



# PRÉPARER UN ENVIRONNEMENT DE TEST

Titre Professionnel : Administrateur Système  
DevOps (niveau 6)

22/08/2023





# INSTALLER ET CONFIGURER SON ENVIRONNEMENT DE TRAVAIL EN FONCTION DU PROJET

Titre Professionnel : Concepteur Développeur  
d'Applications (niveau 6)







# INSTALLER ET CONFIGURER SON ENVIRONNEMENT DE TRAVAIL EN FONCTION DU PROJET WEB OU WEB MOBILE

Titre Professionnel : Développeur Web et Web  
Mobile (niveau 5)



# LESLY LODIN



- Développeur et formateur sur les premiers langages de programmation (BASIC, Pascal, Clipper), les bases de données (dBASE, SQL Server, Oracle, MySQL) et les technologies Web depuis près de 30 ans.
- Je suis aujourd'hui dirigeant de la société Baobab Ingénierie, spécialisée dans les formations informatiques et la création d'applications Web et j'exerce toujours en tant que formateur.
- Au travers de ce document, je partage toute mon expérience et mon savoir-faire, acquis depuis de nombreuses années, en particulier auprès des professionnels du secteur des médias (Sony Music, Métropole Télévision, Canal+) et d'autres services de grandes entreprises.

# GESTION DE VERSIONS

Version	Date	Rédacteur	Description
1.0	14/10/2020	Lesly	Description des principales commandes de Git
1.1	03/10/2021	Lesly	Application de la charte Baobab et de la traçabilité
2.0	18/08/2023	Lesly	Accès SSH et usage avec Visual Studio Code





# GÉNÉRALITÉS

# DÉFINITIONS

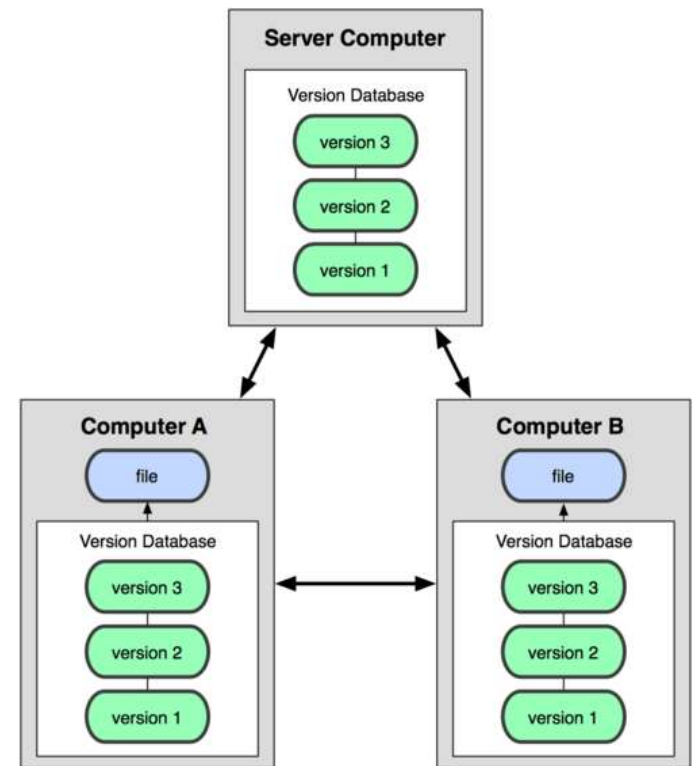
## LOGICIEL DE GESTION DE VERSIONS

- C'est un logiciel qui permet de stocker un ensemble de fichiers en conservant la chronologie de toutes les modifications qui ont été effectuées dessus
- Il permet notamment de retrouver les différentes versions d'un lot de fichiers connexes
- **Git** et Mercurial sont des logiciels et services de gestion de versions décentralisés (ou **DVCS** en anglais, pour Distributed Version Control System) et sont disponibles sur la plupart des systèmes Unix et Windows
- Ils sont notamment utilisés en développement logiciel pour conserver le code source relatif aux différentes versions d'un logiciel

# DÉFINITIONS

## POURQUOI UN SYSTÈME DE GESTION DE VERSION ?

- Revenir aisément à une **version précédente**
- Suivre **l'évolution du projet** au cours du temps
- Permettre le travail en parallèle sur des **parties disjointes du projet** et gérer les modifications concurrentes
- Faciliter la détection et la correction d'erreurs





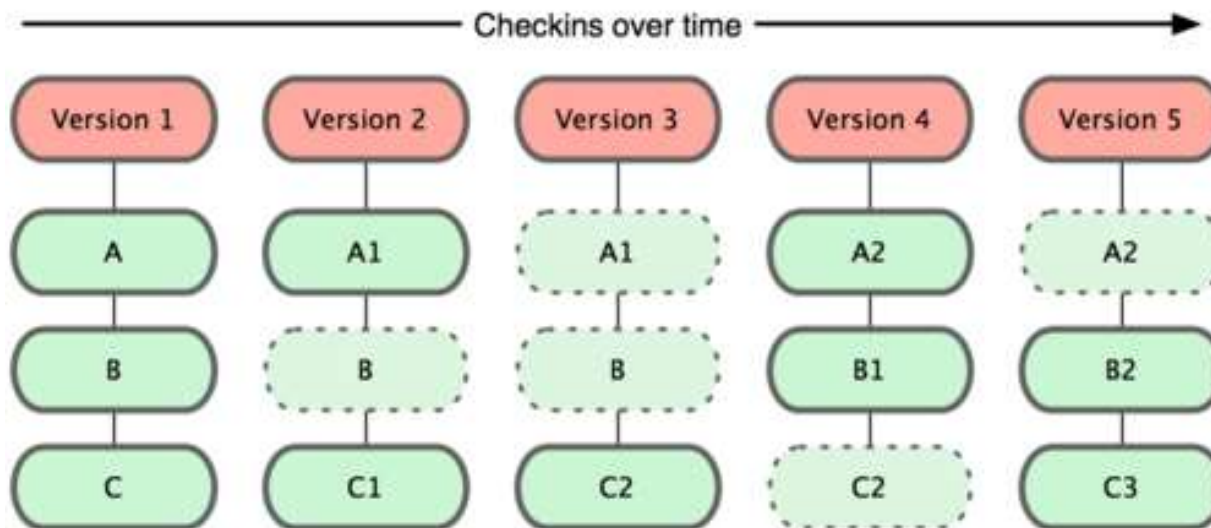
# BREF HISTORIQUE



- De 1991 à 2002, le noyau Linux était développé sans utiliser de système de gestion de version
- A partir de 2002, la communauté a commencé à utiliser **BitKeeper**, un DVCS propriétaire
- En 2005, suite à un contentieux, BitKeeper retire la possibilité d'utiliser gratuitement son produit
- **Linus Torvalds** lance le développement de **Git** et après seulement quelques mois de développement, Git héberge le développement du noyau Linux

# PRINCIPES DE BASE

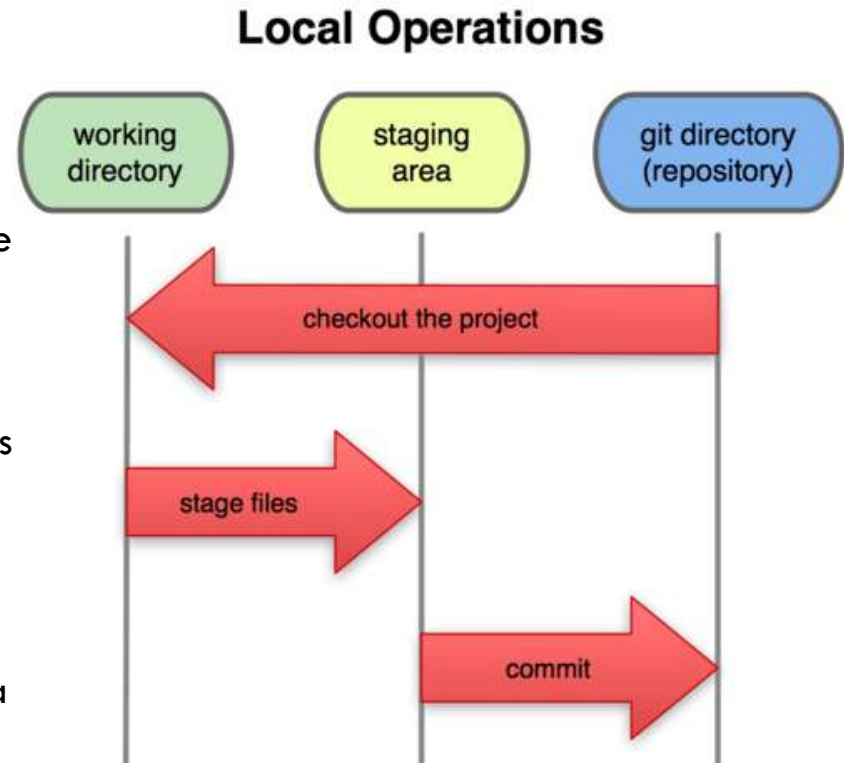
- Un **dépôt Git** est une sorte de système de fichiers (base de données) enregistrant les versions de fichiers d'un projet à des moments précis au cours du temps sous forme **d'instantanés** (ou **snapshots**)



# PRINCIPES DE BASE

## 3 SECTIONS D'UN PROJET GIT

- **Le répertoire Git/dépôt local :**
  - Contient les métadonnées et la base de données des objets du projet
- **Le répertoire de travail :**
  - Extraction unique d'une version du projet depuis la base de données du dépôt
- **La zone de transit/d'index :**
  - Simple fichier contenant des informations à propos de ce qui sera pris en compte lors de la prochaine soumission

