

Version control, Git, GitHub and GitFlow

Criscely Luján^{1,2}

criscely.lujan@ird.fr

Nicolas Barrier²

nicolas.barrier@ird.fr

¹Université Paris-Sud, UMR MARBEC

²IRD, UMR MARBEC

April 11, 2019

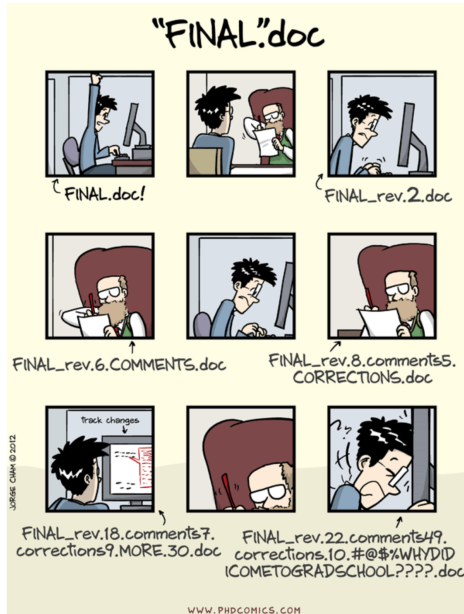


Also known as **revision control** or **source control**.

... *“is the management of changes:*

- *documents*
- *computer programs*
- *large web sites*
- *other collections of information ... ”*

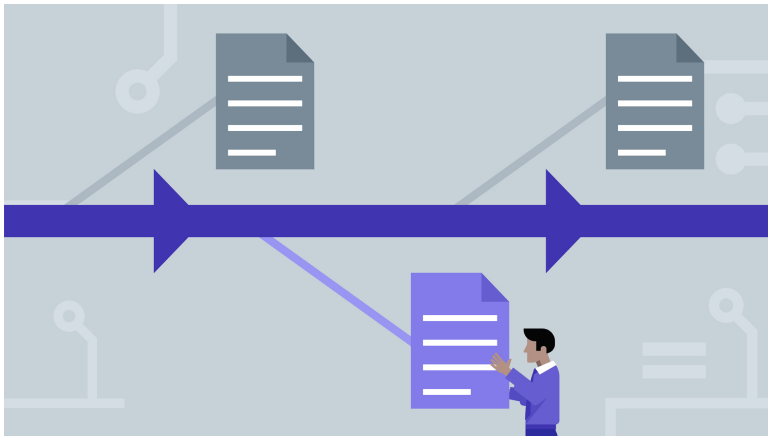
Why version control is important?



Why version control is important?

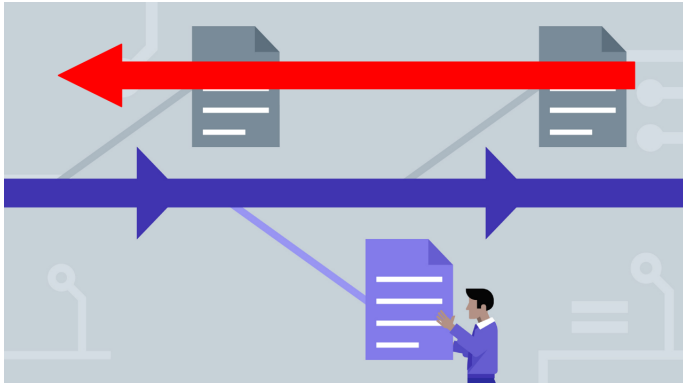
Storing **version** (properly).

- Saving successive changes (“commit”)
- Versioning (v0.1)



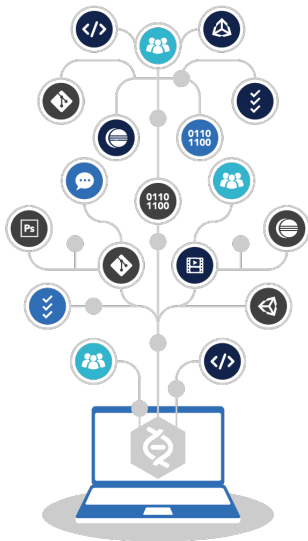
Why version control is important?

