify the way of contributing to the content. As we get more experienced with the way of working we can go for a local installation which has several advantages.

Tools used

Generation of Documentation

For the generation of the documentation we chose for a document utility that is based on python code. The document generation tool is called Sphinx : https://www.sphinx-doc.org/en/master/

It uses rst files (simple structured text files) that can be used to generate different formats of the same content : pdf, html, latex, epub etc

To make this work you need to install a python environment with the needed sphinx libraries.

Quick references and guidance can be found in the document directory of wg-tech-man repo of intermagnet.

For the online reviews we will not do this ourself (see automated build and release). This will be automatically done for us.

Versioning

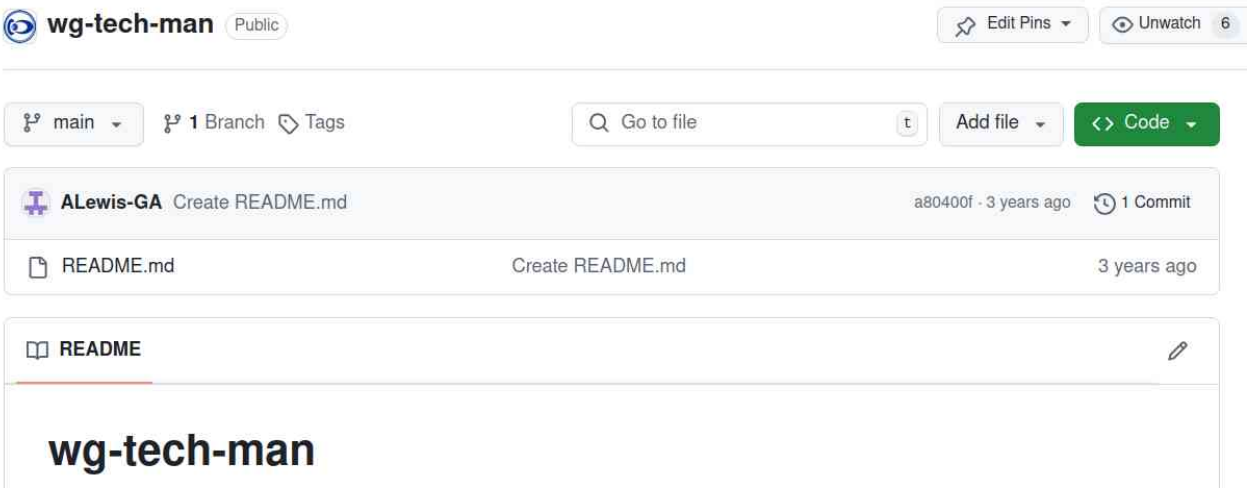
For the versioning tool intermagnet chose to use GIT repositories. It was basically chosen because :

- Each repository can also be linked to a GITHUB pages website generation
    - This has been used to expose the intermagnet.org pages which are re-generated each time we commit to the main branch of the repository : **intermagnet.github.io** https://github.com/INTERMAGNET/intermagnet.github.io.
    - These pages are limited to simple html and based on the yekyll framework (written in RUBY)
    - We use them for the issues and use the issues as discussion forums

GIT specifics

To use GIT you need to understand some specifics.

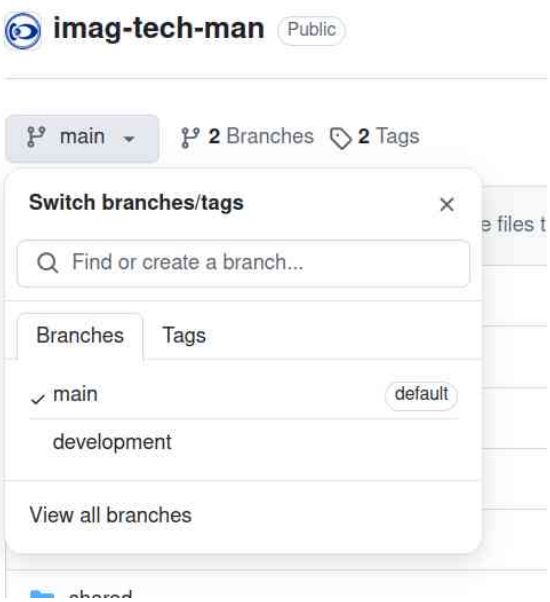
When creating a GIT repository it will create a main branch. This is the starting point of where you will put your versioned files.



