Git For Analysis

ATLAS Software Tutorials October 2019

Adam Parker

Based on slides from K. Suruliz

Introduction

In the previous talk, you learned about git, what it's used for, its basic features and commands (git clone, git add, git commit)

- The focus was mainly on offline release development
- The workflow for user analysis code is a bit different (simpler!)
- The main ATLAS git documentation can be found HERE

In this talk:

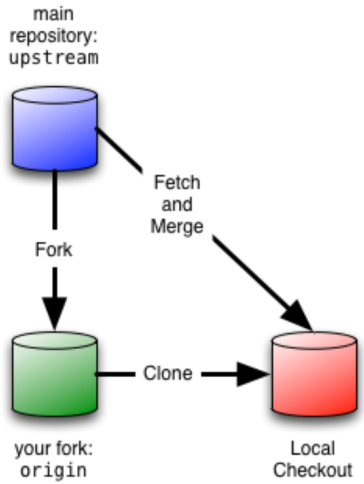
- Focus on git for analysis software
- An example of how you would version control your personal projects
- You will be able to try things out yourself in the hands-on session
- The focus was mainly on offline release development
- Detailed documentation (+ hands-on SoftwareTutorialAnalysisInGitReleases

Development Workflow

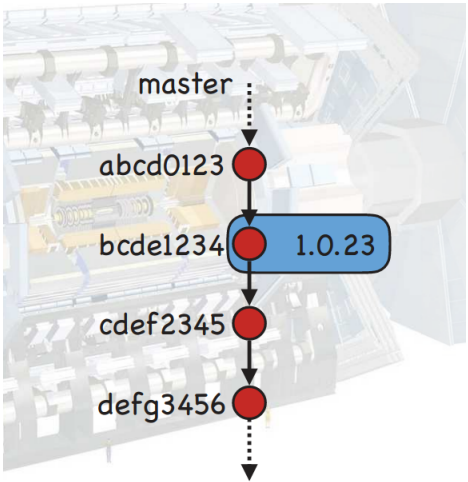
- For Athena development, we rely on forks a lot.

  Each user has their fork (~copy) of the entire athena repository

- This is necessary when you have a large number of developers all contributing to one project



Managing a Private Project



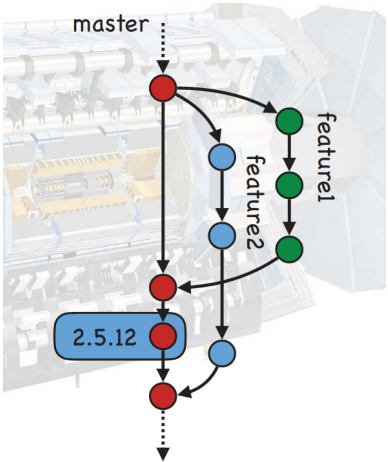
- If you're the only one managing a project, you don't even need to know about git branches too much

  It is quite rare you will need multiple "production versions" of your own code

- You should just keep developing your code in the master branch of your repository

  Possibly making some tags as you go along

Managing a Group Project



- If the group is small and the development is in an absolutely one track way you can still use the private repository workflow

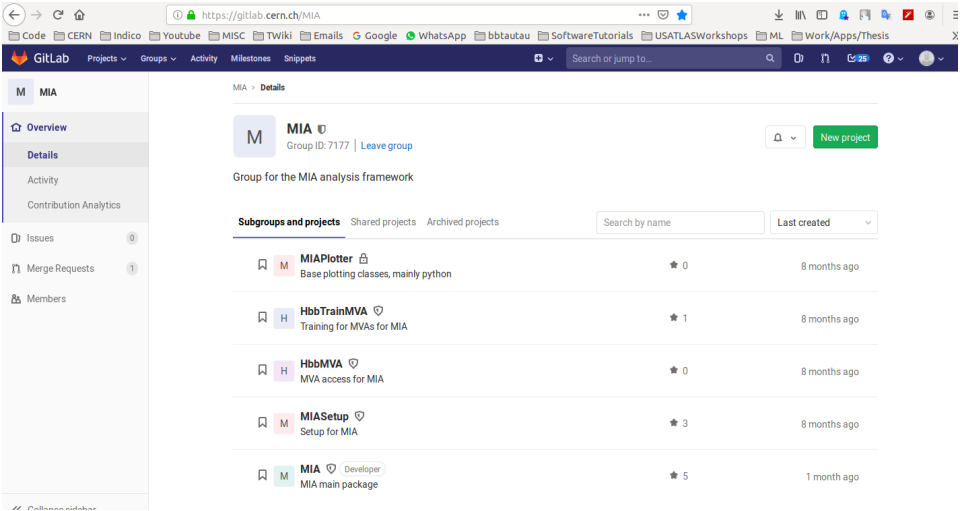
  Note, all developers require the correct access to the repository

- A better idea would be to have developers work in feature branches

  This is not a separate fork but inside the *main* repository

Bottom line is, you should not use forks unless you have >50 developers

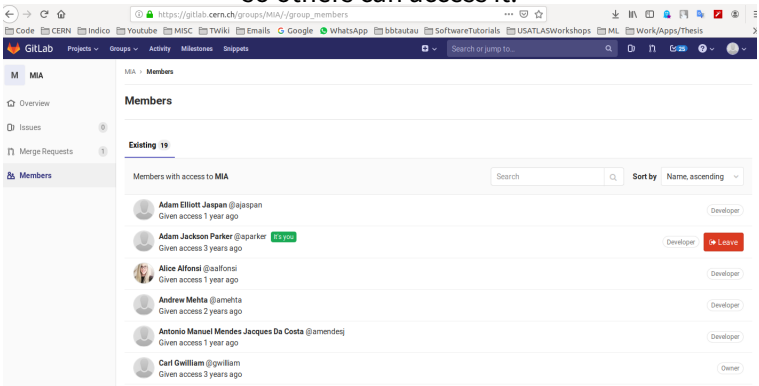
Groups and Projects



Can control access on the level of groups or the level of projects contained within those groups

Permissions

You have seen already how to create a new project in GitLab. You need to set permissions correctly so others can access it.