Restoring previous versions.



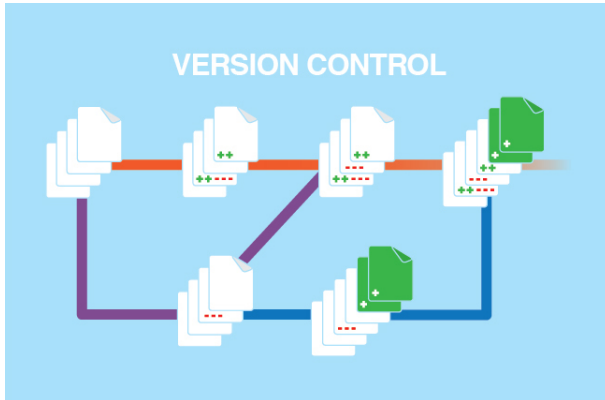
Why version control is important?

Collaborations (networking).



Why version control is important?

Save **time**.



Version control software

Version control software			[hide]
Years, where available, indicate the date of first stable release. Systems with names <i>in italics</i> are no longer maintained or have planned end-of-life dates.			
Local only	Free/open-source	RCS (1982) · SCCS (1972)	
	Proprietary	PVCS (1985) · QVCS (1991)	
Client–server	Free/open-source	CVS (1986, 1990 in C) · CVSNT (1998) · QVCS Enterprise (1998) · Subversion (2000)	
	Proprietary	AccuRev SCM (2002) · ClearCase (1992) · CMVC (1994) · Dimensions CM (1980s) · DSEE (1984) · Endevor (1980s) · Integrity (2001) · Panvalet (1970s) · Perforce Helix (1995) · SCLM (1980s?) · Software Change Manager (1970s) · StarTeam (1995) · Surround SCM (2002) · Synergy (1990) · Team Concert (2008) · Team Foundation Server (2005) · Visual Studio Team Services (2014) · Vault (2003) · Visual SourceSafe (1994)	
Distributed	Free/open-source	ArX (2003) · BitKeeper (2000) · Codeville (2005) · Darcs (2002) · DCVS (2002) · Fossil (2007) · Git (2005) · GNU arch (2001) · GNU Bazaar (2005) · Mercurial (2005) · Monotone (2003) · Pijul (2015) · SVK (2003) · Veracity (2010)	
	Proprietary	TeamWare (1990s?) · Code Co-op (1997) · Plastic SCM (2006) · Team Foundation Server (2013) · Visual Studio Team Services (2014)	
Concepts	Baseline · Branch · Changeset · Commit · Data comparison · Delta compression · Fork (Gated commit) · Interleaved deltas · Merge · Repository · Tag · Trunk		
Category · Comparison · List			

The diagram shows a sequence of numbered boxes (1-10) connected by arrows. Box 1 is labeled 'Trunk'. Box 2 is labeled 'Branches'. Box 3 is labeled 'Merges'. Box 4 is labeled 'Tags'. Box 5 is labeled 'Tags'. Box 6 is labeled 'Tags'. Box 7 is labeled 'Tags'. Box 8 is labeled 'Tags'. Box 9 is labeled 'Tags'. Box 10 is labeled 'Tags'. A note at the bottom right says 'This could be a distributed or centralized system'.



What is Git?

Git is a distributed version control system for tracking changes in source code during the development of software.



Why use Git?

- **Popular and successful**
 - Active development
 - Fast
- **Distributed**
 - Work online and offline
 - Collaborate with large groups
- **Tracks any type of file**
 - Works best with text
- **Branching**
 - Smarter merges

What is GitHub Inc.?

GitHub is a web-based hosting service for version control using **Git**.

The GitHub logo, which consists of the word "GitHub" in a bold, black, sans-serif font.

[📄 Download logo](#)



[📄 Download mark](#)



[📄 Download Octocat](#)

- Access to the control and collaboration features for every project.

