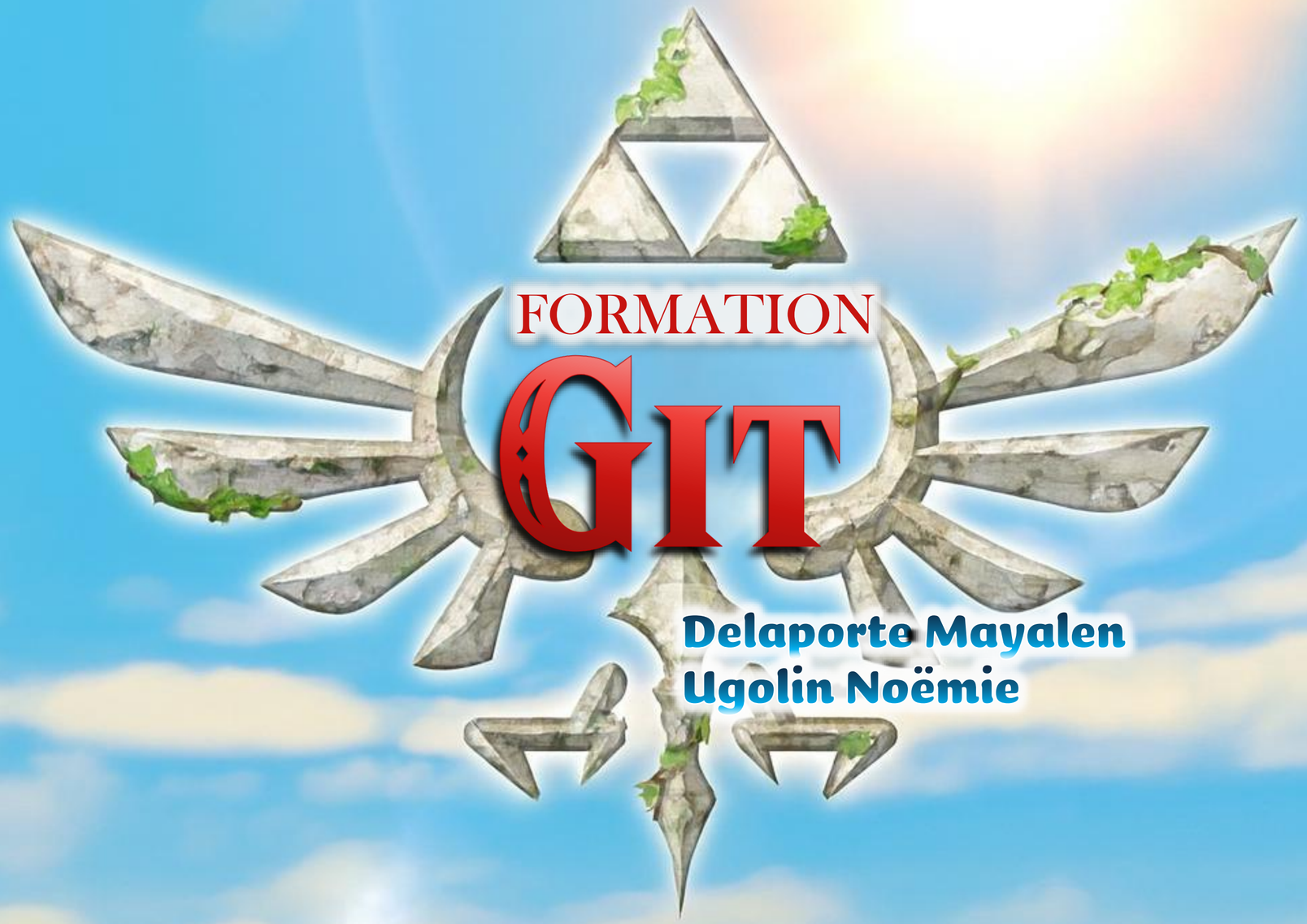
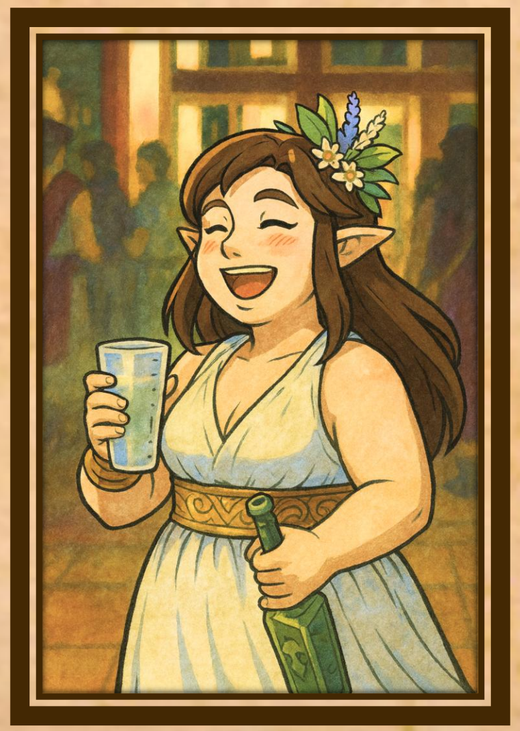