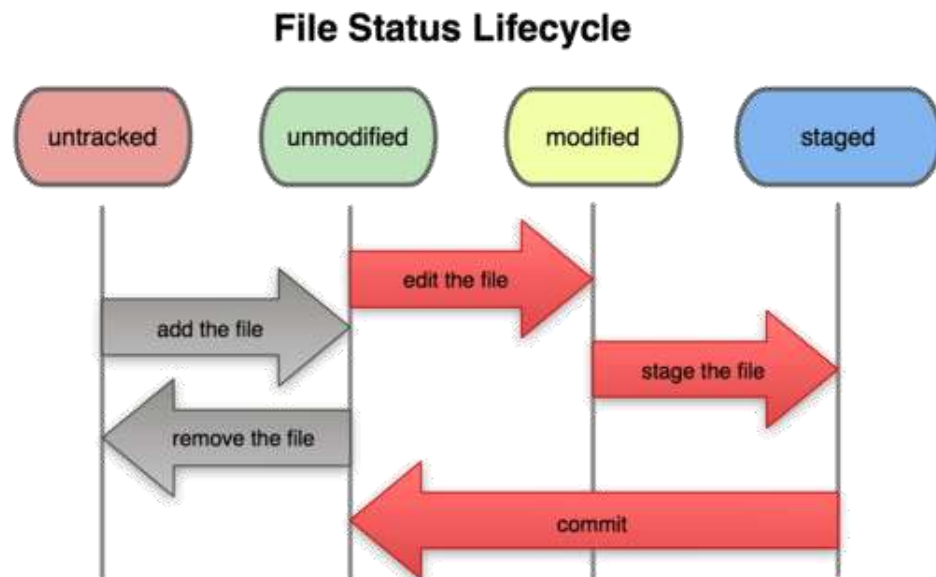




# PRINCIPES DE BASE

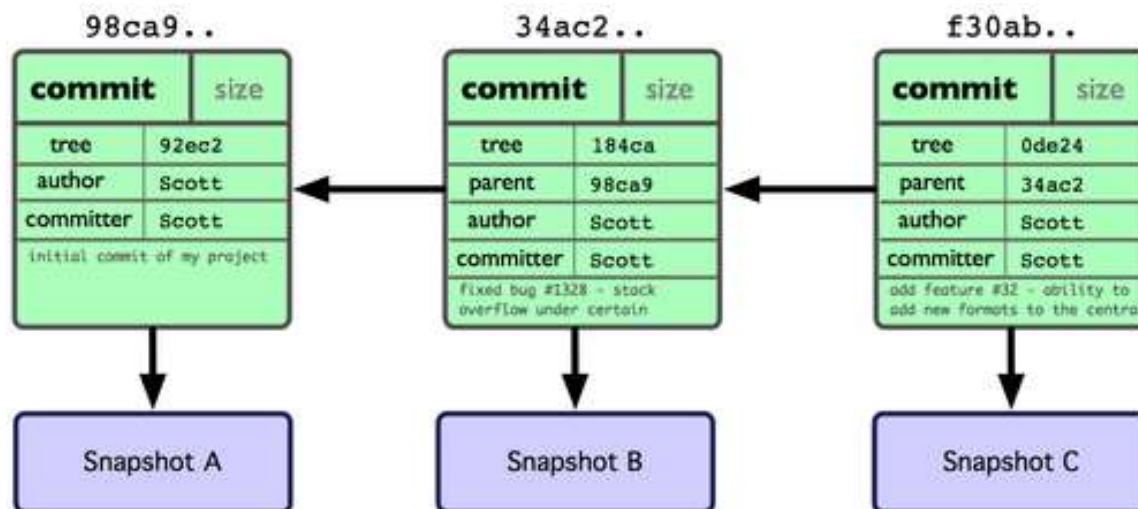
## 4 ETATS D'UN FICHIER DANS GIT

- **Non versionné (untracked) :**
  - fichier n'étant pas ou plus géré par Git
- **Non modifié (unmodified) :**
  - fichier sauvegardé de manière sûre dans sa version courante dans la base de données du dépôt local
- **Modifié (modified) :**
  - fichier ayant subi des modifications depuis la dernière fois qu'il a été soumis
- **Indexé (staged) :**
  - idem, sauf qu'il en sera pris un instantané dans sa version courante lors de la prochaine soumission (commit)



# PRINCIPES DE BASE

- Chaque **soumission** (ou **commit**) donne lieu à la création d'un objet "**commit**" contenant :
  - Un pointeur vers un instantané (ou **snapshot**) du contenu dont les modifications étaient indexées au moment de la soumission
  - Quelques métadonnées (auteur, message)
  - Un pointeur vers l'objet "commit" précédent



# PRINCIPES DE BASE

- Avoir à la fois un **dépôt local** et un **dépôt à distance** est tout simplement le meilleur des deux mondes !
- On peut modifier son code sans être connecté à Internet tout en présentant son projet fini sur une **plateforme distante** afin que tout le monde puisse y accéder.
- Cette configuration facilite aussi le fait d'avoir **plusieurs développeurs** travaillant sur le **même projet**.
- Chacun peut travailler seul sur son propre ordinateur mais **téléverser** (ou **push**) ses modifications vers le dépôt distant quand elles sont prêtes.



# PLATEFORMES



- **GitHub** est un service web d'hébergement et de gestion de développement de logiciels utilisant le logiciel de gestion de versions **Git**
- [GitHub](https://github.com) propose des comptes professionnels payants ainsi que des comptes gratuits pour les projets de logiciels libres.
- Depuis quelques années, GitHub est devenu le **book/portfolio** de nombreux développeurs !
- Outre la plateforme Github, l'entreprise Github développe l'éditeur de texte **Atom** ou encore le framework **Electron**
- Le 4 juin 2018 **Microsoft** annonce l'acquisition de l'entreprise pour la somme de 7,5 milliards de dollars américains

# PLATEFORMES



- [GitLab](#) est la **principale alternative à GitHub** depuis le rachat de GitHub par Microsoft !
- Les anti-Microsoft ont même lancé le hashtag **#MovingToGitLab** !
- GitLab propose une version gratuite hébergée par ses soins ou sur vos propres serveurs.
- Il existe aussi des versions payantes avec plus d'options.

# PLATEFORMES



- [Bitbucket](#) est la version de Atlassian.
- Elle plaira aux habitués de la **gestion de projet sous Atlassian**.
- Bitbucket conviendra aussi bien aux étudiants ou petites équipes qu'aux grands groupes.
- Une version gratuite est disponible.





# INSTALLER GIT

# INSTALLATION DE GIT

- Choisir et télécharger la version de Git qui correspond à son système d'exploitation : MacOS, Windows ou Linux/Unix.

## Downloads



macOS



Windows



Linux/Unix

Older releases are available and the [Git source repository](#) is on GitHub.

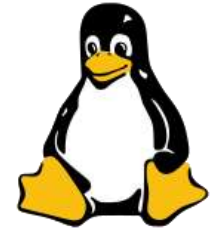
Latest source Release

**2.42.0**

[Release Notes \(2023-08-21\)](#)

Download for Windows

# INSTALLATION DE GIT



## INSTALLATION SUR LINUX

- Pour installer les outils basiques de Git sur Linux via un installateur binaire, il faut généralement le faire au moyen de l'outil de gestion de paquet fourni avec la distribution Linux
- Sur Fedora, on utilise **dnf** :
- **\$ dnf install git-all**
- Sur une distribution basée sur Debian, comme Ubuntu, ce sera **apt-get** :
- **\$ apt-get install git-all**
- Sinon : <https://git-scm.com/download/linux>



# INSTALLATION DE GIT

## INSTALLATION SUR MAC OS

- La méthode la plus facile est probablement d'installer les Xcode Command Line Tools puis d'essayer de lancer git dans le terminal la première fois :
- `$ git --version`
- S'il n'est pas déjà installé, il vous demandera de le faire
- On peut aussi l'installer à partir de l'installateur binaire de Git pour macOS qui est maintenu et disponible au téléchargement sur le site web de Git à <http://git-scm.com/download/mac>

# INSTALLATION DE GIT



## INSTALLATION SUR WINDOWS

- L'application officielle est disponible au téléchargement sur le site web de Git
- C'est un projet nommé **Git for Windows** (appelé aussi msysGit) qui est séparé de Git lui-même
- Rendez-vous sur <http://git-scm.com/download/win> pour télécharger les versions 32-bit ou 64-bit via un installeur ou en version portable
- Il existe aussi des versions avec interface graphique mais nous n'aborderons pas ce sujet ici

