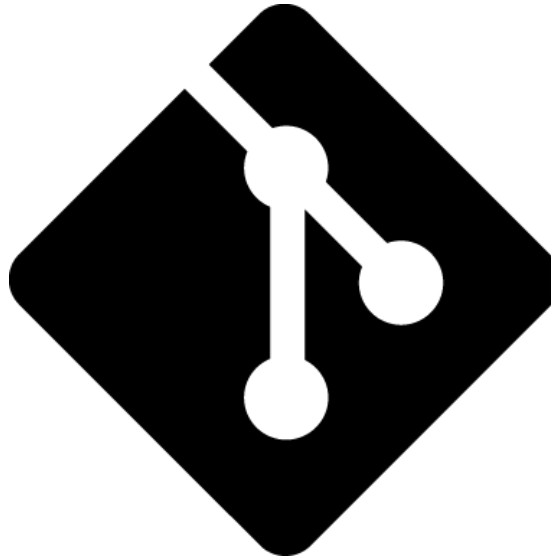


Le versioning

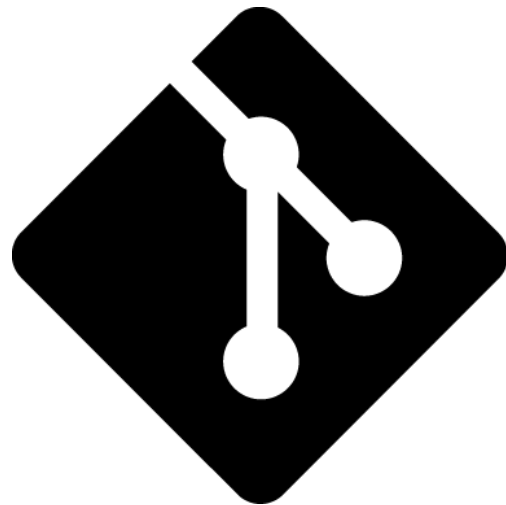


Prérequis avant d'attaquer ce cours :

- ✓ Notions de base Linux.

Objectifs:

- ✓ Comprendre le fonctionnement et la syntaxe de base du versionning.



Introduction

Qu'est-ce que la gestion de versions ?

Un système de gestion de versions (**VCS**, **V**ersion **C**ontrol **S**ystem) est un programme ou un ensemble de programmes qui effectue le **suivi** des **changements** apportés à **une collection de fichiers**.

Les objectifs d'un système de gestion de versions est :

- ✓ de rappeler facilement les **versions antérieures** de fichiers ou d'un projet entier,
- ✓ permettre à **plusieurs membres** (**Collaboratif**) d'une équipe de travailler simultanément sur un même projet et même sur les mêmes fichiers, sans impacter le travail des uns et des autres.

Avec un système VCS, vous pouvez :

- ✓ Consulter **tous les changements** apportés à votre projet, **à quel moment** ils ont été apportés et qui les a effectués.
- ✓ Inclure un **message** pour chaque changement pour expliquer le raisonnement qui est derrière.
- ✓ **Récupérer** les **versions antérieures** du projet entier ou de certains fichiers.
- ✓ Créer des **branches**, où les changements peuvent être apportés à des fins d'expérimentation. Cette fonctionnalité permet à un ou plusieurs utilisateurs de travailler sur plusieurs ensembles de changements à la fois (par exemple, des fonctionnalités ou des correctifs de bogues), sans affecter la branche main. Par la suite, vous pourrez fusionner les changements que vous souhaitez conserver dans la branche main.
- ✓ Attachez une étiquette à une version, par exemple pour marquer une nouvelle publication.

Git est un système VCS **open source** rapide, polyvalent, hautement scalable et gratuit. Son principal auteur est Linus Torvalds, le créateur de Linux.

Gestion de versions distribuée

Git est distribué :

ce qui signifie que l'historique **complet** d'un projet est stocké à la fois :

- ✓ sur le **client**
- ✓ Et sur le **serveur**.

