

- Fiche de cours : GIT (outil de versionning)
  - En pratique
    - Création d'un dépôt git
    - Ajout des fichiers
    - Faire un commit
    - Ajouter un dépôt distant (remote)
    - Synchronisation des dépôt
  - ATTENTION :

# Fiche de cours : GIT (outil de versionning)

---

Permet de faciliter le travail collaboratif entre plusieurs développeur sur un même projet. L'intérêt est d'avoir un historique des modifications (appelée version) que l'on peut rétablir à tout moment.

Git permet de suivre les évolutions d'un projet en utilisant le principe de "branche".

Chaque branche correspond à une version de votre projet qui évolue de manière INDEPENDANTE ! Cela permet de pouvoir expérimenter des solutions sans impacter le code existant.

Pour commencer, nous travaillerons avec une seule branche afin de mieux comprendre ce que ce principe implique.

**Good to know** : La branche "principale" d'un projet et souvent appelée "master" ou "main". Cette branche doit être la plus saine possible !

## En pratique

---

### Création d'un dépôt git

```
cd project
```

```
git init
```

# Ajout des fichiers

## 2 solutions

Ajouter les fichiers individuellement :

```
git add nomfichier
```

Ajouter tous les fichiers du projet

```
git add .
```

# Faire un commit

Un commit est comme une sorte de point d'ancrage qui permet au développeur de sauvegarder une version du code à un instant T avec une description de son travail.

Un commit est identifié de manière UNIQUE via un hash qui lui est attribué.

```
git commit -a -m "Message du commit"
```

- -a pour add (on ajoute au suivi les nouveaux fichiers créés s'ils y en a)
- -m pour message (on décrit les modifications apportées au code)

Un commit doit toujours être le plus EXPLICITE possible tout en étant CONCIS dans sa description

**ATTENTION** : vous devez au préalable configurer votre git local afin qu'il puisse vous reconnaître avant de pouvoir réaliser un commit. Pour cela, tapez les commandes suivantes (une seule fois) :

```
git config --global user.name "username"
```

```
git config --global user.email "votreemail@email.fr"
```

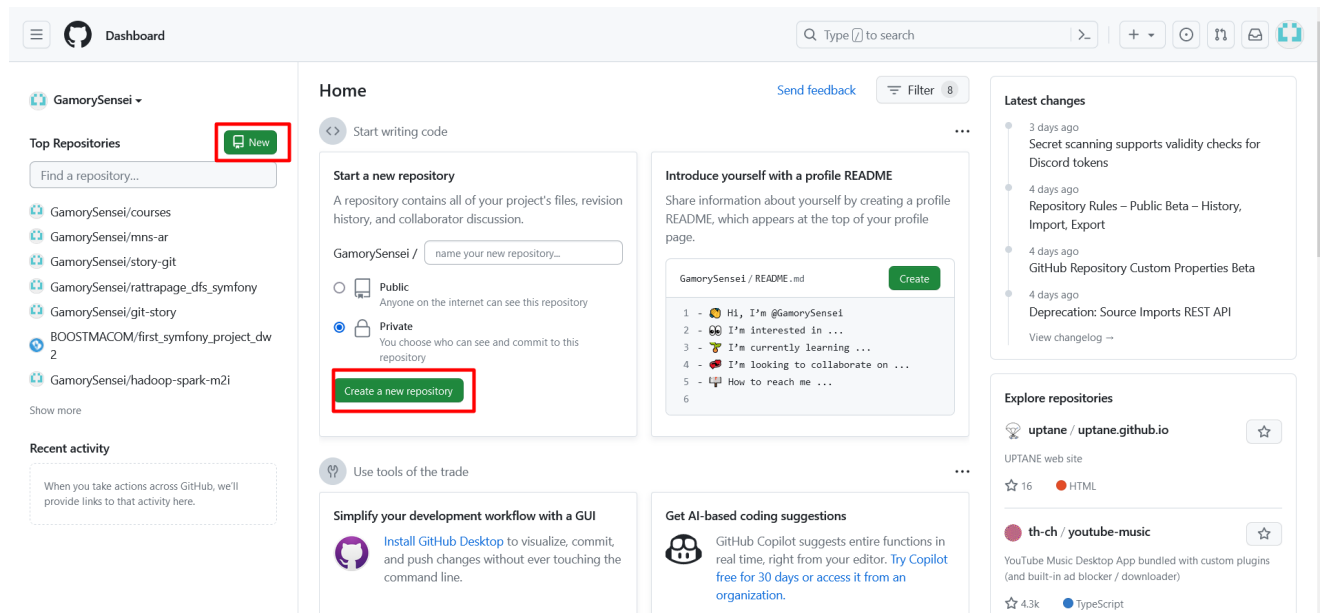
## Ajouter un dépôt distant (remote)

Lorsque l'on crée un dépôt LOCAL, on a souvent besoin de le synchroniser avec un dépôt DISTANT dans le but de pouvoir partager ensuite notre code avec d'autres dévs. Pour cela nous avons besoin d'avoir un compte chez un hébergeur de dépôt et 3 choix s'offrent à nous :

1. GitHub
2. GitLab
3. BitBucket

Nous allons partir sur GitHub car c'est le plus utilisé.