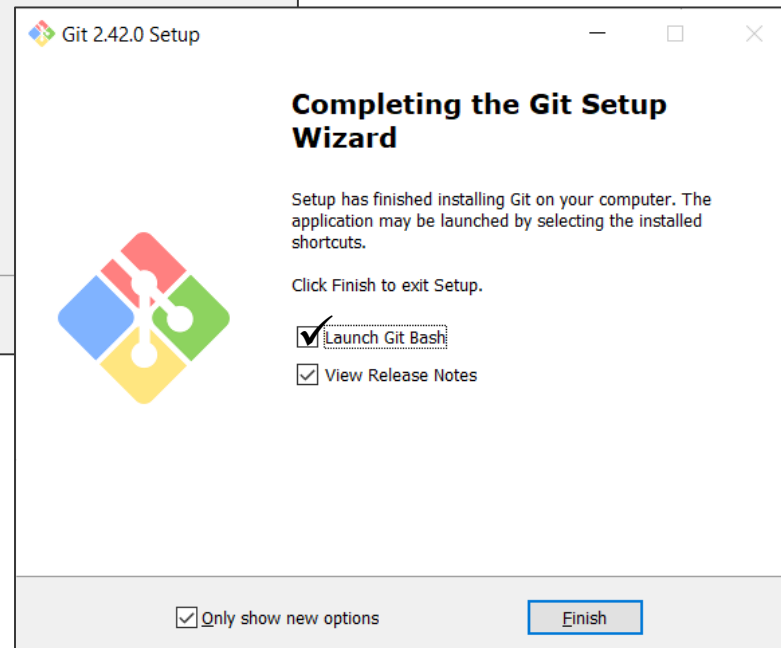
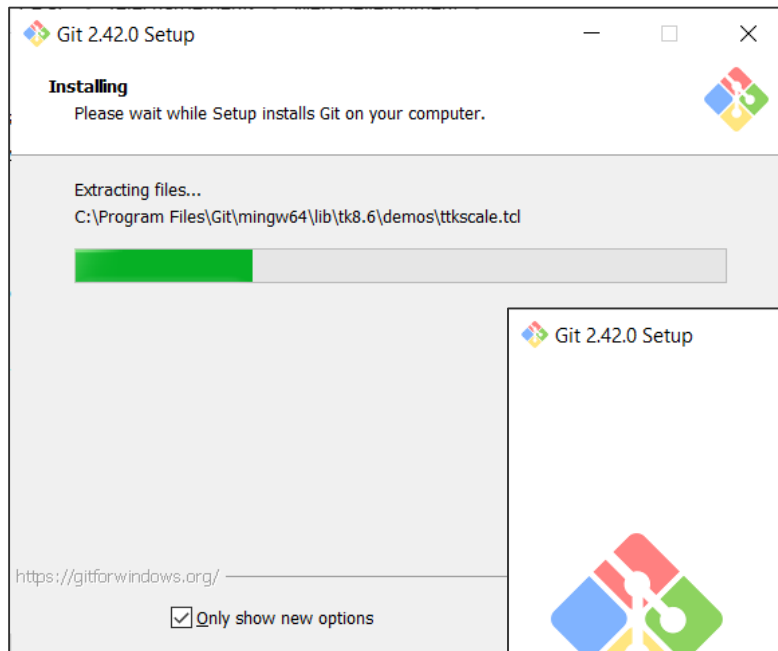


# INSTALLATION DE GIT



- **Exécuter** le fichier qui vient d'être téléchargé.
- Appuyer sur "**Suivant**" à chaque fenêtre puis sur "**Installer**".
- Lors de l'installation, **laisser toutes les options par défaut**, elles conviennent très bien !
- Sous Windows, on pourra cocher ensuite "**Launch Git Bash**".
- Pour les utilisateurs de Mac ou Linux, le terminal suffira amplement.
- **Git Bash** est l'interface permettant d'utiliser Git en ligne de commande.

# INSTALLATION DE GIT



# INSTALLATION DE GIT



```
MINGW64:/  
  
Baobab@BAOBAB-1 MINGW64 /  
$ ls  
LICENSE.txt          etc/                 tmp/  
ReleaseNotes.html   git-bash.exe*       unins001.dat  
bin/                 git-cmd.exe*        unins001.exe*  
cmd/                 mingw64/            unins001.msg  
dev/                 proc/               usr/  
  
Baobab@BAOBAB-1 MINGW64 /  
$ |
```



# CRÉER UN DÉPÔT DISTANT SUR GITHUB

# CRÉER SON COMPTE SUR GITHUB

- A chaque commit réalisé avec **Git**, un instantané est stocké dans un dépôt distant (aussi appelé "**repo**" pour "**repository**")
- Pour déposer son projet sur **Github**, on doit donc disposer d'un dépôt distant pour le faire vivre
- Un dépôt distant est exactement le même qu'un dépôt local : il est juste stocké sur un serveur ou un ordinateur distant pour faciliter la collaboration, les sauvegardes et d'autres trucs géniaux :o)
- Pour démarrer, il faut se rendre sur <https://github.com/> puis cliquer sur le bouton "**Sign up**" situé dans l'en-tête





# CRÉER SON COMPTE SUR GITHUB

- Renseigner son pseudo, son adresse mail et un mot de passe fort, résoudre l'énigme puis cliquer sur **"Create account"**
- Surveiller ensuite sa boîte mail pour valider sa création de compte
- Une fois fini, on se retrouve sur son **"Dashboard"**



Wellcome to GitHub!  
Let's begin the adventure

Enter your email\*  
✓ baobab-tree@gmail.com

Create a password\*  
✓ ●●●●●●●●

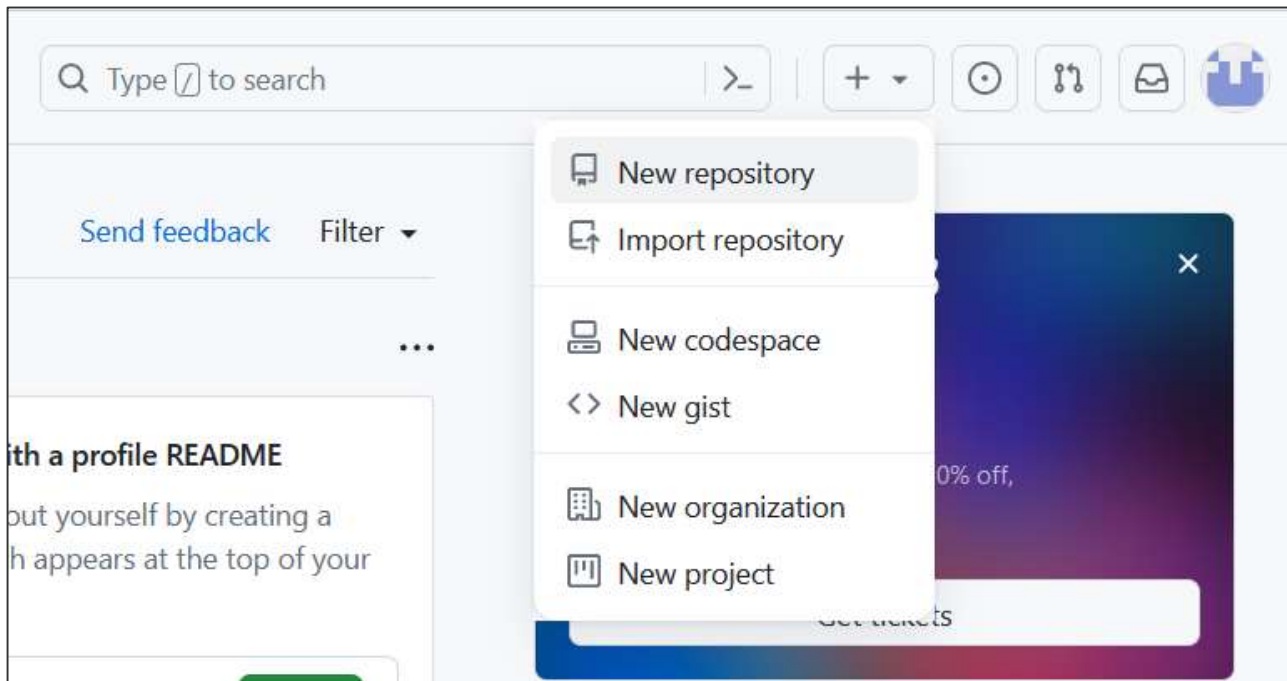
Enter a username\*  
→ baobab-tree

Continue

Create account

# CRÉER UN DÉPÔT SUR GITHUB

- Cliquer ensuite sur "**New Repository**" à la fin de la création du compte ou à partir du menu situé à droite de l'en-tête



# CRÉER UN DÉPÔT SUR GITHUB

- Remplir le formulaire puis une fois terminé cliquer sur **"Create repository"**


Owner \* / Repository name \*


daron-coder / my-first-project ✓

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

Description (optional)

Mon premier projet

☒  **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.](#)

☐ **Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

# FICHER README

- Bien qu'un fichier **README** ne soit pas un élément obligatoire d'un repo Github, c'est une très bonne habitude à prendre que d'en placer un
- Ce fichier est un endroit génial pour décrire son projet ou ajouter un peu de documentation comme expliquer comment installer ou utiliser son projet, quelles technologies il utilise etc.
- Le fichier **README.md** porte l'extension .md comme **markdown** dont voici un petit memento : <https://github.com/Simplonline-foad/utiliser-markdown/blob/master/README.md>



