



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Introduction to Git



Kevin Balem - LOPS - Ifremer

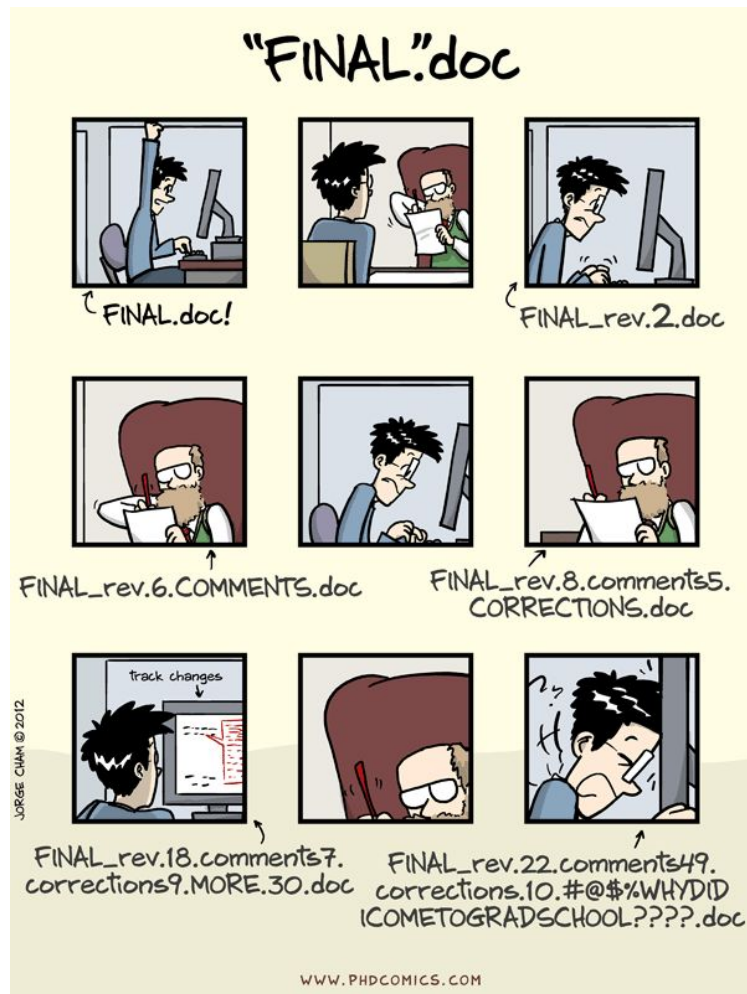


kevin.balem@ifremer.fr * github.com/quai20 * [@balemkevin](https://twitter.com/balemkevin)

1. What's GIT ?
2. What's a repository ?
3. What's a remote ?
4. Introduction to branches
5. Conflicts
6. Pull/Merge Request
7. Projects
8. If time, quick intro to CI/CD

1. What's & why **Git** ?

“Git is a **versioning** software for **tracking changes** in any set of **files**, usually used for coordinating work among **programmers** collaboratively **developing** source **code** during software development. [...] Git was created by Linus Torvalds in 2005 for development of the Linux kernel, with other kernel developers contributing to its initial development.”