1. Créer un compte sur GitHub : <https://github.com/signup>
2. Créer un nouveau dépôt :



3. Saisir le nom du dépôt (nom du projet en général) et la description

New repository

Q Type to search

+ ↕ ⌚ ⚙️ 📁 🔄

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

*Required fields are marked with an asterisk (\*).*

Owner \*

Repository name \*

GamorySensei

/

Great repository names are short and memorable. Need inspiration? How about [laughing-octo-tribble](#) ?

Description (optional)

☒ **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:**  
☐ **Add a README file**  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

**Add .gitignore**  

.gitignore template: None

  
Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

**Choose a license**  

License: None

  
A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

You are creating a public repository in your personal account.

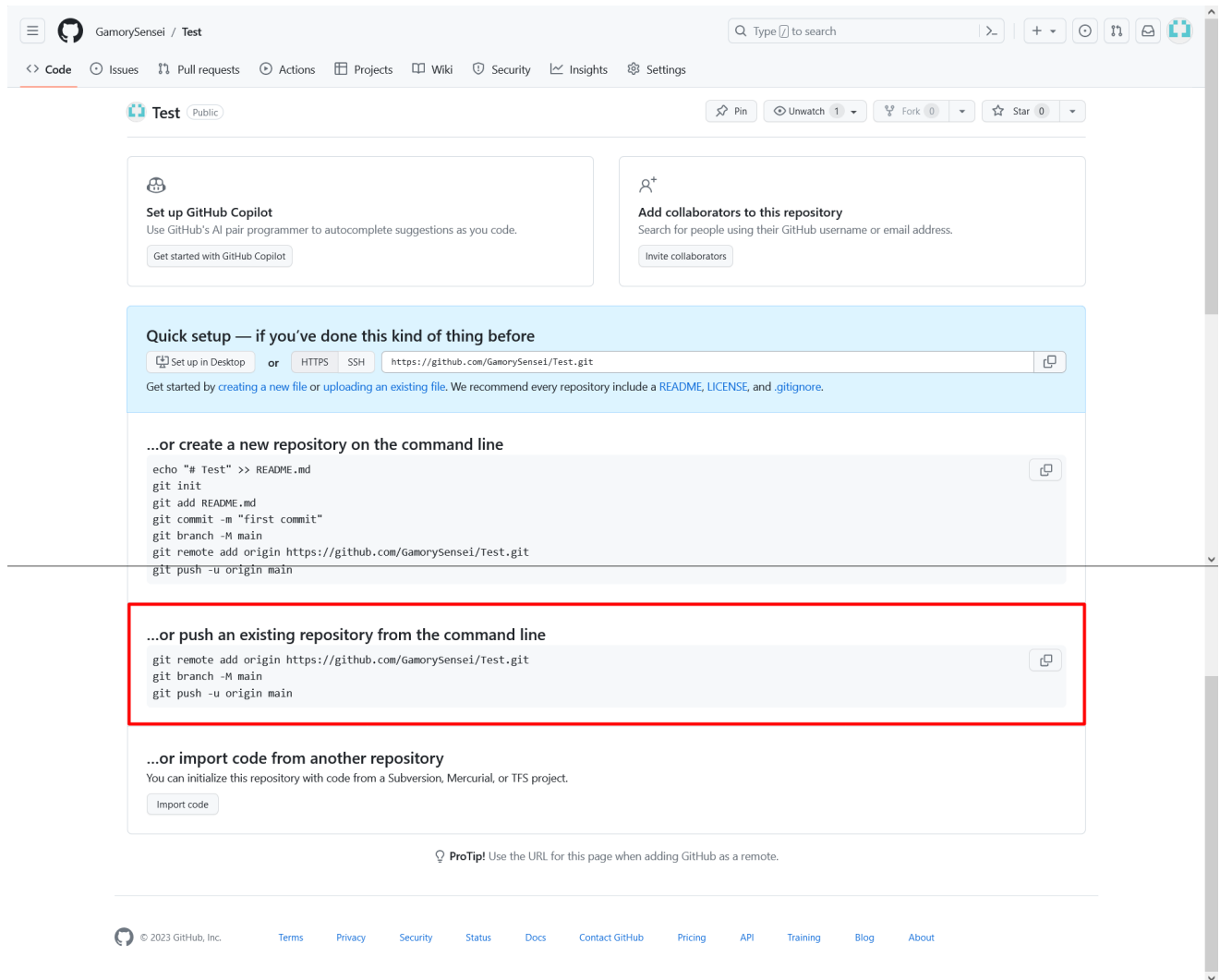
Create repository

© 2023 GitHub, Inc.

[Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

Laissez le en PUBLIC pour le moment puis "Create repository"

4. Suivez les commande en renommant la branche "main" en "master" si vous avez commencé par créer votre dépôt local :



# Synchronisation des dépôt

Récupération du code du dépôt distant :

```
git pull
```

Envoi du code local vers le déppôt distant :

```
git push
```

## ATTENTION :

Vous devez toujours effectuer un commit avant de pouvoir "push" ! Il est également recommandé de toujours "pull" avant de "push" afin d'éviter de "casser" le dépôt distant si vous êtes plusieurs à travailler sur un même branche.