When you open a repository (in this case the wg-tech-man) you view the main branch.
The wg-tech-man only has this branch and no tags. We currently only use it to discuss on items.
Each work group has a similar repository. You also remark the number a80400f which are the first 6 digits of the unique code of the last commit (change to the branch) accepted.

Branches

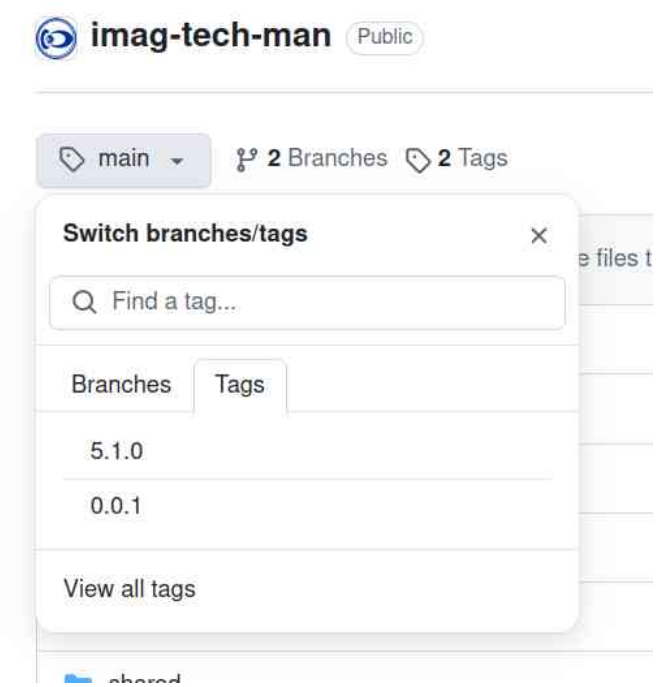
Besides the main branch we can create other branches with a meaningful name. Normally we will create a branch for each item that a developer can work on. A branch is a copy of the main branch on the moment that the branch was created. From then on we have a starting point with it's own lifecycle and this can evolve independently of the main branch. A developer will never work directly on the main branch



When you click on the dropdown of the main branch you will see other branches if there are other branches. In the case of the imag-tech-man repository you will notice one branch created development (which is a special one and we will explain why later on). Normally we would create one for each action item to be introduced : for instance tm_17_despiking could be used as name for a branch indicating the work that will be done within this branch.

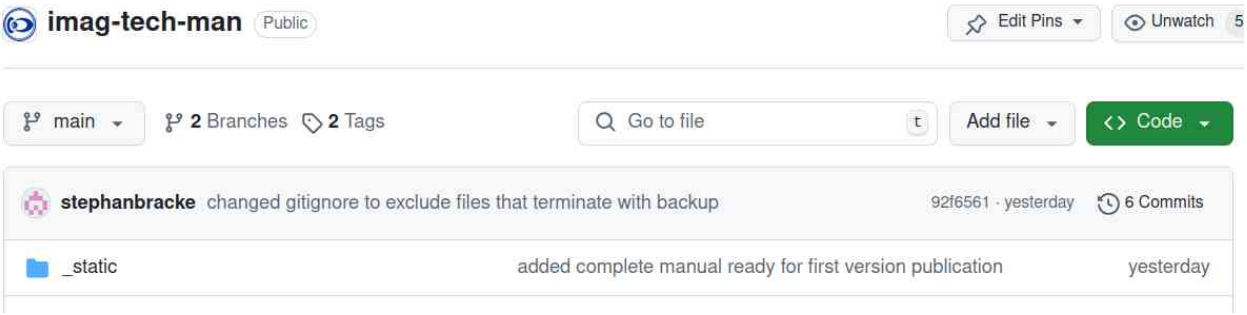
Once the work is done we will create a pull request : A pull request is an action to ask to integrate the changes on the development branch into the main branch. Once reviewed and accepted the change is available on the main Branch.