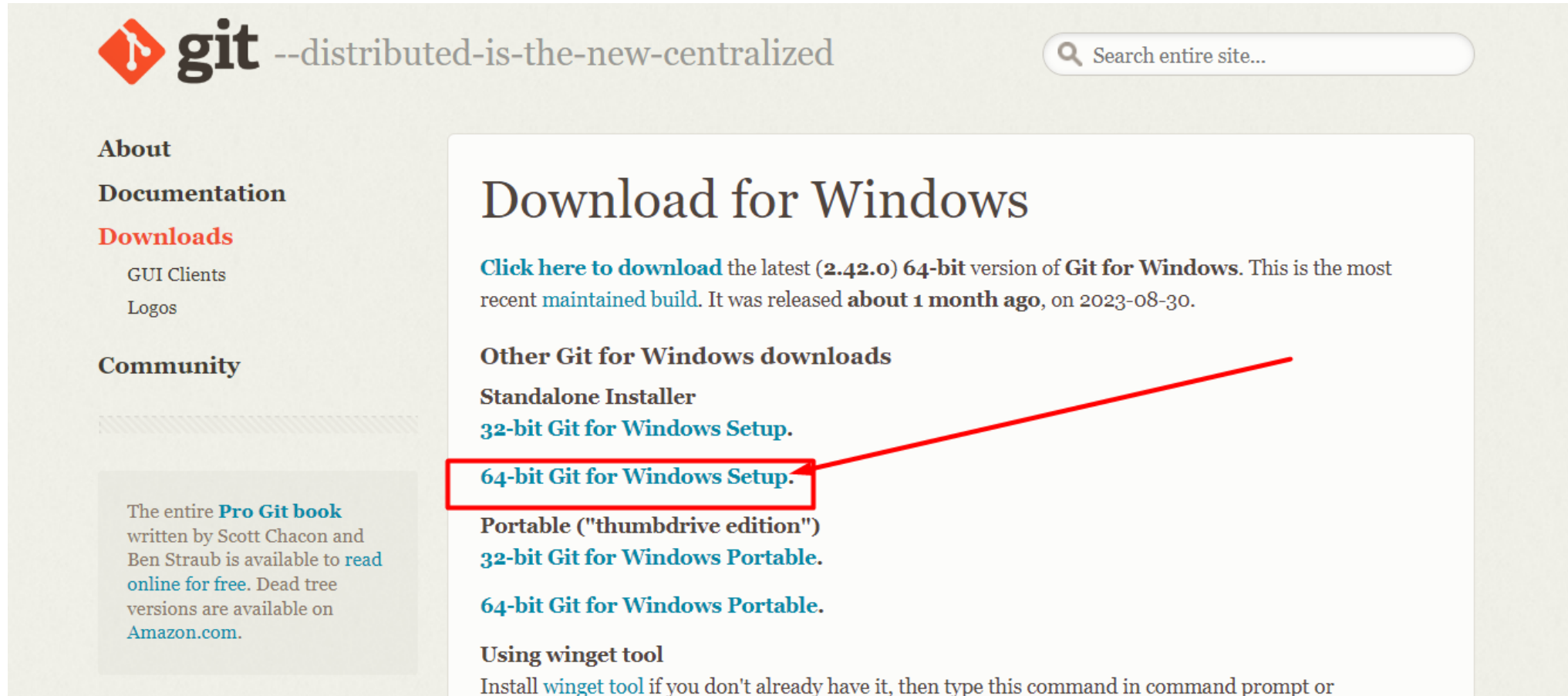
Vous pouvez :


- ✓ modifier les fichiers **sans** connexion réseau,
 - ✓ les archiver **localement**
 - ✓ et les **synchroniser** avec le serveur lorsqu'une connexion est disponible.
- Si un serveur tombe en panne, vous disposez toujours d'une copie locale du projet.

Installation Env

Git Bash :

➤ <https://git-scm.com/download/win>



 **git** --distributed-is-the-new-centralized

Search entire site...

About

Documentation

Downloads

GUI Clients

Logos

Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Download for Windows

[Click here to download](#) the latest (**2.42.0**) **64-bit** version of **Git for Windows**. This is the most recent [maintained build](#). It was released **about 1 month ago**, on 2023-08-30.

