Versioning your code (and more)

An illustration with git

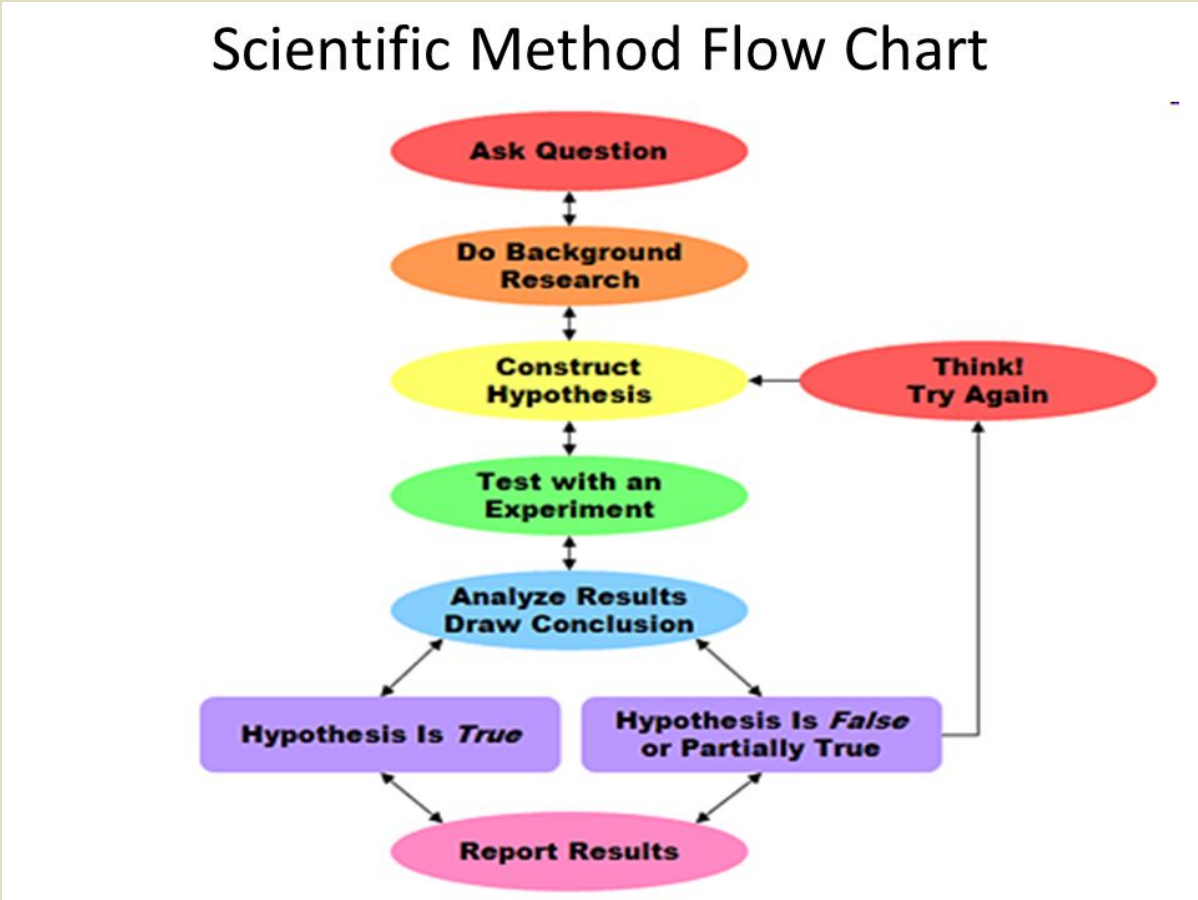
Stéphane DRAY

Lundi 2 novembre 2020

Introduction


UNQUOTE.li
“
Happiness is not at the top of the mountain, but in how to climb.
— Confucius
IMAGE FROM UNSPLASH

Science is a long and step-by-step procedure



https://images.slideplayer.com/27/8939519/slides/slide_4.jpg

Remember the time

Remember the time

Track the changes in a project:

- Data gathering: **Lab notebook**
- Data analysis: **Versioning**
- Article writing: **Versioning**

Remember the time

Track the changes in a project:

- Data gathering: **Lab notebook**
- Data analysis: **Versioning**
- Article writing: **Versioning**

https://miro.medium.com/max/472/1*B0Ykkbhw5bzhHkY79cYrWg.jpeg

An example

https://github.com/sdray/ade4/

Introduction to git

git

There are several tools for versioning. This course focuses on `git`.