ACCÉDER EN SSH AU  
DÉPÔT DISTANT

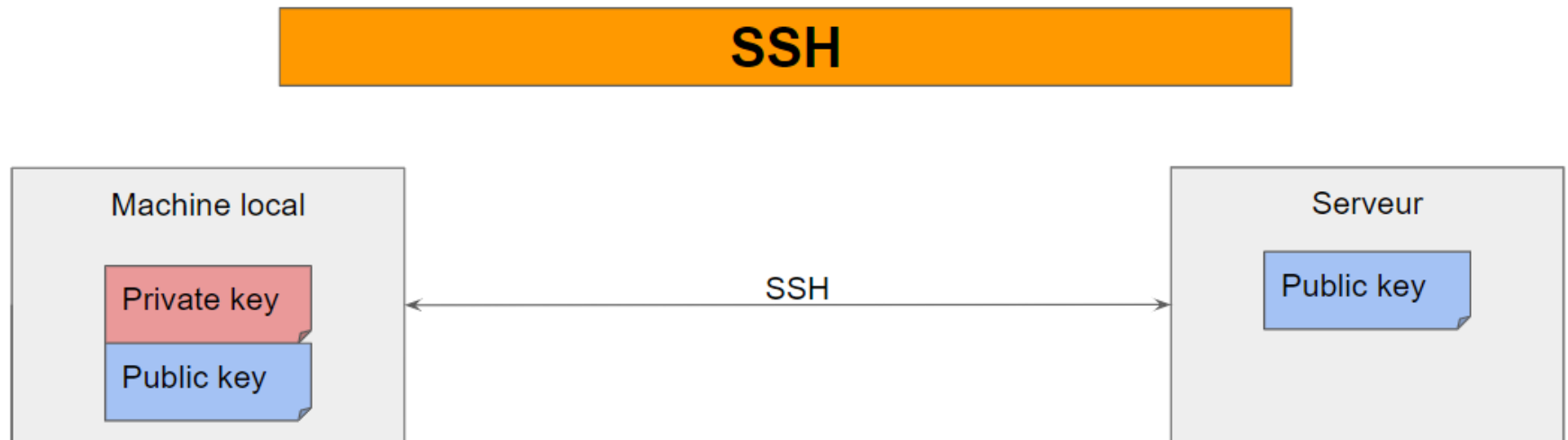


# S'AUTHTENTIFIER AVEC SSH

- Github utilise **SSH** (pour **Secure Shell**) comme protocole de transmission et permet l'accès en lecture et en écriture aux référentiels distants.
- Une connexion chiffrée est utilisée pour permettre l'**authentification** de la machine et pour garantir l'intégrité des données transmises.
- Sans elle, des tiers pourraient apporter des modifications aux référentiels.
- Le chiffrement est **asymétrique** : une paire de clés privée et publique (pour **Public-Private Keypair**) est créée dans un premier temps.
- La clé privée reste sur l'ordinateur personnel, tandis que la clé publique est partagée avec des tiers comme GitHub.

# S'AUTHTENTIFIER AVEC SSH

- Le protocole de connexion SSH impose un échange de **clés de chiffrement** en début de connexion.
- Par la suite, tous les segments TCP sont **authentifiés** et **chiffrés**.



# GÉNÉRER UNE PAIRE DE CLÉS SSH

- Ouvrir si besoin une session **git-bash** (ou un **terminal**).
- Créer la paire de clés avec la commande suivante :  
`ssh-keygen -t rsa -C "your_email@example.com"`
- Ou mieux encore avec le chiffrement RSA :  
`ssh-keygen -t rsa -b 4096 -C "your_email@example.com"`
- À ce stade, deux clés vont être générées dans le dossier caché "home" :
  - **Linux** : /home/[user]/.ssh
  - **Windows**: C:\Utilisateurs\[user]\.ssh
- On y retrouve alors les fichiers suivants :
  - **id\_rsa** : clé privé à conserver sur son PC et à ne surtout pas partager;
  - **id\_rsa.pub** : clé publique à envoyer sur les serveurs distants (GitHub par exemple).

# GÉNÉRER UNE PAIRE DE CLÉS SSH

```
MINGW64:/

Baobab@BAOBAB-1 MINGW64 /
$ ssh-keygen -t rsa -b 4096 -C "baobab@gmail.com"
Generating public/private rsa key pair.

Enter file in which to save the key (/c/Users/Baobab/.ssh/id_rsa): Enter p
assphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Baobab/.ssh/id_rsa
Your public key has been saved in /c/Users/Baobab/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:L5N7YYeuNni5XWcjnmU13kF7UatLwD/F+h0kMEJPBy8 baobab@gmail.com
The key's randomart image is:
+---[RSA 4096]---+
|      .o =..  .|
|      = = . o |
|      E o B   |
|      + B o   |
|      S . *.+o |
|      o+ o.++= |
|      .++..oo.B.o|
|      . ==oo B .|
|      o+=. o   |
+-----[SHA256]-----+

Baobab@BAOBAB-1 MINGW64 /
$ cd c/Users/Baobab/.ssh

Baobab@BAOBAB-1 MINGW64 ~/.ssh
$ ls
id_rsa id_rsa.pub known_hosts
```

# COPIER LA CLÉ PUBLIQUE

- **MacOS :**
  - `pbcopy < ~/.ssh/id_rsa.pub`
- **GNU/Linux (nécessite le package xclip) :**
  - `xclip -sel clip < ~/.ssh/id_rsa.pub`
- **Ligne de commande Windows :**
  - `type %userprofile%\ssh\id_rsa.pub | clip`
- **Git Bash sous Windows/Windows PowerShell :**
  - `cat ~/.ssh/id_rsa.pub | clip`
- Ou enfin ouvrir le fichier avec un éditeur de texte et copier la clé !



# COLLER LA CLÉ PUBLIQUE DISTANTE

- Copier le contenu des clés SSH dans les [paramètres](#) du compte GitHub créé auparavant.

### Add new SSH Key

**Title**

**Key type**

Authentication Key ▾

**Key**

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQAZDZyfgUny5bYjgofQZVXHtHqwek/d30JZP
/Lwa5EZ+aTex9EeYjxZpAkgGBAVhmmiTkNGKE6VC3iuhxfmZo2FLCYds
/hv7PaWohTlcyZtNFSF7fWZctYa+PwkPsQheDm36vbZfVUxpSadiDm8LhY4DOZ95RPxJMk6QjrdsuquzCYZLwjHiNYhwy
NRlOm3ySRmnFfyFzxwHK0Gs1Zm0l3Q2Oc2YR9
/8LHGE49JNc6wHE9ElZuBIhe9MfhM9HCDNSzNh4MxldYuVsiZeoeq4210jsv9ot+61fpOQFkngz6l30w+TN9u16AvB1pon
jNaSzcdBWz8xfhAdr8o99d8b4yyIa+YTs0i+REICyhF6DPLG41
/1b+Rda8WFtRNTJt2UWmp5yq1I+5mKeMbyv4xKAMUs1qnrSDfcec
/stSSn0e2fEPBU2DzZVAn7nRlJQaPtTmkFY4vz5GnOn96CwzD
/8ABWJxJo6A1mJxvplnh6h7z38Dansaz5txTm2iHf6MMMNQ82AV1cij0lF1pl7S69gSSDhjA8C5rz7YL2uZBKuKnbgwmnvtL
```

Add SSH key

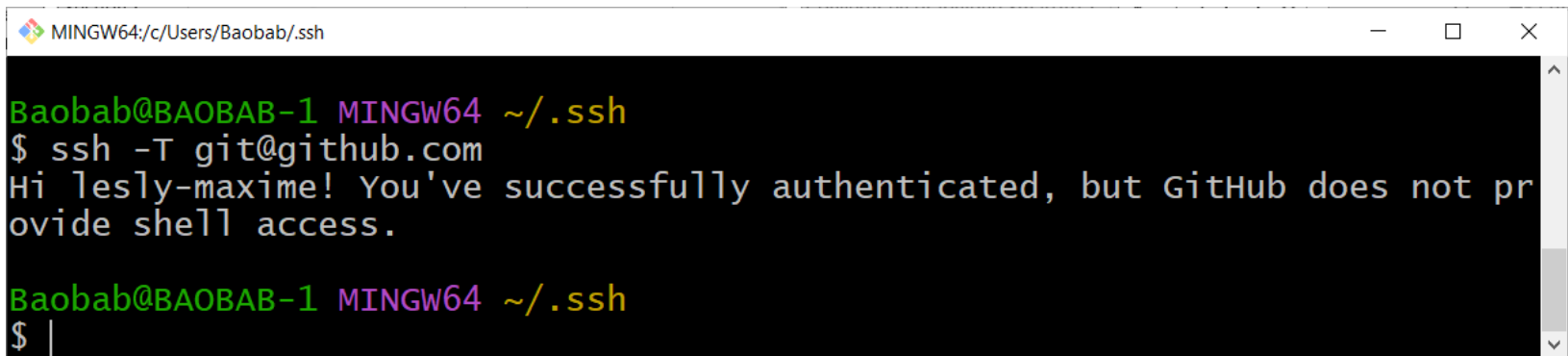
# COLLER LA CLÉ PUBLIQUE LOCALE

- **Effacer les clés en cours pour github.com :**
- `ssh-keygen -R github.com`
- On peut mélanger des entrées hachées/non hachées dans le fichier "known\_hosts".
- **Si on souhaite ajouter une clé github, on utilisera la commande :**
- `ssh-keyscan -t rsa github.com >> ~/.ssh/known_hosts`
- **Si on veut qu'il soit haché, on ajoute l'indicateur -H :**
- `ssh-keyscan -H github.com >> ~/.ssh/known_hosts`

# TESTER LA CONNEXION SSH

- Pour vérifier si la connexion SSH est fonctionnelle, il suffit de taper la commande suivante :

`ssh -T git@github.com`

A screenshot of a Windows terminal window titled 'MINGW64:/c/Users/Baobab/.ssh'. The prompt is 'Baobab@BAOBAB-1 MINGW64 ~/.ssh'. The user enters the command '\$ ssh -T git@github.com'. The output is 'Hi lesly-maxime! You've successfully authenticated, but GitHub does not provide shell access.' The prompt returns to 'Baobab@BAOBAB-1 MINGW64 ~/.ssh' followed by a '\$' and a cursor.

