

# Using Git and GitHub

SSOO

University of Alcalá

## Objectives

1. Understand the need of version control
2. Implement software development workflows with Git and Github

## Bibliography

1. GitHub Guides. ([Link](#))

# Table of Contents

1. Version control
2. Git
  - What is Git?
  - Git sites
  - Git vs. SVN
3. Using Git
  - Conflicts
  - Commits
  - Branches
  - Tags
  - Good practices
4. GitHub
  - Features
  - README
  - Markdown

# Version control

## Version control systems

Version control systems (VCS) keep track of changes to source code. Allows multiple people to edit a project in a predictable manner.

### Main open source VCS

- 1982 RCS
- 1990 CVS
- 2000 Subversion
- 2005 Git/Mercurial

There are many proprietary ones but `Git` is now the most popular one by far.  
All software should be under a version control system, if not, it ain't software!

# Git

## What is Git?



Git is an open source distributed version control system,  
created by Linus Torvald.  
<https://git-scm.com/>  
(Interactive tutorial)



# Git

## Git sites

It is easier to start with free hosting sites instead of maintaining your own server.

- **GitHub**: public repositories (as many as you want), but private ones are not free (except for academia). It is now part of Microsoft
- **Bitbucket**: allow us to keep private repositories limiting the number of collaborators.
- **GitLab**: both public and private without limitations. It is becoming more popular.
- Others ...

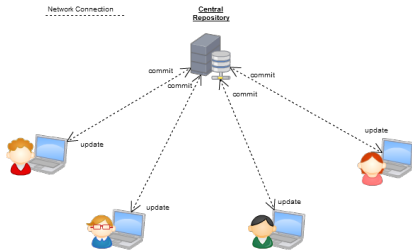
It is typically used as central repository:

- from which everyone pulls other people's changes
- to which everyone pushes changes they have made

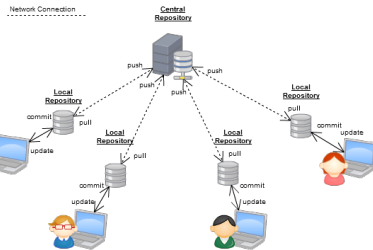
# Git

## Git vs. SVN (I)

### Centralized (SVN)



### Distributed (Git)



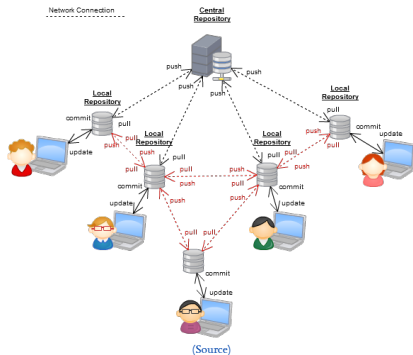
(Source)

Disclaimer: Do not pay attention to the labels of these diagrams

# Git

## Git vs. SVN (II)

### Fully distributed (Git)



### Key Git concepts to know

- repository (local or not)
- clone
- commit, push
- pull, fetch
- remote, origin
- merge



# Using Git

## Conflicts

If two people both modify the same file, the first to push wins. The second person will have to pull and merge before pushing.

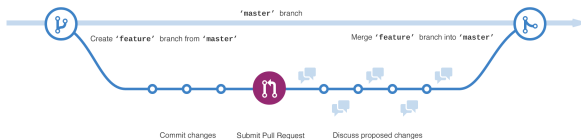
- Changes in different parts of a file are automatically merged
- Changes in the same part of a file cause conflicts
  - Select either your changes or remote, or a mix of the two

# Using Git

## Commits

- Every commit has an ID (its hash)

# Using Git Branches



Branches are used extensively (e.g. some like feature branches).

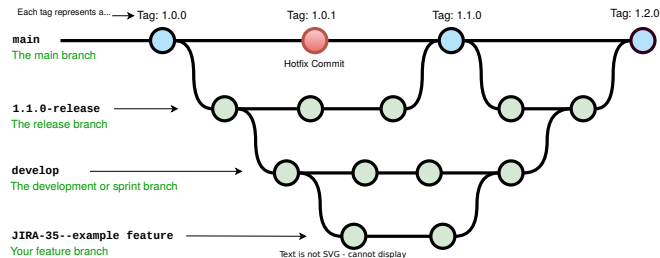
- A repository (local and remote) can have explicit branches
- The default branch is called **master**
- A **merge** is a fusion between two branches

Do not use branches in the project!

# Using Git

## Example diagram for a GiT workflow:

See: <https://nvie.com/posts/a-successful-git-branching-model/>



A tag is a pointer to a specific point in the repository history

- Tags usually have names (e.g. "v1.1")
- Widely used to keep and publish software releases

# Using Git

## Good practices

Learn on the job: the best way to learn it is by using it.

### Best practices

- Regularly push and pull (at least daily, in general)
- $\Rightarrow$  Test before pushing!  $\Leftarrow$
- Don't push half-baked changes
- Don't pull if you're in the middle of a task
- Never commit temporal/intermediate files
- The master must be a clean and functional version of the project

Remember: Git never overwrites local changes without an explicit order

- .. even with a `git pull`

# GitHub

## Features

### Free Git hosting provider

- Free public repositories

### User interface to Git

- Repository browser

### Added value Git operations

- Social network
- Collaborative tools
- Pull requests
- Issue tracking
- Web hosting
- Markdown integration
- Organizations



# GitHub

## Key concepts

### Key GitHub concepts to know

- Pull request
- Fork

# GitHub

## README

### Special file: README.md

- Contains information about the project
- Automatically visualized
- md means Markdown



# Markdown (I)



## Markdown: Trivial markup

- Simple
- Very simple
- Extremely simple
- Did I say it's simple?

## VERY powerful

- Several outputs
- Professional quality
- ... and simple!

# Markdown (II)

## Markdown example

```
# I am a header
```

```
## I am a subheader
```

```
Regular , *italic* and **bold**
```

```
- List item 1
```

```
- List item 2
```

```
[ I am a link ]( http : // foo . com )
```

```
![ I am a pic ]( markdown . png )
```

```
~~~C
```

```
printf( " Hello , world " );
```

```
~~~
```

I am a header

I am a subheader

Regular, *italic* and **bold**

- List item 1
- List item 2

[I am a link](#)

I am a pic

```
printf("Hello, world");
```