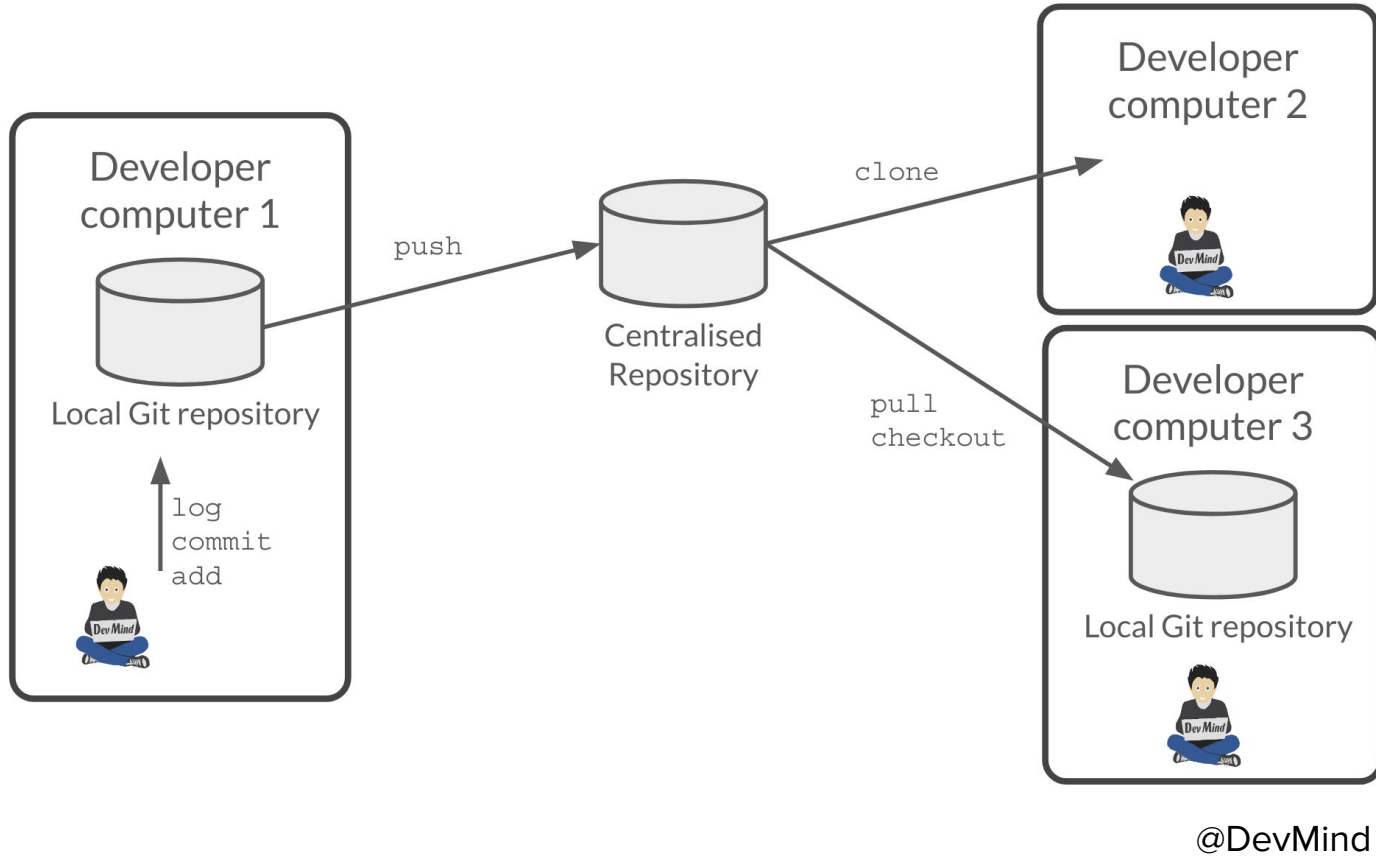


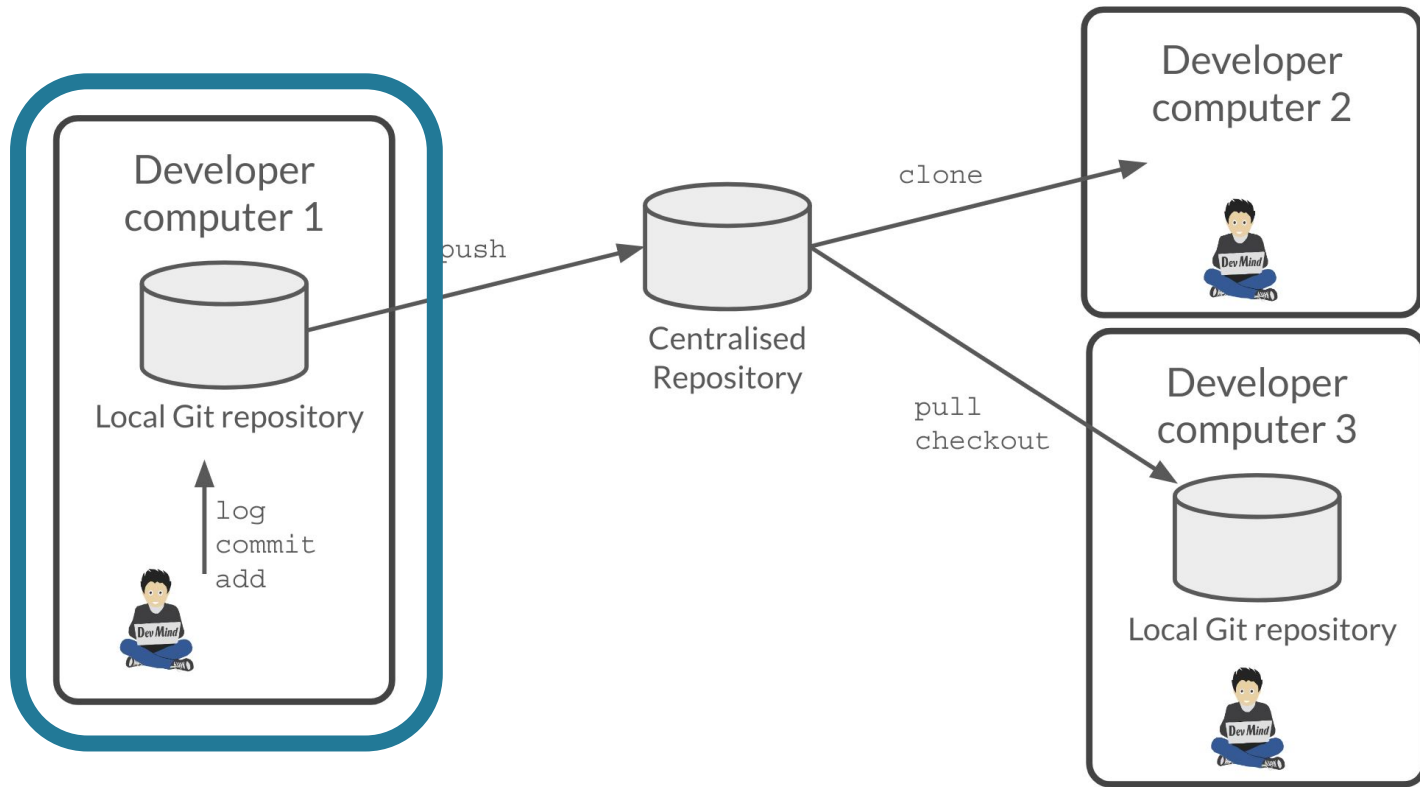
Concretely, it's a set of functions to track file changes and manage versions of your files.

```
kbalem@minicotton: ~  
kbalem@minicotton: ~  
(base) kbalem@minicotton:~$ git  
usage: git [--version] [--help] [-C <path>] [-c name=value]  
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]  
        [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]  
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]  
        <command> [<args>]  
  
Ci-dessous les commandes Git habituelles dans diverses situations :  
  
démarrer une zone de travail (voir aussi : git help tutorial)  
  clone  Cloner un dépôt dans un nouveau répertoire  
  init   Créer un dépôt Git vide ou réinitialiser un existant  
  
travailler sur la modification actuelle (voir aussi : git help revisions)  
  add    Ajouter le contenu de fichiers dans l'index  
  mv     Déplacer ou renommer un fichier, un répertoire, ou un lien symbolique  
  reset  Réinitialiser la HEAD courante à l'état spécifié  
  rm     Supprimer des fichiers de la copie de travail et de l'index  
  
examiner l'historique et l'état (voir aussi : git help revisions)  
  bisect Trouver par recherche binaire la modification qui a introduit un bogue  
  grep   Afficher les lignes correspondant à un motif  