Other Git for Windows downloads

Standalone Installer

[32-bit Git for Windows Setup.](#)

[64-bit Git for Windows Setup.](#)

Portable ("thumbdrive edition")

[32-bit Git for Windows Portable.](#)

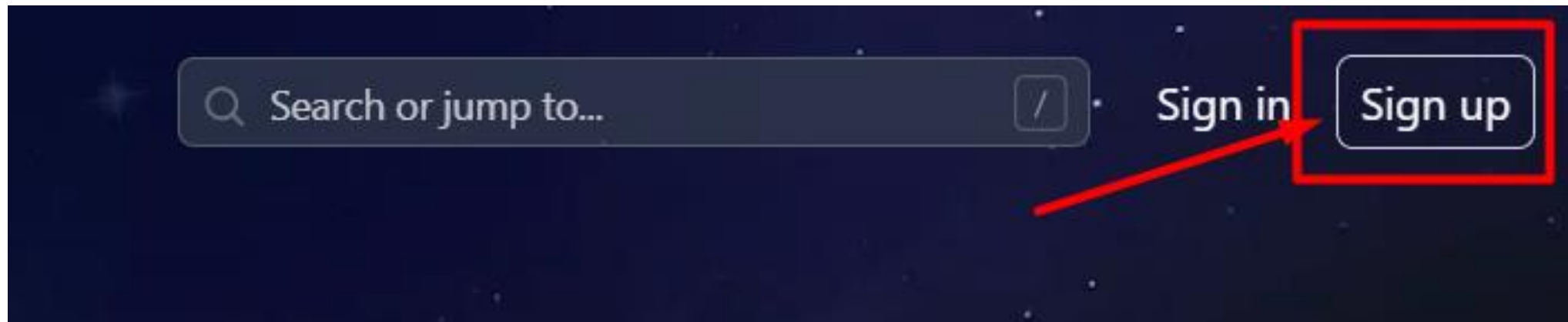
[64-bit Git for Windows Portable.](#)

Using winget tool

Install [winget tool](#) if you don't already have it, then type this command in command prompt or

Github:

➤ <https://github.com/>



Gitlab :

➤ <https://gitlab.com/>



GitLab.com

Prénom

Nom

Nom d'utilisateur

Sécurité

Configuration ultérieures :

✓ Accès ssh : <https://docs.github.com/fr/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account>

✓ Access token :

**Support for password authentication was removed on August 13, 2021.
Please use a personal access token instead**

➤ <https://docs.github.com/fr/enterprise-cloud@latest/authentication/keeping-your-account-and-data-secure/managing-your-personal-access-tokens#creating-a-personal-access-token-classic>

TP : commandes de bases

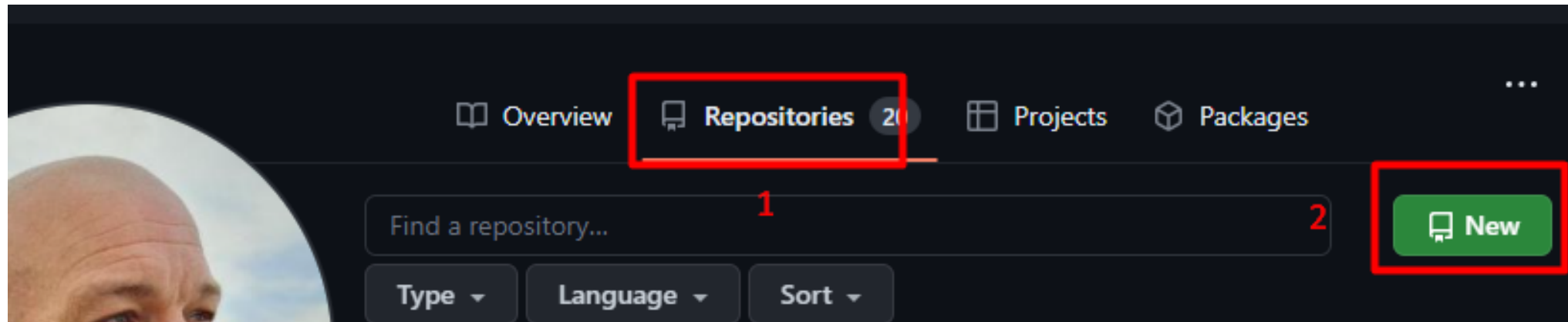
Exemple avec le README :

Comme **tout** projet que vous gèrerez :

1. Préparation du **versioning** (Git pour nous) ;
2. Mise en place du **squelette** (skeleton) ;
3. Branchement **local** => **distant**.

**Etape 1 :
distant (Git).**

Création du repository sur votre compte (ou celui du scrum) :



Création du repository sur votre compte (ou celui du scrum) :

Owner * / Repository name *

chrisalexlegoff / monecommerce 1

✓ monecommerce is available.

Great repository names are short and memorable. Need inspiration? How about **fantastic-system**?

