Accessing OSIRIS and using OSIRIS through GitHub

Anton Helm
ahelm@ipfn.tecnico.ulisboa.pt

Adam Tableman, Ricardo Fonseca

1 GoLP / Instituto de Plasmas e Fusão Nuclear, Instituto Superior Técnico, Lisbon, Portugal
2 UCLA Plasma Simulation Group, University of California, Los Angeles, U.S.A
3 ISCTE - Instituto Universitário de Lisboa, Lisbon, Portugal

epp.tecnico.ulisboa.pt || golp.tecnico.ulisboa.pt
Why should we use a version control system

- **Collaboration**
- **Versioning**
- **Logging**
- **Backup**
GitHub - social coding
Internet hosting service for git repositories

social component

access will be granted by us (private repository)
GitHub - social coding
Internet hosting service for git repositories

issues
- bug reports
- code discussion
- feature request
- project discussions

access will be granted by us (private repository)
GitHub - social coding
Internet hosting service for git repositories

Pull request
- code pushes
- bug fixes
- publishing features
- code discussion section

access will be granted by us (private repository)
GitHub - social coding
Internet hosting service for git repositories

private repo/copy

access will be granted by us (private repository)
GitHub - social coding
Internet hosting service for git repositories

OSIRIS release info

access will be granted by us (private repository)
versioning numbering of OSIRIS A.B.C:

- sequence-based software versioning scheme
- A ... main version of OSIRIS (major rework)
- B ... minor changes (features and extensions)
- C ... hot fixes (bugs at runtime/compile time)
- each release matches to a tag on master branch
OSIRIS versioning and changelog

versioning numbering of OSIRIS A.B.C:

✦ sequence-based software versioning scheme
✦ A ... main version of OSIRIS (major rework)
✦ B ... minor changes (features and extensions)
✦ C ... hot fixes (bugs at runtime/compile time)
✦ each release matches to a tag on master branch

✦ changelog for each new release
✦ references to pull request and issues
OSIRIS versioning and changelog

versioning numbering of OSIRIS A.B.C:

✦ sequence-based software versioning scheme
✦ A ... main version of OSIRIS (major rework)
✦ B ... minor changes (features and extensions)
✦ C ... hot fixes (bugs at runtime/compile time)
✦ each release matches to a tag on master branch

✦ changelog for each new release
✦ references to pull request and issues

✦ Download source code of OSIRIS
✦ recommended way: use git to obtain OSIRIS
Branching model for OSIRIS
master branch for using OSIRIS and development branch for extending

**master branch**
- stable version with supported features
- for simulation and for using OSIRIS
- tags for different versions

**dev branch**
- newest features and current development
- mirror of master but ahead in commits
- can be unstable to perform simulations

**extension and hotfixes**
- adding changes to OSIRIS through pull request
- pull requests for features - base is dev
- pull requests for hotfixes - base is master
What is git

git is not subversion on steroids

In many ways you can just see git as a filesystem — it’s content-addressable, and it has a notion of versioning, but I really really designed it coming at the problem from the viewpoint of a filesystem person (hey, kernels is what I do), and I actually have absolutely zero interest in creating a traditional SCM system.

Linus Torvalds
Installing and setting up git

obtaining git:
https://git-scm.com/downloads

make sure git uses your name/email:

$ git config --global user.name Anton Helm
$ git config --global user.email anton.helm@tecnico.ulisboa.pt

otherwise setup git to use your name/email:

$ git config --global user.name "Your Name"
$ git config --global user.email "your_email@somewhere.com"
clone the OSIRIS repository (GitHub credentials required):
$ git clone https://github.com/GoLP-IST/osiris.git

enter the “osiris” directory and check that you are on master
$ cd osiris && git branch
* master

show remotes
$ git remote -v
origin https://github.com/GoLP-IST/osiris.git (fetch)
origin https://github.com/GoLP-IST/osiris.git (push)

and rename from origin to upstream
$ git remote rename origin upstream && git remote -v
upstream https://github.com/GoLP-IST/osiris.git (fetch)
upstream https://github.com/GoLP-IST/osiris.git (push)

update master branch to match the official OSIRIS master branch
$ git checkout master && git pull --rebase upstream master
...
Your branch is up-to-date with ‘upstream/master'.
...
check for available OSIRIS versions (master branch has tags)