```
MINGW64:/c/Users/Baobab/.ssh
Baobab@BAOBAB-1 MINGW64 ~/.ssh
$ ssh -T git@github.com
Hi lesly-maxime! You've successfully authenticated, but GitHub does not provide shell access.
Baobab@BAOBAB-1 MINGW64 ~/.ssh
$ |
```



# CRÉER UN DÉPÔT LOCAL AVEC GIT

# INITIALISER UN DÉPÔT GIT

- Pour commencer à suivre un projet existant avec Git, il faut lancer une session **git-bash** (ou terminal) à partir de la console puis se positionner dans le répertoire du projet et saisir :
- **\$ git init**
- Cela crée un nouveau sous-répertoire nommé **".git"** qui contient tous les fichiers nécessaires au dépôt — un squelette de dépôt Git

Nom	Modifié le	Type	Taille
.git	06/11/2020 20:39	Dossier de fichiers	
data	06/11/2020 20:35	Dossier de fichiers	
fullcalendar	06/11/2020 20:35	Dossier de fichiers	
constants_inc.php	04/06/2020 16:54	Fichier PHP	1 Ko
db_connect_inc.php	04/06/2020 13:27	Fichier PHP	1 Ko

MINGW64:/d/Apps/uwamp/www/my-second-project

Baobab@BAOBAB-1 MINGW64 /

```
$ cd /d/Apps/uwamp/www/my-second-project/
```

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-second-project

```
$ git init
```

Initialized empty Git repository in D:/Apps/uwamp/www/my-second-project/.git/

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-second-project (master)

# CONFIGURER SON IDENTITÉ

Il est essentiel de **configurer son identité** car ce sont des informations vitales dont on aura besoin pour les validations dans Git.

▪ **On peut le faire globalement, par repo ou lors du commit :**

1. Changer les informations du **committer** globalement :

- `$ git config --global user.name "Baobab Tree"`
- `$ git config --global user.email baobab@gmail.com`

2. Changer les informations du **committer** par repository :

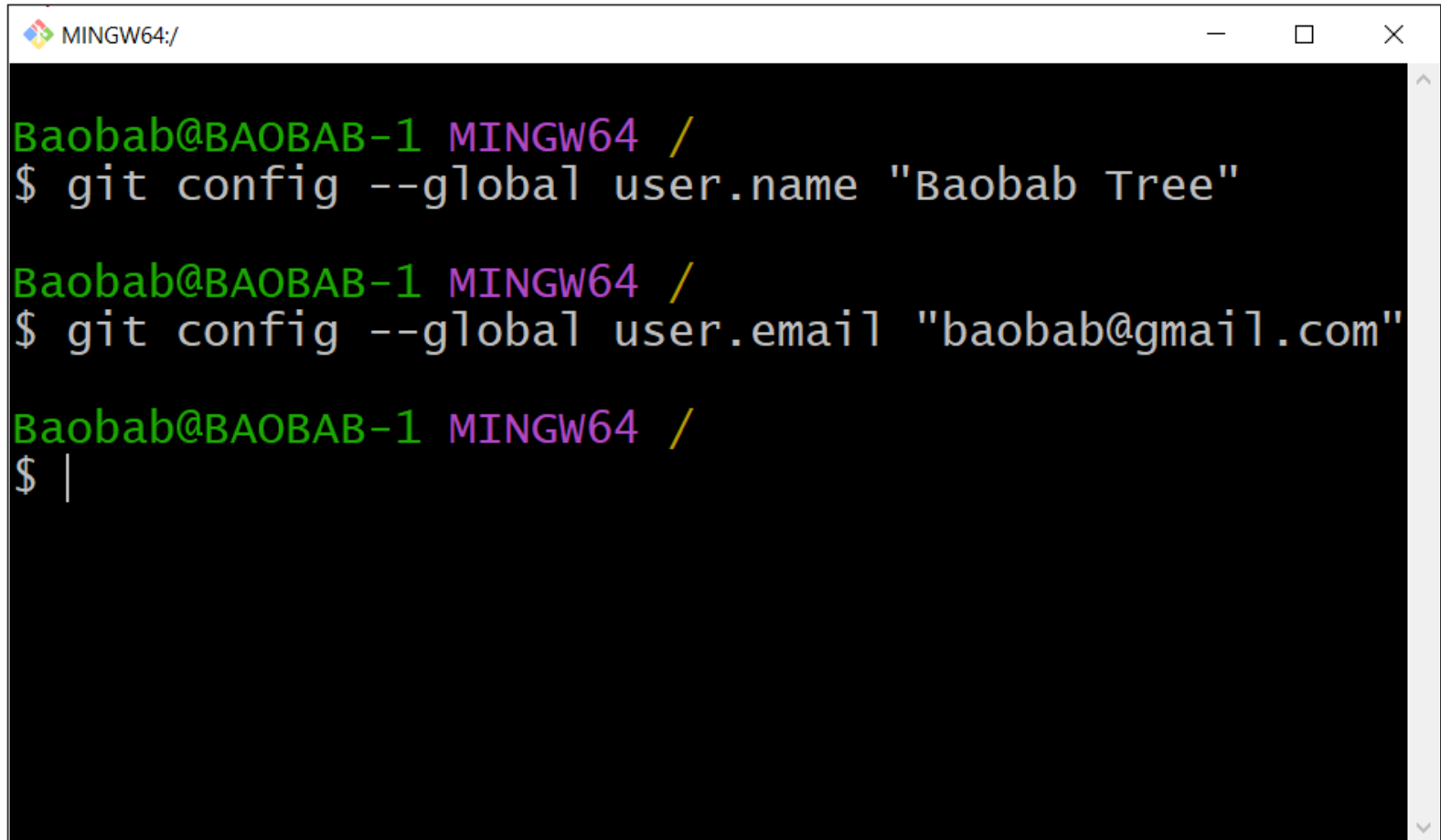
- `$ git config user.name Baobab Tree"`
- `$ git config user.email baobab@gmail.com`

3. Changer les informations du **committer** lors du commit :

- `git commit --author="Baobab Tree <baobab@gmail.com>"`



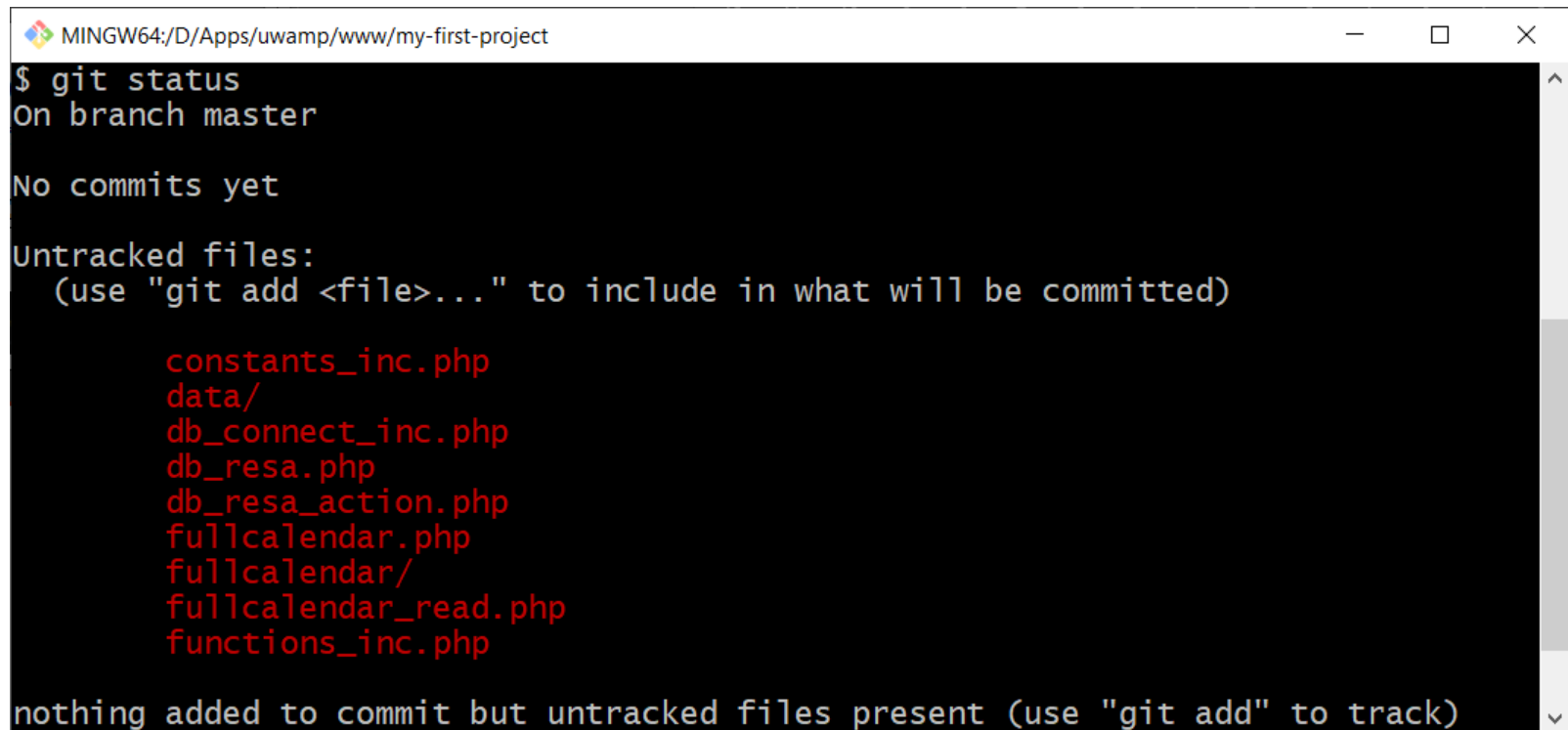
# CONFIGURER SON IDENTITÉ



```
MINGW64:/  
  
Baobab@BAOBAB-1 MINGW64 /  
$ git config --global user.name "Baobab Tree"  
  