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

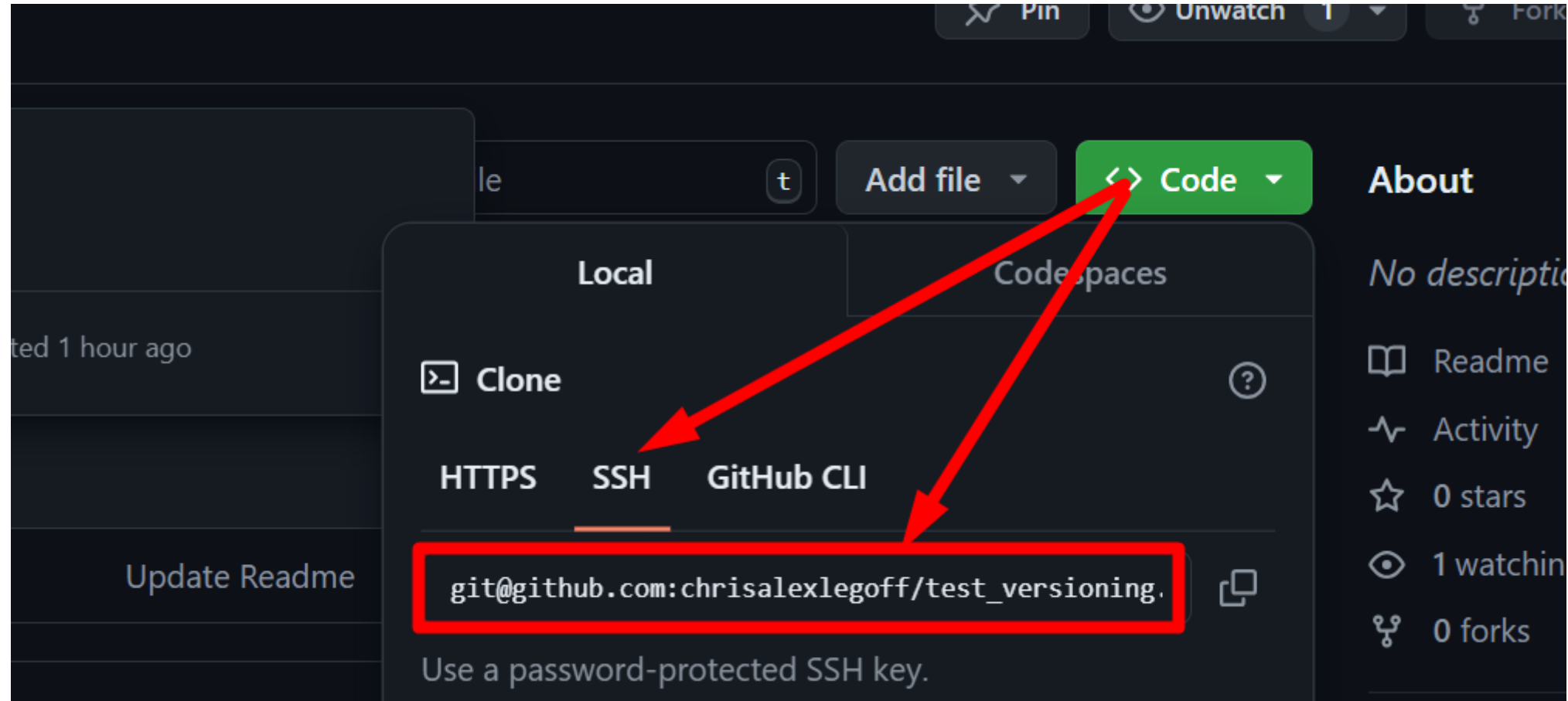
License: None

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

i You are creating a public repository in your personal account.