  log    Afficher l'historique des validations  
  show   Afficher différents types d'objets  
  status Afficher l'état de la copie de travail  
  
agrandir, marquer et modifier votre historique  
  branch Lister, créer ou supprimer des branches  
  checkout Basculer de branche ou restaurer la copie de travail  
  commit  Enregistrer les modifications dans le dépôt  
  diff    Afficher les changements entre les validations, entre validation et copie de travail, etc  
  merge   Fusionner deux ou plusieurs historiques de développement ensemble  
  rebase  Reporter les validations locales sur le sommet mis à jour d'une branche amont  
  tag     Créer, lister, supprimer ou vérifier un objet d'étiquette signé avec GPG  
  
collaborer (voir aussi : git help workflows)  
  fetch  Télécharger les objets et références depuis un autre dépôt  
  pull   Rapatrier et intégrer un autre dépôt ou une branche locale  
  push   Mettre à jour les références distantes ainsi que les objets associés  
  
'git help -a' et 'git help -g' listent les sous-commandes disponibles et  
quelques concepts. Voir 'git help <commande>' ou 'git help <concept>'  
pour en lire plus à propos d'une commande spécifique ou d'un concept.  
(base) kbalem@minicotton:~$
```

```
git --version  
sudo add-apt-repository -y ppa:git-core/ppa  
sudo apt-get update  
sudo apt-get install git -y  
git --version
```