Baobab@BAOBAB-1 MINGW64 /  
$ git config --global user.email "baobab@gmail.com"  
  
Baobab@BAOBAB-1 MINGW64 /  
$ |
```

# AFFICHER LE STATUT DES FICHIERS

- Pour voir ce qui a changé depuis le dernier commit : `$ git diff`
- Pour lister tous les fichiers modifiés (ou nouveaux) : `$ git status`



```
MINGW64:/D:/Apps/uwamp/www/my-first-project
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        constants_inc.php
        data/
        db_connect_inc.php
        db_resa.php
        db_resa_action.php
        fullcalendar.php
        fullcalendar/
        fullcalendar_read.php
        functions_inc.php

nothing added to commit but untracked files present (use "git add" to track)
```

# EXÉCUTER UN COMMIT

- Effectuer les modifications de code et effectuer les commits :
  - Pour ajouter l'ensemble des fichiers à la liste des fichiers à commiter : `$ git add .`
  - Ou sinon pour un fichier/dossier : `$ git add nom_dossier nom_fichier.ext`
  - Puis commiter avec : `$ git commit -m "commentaire associé au commit"`

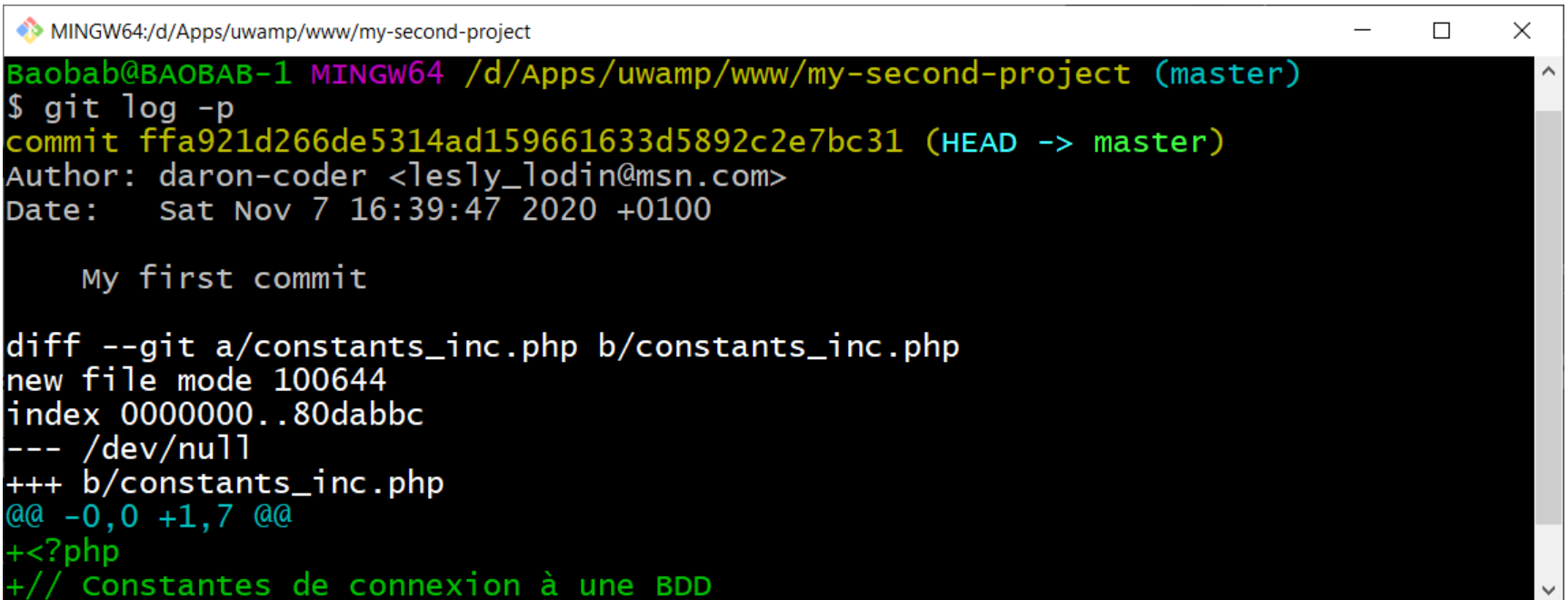
```
MINGW64:/d/Apps/uwamp/www/my-second-project

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-second-project (master)
$ git add .
warning: LF will be replaced by CRLF in data/aston_bdd.sql.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in fullcalendar/core/main.min.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in fullcalendar/daygrid/main.min.js.
The file will have its original line endings in your working directory

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-second-project (master)
$ git commit -m "My first commit"
[master (root-commit) 53bd780] My first commit
13 files changed, 1946 insertions(+)
create mode 100644 constants_inc.php
create mode 100644 data/aston_bdd.sql
```

# AFFICHER LE JOURNAL

- Pour consulter les historiques des commits :
  - `$ git log`
  - `$ git log -p` (pour avoir plus détails)



```
MINGW64:/d/Apps/uwamp/www/my-second-project
Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-second-project (master)
$ git log -p
commit ffa921d266de5314ad159661633d5892c2e7bc31 (HEAD -> master)
Author: daron-coder <lesly_lodin@msn.com>
Date: Sat Nov 7 16:39:47 2020 +0100

    My first commit

diff --git a/constants_inc.php b/constants_inc.php
new file mode 100644
index 0000000..80dabbc
--- /dev/null
+++ b/constants_inc.php
@@ -0,0 +1,7 @@
+<?php
+// Constantes de connexion à une BDD
```

# INITIALISER UN DÉPÔT DISTANT

- La commande "**git remote add**" est utilisée pour créer un dépôt distant :
  - Le premier argument est le **nom du dépôt distant**
  - Le deuxième argument est son **URL**
- Créer le dépôt distant à l'aide des instructions suivantes :
  - `$ git branch -M main`
  - `$ git remote add origin https://github.com/daron-coder/my-second-project.git`
  - `$ git push -u origin main`

# INITIALISER UN DÉPÔT DISTANT

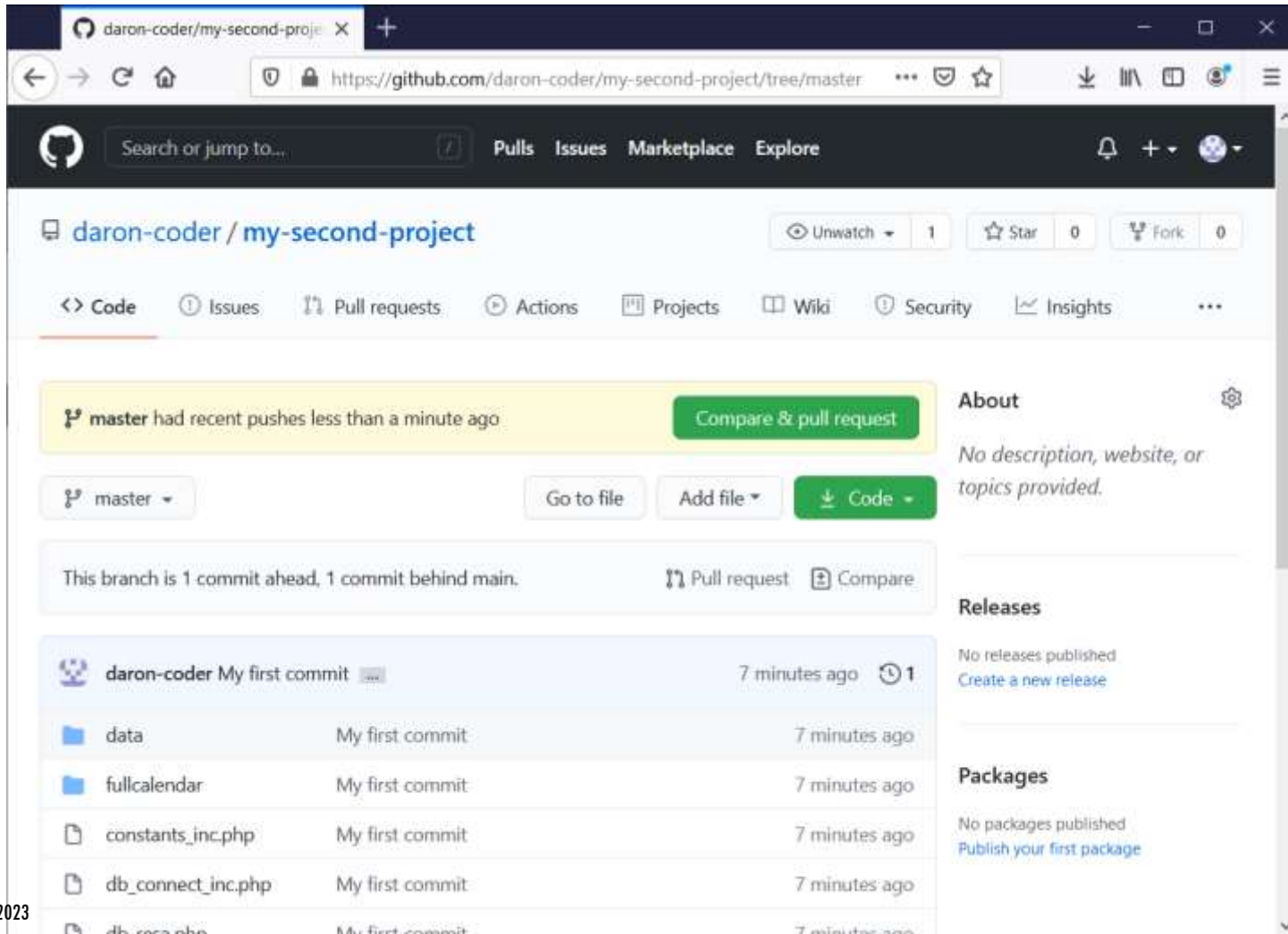
```
MINGW64:/d/Apps/uwamp/www/my-second-project
Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-second-project (master)
$ git push -u prj2 master
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 8 threads
Compressing objects: 100% (19/19), done.
Writing objects: 100% (19/19), 1.27 MiB | 1.01 MiB/s, done.
Total 19 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/daron-coder/my-second-project/pull/new/master
remote:
To https://github.com/daron-coder/my-second-project.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'prj2'.