# FORMATION GIT

Delaporte Mayalen
Ugolin Noëmie

# Presentation



## Noëmie Ugolin

Zelda niveau 100
Git niveau 100

## Mayalen Delaporte

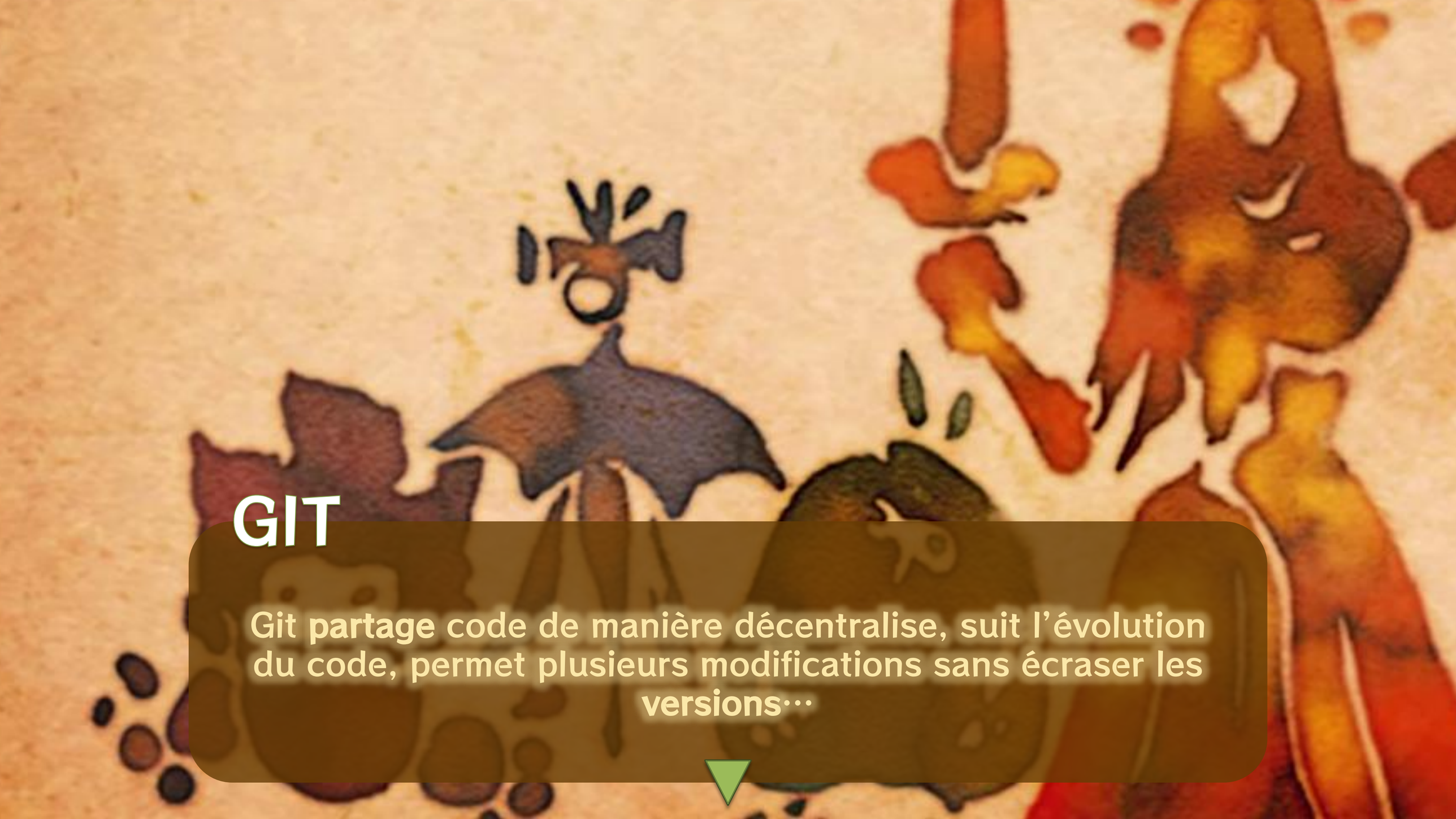
Zelda niveau 10
Git niveau 50