The screenshot shows the GitHub repository settings page for a repository named 'osmose_configurations'. At the top, there is a navigation bar with links for Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings (which is highlighted with an orange underline). On the left side, there is a sidebar menu with the following options: Options (highlighted with an orange bar), Collaborators, Branches, Webhooks, Notifications, Integrations & services, and Deploy keys. The main content area is titled 'Settings' and contains two sections: 'Repository name' and 'Features'. The 'Repository name' section shows the current name 'osmose_configurations' in a text input field and a 'Rename' button. The 'Features' section lists several features that are all checked: 'Wikis' (with a description: 'GitHub Wikis is a simple way to let others contribute content. Any GitHub user can create and edit pages to use for documentation, examples, support, or anything you wish.'), 'Restrict editing to collaborators only', 'Issues' (with a description: 'Issues integrate lightweight task tracking into your repository. Keep projects on track with issue labels and milestones, and reference them in commit messages.'), and 'Projects' (with a description: 'Project boards on GitHub help you organize and prioritize your work. You can create project boards for specific feature work,'). Below the 'Issues' feature, there is a light blue box with the text 'Get organized with issue templates' and 'Give contributors issue templates that help you cut through the noise and help them push your project forward.', followed by a green button labeled 'Set up templates'.

<> Code ① Issues 0 🔄 Pull requests 0 📁 Projects 0 📖 Wiki 📊 Insights ⚙️ Settings

Options

- Collaborators
- Branches
- Webhooks
- Notifications
- Integrations & services
- Deploy keys

Settings

Repository name


osmose_configurations **Rename**

Features


- ☒ **Wikis**
GitHub Wikis is a simple way to let others contribute content. Any GitHub user can create and edit pages to use for documentation, examples, support, or anything you wish.
- ☒ **Restrict editing to collaborators only**
- ☒ **Issues**
Issues integrate lightweight task tracking into your repository. Keep projects on track with issue labels and milestones, and reference them in commit messages.

Get organized with issue templates
Give contributors issue templates that help you cut through the noise and help them push your project forward. **Set up templates**
- ☒ **Projects**
Project boards on GitHub help you organize and prioritize your work. You can create project boards for specific feature work,

- Work with public and private **repositories**.


PUBLIC 

Owner

 **hubot** ▾

/


Repository name

hello-world 


Great repository names are short and memorable. Need inspiration? How about **petulant-shame**.

Description (optional)

Just another repository

☒  **Public**

Anyone can see this repository. You choose who can commit.


☐  **Private**

You choose who can see and commit to this repository.

☒ **Initialize this repository with a README**

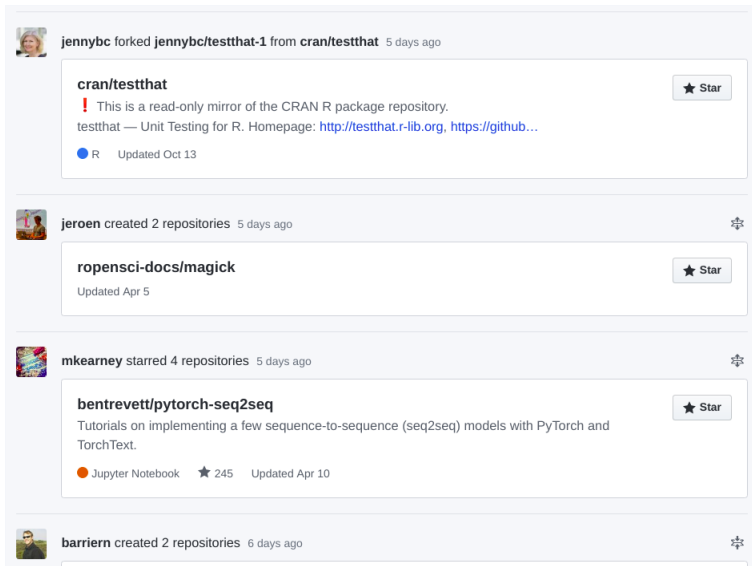
This will allow you to `git clone` the repository immediately. Skip this step if you have already run `git init` locally.