2. What's a repository ?





@DevMind

Let's create a local **repo** ...

Create a repository

Repository : a place that stores files and keeps track of any changes

```
> mkdir mycoolproject_<your_name>
> cd mycoolproject_<your_name>
> git status
??
> git init
??
```

```
> git config --global user.name "FIRST_NAME
LAST_NAME"
> git config --global user.email
"MY_NAME@example.com"
> git config --list
```

Add some files in our repo

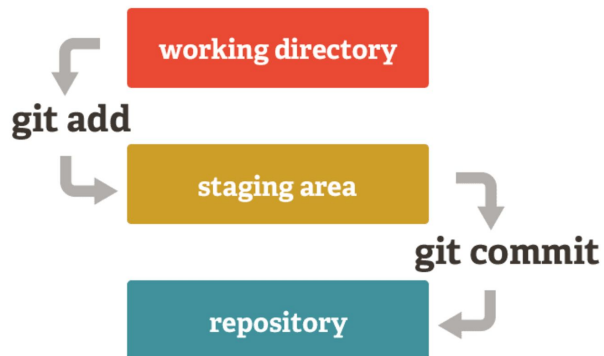
```
> echo "this is my cool project" > readme.txt  
> git status  
??
```

```
> git add readme.txt  
> git status  
??
```

Your first commit

Commit : a snapshot of the repository at a certain point in time. Every commit must be documented by a message.

BE EXPLICIT !



```
> git commit -m "[add] readme.txt"
??
> git status

> git log
> git log --patch
```

Staging area : Intermediate layer of your repo where tracked files are stored before commit. Allow us to add multiple files before ONE commit.

.gitignore : This file will list what you don't want to track.

.keep : This is a convention if you want to commit an empty folder

```
> touch licence.txt
> touch contribute.txt
> git add .
```

```
> touch secret.txt
> git status
> echo "secret.txt" > .gitignore
> git add .gitignore
> git status
```

```
> git commit -m "[add] license and contribute files"
```

```
> mkdir docs
> git status
??
> touch docs/.keep
> git status
> git add docs
> git commit -m "[add] docs folder"
```

A **tracked file** doesn't mean that every change is automatically taken into account. If a file change, you have to **add & commit** the changes

```
> echo "MIT Licencing" >> licence.txt
```

```
??
```

To **delete** a tracked file, you need to remove it from the file system and commit this change.

To **delete** some work that you haven't commit yet (to switch branch or do something else), you can **restore** the file(s) to its previous state. You can also **stash** the work. Then you can the recover it with **git stash pop** or clear the stash with **git stash clear**.

To go back to a previous **commit** and delete what you've done since (**be careful**)

```
> rm contribute.txt  
> git add .  
> git commit -m "[rm] contribute.txt"
```

OR

```
> git rm contribute.tx  
> git commit -m "[rm] contribute.txt"
```

```
> git restore <file>  
> git stash
```