`git` is a program that takes **snapshots** of a directory tree and helps navigating between those snapshots.

- archiving/documentation of development steps
- backtracking to previous versions
- handling of multiple versions
- collaborative development
- backup (using external servers)

Especially useful when developing software, writing papers. . .

Some definitions

- **repository**: a directory including with a `.git` directory
- **commit**: a snapchot of the repository (a date, an author, a description, a parent)
- **working copy**: a version of the files of the repository on your computer
- **index**: a temporary space including the modficiations to be commited

All git commands follow the same pattern:

```
git SUBCMD ARGS *
```

Starting with git

Create a repository

```
git init
```

Demo

- Create a directory
- Move to the directory
- Initialize the repostitory

Checking the state of the repository

```
git status
```

Demo

- Check the status of your repository

Put a file under version control

```
git add FILE
```

Files in the directory are ignored by git unless they are explicitly added to the repository

Demo

- Create a text file
- Check the status of your repository
- Add the file to your repository
- Check the status of your repository

Take a snapshot

```
git commit -a -m MSG
```

A snapshot in git is called a commit. The -m option introduces a message describing the changes in the new snapshot.

Demo

- Take a snaphsot of your repository
- Check the status of your repository

Print commit history

```
git log
```

Demo

- Display the history

Print an old version of a file

```
git show COMMIT ID:PATH
```

You can retrieve commit id using git log. Only the 6-8 letters of the id are generally enough

Demo

- Display the content of your text file at the previous commit

Summary

Here are the basic commands to archive your work in a git repo:

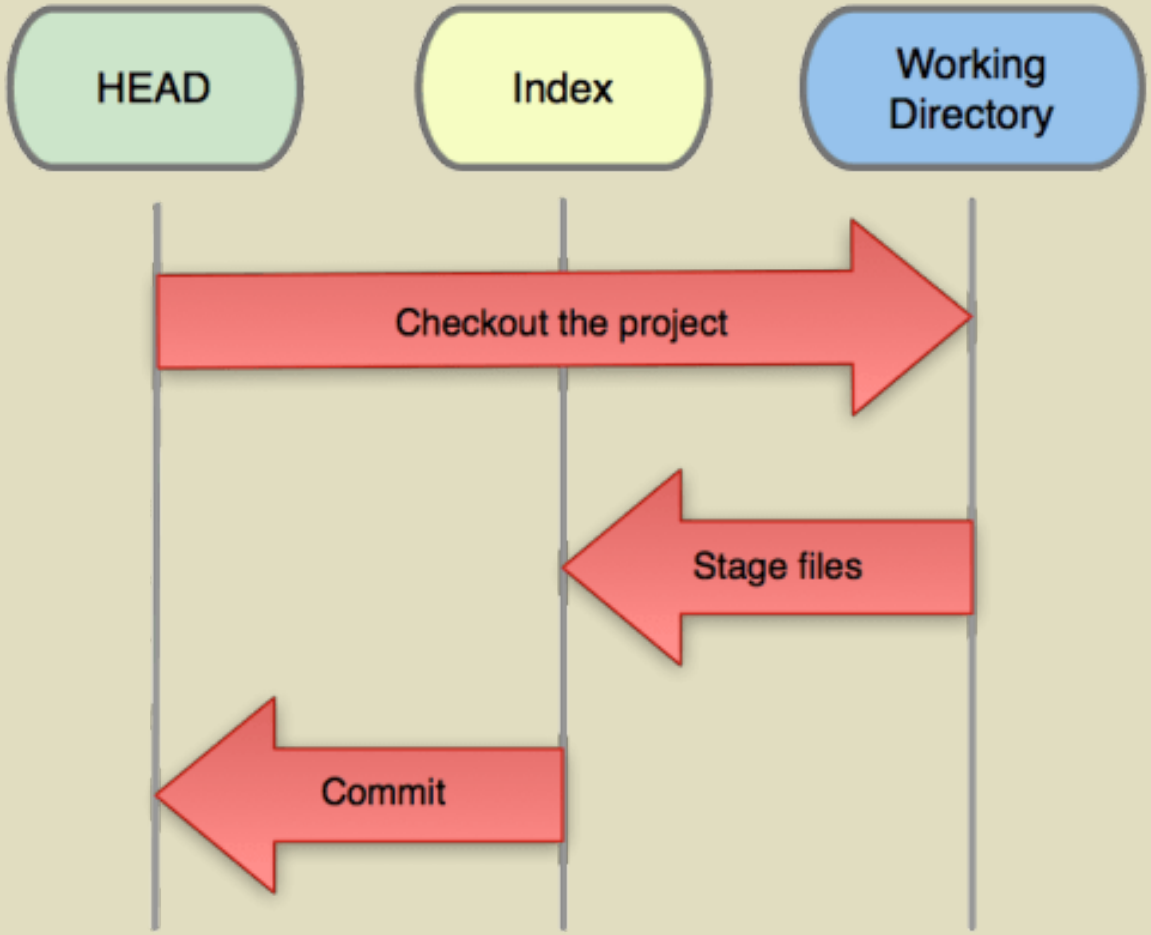
I want to	I use
Initialize a new (empty) git repo	git init
Put files under version control	git add file1 file2 ...
Take a snapshot	git commit -a -m "commit descr"
See commit history	git log
See previous version of a file	git show aazef345:src/foo.c