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-second-project (master)
$ git remote
prj2

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-second-project (master)
$ |
```



# INITIALISER UN DÉPÔT DISTANT



# EN RÉSUMÉ...



## Séquence d'envoi de code vers le dépôt distant :

- `cd /var/www/html/<dossier_du_projet>`
- `echo "# Titre du README" >> README.md`
- `$ git init`
- `$ git status`
- `$ git add .`
- `$ git status`
- `$ git commit -m "My first commit"`
- `$ git log -p`
- `$ git branch -M main`
- `$ git remote add origin https://github.com/<username>/<projectname>.git`
- `$ git push -u origin main`



# CLONER UN PROJET EXISTANT

# CLONAGE D'UN PROJET EXISTANT

- La commande **"git clone"** permet de cloner un dépôt existant (en local ou à travers un réseau si le chemin est une URL) à un autre endroit :
  - La métaphore du clonage doit être comprise ici comme une copie exacte
  - A la fin du clonage, les deux dépôts sont les images parfaites l'une de l'autre
  - Ils possèdent le même historique et il n'y en a pas un qui est plus légitime que l'autre
- Il faut tout d'abord se placer dans le répertoire racine du serveur web puis lancer la commande suivante dépôt public anonyme :
  - **\$ git clone** <https://github.com/daron-coder/my-first-project.git>
- Ou en SSH bien pour un dépôt privé avec autorisation :
  - **\$ git@github.com:daron-coder/my-first-project.git**

# CLONAGE D'UN PROJET EXISTANT

```
MINGW64:/d/Apps/uwamp/www/my-first-project

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www
$ git clone https://github.com/daron-coder/my-first-project.git
Cloning into 'my-first-project'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www
$ cd my-first-project/

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-first-project (main)
$ ls
README.md

