

# Git version control

Benjamin Audren

École Polytechnique Fédérale de Lausanne

October 23, 2013

“This must be Thursday. I never could get the hang of Thursdays”  
(*Arthur Dent, a few minutes before the Earth gets destroyed*)

# Motivation

## What is Version Control ?

- System that keeps tracks of different **versions** of a code (several file), tracking its entire **history** of creation.
- Accessible from a **central repository**, that stores all the versions, and allow communication between developers.
- Can revert to a previous version to **reproduce** exactly the behaviour at a certain time (bug hunting)

# Softwares

## Version Control Softwares

- CVS (Concurrent Versions System)
- Apache Subversion (SVN)
- Git

# Softwares

## Version Control Softwares

- CVS (Concurrent Versions System)
- Apache Subversion (SVN)
- **Git**

*Linus Torvalds*: “Subversion used to say CVS done right: with that slogan there is nowhere you can go. There is no way to do cvs right.”

# Differences between svn and git

git vs svn

- remote repositories **both**

# Differences between svn and git

## git vs svn

- remote repositories **both**
- merging capabilities **both**

# Differences between svn and git

## git vs svn

- remote repositories **both**
- merging capabilities **both**
- ok, good merging capabilities **git**



# Differences between svn and git

## git vs svn

- remote repositories **both**
- merging capabilities **both**
- ok, good merging capabilities **git**
- local repository is an entire copy **git**

# Differences between svn and git

## git vs svn

- remote repositories **both**
- merging capabilities **both**
- ok, good merging capabilities **git**
- local repository is an entire copy **git**
- offline work ! **git**

# Differences between svn and git

## git vs svn

- remote repositories **both**
- merging capabilities **both**
- ok, good merging capabilities **git**
- local repository is an entire copy **git**
- offline work ! **git**
- pull-request system (user submitted patch) **git**

# Workflow

**Basic**, master branch only

Into an existing or newly created directory

- `git init`

# Workflow

**Basic**, master branch only

Into an existing or newly created directory

- `git init`
- `$EDITOR file.py`

# Workflow

**Basic**, master branch only

Into an existing or newly created directory

- `git init`
- `$EDITOR file.py`
- `git status`

# Workflow

**Basic**, master branch only

Into an existing or newly created directory

- `git init`
- `$EDITOR file.py`
- `git status`
- `git add file.py`

# Workflow

Basic, master branch only

Into an existing or newly created directory

- `git init`
- `$EDITOR file.py`
- `git status`
- `git add file.py`
- `git commit -m 'file added'`



# Reminder: Staging Area

After modifying

- Local version is affected: **Working Tree**

# Reminder: Staging Area

## After modifying

- Local version is affected: **Working Tree**
- **add** files to the index (staging area)

# Reminder: Staging Area

## After modifying

- Local version is affected: **Working Tree**
- **add** files to the index (staging area)
- **commit them** to the repository

# Reminder: Staging Area

## After modifying

- Local version is affected: **Working Tree**
- **add** files to the index (staging area)
- **commit them** to the repository

## Beware

- You can easily **un-add** files from the staging area

# Reminder: Staging Area

## After modifying

- Local version is affected: **Working Tree**
- **add** files to the index (staging area)
- **commit them** to the repository

## Beware

- You can easily **un-add** files from the staging area
- You can easily **un-commit** files once committed (**amend**, **rebase**)...

# Reminder: Staging Area

## After modifying

- Local version is affected: **Working Tree**
- **add** files to the index (staging area)
- **commit them** to the repository

## Beware

- You can easily **un-add** files from the staging area
- You can easily **un-commit** files once committed (**amend**, **rebase**)...
- But you should probably **avoid** doing it...

# Reminder: Staging Area

## After modifying

- Local version is affected: **Working Tree**
- **add** files to the index (staging area)
- **commit them** to the repository

## Beware

- You can easily **un-add** files from the staging area
- You can easily **un-commit** files once committed (**amend**, **rebase**)...
- But you should probably **avoid** doing it...
- Except **locally only**

# Important other things

## Configuration

### Important files to be present

- a `.gitignore`
- a `README.md` or any other markup language

The `git.config` file should contain at minima:

```
[user]
```

```
name = First Last
```

```
email = first.last@somewhere.com
```



# Important other things

More basic commands

See the history  
with `git log`

# Important other things

## More basic commands

See the history

with `git log`

Recover previous version of a file

with `git checkout file.py`

# Important other things

## More basic commands

See the history

with `git log`

Recover previous version of a file

with `git checkout file.py`

Unstage a file for commit

with `git reset HEAD file.py`

# Hands-on

go ahead !

## Exercise

Go to last week directory where you created a **very simple module**, and turn it into a git repository.

# Hands-on

go ahead !

## Exercise

Go to last week directory where you created a **very simple module**, and turn it into a git repository.

## Do some work

Make some changes, commit, try to see the history