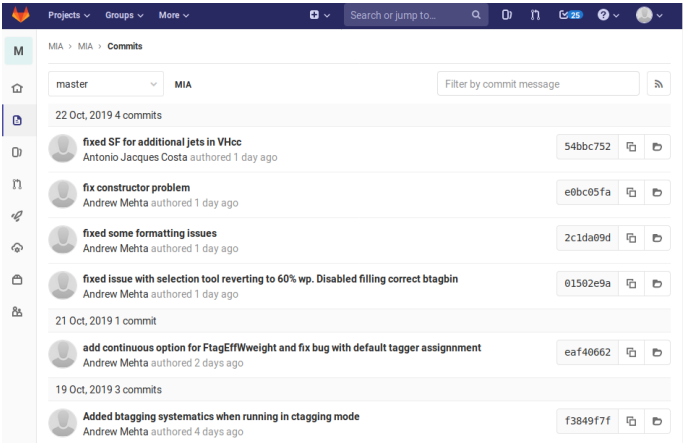
Note the different levels of permissions. Roughly speaking, reporters have read-onlyaccess (should include the rest of ATLAS, generally), developers can push code/createmerge requests, and maintainers can manage the repository

Commits, Tags and Hashes

- Each commit of a repository (athena or your project) is uniquely identified by a **hash**.

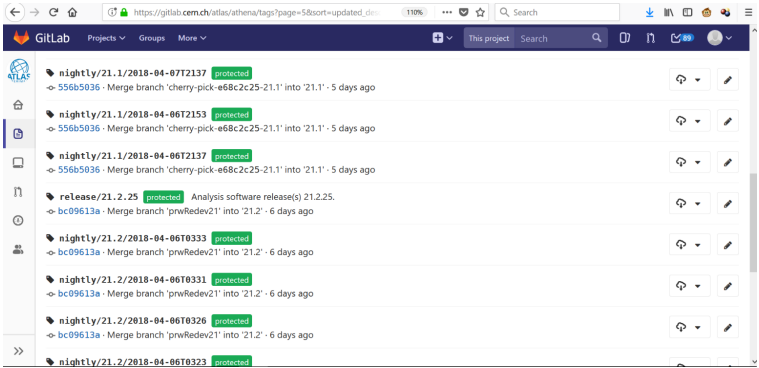
  You can think of this as corresponding to a particular snapshot of the repository

- It is possible to checkout a package at a certain commit point using this hash.



Commits, Tags and Hashes

Tags can be used to mark release points



Syntax for checking out a branch or particular tag:
git checkout <branch or tag name>, e.g.
git checkout release/21.2.93

Project Structure: SubModules

The basic philosophy is that projects should be entirely self-contained. There should be no need to tell users to manually check out additional packages needed for the software to work

There are often situations where you need to use a:

- a package from another ATLAS member
- an updated R&D version of a CP package
- your own helper library or executables/macros.

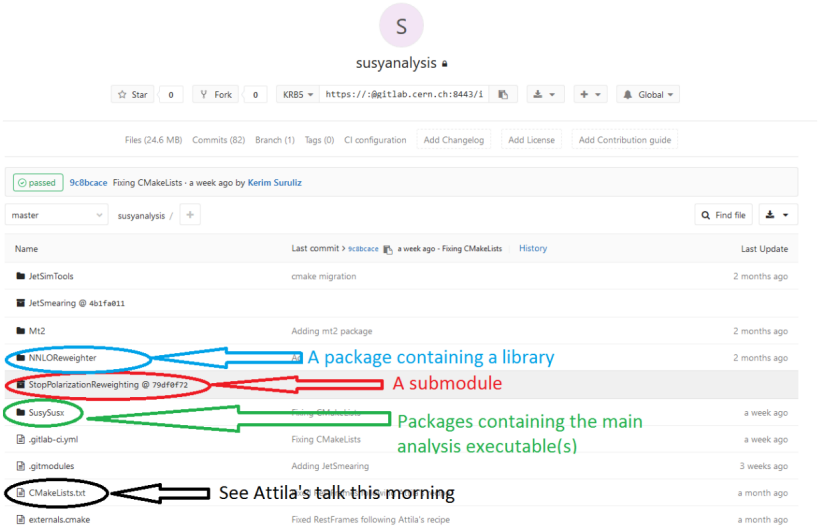
This is where the concept of a **submodules** comes in.

Submodules are nested repositories inside a parent git repository at a specific tag or snapshot of time.

This means packages can work out of the box and be completely self-contained.

You will see this in action in the hands-on

Project Structure: Example



Conclusion

Git is very powerful and has a lot of great features useful for analysis software development.

- However there is a lot to learn and the terminology is not always easy to understand
  forks, branches, tags, hashes, commits, origin, upstream, master...
- Many good online tutorials in addition to the ATLAS git tutorial, I suggest going through some of these
- Do spend a bit of time learning about git and acquiring good habits when you're starting

More on git/gitlab in Giordons talk later this afternoon

Backup