Baobab@BAOBAB-1 MINGW64 /d/Apps/uwamp/www/my-first-project (main)
$
```





# GITHUB VIA VISUAL STUDIO CODE



# INTRODUCTION

- **Visual Studio Code** (VS Code) est devenu l'un des éditeurs les plus populaires dans le domaine du développement web.
- Il a acquis une telle popularité grâce à ses nombreuses fonctionnalités intégrées telles que **l'intégration du contrôle de source**, notamment avec Git.
- Exploiter la puissance de Git à partir de VS Code peut rendre le flux de travail plus **efficace**, plus **rapide** et plus **robuste**.

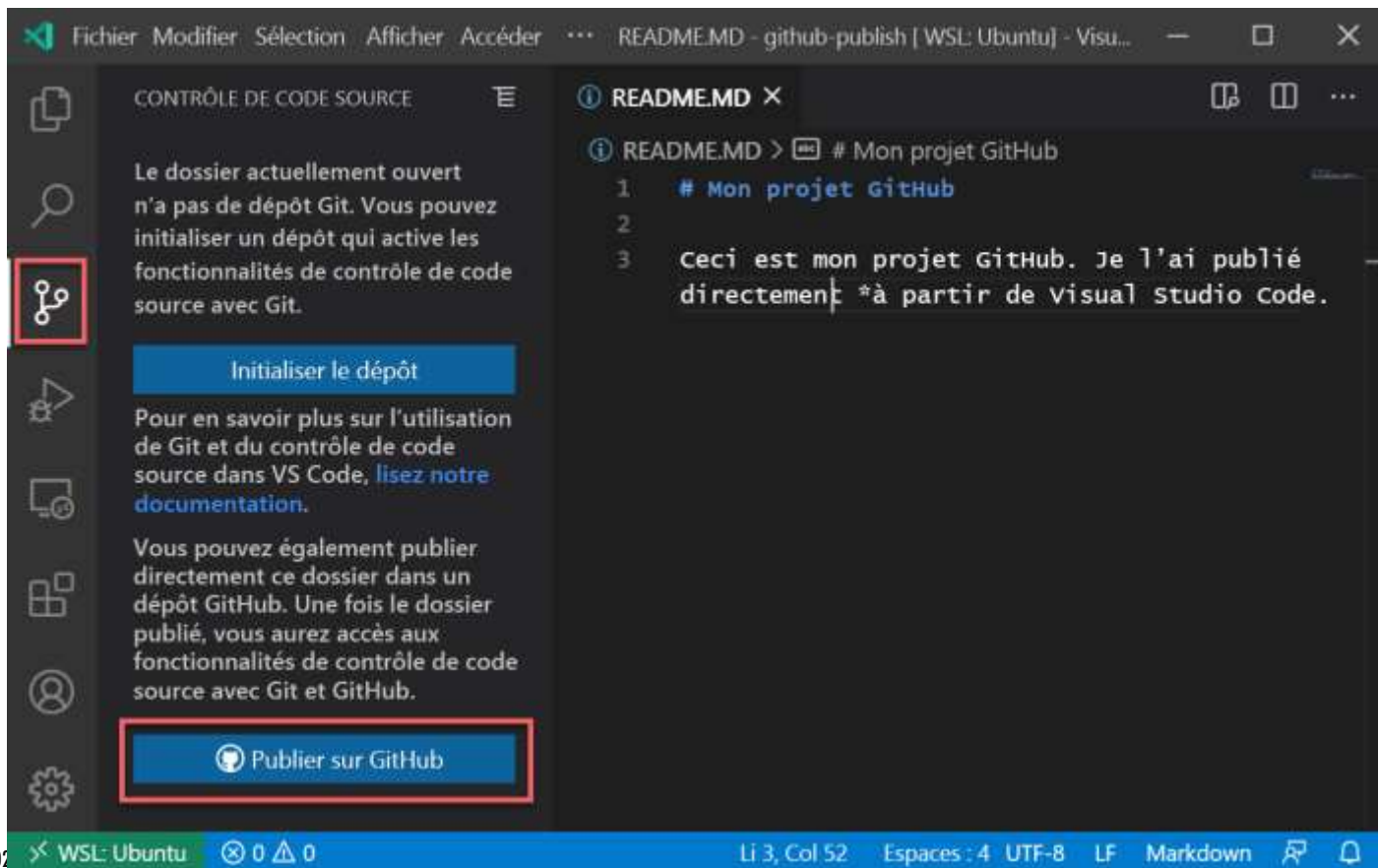


# CRÉER UN RÉFÉRENTIEL

- Il faut normalement créer un référentiel sur GitHub pour pouvoir y publier du code.
- Visual Studio Code permet **d'effectuer cette opération directement** à partir de l'éditeur.
- Il suffit pour cela de :
  - Créer un **dossier** et l'ouvrir dans VS Code;
  - Créer un fichier **README.md** au format Markdown;
  - Créer éventuellement un fichier **.env.development** dans lequel on pourra définir des informations confidentielles concernant l'application, par exemple :  
`CONNECTION_STRING=secret-that-should-not-be-pushed-to-github`
  - Créer éventuellement un fichier **.gitignore** dans lequel on indiquera à Git quels fichiers ou répertoires ignorer dans le projet, par exemple :  
`*.txt`

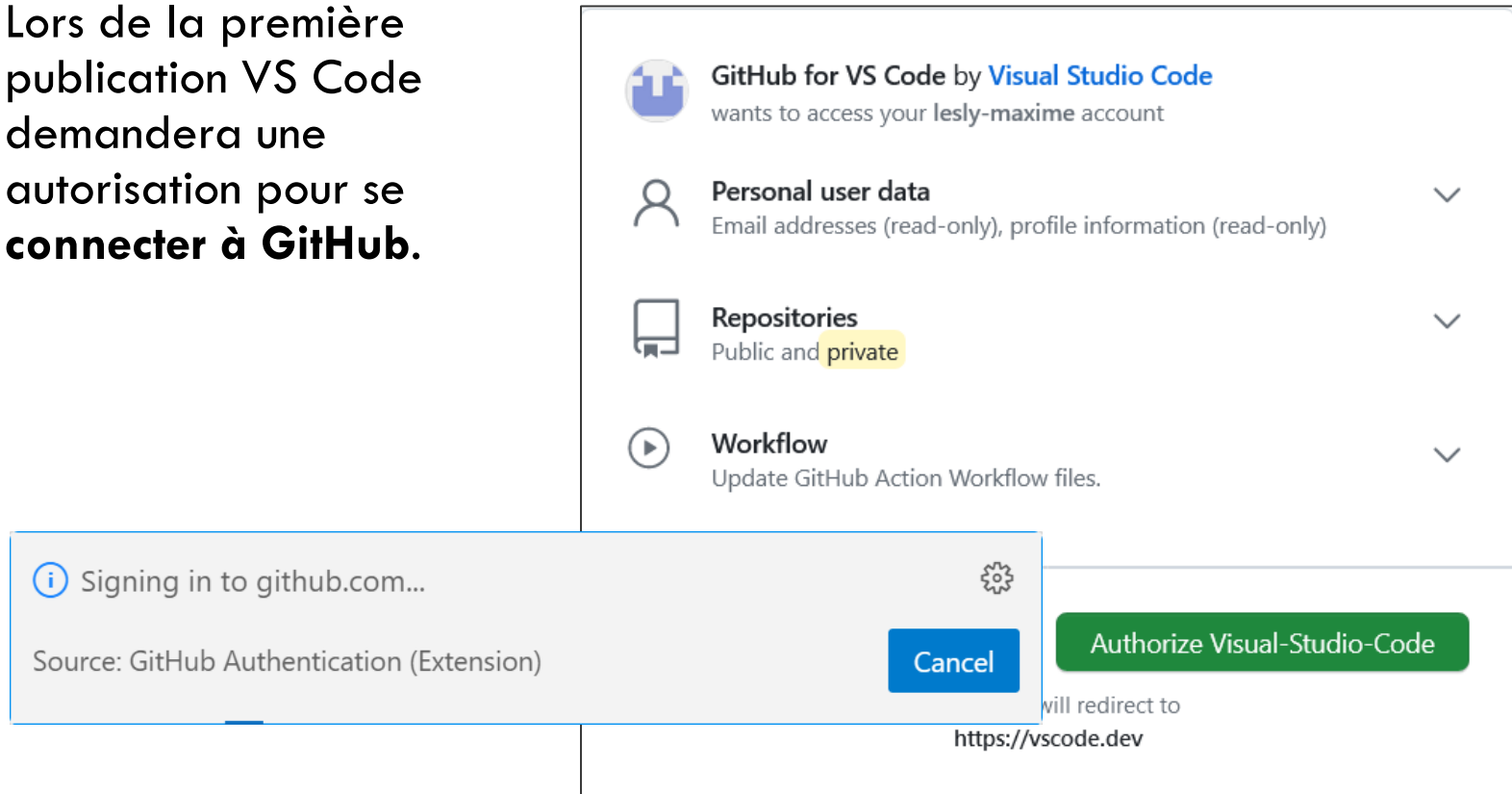
# PUBLIER UN RÉFÉRENTIEL

- Ouvrir la vue de **gestion du contrôle de code source (SCM)** via l'icône SCM dans la barre d'activité puis sélectionner **"Publier sur GitHub"**.



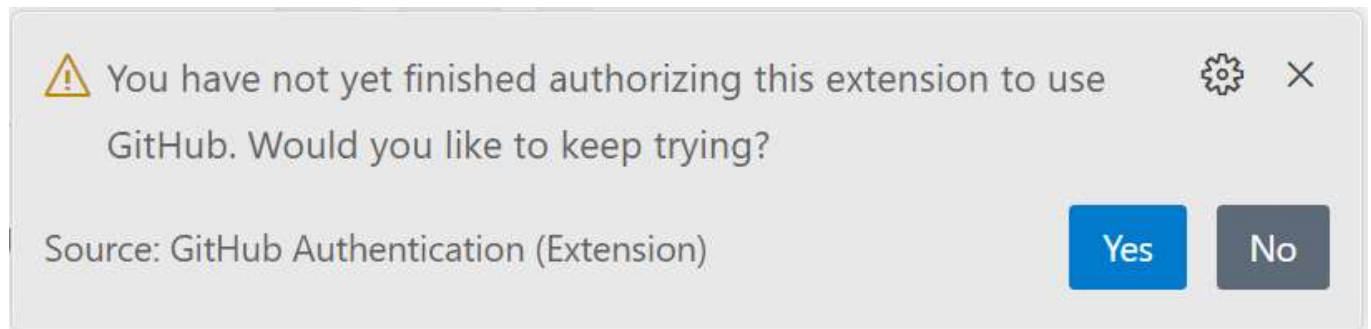
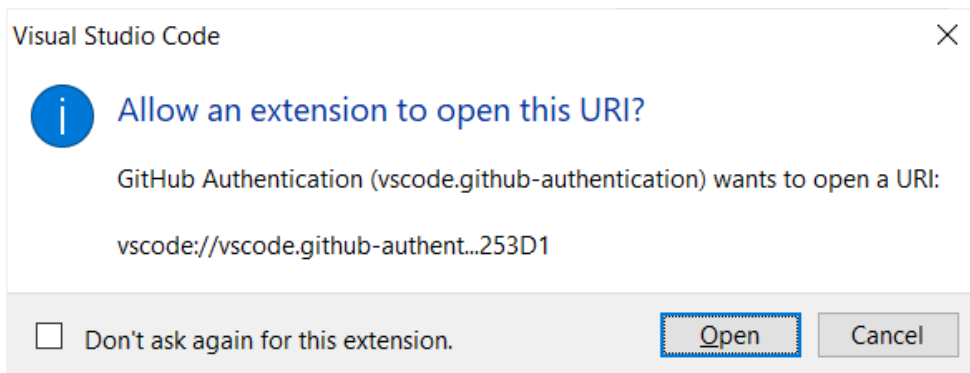
# PUBLIER UN RÉFÉRENTIEL

- Lors de la première publication VS Code demandera une autorisation pour se connecter à GitHub.



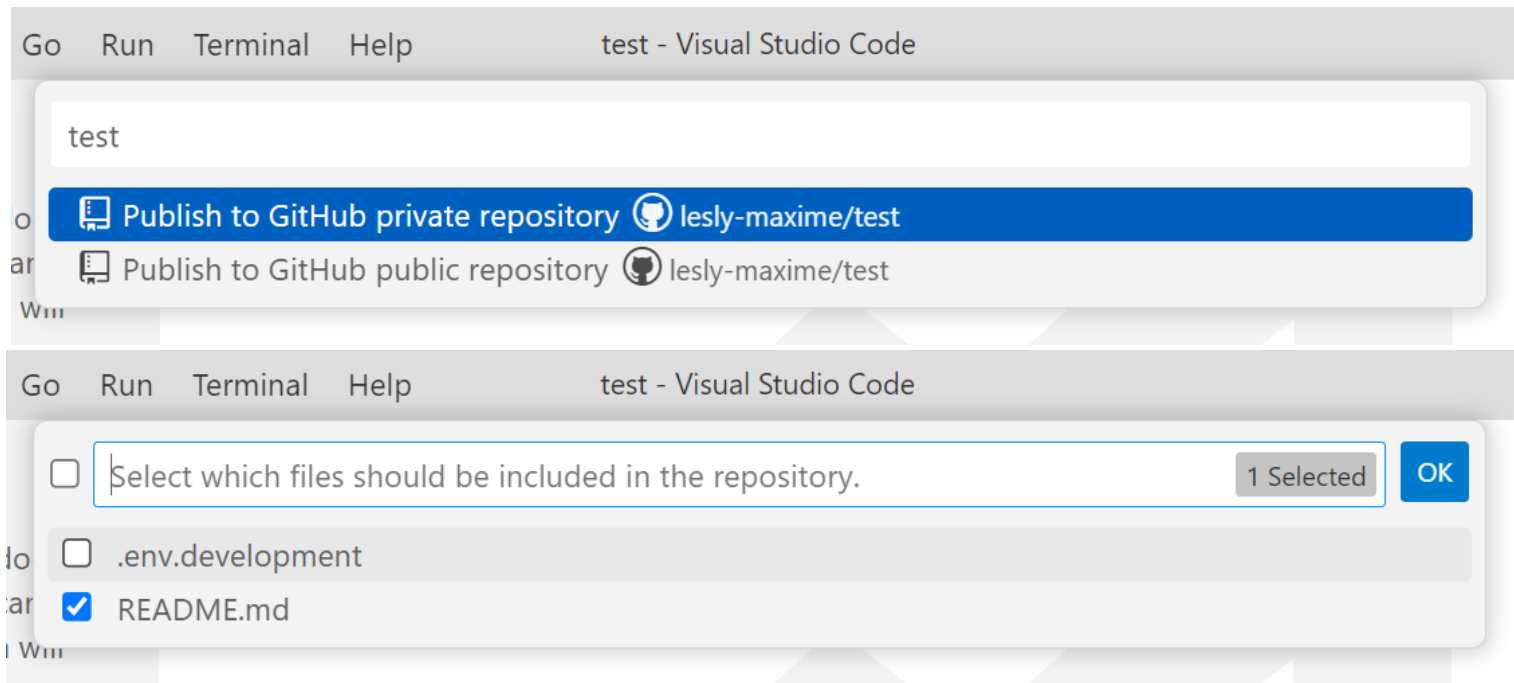
# PUBLIER UN RÉFÉRENTIEL

- Puis GitHub demande l'autorisation de **communiquer avec VS Code**.



# PUBLIER UN RÉFÉRENTIEL

- Indiquer ensuite si le projet doit être **public** ou **privé** et les fichiers à publier.





# PUBLIER UN RÉFÉRENTIEL

The screenshot shows the GitHub interface for a repository named 'baobab-project' owned by 'lesly-maxime'. The repository is public and has 1 watch, 0 stars, and 0 forks. The main content area displays the 'master' branch with an 'Initial commit' made 1 hour ago. Below the commit, there is a 'README.md' file, also committed 1 hour ago. The README content shows the title 'baobab-project'. On the right sidebar, there is an 'About' section with the description 'My first project for Baobab using GitHub', and links to 'Readme', 'Activity', '0 stars', '1 watching', and '0 forks'. At the bottom of the sidebar, there is a 'Releases' section.

lesly-maxime / baobab-project

Code Issues Pull requests Actions Projects Wiki Security

baobab-project Public Pin Unwatch 1 Fork 0 Star 0

master Go to file Add file <> Code About

My first project for Baobab using GitHub

Readme Activity 0 stars 1 watching 0 forks

Releases

lesly-maxime Initial commit 1 hour ago 1

README.md Initial commit 1 hour ago

README.md

baobab-project