LMSAL HUB Description

Release 1.0

2018, Juan Martinez-Sykora

Feb 07, 2020

Contents

CHAPTER 1

LMSAL_HUB Vision

LMSAL_HUB is a gitlab group for solar physics focused on maintaining, sharing, distributing, debugging, and providing documentation to any code developed at Lockheed Martin Solar and Astrophysical Lab (LMSAL) or Bay Area Environmental Research Institute (BAERI). The repositories are mainly, but not only, in Python. The repositories can be used also as a platform for develop the codes to other more versatile and state-of-the-art repositories (e.g., sunpy, astropy or irispy).

The repositories could be a collection of different scripts, codes and classes with various degrees of portability and capabilities. Each repository will contain requirements, installation instructions and other useful information either in the README files, wiki or associated readthedocs documents.

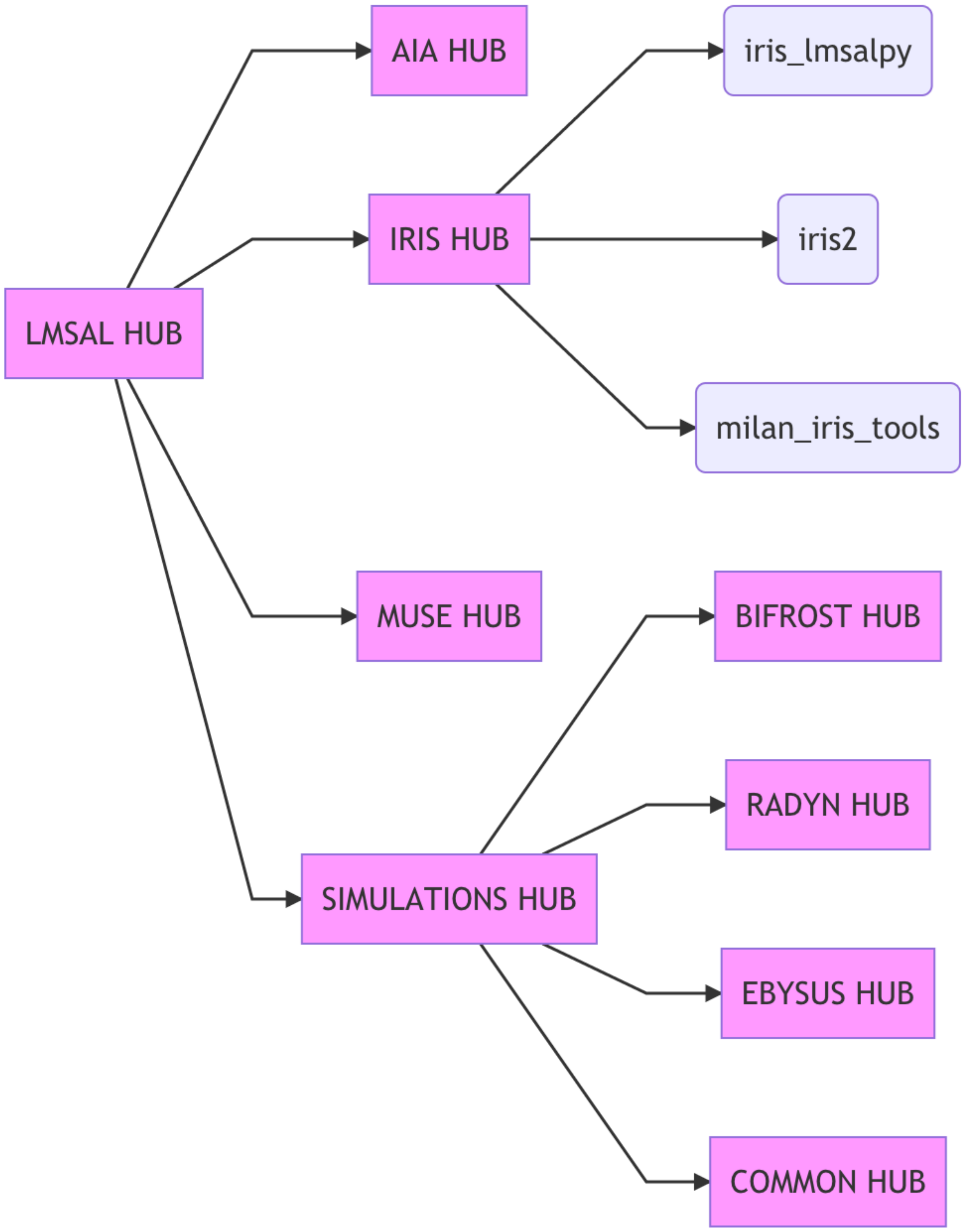
CHAPTER 2

LMSAL_HUB Structure (gitlab)

LMSAL_HUB is a gitlab group which contains subgroups, repositories, libraries, Scripts, documentation and other related materials.