# Timeline de la séance

# GIT

# GIT

Git=  Logiciel de version en local

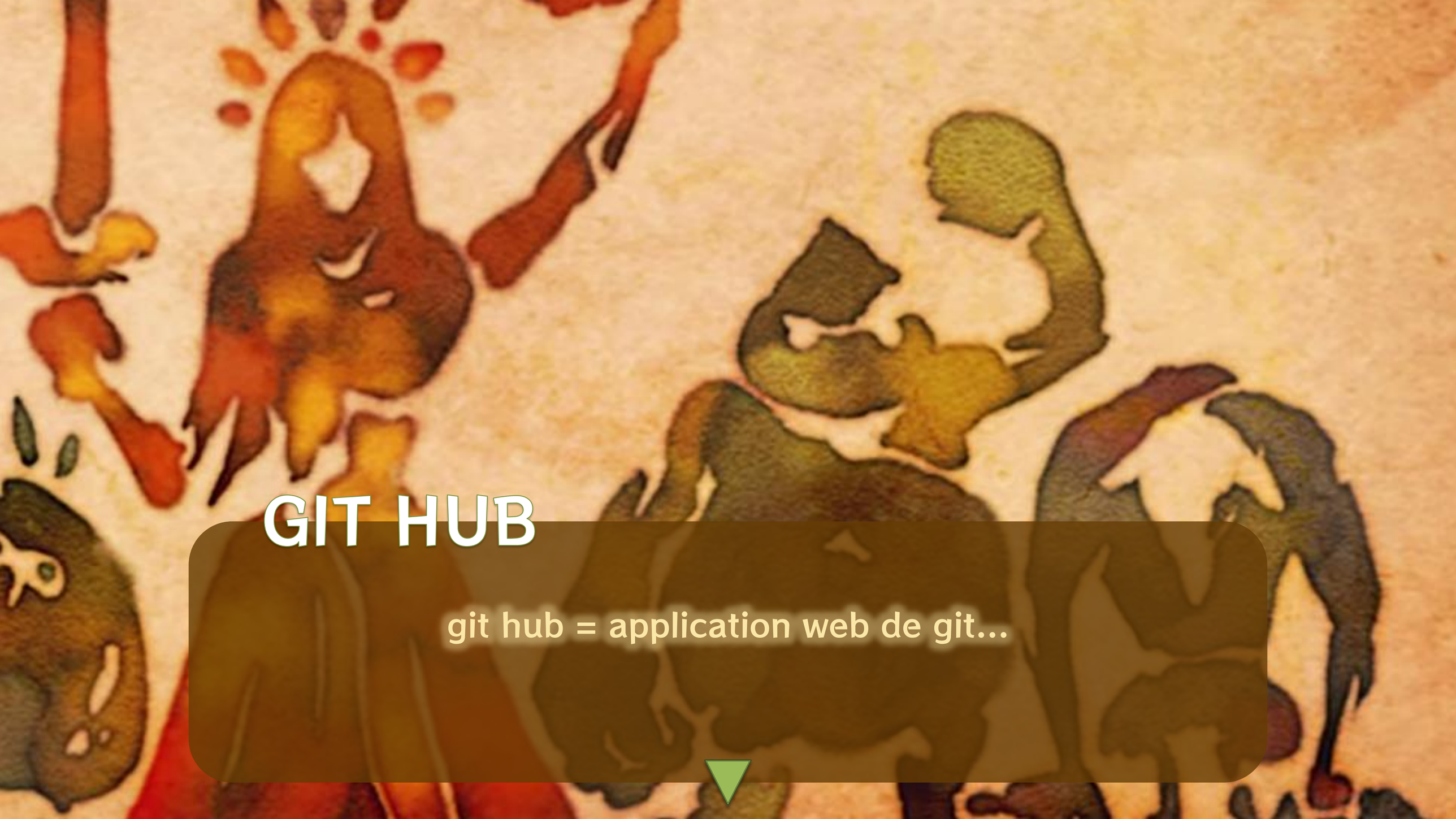


# GIT

Git **partage** code de manière décentralise, suit l'évolution du code, permet plusieurs modifications sans écraser les **versions**…