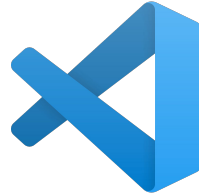
```
> git reset --hard 0d1d7fc32
```

Integration with **IDE** & **Git** desktop clients

<https://github.atom.io/>



<https://code.visualstudio.com/docs/editor/versioncontrol>



<https://www.gitkraken.com/>



<https://github.com/jupyterlab/jupyterlab-git>



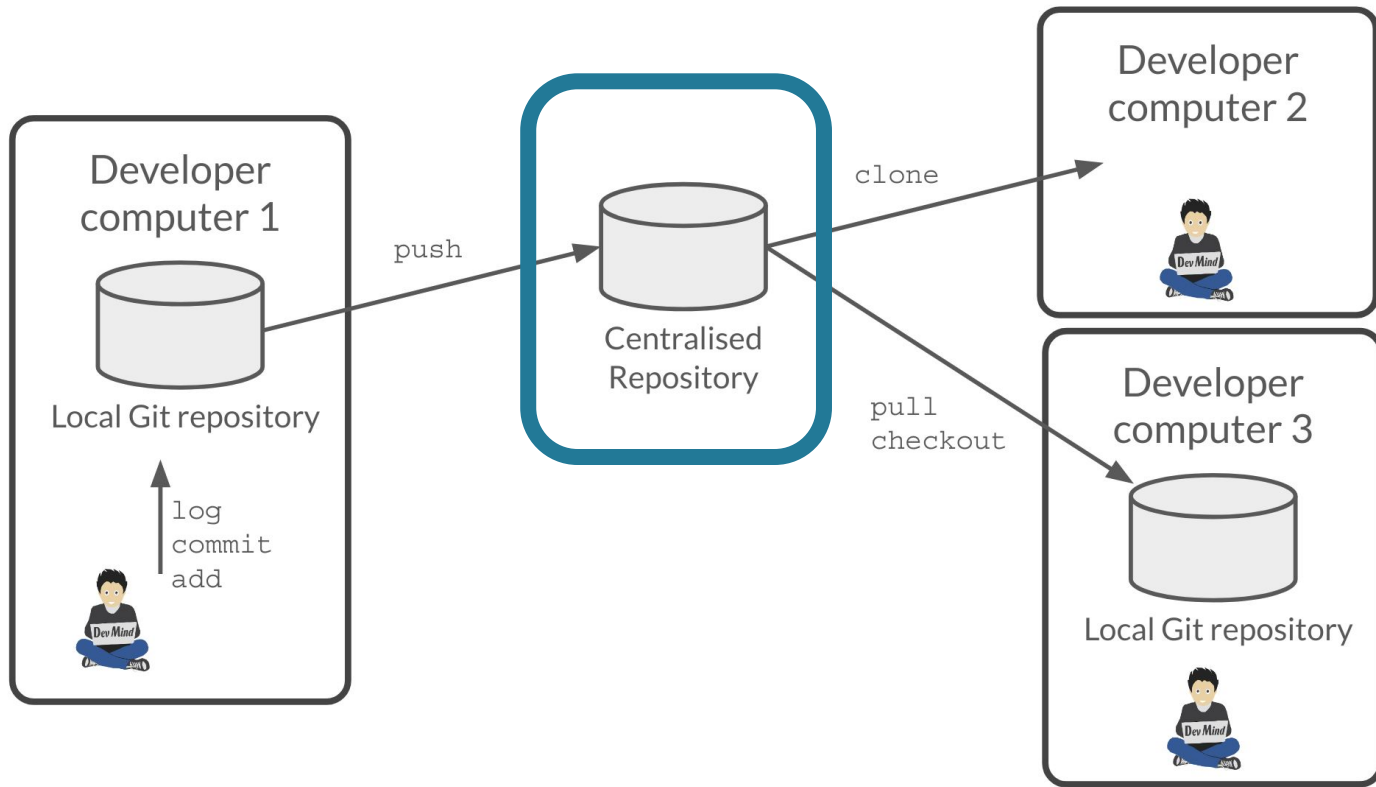
<https://desktop.github.com/>



Let's look at 2 examples



3. What's a **remote** ?



@DevMind



...



<https://github.com>



<https://gitlab.imt-atlantique.fr>

- Create a project
- How to **push** our project on the **remote** ?
- How to **pair** your computer and Gitlab (**SSH Keys**)
- The Gitlab interface to edit files
- How to **pull** changes ?














```
> git remote -v  
> git remote add/set-url [...]  
> git push
```

```
> git pull
```


- Cloning ?