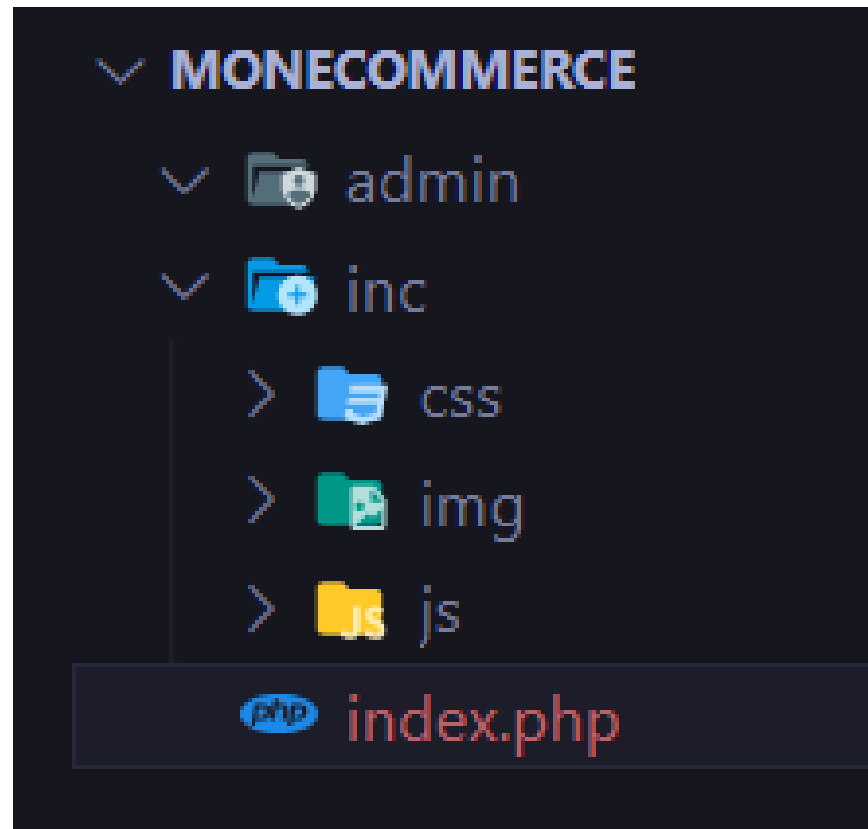
2 **Create repository**

Création du repository sur votre compte (ou celui du scrum) :

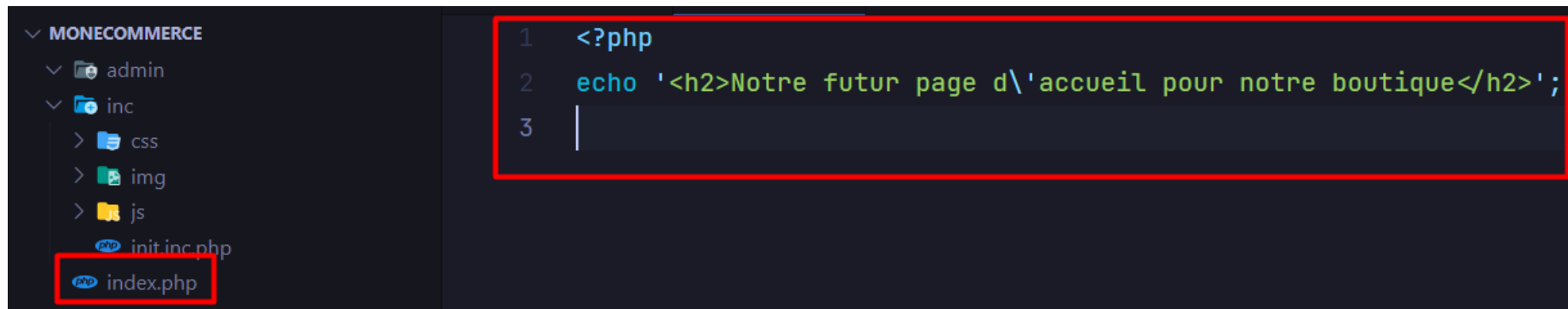


**Etape 2 :
local (Votre PC).**

Création du squelette (**skeleton**) :



Création du squelette (**skeleton**) :



**Etape 3 :
Versioning (Synchronisation).**

Mise en place du versioning (synchronisation) :

- ✓ Initialisation de git dans notre projet (création dossier config « .git » à la racine)

```
• PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce> git init
Initialized empty Git repository in E:/AFCI/cours/PHP/MonEcommerce/MonEcommerce/.git/
○ PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce>
```

- ✓ Ajout du **repo** distant

```
• PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce> git remote add origin git@github.com:chrisalexlegoff/monecommerce.git
○ PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce>
```

HTTPS pour vous!