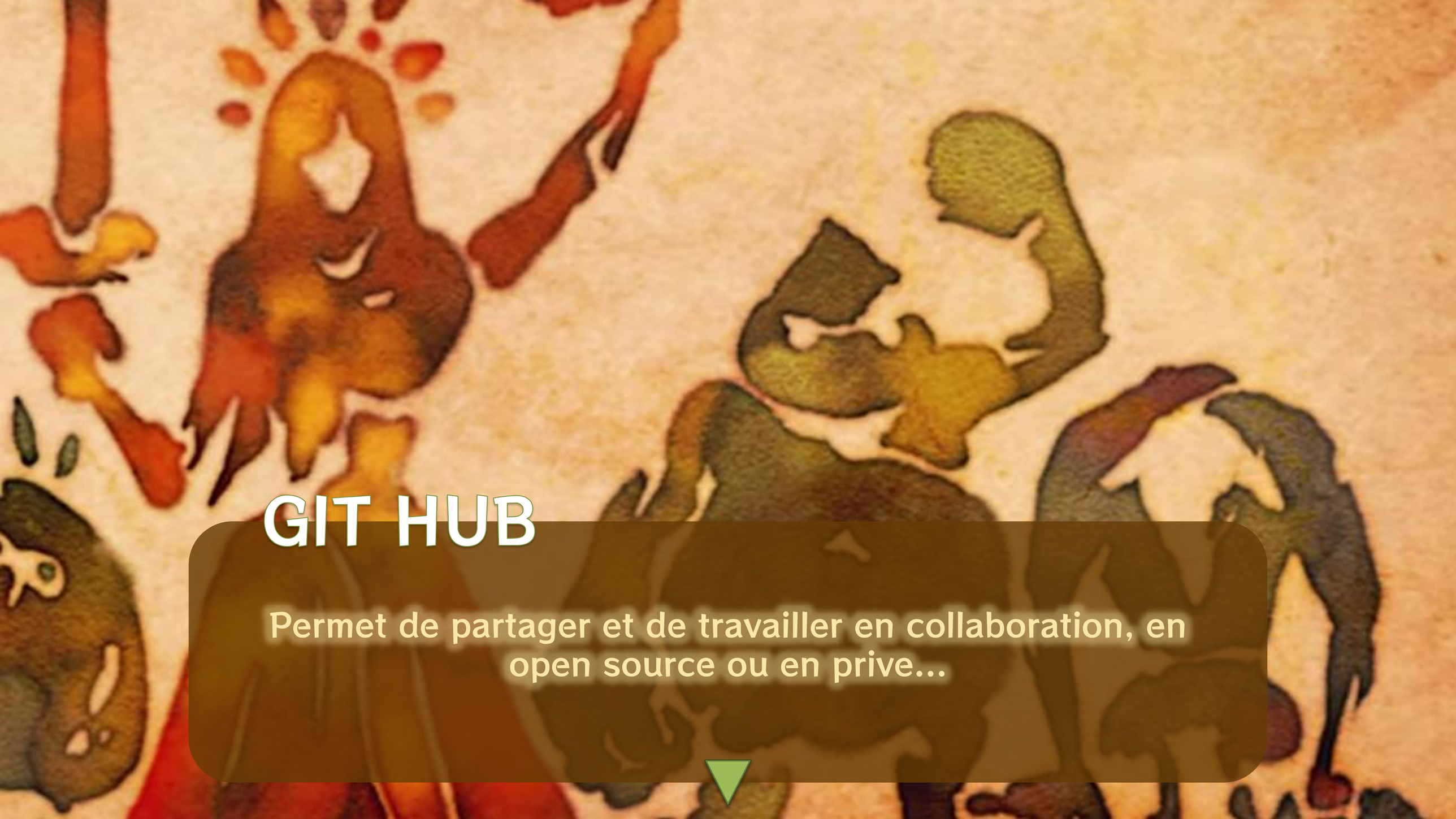
# GIT

…local, crée des branches, fonctionne sans connexion internet



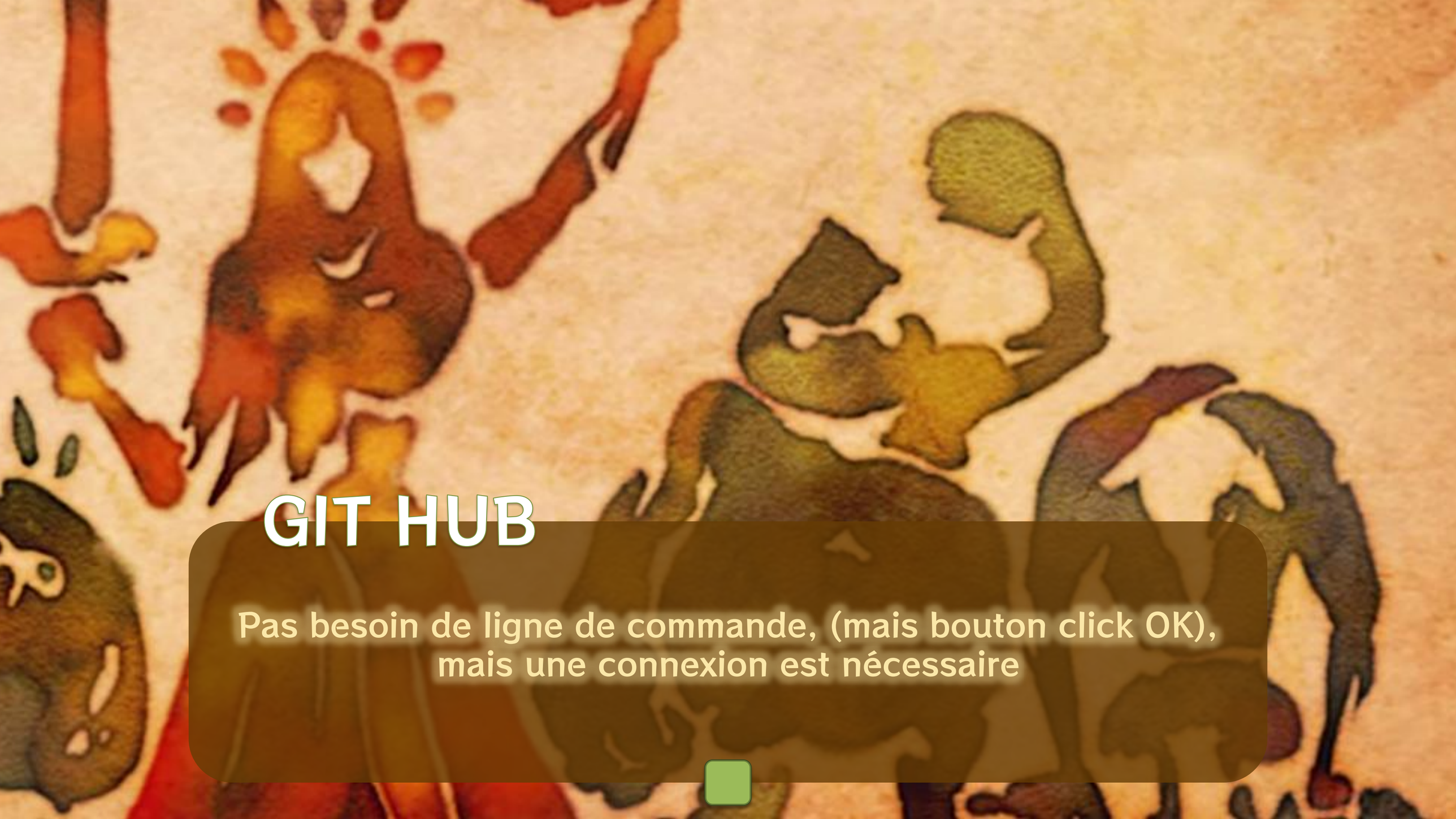
# GIT HUB

git hub = application web de git...



# GIT HUB

Permet de partager et de travailler en collaboration, en open source ou en prive...



