Bases Github

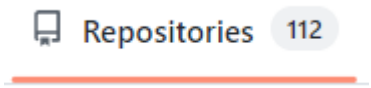
Github est une plateforme Web collaborative reposant sur Git, logiciel de gestion de versions.

Créer un repository

Un repository est un projet hébergé sur la plateforme, permettant de travailler à plusieurs.

La solution la plus simple, surtout pour les débutants, et de créer le repository sur github :

1. Créer au besoin un compte guthub si vous n'en avez pas ;
2. Allez dans **Repositories**, puis cliquez sur **New** :



Cocher la case Add a README file, pour que le système de fichier du repository existe (ce qui va permettre de le cloner ensuite).

Owner * Repository name *

jcheron / MyFirstRepository

Great repository names are short and memorable. Need inspiration? How about bug-free-potato?

Description (optional)

Mon premier repository

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. Learn more.

Add .gitignore
Choose which files not to track from a list of templates. Learn more.

.gitignore template: None

Choose a license
A license tells others what they can and can't do with your code. Learn more.

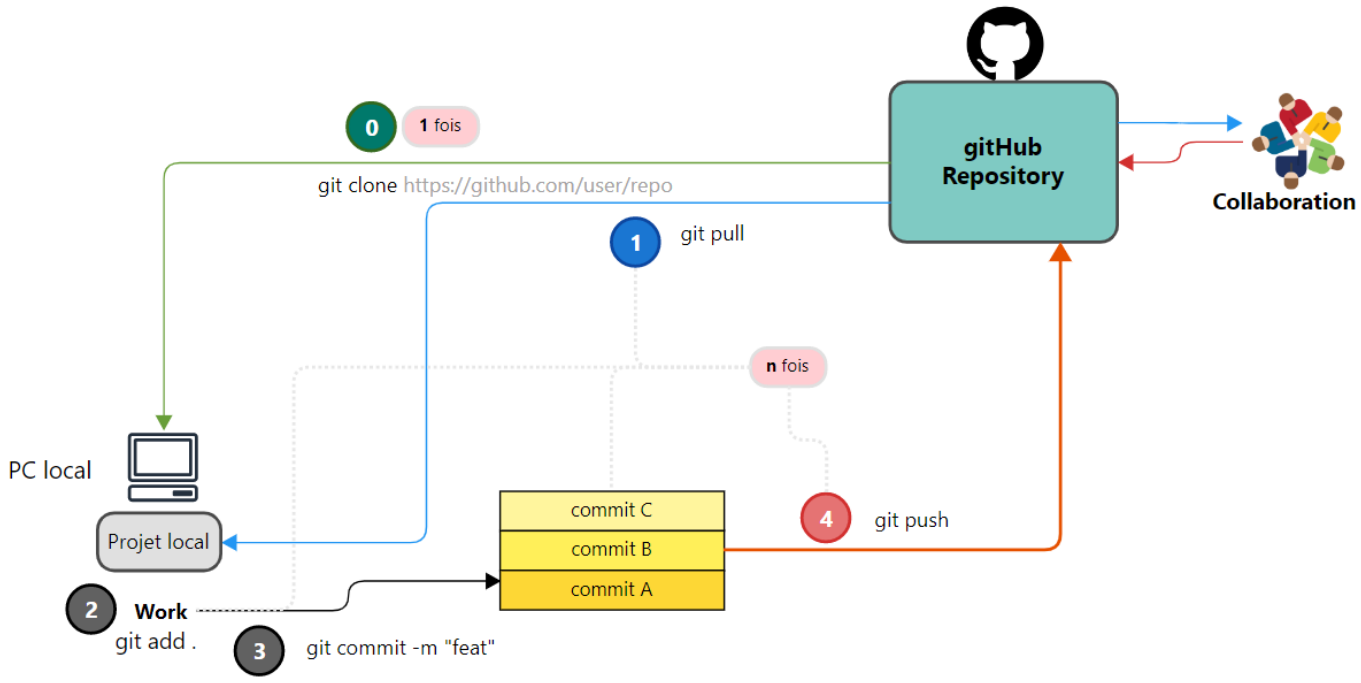
License: None

This will set main as the default branch. Change the default name in your settings.

You are creating a public repository in your personal account.

Create repository

Principe



?900

Pré-requis

Paramètres Git locaux

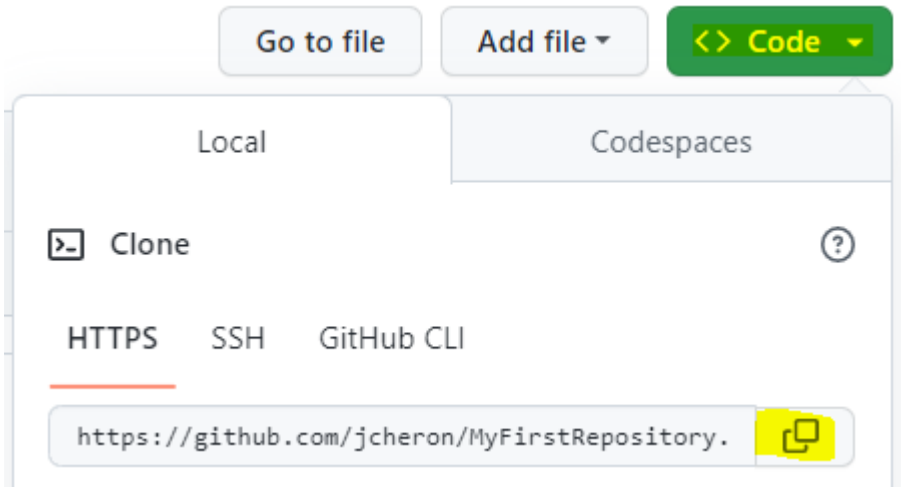
Git ayant besoin de vous identifier, ajouter vos identifiants à la configuration locale de git

Travailler en local

0 - Clone

Pour travailler en local sur le projet, il faut déjà en créer une copie en local : un `clone` :

Copier l'adresse du repository sur Github, avec le bouton **Code** :




```
git clone https://github.com/username/repositoryName.git
cd repositoryName
```

Ignorer des modifications

Créer le fichier **.gitignore** (il n'y a pas de commande git pour ça)
Il va permettre d'ajouter les fichiers ou les dossiers à ne pas “tracker”, en respectant le principe :

- une ligne par exclusion :

```
/conf
main.conf
/lib
/cache
```

1 - Commit

- Vous avez travaillé et modifié des fichiers (ajout/modification/suppression)

- Vous pouvez créer un commit, qui représente votre progression (1 commit = 1 unité de code)

```
git status
```

```
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   pom.xml
        modified:   src/main/kotlin/edu/spring/btp/BtpApplication.kt

no changes added to commit (use "git add" and/or "git commit -a")
```

Pour ajouter un fichier à tracker (sera ajouté aux prochains commits) :

```
git add filename
```

Pour ajouter tous les fichiers à tracker :

```
git add .
```

Refaire un **git status** pour vérifier :

```
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   pom.xml
        modified:   src/main/kotlin/edu/spring/btp/BtpApplication.kt
```

Commiter les modifications :

```
git commit - m "feat: ajout de [NomFonctionnalite]"
```