- ✓ Ajout de **tous** les dossiers fichiers **modifiés, ajoutés** ou **supprimés**

```
• PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce> git add .
○ PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce>
```

Mise en place du versioning (suite):

- ✓ Message (**obligatoire**) pour décrire notre commit

```
• PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce> git add .
• PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce> git commit -m "initial commit"
• [master (root-commit) 0719746] initial commit
  2 files changed, 2 insertions(+)
  create mode 100644 inc/img/test-image.jpg
  create mode 100644 index.php
```

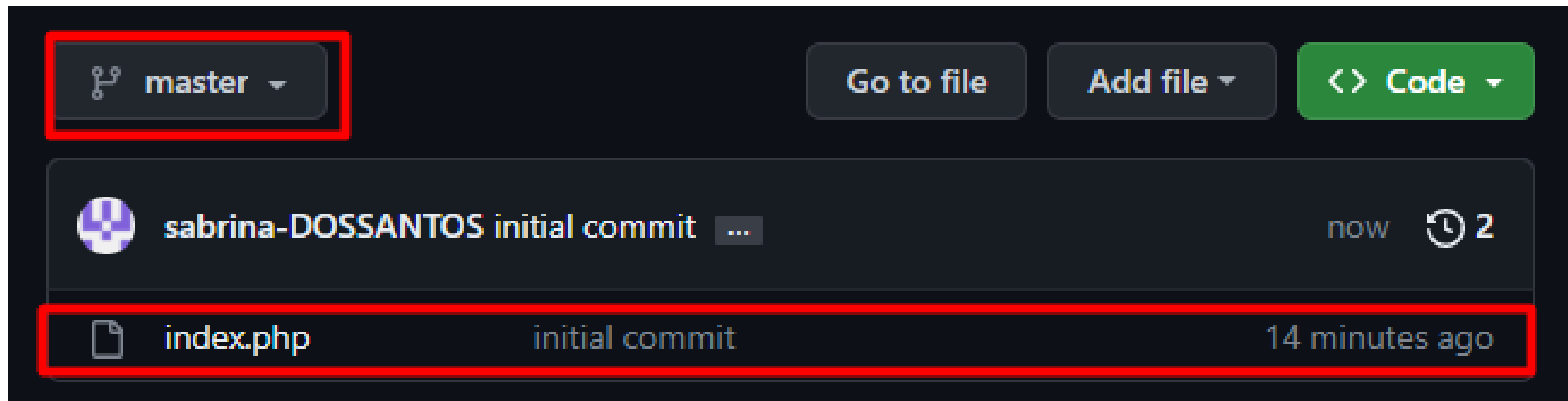
vous le l'avez pas

Mise en place du versioning (suite):

✓ On pousse (**PUSH**)

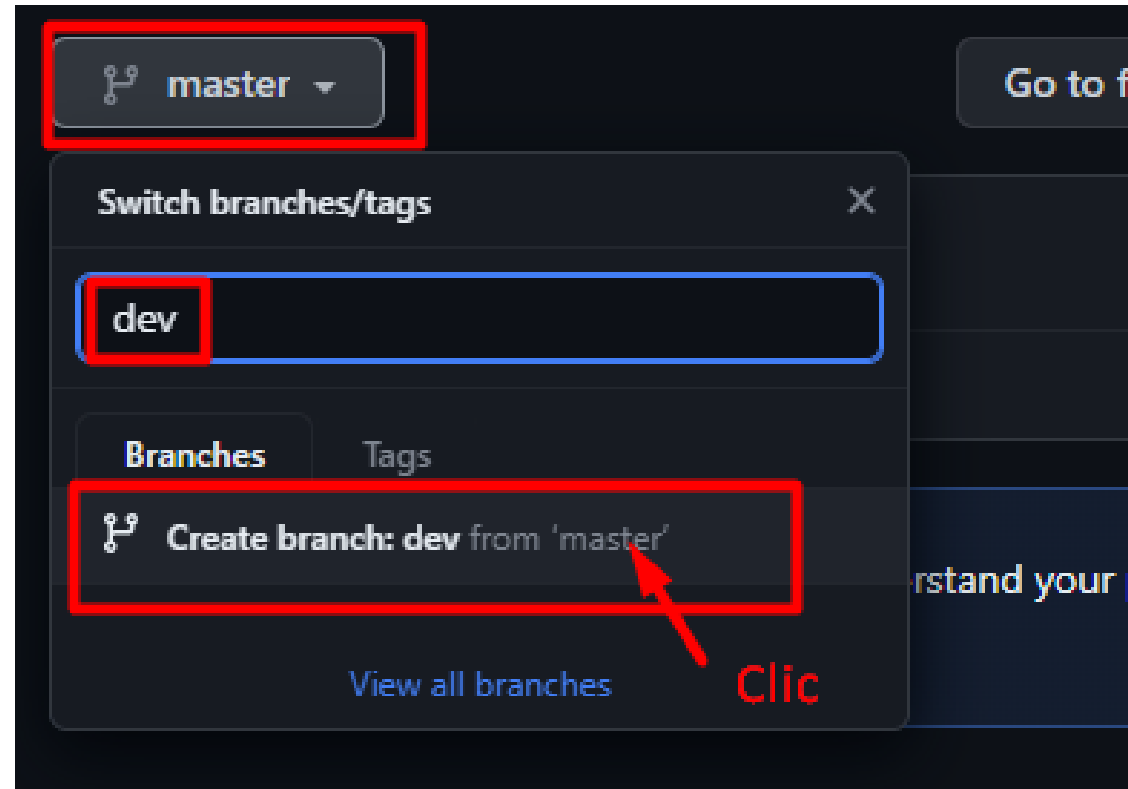
```
● PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce> git push -u -f origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 8.57 KiB | 975.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:chrisalexlegoff/monecommerce.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
PS E:\AFCI\cours\PHP\MonEcommerce\MonEcommerce>
```