Add .gitignore: **None** ▾

Add a license: **None** ▾ 

Create repository


- Develop a **networking**.

**jennybc** forked **jennybc/testthat-1** from **cran/testthat** 5 days ago

cran/testthat ★ Star


! This is a read-only mirror of the CRAN R package repository.
testthat — Unit Testing for R. Homepage: <http://testthat.r-lib.org>, <https://github.com/r-lib/testthat>

R Updated Oct 13

**jeroen** created 2 repositories 5 days ago 🔔

ropensci-docs/magick ★ Star


Updated Apr 5

**mkearney** starred 4 repositories 5 days ago 🔔

bentrevett/pytorch-seq2seq ★ Star

Tutorials on implementing a few sequence-to-sequence (seq2seq) models with PyTorch and TorchText.


Jupyter Notebook ★ 245 Updated Apr 10

**barriern** created 2 repositories 6 days ago 🔔

- **Plans** for enterprise, teams, pro and free accounts.

Plans for every developer

Whether you're starting an open source project or choosing new tools for your team, we've got you covered.



Individuals

Free	Pro
\$0	\$7
Per month The basics of GitHub for every developer	Per month Pro tools for developers with advanced requirements
<ul style="list-style-type: none">Unlimited public repositoriesUnlimited private repositories3 collaborators for private repositoriesIssues and bug trackingProject management	<ul style="list-style-type: none">Unlimited public repositoriesUnlimited private repositoriesUnlimited collaboratorsIssues and bug trackingProject managementAdvanced tools and insights
Included in Pro	Already signed up

Included free alongside other real-world development tools in the [GitHub Student Developer Pack](#)

Teams

Team	Enterprise
\$9 Per user / month Advanced collaboration and management tools for teams	Contact Sales for pricing Security, compliance, and deployment controls for organizations
<ul style="list-style-type: none">Unlimited public repositoriesUnlimited private repositoriesTeam access controlsUser management and billingIssues and bug trackingProject managementAdvanced tools and insights	<ul style="list-style-type: none">Everything included in TeamSelf-hosted or cloud-hostedSAML single sign-onAccess provisioningSimplified account administrationUnified search and contributionsPriority support99.95% uptime SLA for Enterprise CloudInvoice billingAdvanced auditing
<small>Starts at \$25 / month and includes your first 5 users</small> <small>Free to academic faculty for teaching or non-profit research</small>	<small>Questions? Learn more about Enterprise</small> <small>Free for educational institutions participating in the GitHub Education program</small>

- Is the **largest** host of source code in the world! (*28 million users, 57 million repositories (28 million public) - June 2018*).

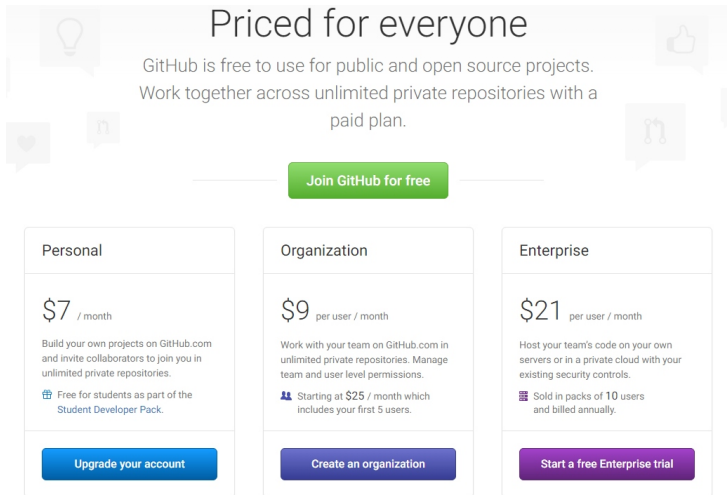


Register a GitHub account

- Create an account in [GitHub](#) is free!
- Free private repositories
 - Students, faculty, and educational / research staff: [GitHub Education](#).
 - Official nonprofit organizations and charities: [GitHub for Good](#).