Git receives a full copy of nearly all data that the server has. Every version of every file for the history of the project is pulled down by default when you run git clone.

```
> cd ..  
> rm -rf mycoolproject  
  
> git clone <url>
```

- M mycoolproject
-  Project information
-  Repository
-  Issues 0
-  Merge requests 0
-  CI/CD
-  Security & Compliance
-  Deployments
-  Monitor
-  Infrastructure
-  Packages & Registries
-  Analytics
-  Wiki
-  Snippets
-  Settings

BALEM Kevin > mycoolproject

 mycoolproject 
Project ID: 726

  Star 0  Fork 0

16 Commits 1 Branch 0 Tags 164 KB Files 164 KB Storage

master mycoolproject / +


History Find file Web IDE  Clone

 upd readme
quaiz20 authored 5 minutes ago

3042ea1e 

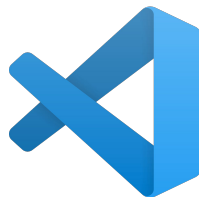
-  README  Other  Add CHANGELOG  Add CONTRIBUTING  Enable Auto DevOps  Add Kubernetes cluster
-  Set up CI/CD

Name	Last commit	Last update
docs	[add] docs folder	1 day ago
.gitignore	[add] license and contribute files	1 day ago
README.txt	upd readme	4 minutes ago
contribute.txt	[add] new contribute file	19 hours ago
licence.txt	[upd] licence	1 day ago

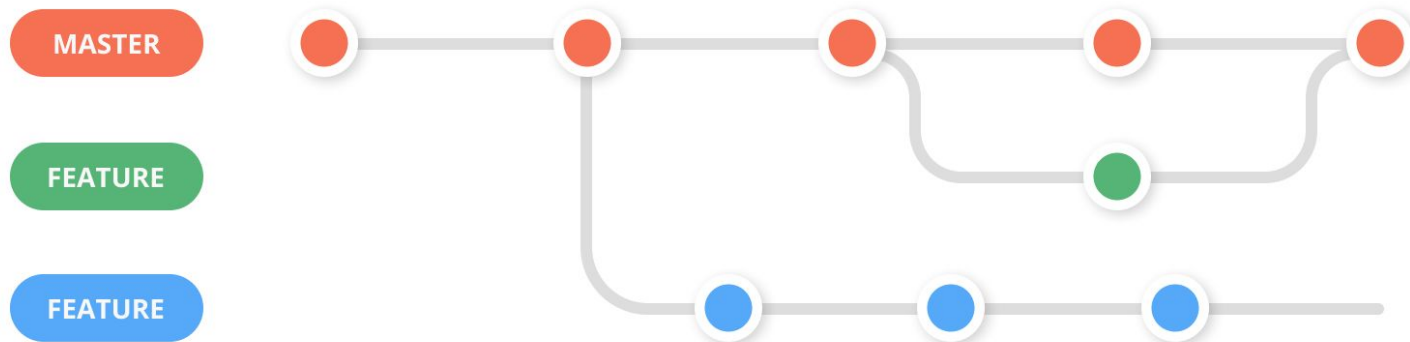
 README.txt

```
## MY COOL PYTHON PROJECT
this is my cool project in python v3.6
please share worldwide !
do it now !
```


Let's look at 1 example



4. Intro to branches



- Working on a feature
- Fixing a bug
- Trying something experimental
- But still want to take advantage of git :
 - Track changes of that branch
 - Work with others

- Create a file with a bug in it
- Let's create a **branch** and go (**checkout** or **switch**) into that new branch
- If we go back to the master branch, what does the file look like ?

```
# You can do that now
```

```
> git checkout -b bugfix/bad_print
```

```
OR
```

```
> git branch bugfix/bad_print
```

```
> git checkout bugfix/bad_print
```

```
OR
```

```
> git switch -c bugfix/bad_print
```

```
... fixing our bug, edit, add, commit
```

```
> git branch
```

```
> git status
```

```
> ??
```

```
> cat mybuggedfile
```

What do I do with my branch ?

- If you no longer need the branch, you can delete it
- If you want to integrate the changes in the **master** branch, you're going to do a **merge**. You have to be on **master** to merge your branch. The other way is also possible but we'll see that later.

```
> git checkout bugfix/bad_print
> git branch -d bugfix/bad_print
??
# git branch -D bugfix/bad_print
```

```
> git checkout master
> git merge bugfix/bad-print
> git log
??
```

```
> git branch
> git branch -d bugfix/bad_print
# fast-forward merge ?
```

What if my master change on the way?