$ git tag

...  
4.2.2  
4.3.0  
4.3.1  
...

get a state of OSIRIS at a specific version

$ git checkout 4.2.2

get latest version of OSIRIS

$ git checkout master
developing and contributing to OSIRIS
creating a fork and adding it as a remote to git

- fork - private GitHub repo
  - private copy of a project on GitHub
  - can be added as a git remote
  - can be used as a backup
  - can be used to collaborate with others

add your fork to your remotes on local git repository

$ git remote add origin \\
https://github.com/ahelm/osiris.git

$ git remote -v
origin  https://github.com/ahelm/osiris.git
origin  https://github.com/ahelm/osiris.git
developing and contributing to OSIRIS
adding official OSIRIS dev branch and starting a new feature

checking the dev branch on your git repo

$ git checkout -b dev upstream/dev
Switched to a new branch 'dev'

getting latest changes on dev branch

$ git pull --rebase upstream dev
From https://github.com/GoLP-IST/osiris
 * branch            dev       → FETCH_HEAD
 ...

checking git log (latest commits/associated branches)

$ git log

checkout a new feature branch

$ git checkout -b new_feature upstream/dev
Switched to a new branch 'new_feature'
developing and contributing to OSIRIS

basic git commands

list branches (* - currently selected branch)

$ git branch
  dev
  master
  * new_feature

list changes on working copy (short list)

$ git status --short
  M source/pgc/os-emf-pgc.f03
  ?? source/pgc/new_file.f03

track changes (specify individual files or all changes with .) and commit

$ git add .
$ git commit # pops for a commit msg

pull changes from dev branch on new feature - might lead to conflicts

$ git pull --no-commit upstream dev

push changes to a branch on remote (e.g. your fork set up as “origin”)

$ git push <remote_name> <branch> # general example
$ git push origin new_feature

create new pull request with new features
fixing an issue with OSIRIS

taking current stable version of OSIRIS and implementing a fix

create a new branch from official OSIRIS master
$ git checkout -b some_hotfix upstream/master

implementing the fix (in file os-some-buggy-file.f03)
$ git status --short
  M source/os-some-buggy-file.f03

track changes and commit
$ git add .
$ git commit # pops for a commit msg

pull changes from master branch on hotfix - might lead to conflicts
$ git pull --no-commit upstream master

push changes to a branch on remote (e.g. your fork set up as “origin”)
$ git push origin some_hotfix

create new pull request with fixed changes
developing and contributing to OSIRIS
create a pull request to add your changes to main OSIRIS repo

select pull request section
developing and contributing to OSIRIS
create a pull request to add your changes to main OSIRIS repo

select pull request section

select to create new PR

New pull request
developing and contributing to OSIRIS
create a pull request to add your changes to main OSIRIS repo

select pull request section

select to create new PR

Open a pull request
Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

this is a test

Leave a comment

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Create pull request
developing and contributing to OSIRIS
create a pull request to add your changes to main OSIRIS repo

- select pull request section
- select to create new PR
- master only for hotfixes
- select your branch for contribution
developing and contributing to OSIRIS
create a pull request to add your changes to main OSIRIS repo

select pull request section

select to create new PR

master only for hotfixes

select your branch for contribution

use comment section as part of documentation
Ensuring OSIRIS is stable
continues integration and buildbot

- your fork
- pull request
- OSIRIS@GitHub

Buildbot

Compile tests:
- gfortran
- intel
- IBM

Unit testing:
- vdf tests
- subroutine tests

Physics tests (small runs):
- LWFA
- Weibel
- warm plasma
Thank you for attention!

**further references**

- good and interactive git tutorial: [http://try.github.com/](http://try.github.com/)
- collection of git references: [https://gist.github.com/jaseemabid/1321592](https://gist.github.com/jaseemabid/1321592)
- git cheat sheet: [https://www.git-tower.com/blog/git-cheat-sheet/](https://www.git-tower.com/blog/git-cheat-sheet/)