Tags

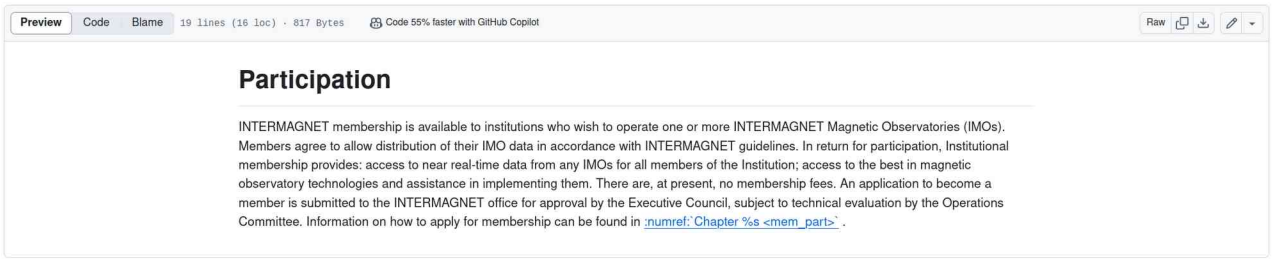


A tag is created from the main branch and is again a copy of the main branch on that moment. The difference with a branch is that you can’t change anything in this copy. It is typically used as versioning so once we have a version ready we create a version release and we tag the main branch. In the picture you see what is currently available on the imag-tech-man repository.

Commit



Each time when you are on a branch you can commit to the branch.Each commit will get a unique number.



You can edit directly on a branch via the online utility by clicking on the pencil(if you have the correct rights to do so). Each edit will automatically be transformed in a commit when saving the changes. You need always to give a short description to the changes you made (which is important to trace changes afterwards).

When you work locally you can do many changes and group them into one commit however when working online this is not possible.

This knowledge of GIT is sufficient to participate online on the technical manual.

If you want to work locally and get more insides of GIT. It is useful to go through this tutorial