# GIT HUB

Pas besoin de ligne de commande, (mais bouton click OK),
mais une connexion est nécessaire

# INTERETS

versioning= gestion des versions, traçabilité et historique

# Versioning

Chaque modification est enregistrée via un COMMIT
Création de branches pour développer des fonctionnalités en parallèle
Retour possible à une version antérieur

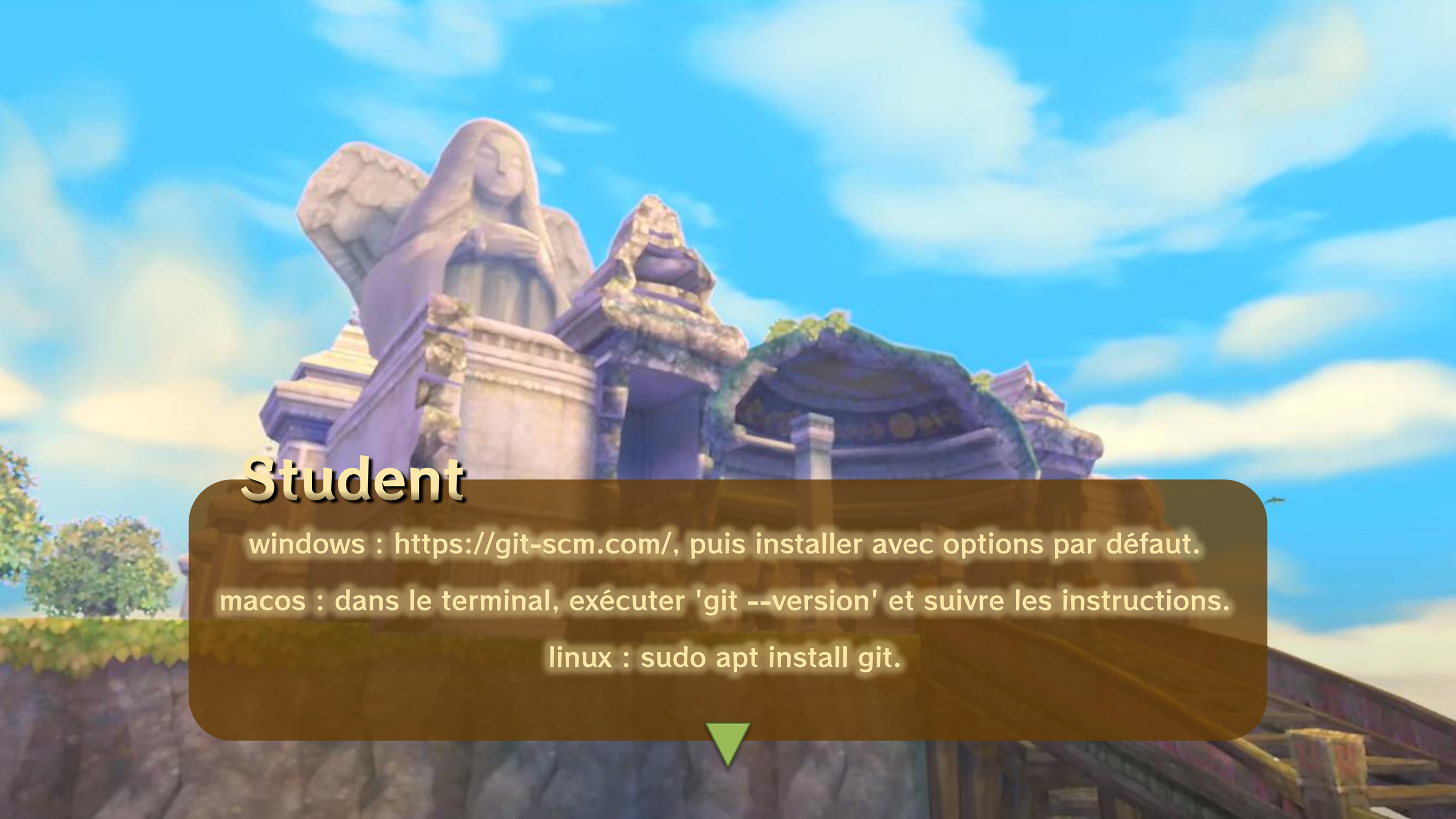
# Branche

Travailler sur plusieurs éléments en parallèle
Séparer les expérimentations du code stable
Faciliter le travail en équipe
Réduire les conflits et erreurs lors des modifications

# Branche

Création d'une **ligne parallèle de développement**
Issue de la branche **main**
Travail isolé **sans impacter la branche principale**
Permet de **tester et intégrer proprement** les nouveautés

# Installation



## Student

windows : https://git-scm.com/, puis installer avec options par défaut.

macos : dans le terminal, exécuter 'git --version' et suivre les instructions.

linux : sudo apt install git.