Vérification sur notre **repository** (GitHub) :



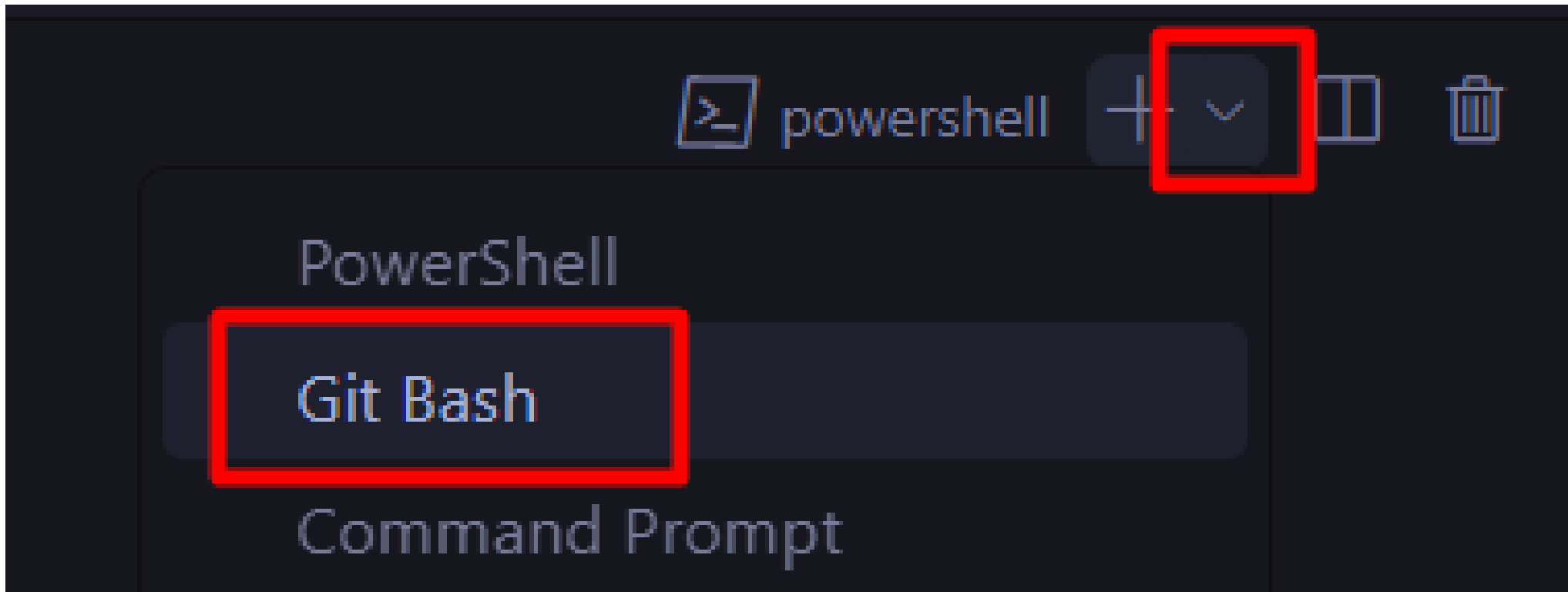
Info : les dossiers vides ne sont pas poussées.