The 3 areas of git

Revisiting git

- **Working directory** : your files, this is the place where you edit contents
- **Commit graph** : snapshot storage
- **Index** : a place in-between to prepare commits. It contains a list of modifications selected for the next commit

Moving between the three areas



https://git-scm.com/blog/2011/07/11/reset.html

Moving between the three areas

- `git add` put updates of the working directory in the index
- `git commit` builds a commit from the contents of the index (option -a selects all modifications and put them in the index before committing)
- `git diff` shows differences between working directory and the index. With --cached option, shows differences between index and last commit
- `git checkout` update the working directory with the content of the repo
- `git reset HEAD` remove the last changes from the index

Using a distant server

Why using a distant server

- A backup of your repository
- Work with others
- Increase your visibility
- Increase interactions with users
- Easy distribution of devel versions

There are several source hosting services run by academic or private institutions.

Cloning a repo

```
git clone URL
```

This command create an (independent) copy of an existing repo, with all its history.

Demo

- Try to clone the repository of the course
- Display the history of the repository

Interact with the distant repository

```
git pull
```

This command will fetch commits from the cloned repo and update the working directory to latest commit.

```
git push
```

This command will send commits to the cloned repo. It requires write access.

Demo

- Try to obtain the last version of the course (after I update it)
- Modify the course and send your commit

Summary

Here are the basic commands to work with a distant repo:

I want to	I use
Clone a distant repo	git clone https://github.com/...
Update local repo	git pull
Send commits to distant repo	git push

Starting a new project on GitHub

Identification

It is possible to use GitHub with https protocol but you will be asked for your GitHub username and password for each action.

ssh provides another way to access to a Git repository. If you generate ssh public and private keys, you can use git without username/password.

```
cd ~/
ssh -keygen - t rsa
```

Demo

- Copy the public key on GitHub (Settings)
- Create a new project
- Clone the repo
- Do a few commits
- Push your commits
- Modify and have a look to README.md
- Create an issue

Dealing with branches

Commit graph and branches

Sometimes, you want to do experimental developments that diverge from the main line of development. Branches allow to "play" without messing with that main line.

- The main feature of branches is to allow switching between different versions very easily
- A branch should correspond to a topic, e.g. a bug fix, new feature, etc.
- especially useful when unsure when/if some modifications will be included (that is, very often)
- it permits several concurrent development processes

Understanding branches

- commits form a graph
- each commit is linked to its parent(s)
- a branch is a pointer to a commit
- after repo init, there is a single branch called master
- at all time, one branch is selected as the ”current branch”

Demo https://learngitbranching.js.org/?NODEMO

Branches in practice

- After a commit, a new commit is created in the graph and the current branch is moved to the new commit
- A new branch can be created by

```
git branch BRANCH_NAME
```

The new branch is located at the same location than the current branch. Commits act on the current branch

- It is possible to switch to a branch by

```
git checkout BRANCH_NAME
```

Current branch (and the working directory) is updated to reflect the state of the corresponding commit

Branches in practice

- A branch can be deleted by

```
git branch -d BRANCH_NAME
```

Only the pointer is deleted, not the pointed commits but commit may be hard to find after branch deletion.