- Let's create a new branch to add a feature
- In the meantime, someone modified our master branch
- This time the **merge** is not automatic (fast-forward) because the parent changed, **git** does a **merge commit**

```
> git checkout -b feature/new_print  
# adding feature
```

```
> git add .  
> git commit -m "[add] new print line"
```

```
> git checkout master  
# modify README for example  
> git add .  
> git commit -m "[upd] readme"
```

```
> git merge feature/new_print  
??
```

```
> git log
```

5. Resolving conflicts

- Let's create a new branch to add a feature
- In the meantime, let's do a commit in the master, on the same file, and same line
- Let's try to **merge**
 - conflicts means you have to do something, you can't leave your repo in the state
- ❖ You can **abort** the merge
- ❖ Or you can **solve** your conflicts

```
> git checkout -b feature/new-print-2
# adding feature
> git add .
> git commit -m "[add] new feature to print"
```

```
> git checkout master
# edit file
> git add .
> git commit -m "[upd] print func"
```

```
> git merge feature/new-print-2
??
```

```
> git status
```

```
> git merge --abort
```


- Let's look at the file that causes the conflict
- Let's see how VS Code can help us resolve the conflict
- Finish the merge process

```
Chemins non fusionnés :  
(utilisez "git add <fichier>..." pour marquer comme  
résolu)  
modifié des deux côtés : myprint.py
```

```
> cat myprint.py  
??
```

```
> code .
```

```
> git add myprint.py  
> git commit -m "[resolve] conflict in print func"
```

THE PROPER WAY

- **Never** solve conflicts on your master branch
- Instead, do a merge on your feature branch to retrieve changes from master and solve conflicts there.

If you want to keep commits in historical order so that your branch starts with the latest master commit, you can do a rebase instead.

(*git rebase master*)

- Then merge the branch into master

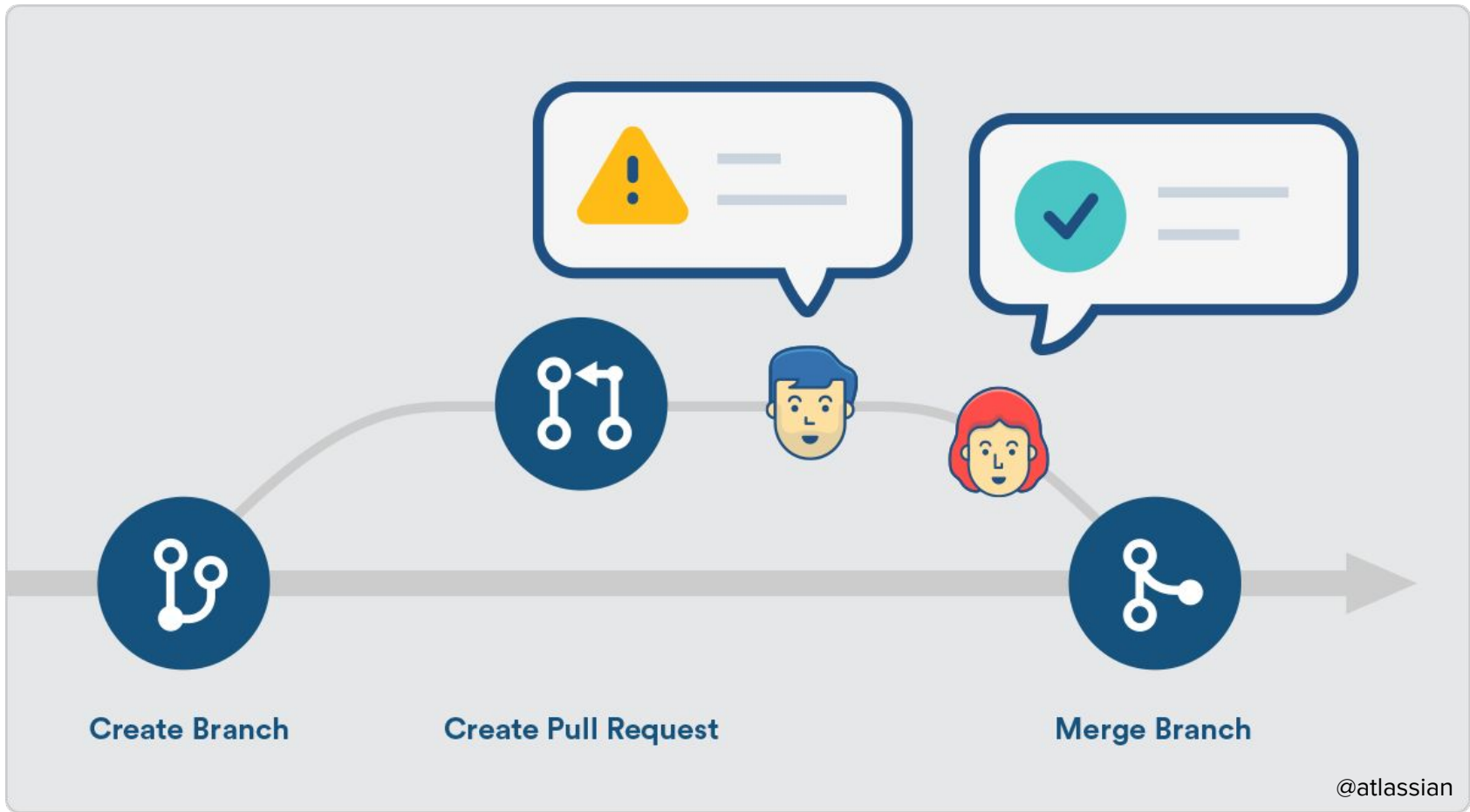
```
> git checkout -b feature/new-print-3
# adding feature
> git add .
> git commit -m "[add] new feature to print"
```

```
> git checkout master
# edit file
> git add .
> git commit -m "[upd] print func"
```

```
> git checkout feature/new-print-3
> git merge master
# solve conflicts & commit
```

```
> git checkout master
> git merge feature/new-print-3
```