Register a GitHub account

- Pay for private repositories
 - Individual cost is 7 dollars per month: [GitHub Pricing](#).






The screenshot shows the GitHub Pricing page. At the top, it says "Priced for everyone". Below this, it states "GitHub is free to use for public and open source projects. Work together across unlimited private repositories with a paid plan." There is a green button that says "Join GitHub for free". Below this, there are three pricing tiers: Personal (\$7/month), Organization (\$9 per user/month), and Enterprise (\$21 per user/month). Each tier has a description of what you can do and a button to upgrade or create an account.

Priced for everyone

GitHub is free to use for public and open source projects. Work together across unlimited private repositories with a paid plan.

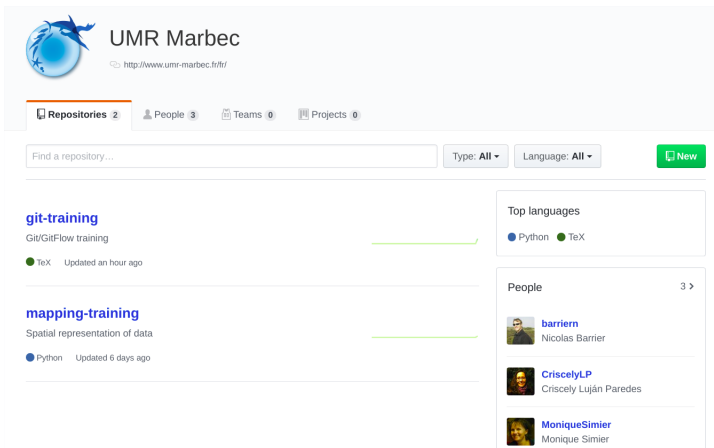
[Join GitHub for free](#)

Personal	Organization	Enterprise
\$7 / month	\$9 per user / month	\$21 per user / month
Build your own projects on GitHub.com and invite collaborators to join you in unlimited private repositories.	Work with your team on GitHub.com in unlimited private repositories. Manage team and user level permissions.	Host your team's code on your own servers or in a private cloud with your existing security controls.
 Free for students as part of the Student Developer Pack.	 Starting at \$25 / month which includes your first 5 users.	 Sold in packs of 10 users and billed annually.
Upgrade your account	Create an organization	Start a free Enterprise trial

Marbec in GitHub

All the materials of Pole Modelisation's technical "workshop" are now stored in an institutionnal GitHub account:

<https://github.com/umr-marbec>.



The screenshot shows the GitHub profile page for UMR Marbec. At the top, there is a profile picture of a blue globe with a bird, the name "UMR Marbec", and the website "http://www.umr-marbec.fr/fr/". Below this, navigation tabs show "Repositories 2", "People 3", "Teams 0", and "Projects 0". A search bar "Find a repository..." is present, along with filters for "Type: All" and "Language: All", and a green "New" button. The repository list shows two items: "git-training" (Git/GitFlow training, TeX, Updated an hour ago) and "mapping-training" (Spatial representation of data, Python, Updated 6 days ago). On the right, a "Top languages" section shows Python and TeX, and a "People" section lists three contributors: barriern (Nicolas Barrier), CriscelyLP (Criscely Luján Paredes), and MoniqueSimier (Monique Simier).

GitHub is a private US company. There are also *institutional* repositories on which Git can be used:

- [Sourcesup](#): this is a Renater platform (login possible from any French research institute or through CRU accounts)
- [Forge Ifremer](#): very close to SourceSup (Ifremer extranet account required)
- [IRD GitLab](#): GitLab IRD platform (IRD account required).

However, the projects hosted on these repositories may have less visibility...

Git clients

Git and Git client **are not** the same! Like R and RStudio is not the same thing!

Git client:

- IDE (Integrated development environment)!
- Make the experience more pleasant providing a richer visual representation.