# Download for Windows

**Click here to download** the latest (**2.51.0(2)**) **x64** version of **Git for Windows**. T
recent maintained build. It was released **3 days ago**, on 2025-09-29.

**Other Git for Windows downloads**

**Standalone Installer**

**Git for Windows/x64 Setup.**

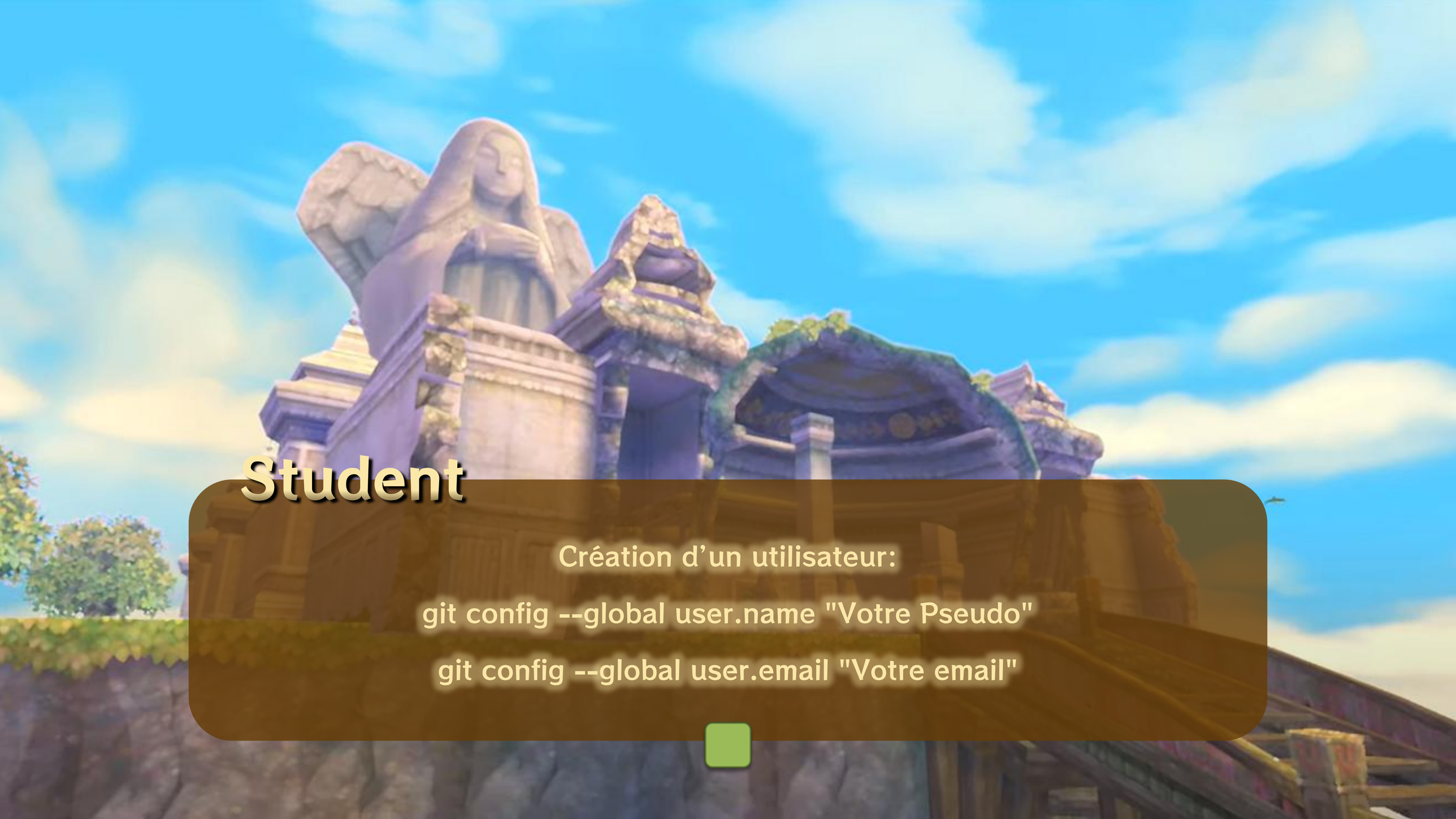
**Git for Windows/ARM64 Setup.**

**Portable ("thumbdrive edition")**

**Git for Windows/x64 Portable.**

# Student

windows : https://git-scm.com/



# Student

Création d’un utilisateur:

```
git config --global user.name "Votre Pseudo"
git config --global user.email "Votre email"
```

# COMMANDE DE BASE

# GIT INIT

Création d’un nouveau dépôt dans un dossier
crée un .git qui contient l’historique

```
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur> mkdir Personne1
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur> cd Personne1
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne1> cat > Monfichier.txt

PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne1> git init
Initialized empty Git repository in C:/Users/mayal/OneDrive/Bureau/M2/Formation_Formateur/Personne1/.git/
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne1>
```

cd = change directory
mkdir = creation de nouveau dossier

```
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur> mkdir Personne1
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur> cd Personne1
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne1> cat > Monfichier.txt

PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne1> git init
Initialized empty Git repository in C:/Users/mayal/OneDrive/Bureau/M2/Formation_Formateur/Personne1/.git/