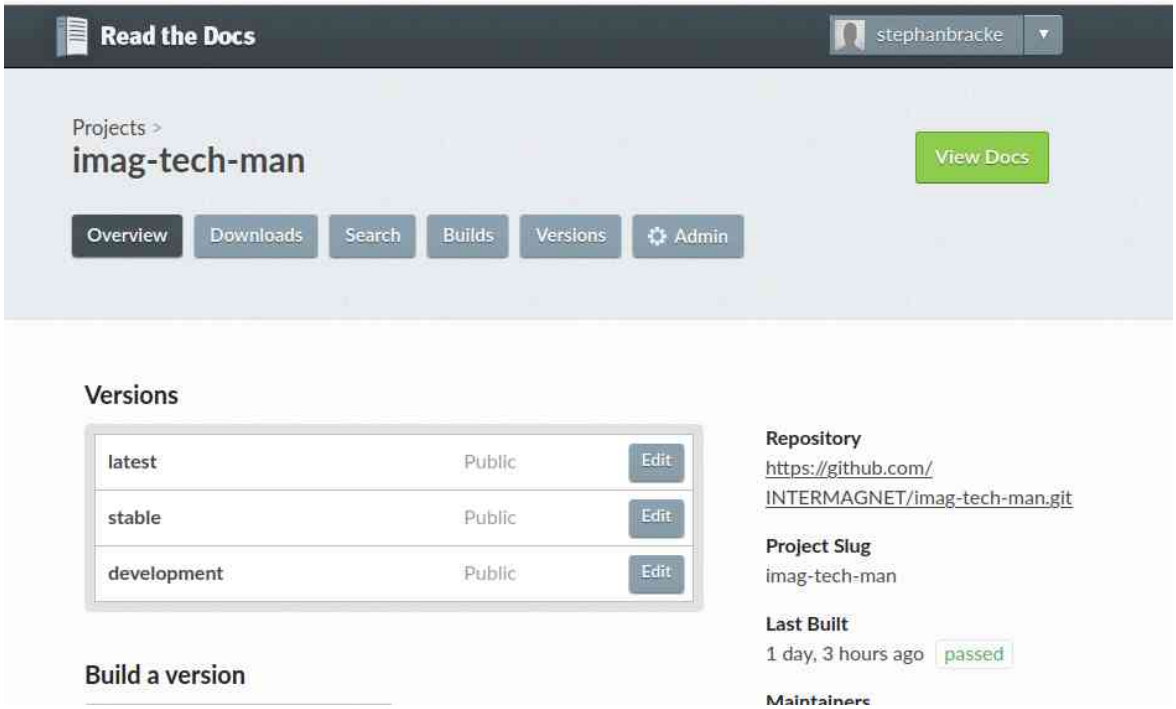
https://www.atlassian.com/git/tutorials

Automated build and release

When we look what is inside the github repositories, you will only find building blocks : rst files , config files etc. No pdfs or html. For this we need to generate it based on a branch or tag. readthedocs.org provides this possibility. (https://about.readthedocs.com/)

Read the docs will provide hooks to your github repository (in this case imag-tech-man).

Each time there is a new commit on a branch or a new tag created readthedocs will copy over the code and start to generate documentation that you asked for.



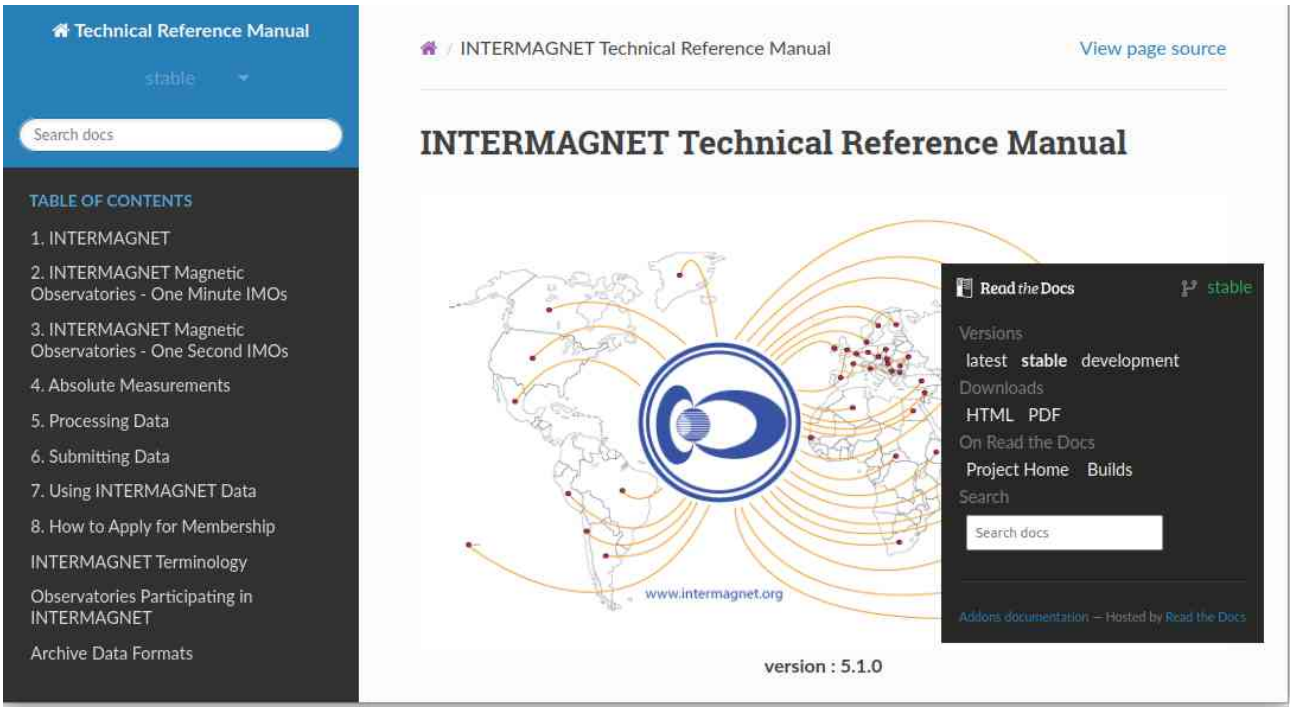
In this configuration of readthedocs you see the repository which he is linked to in this case imag-tech-man.