Some example of Git clients:

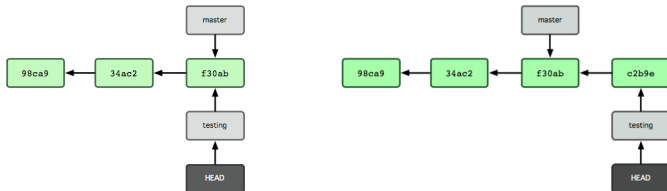
- [SourceTree](#)
- [GitKraken](#)
- [GitUp](#)
- [SmartGit](#)
- [git-cola](#)
- [RStudio](#)

Git branches

One main advantage of Git is the use of *branches*, which allow multiple developments of the same code at the same time.

Definition

A branch in Git is simply a lightweight movable pointer to one of the commits.



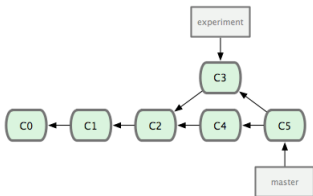
In this example, the `master` branch points to the `f30ab` commit, while the `testing` branch points to the `c2b9e` one. `HEAD` points to the active branch (here, `testing`).

Source: <https://git-scm.com/book/en/v1/Git-Branching-What-a-Branch-Is>

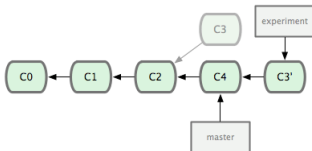
Merging branches

To merge a branch (for instance a feature branch) to another branch (for instance the main one), several options are offered.

- merge: Three-points branch (common ancestor + tips of the two branches)
- rebase: Compresses all the changes into a single patch.



(a) Merge



(b) Rebase

Figure: Merging versus rebasing

There are several ways to use Git branches (we talk about **workflows**).

- *Centralized workflow*: one main branch, everyone commit in the same place.
- *Feature Branch Workflow*: developments are made in dedicated branches (feature branches), which are regularly merged into the master one.
- **Gitflow Workflow**: Strict branching model designed around the project release.

Source: <https://www.atlassian.com/git/tutorials/comparing-workflows>

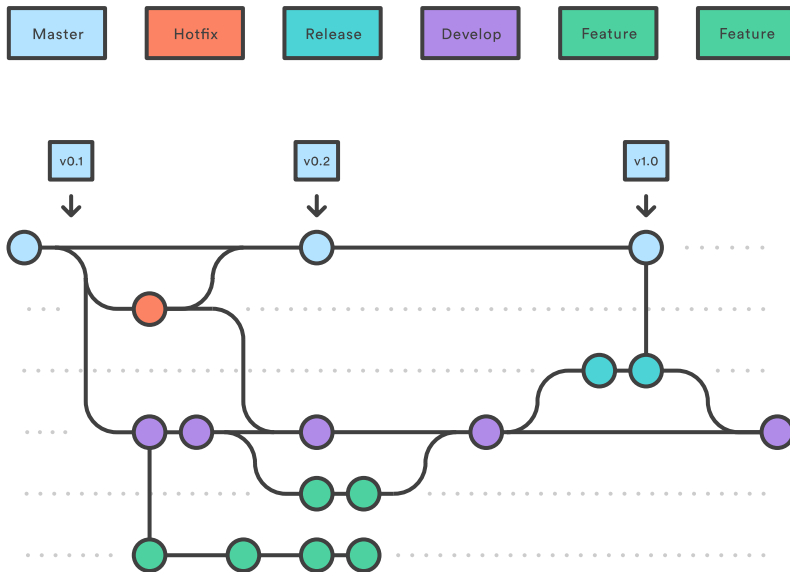
GitFlow workflow contains two main branches:

- **master**: official release history. Branch which is shared to the world!
- **develop**: integration branch for features

It also contains additional temporal branches:

- **feature**: feature branches (one for each new feature to add to the code)
- **release**: branch created when enough features have been added (new version of the code) to develop
- **hotfix**: branch for maintenance and bug correction of the production release

In summary...



Source: <https://www.atlassian.com/git/tutorials/comparing-workflows>

Thanks for your attention

Now, let's crack on it!



Source: <https://www.pinterest.fr/pin/447263806724736402/>