In the LMSAL_HUB group you will find different subgroups related to instruments, projects or subjects. Each of these subgroups may contain other subgroups, repositories, libraries, scripts and/or documentation.

As mentioned in the vision section, e once the projects in the group are mature enough, they might be moved to elaborate and state-of-the-art repositories (e.g., sunpy, astropy or irispy). For this reason, each subgroup will have a repository which could be used as platform for the integration.


LMSAL HUB
AIA HUB
IRIS HUB
iris_lmsalpy
iris2
milan_iris_tools
MUSE HUB
SIMULATIONS HUB
BIFROST HUB
RADYN HUB
EBYSUS HUB
COMMON HUB

CHAPTER 3

LMSAL_HUB Git commits

(i.e., Mind your Git manners)

The commit message is mainly for the other people, so they should be able to understand it now and six months later. Commit msg must only be one sentence and should start with a tag identifier (see end of this section).

Use the imperative form of verbs rather than past tense when referring to changes introduced by commit in question. For example “Remove property X”, not “Removed property X” and not “I removed. . . ”.

Use the following tags for commit msg:

```
[ADD] : adding new feature
[FIX] : a bug
[DOC] : documentation only
[REF] : refactoring that does not include any changes in features
[FMT] : formatting only (spacing, removes blank lines etc.)
[REP] : repository related changes (e.g., changes in the ignore list, remove files )
```

Example commit msg:

```
[FIX] Adds missing initialization to tg array
[REF] Removes unnecessary allocation for dynamic arrays
```

Note: Avoid to commit more than one file every time. This facilitates to follow the commit documentation.

Note: Avoid to do merge unless it is between different branches.

Note: Avoid to add binary, executables files. In fact, mainly include only documentation, license, and code files.

CHAPTER 4

LMSAL_HUB: Gitlab minimum help

Gitlab allows to create groups to work on various projects. Most of you have guest access to LMSAL HUB. Guests can see the group structure, repositories and wiki, but not the code unless the repository is open to the public or similar.

Note: Guests do not have permission to add new subgroups, add new repositories or commit changes

If you are interested to have access to a specific subgroup or repository you need the permission of the owners or PIs of the projects.

Once you are a member of a subgroup you can, create branches, add new repositories, or commit changes. Gitlab allows to import a repository from, for instance, github.

Gitlab allows to have protected branches which allow to secure stable branches secure and force developers to use merge requests.

In addition, Gitlab has integrated pipeline configurations useful for adding debugging options, rerun code after any substantial change in the code etc.

The PI or owner of the repository or group will manage them and be in charge of the accessibility. If any of you are interested to have a new subgroup, contact directly the owners of LMSAL HUB.

In case any of you want to create new repositories with no access to the code for any of the owners of the LMSAL HUB, then you should create your own private independent repository/group. This could be moved eventually to LMSAL HUB if the accessibility status changes and, at least, the owners or LMSAL HUB can have access.

CHAPTER 5

LMSAL_HUB: How to prepare a repository

A repository should have at minimum a README file, a LICENSE file and documentation. The later can be easily created with the gitlab wiki, or using notebook instructions (and converted, for instance, to html or pdf using readthedocs) in a ‘‘docs” folder.

Python repositories should have a setup.py file and in each folder within the repository that contains the code an __init__.py file.

5.1 Install a repository

With this type of setup, it is very easy to install a new Python library (after cloning the repository).

> python setup.py install

5.2 Non-root install

If you do not have write permission to your Python packages directory, use the following option with setup.py:

> python setup.py install –user

This will install the Python library under your home directory (typically ~/.local).

5.3 Developer install

If you want to install the library but also actively change the code or contribute to its development, it is recommended that you do a developer install instead:

> python setup.py develop

This will set up the package such as the source files used from the git repository (only a link to it is placed on the Python packages directory). It can also be combined with the –user flag for local installs.