Création branche **dev** (GitHub) :



Info : Conseillé pour garder la master « **propre** ».

Récupération **dev** (local) :

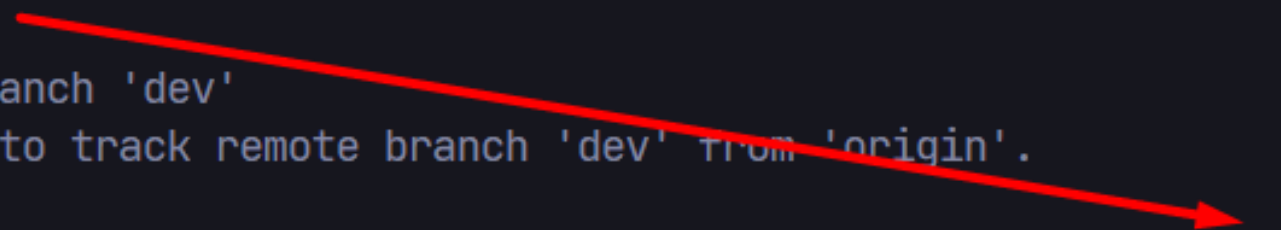


Récupération **dev** (local) :

```
christophe@Win11Chris MINGW64 /e/AFCI/cours/PHP/MonEcommerce/MonEcommerce (master)
$ git fetch
From github.com:chrisalexlegoff/monecommerce
* [new branch]      dev      -> origin/dev
```

```
christophe@Win11Chris MINGW64 /e/AFCI/cours/PHP/MonEcommerce/MonEcommerce (master)
$ git checkout dev
Switched to a new branch 'dev'
Branch 'dev' set up to track remote branch 'dev' from 'origin'.

christophe@Win11Chris MINGW64 /e/AFCI/cours/PHP/MonEcommerce/MonEcommerce (dev)
$
```



La commande « **git branch** » vous permet de savoir où vous êtes (en cmd par exemple)

Etape 4 : Git flow (non obligatoire)

Pour **protéger** votre travail en local et pouvoir gérer les (éventuels) **conflits** avant de **pusher** je vous conseille d'utiliser **Git Flow** :

```
christophe@Win11Chris MINGW64 /e/AFCI/cours/PHP/MonEcommerce/MonEcommerce (dev)
$ git flow init
```

Which branch should be used for bringing forth production releases?

- dev
- master

Branch name for production releases: [master]

Défaut

Which branch should be used for integration of the "next release"?

- dev

Branch name for "next release" development: [] dev

A préciser

How to name your supporting branch prefixes?

Feature branches? [feature/]

Bugfix branches? [bugfix/]

Release branches? [release/]

défaut

Hotfix branches? [hotfix/]

Support branches? [support/]

Version tag prefix? []

Hooks and filters directory? [E:/AFCI/cours/PHP/MonEcommerce/MonEcommerce/.git/hooks]

Pour **protéger** votre travail en local et pouvoir gérer les (éventuels) **conflits** avant de **pusher** je vous conseille d'utiliser **Git Flow** :

```
christophe@Win11Chris MINGW64 /e/AFCI/cours/PHP/MonEcommerce/MonEcommerce (dev)
```

```
$ git flow feature start 01
```

```
Switched to a new branch 'feature/01'
```

```
Summary of actions:
```

- A new branch 'feature/01' was created, based on 'dev'
- You are now on branch 'feature/01'

```
Now, start committing on your feature. When done, use:
```

```
git flow feature finish 01
```

```
christophe@Win11Chris MINGW64 /e/AFCI/cours/PHP/MonEcommerce/MonEcommerce (feature/01)
```

```
$
```

Voilà, tout est **prêt**, nous pouvons commencer à développer notre projet.

Quelques conseils :

- ✓ Créez toujours à **minima** une **deuxième** branche (ex: **dev**) ;
- ✓ Utilisez **git flow** en local (simple conseil) ;
- ✓ **Puller** le matin avant de commencer (**PULL**) ;
- ✓ **Pusher** le soir ou dès votre tâche terminée (**PUSH**) ;

Le **versioning** est très délicat à **comprendre/gérer** au début, rassurez-vous on s'habitue.

- Plus vous mettrez à jour **souvent**, au **mieux** ça se déroulera !

Questions ??

Prochain chapitre:
Outils pour le travail collaboratif.