There are currently three version builds configured :

1. stable : The latest official release (the most up to date version to be used as a reference) it corresponds with the latest version tag created in GIT.
2. latest : The current working version with accepted changes but not ready for an official release. It corresponds with the latest commit on the main branch.
3. development : This version may contain errors and/or proposed changes not yet accepted. It corresponds with the latest commit on the development branch.

Each time we commit something to the main branch or the development branch. Read the docs will start an automated build and release cycle and deploy this change on to the url tech-man.intermagnet.org. The same thing will happen eaach time when we create a version tag.

Stable Version



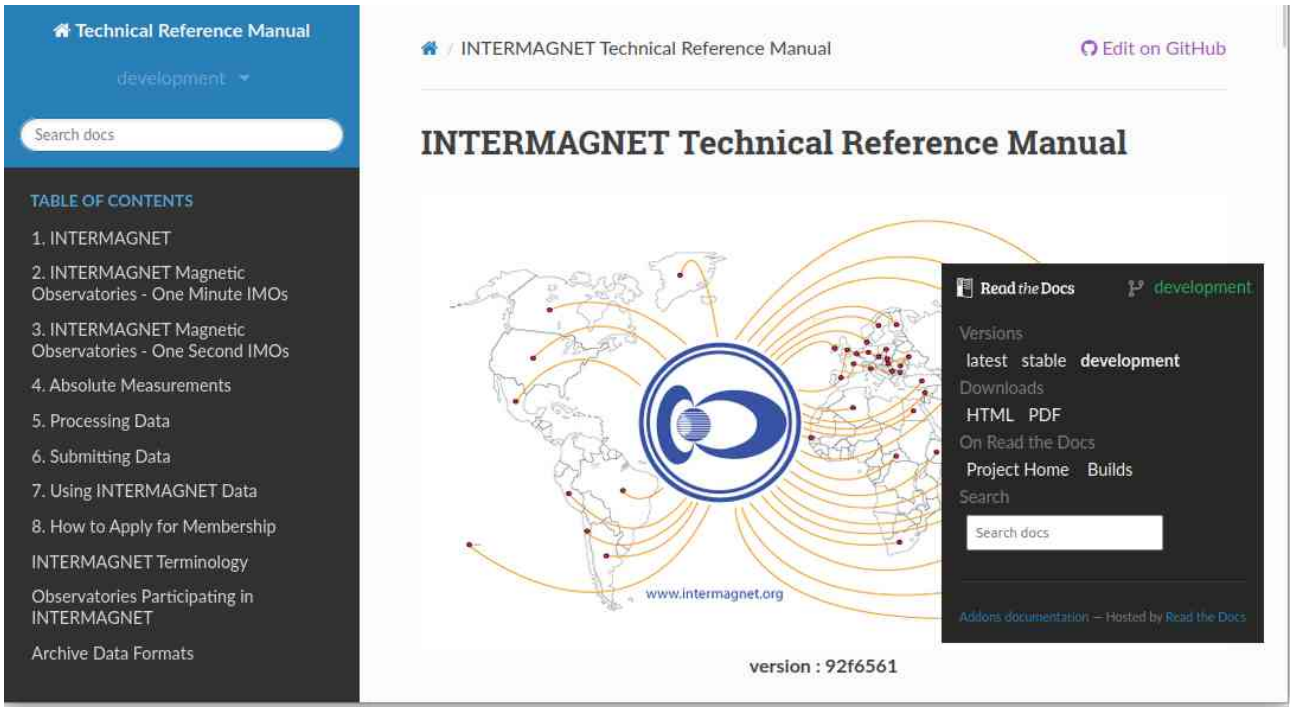
The Stable version will have the version number of the latest version tag. In this example 5.1.0.

The url is linked with tech-man.intermagnet.org.

The stable version is not editable so on the top right corner there is no possibility to edit it on GiTHub only a link to view the source.