- Two branches can be merged by

```
git merge BRANCH_NAME
```

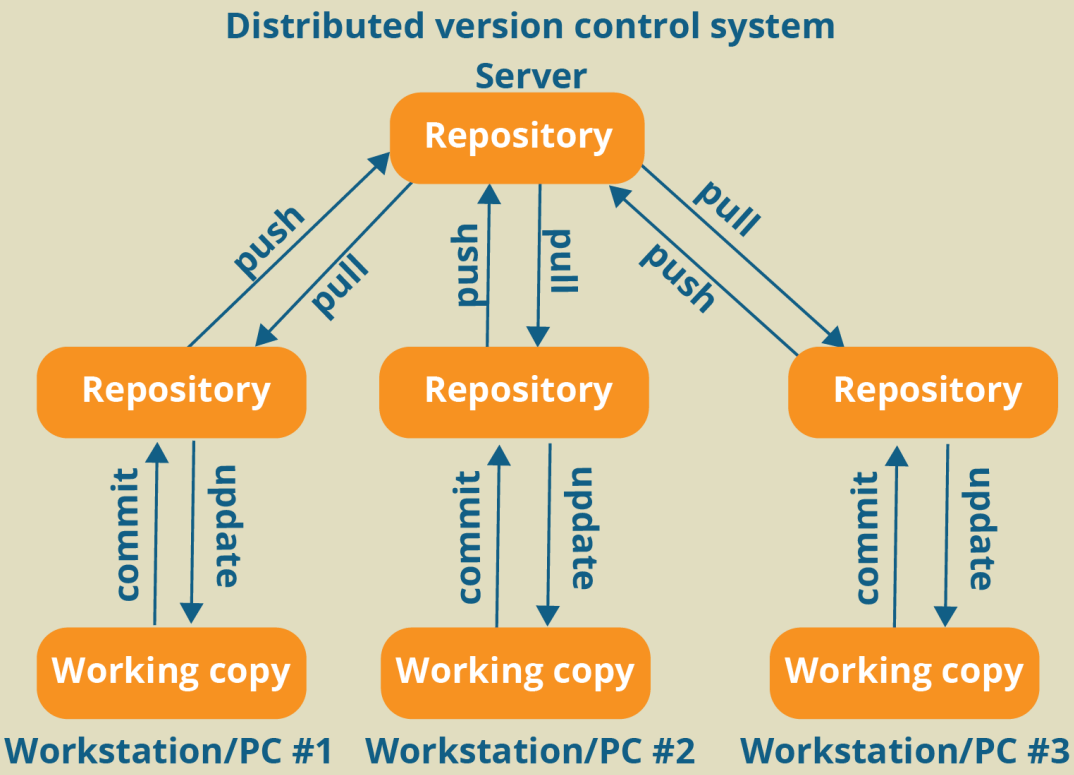
git will try to merge the snapshot pointed by current branch and the one pointed by the branch given in argument.

It creates a new commit and moves the current branch to the newly created commit.

The newly created commit has two parents!

Work with others

A distributed system



https://miro.medium.com/max/1698/1*GgaGcwh5L246YcU5NVDA5A.png

Dealing with conflicts

What happens if someone push before me? A conflict.

Two commits are in conflict when they contain different updates to the same area of a file. They should be resolved manually after merging

To avoid conflicts, update your branch before you start working on something and make frequent and small commits

Demo

- Invite a colleague in your project
- Clone the repo
- Create commit, branches, etc.
- Push
- If they are conflicts, try to solve them

Git and RStudio

GUIs for git

There are several GUIs for git. RStudio provides a convenient way to deal with R projects

Demo

- Create a project and configure RStudio

Acknowledgments

This presentation is strongly inspired from the slides of Philippe Veber (LBBE)

A good summary:

https://github.github.com/training-kit/downloads/fr/github-git-cheat-sheet.pdf