6. Pull/merge requests





M mycoolproject

Project information

Repository

Issues 0

Merge requests 0

CI/CD

Security & Compliance

Deployments

Monitor

Infrastructure

Packages & Registries

Analytics

Wiki

Snippets

Settings

BALEM Kevin > mycoolproject



mycoolproject

Project ID: 726



Star

0



Fork

0

16 Commits 1 Branch 0 Tags 164 KB Files 164 KB Storage

master

mycoolproject /



History

Find file

Web IDE



Clone



upd readme

quai20 authored 5 minutes ago

3042ea1e



README

Other

Add CHANGELOG

Add CONTRIBUTING

Enable Auto DevOps

Add Kubernetes cluster

Set up CI/CD

Name	Last commit	Last update
docs	[add] docs folder	1 day ago
.gitignore	[add] license and contribute files	1 day ago
README.txt	upd readme	4 minutes ago
contribute.txt	[add] new contribute file	19 hours ago
licence.txt	[upd] licence	1 day ago

README.txt

```
## MY COOL PYTHON PROJECT
this is my cool project in python v3.6
please share worldwide !
do it now !
```

7. Mini-Projets

Project simple workflow

1. **Start a project** on gitlab with some developers : readme, add users to the project, clone to local repo, first lines of code, push
2. **Add issues** to request some features, and **assign it** to someone
3. Create **Pull/Merge requests** (you can do it from the issue, or local branch then push & PR, ...) → Discussion/Review/Conflicts → Merge from the PR
4. **Test** the features and go back to (3) if needed.
5. Release the project

Let's take a look at one example : <https://github.com/euroargodev/argopy>

Let's build 3 simple python projects following the previous workflow

(last year's projects : dice roll simulator, currency converter, password generator)

Team	Project	Features

- Create the project (add me to the **developers** : @k17balem)
- Post one issue to request a new feature and assign it to a developer
- Code one merge request, taking care of one issue, and asking for the review of another developer, then merge the changes into master
- Review the merge request of somebody else
- The readme should follow the template

README.md template

RAPPORT DE TD GITLAB

- Noms (login gitlab-imt) :
- Date :
- Nom du projet :
- URL du projet :

1. Objectifs et description du projet

Décrire les objectifs du projet (en termes d'apprentissage et de fonctionnalités), et décrire simplement les fonctionnalités.

2. Description du workflow

Décrire les différentes étapes de construction de votre projet en faisant apparaître les termes vus en cours (add, clone, push, pull, merge, merge request, issue, gitlab, repo, remote, ...)