In the right corner you will have a clickable band. In the picture it is expanded and it is there you can switch to the latest or development version. It also here that you can download the different formats available: for the moment HTML and PDF. Further on you can inspect the builds and look at the log files. This can be useful in case of failure. If you download the PDF the version number on the first page should correspond. I noticed in the past that this could take up some time before it is updated.

The Development Branch



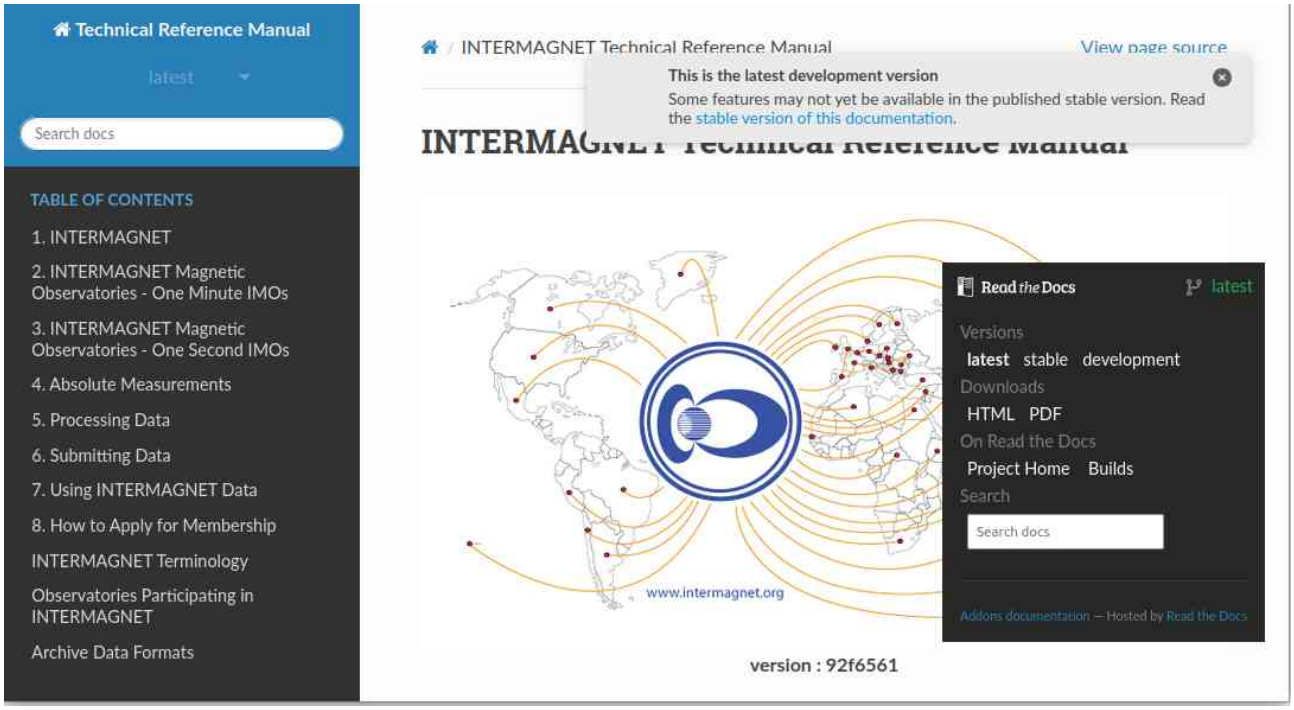
It is quite similar to the stable version but we remark two major differences :

- The version number is the short version of the unique code of the latest commit on the development branch (you can verify this by looking into GIT)
- This time there is the possibility to click on edit on GIT. If you click on this you should be directed to the corresponding rst file in GIT which gives you the possibility to edit it there change it and commit. A new version will be generated on readthedocs when committing to the development branch

This development branch was only created to ease up reviews and to see impact wthout needing to install it locally.

We need to evaluate if this is workable because when more people work on the same development stream for different action items you will not be able to seperate them into different pull requests.

The latest version



This is the one generated whenever there are commits added to the main branch (for exemple after an accept of the pull request) . It is not recommended to edit directly on the main branch so also here there is not the button to edit on Git displayed