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne1>
```

## Student

acceder au terminal
créer un dossier pour deux personne fictive
dans le premier réaliser un git init

# GIT CLONE + URL

Copie un dépôt git existant (github) en local
Sert à rejoindre un projet existant

# GIT CLONE + URL

## Student

Aller dans le second dossier
réaliser un git clone de votre premier dossier
vérifier que votre dossier soit présent

# GIT CLONE + URL

```
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne1>
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne1> cd ..
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur> cd Personne2
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne2> git clone Personne1/.git/
fatal: repository 'Personne1/.git/' does not exist
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne2> git clone /Personne1/.git/
fatal: repository '/Personne1/.git/' does not exist
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne2> git clone C:\Users\mayal\OneDrive\Bureau\M2\Fo
rmation_Formateur\Personne1
Cloning into 'Personne1'...
warning: You appear to have cloned an empty repository.
done.
PS C:\Users\mayal\OneDrive\Bureau\M2\Formation_Formateur\Personne2>
```

# GIT STATUS

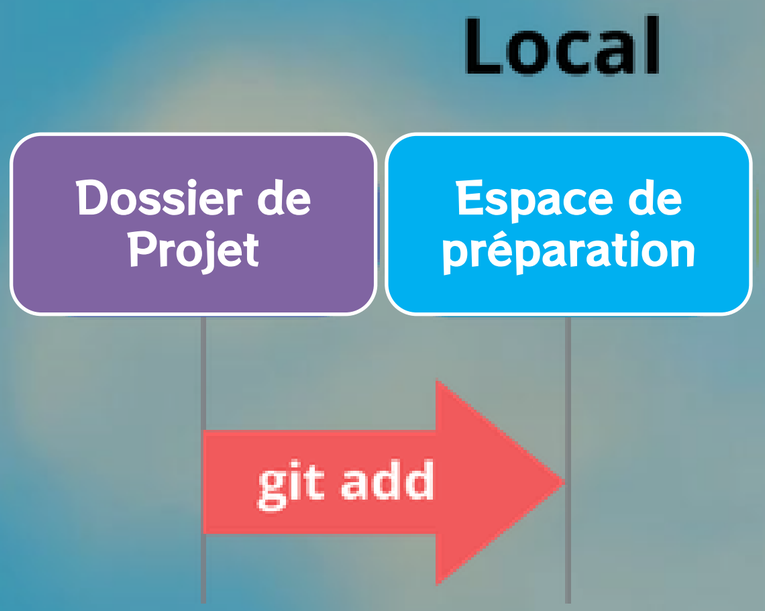
Voir état actuel du projet
utile avant un commit pour voir l'état

# GIT STATUS

## Student

Réalisez un git status afin de voir l'état de votre dossier

# GIT ADD .




## Student

Placer les fichiers modifier en zone de préparation avant commit
Sélectionner les documents à sauvegarder dans la prochaine version

# GIT ADD .

**Student**

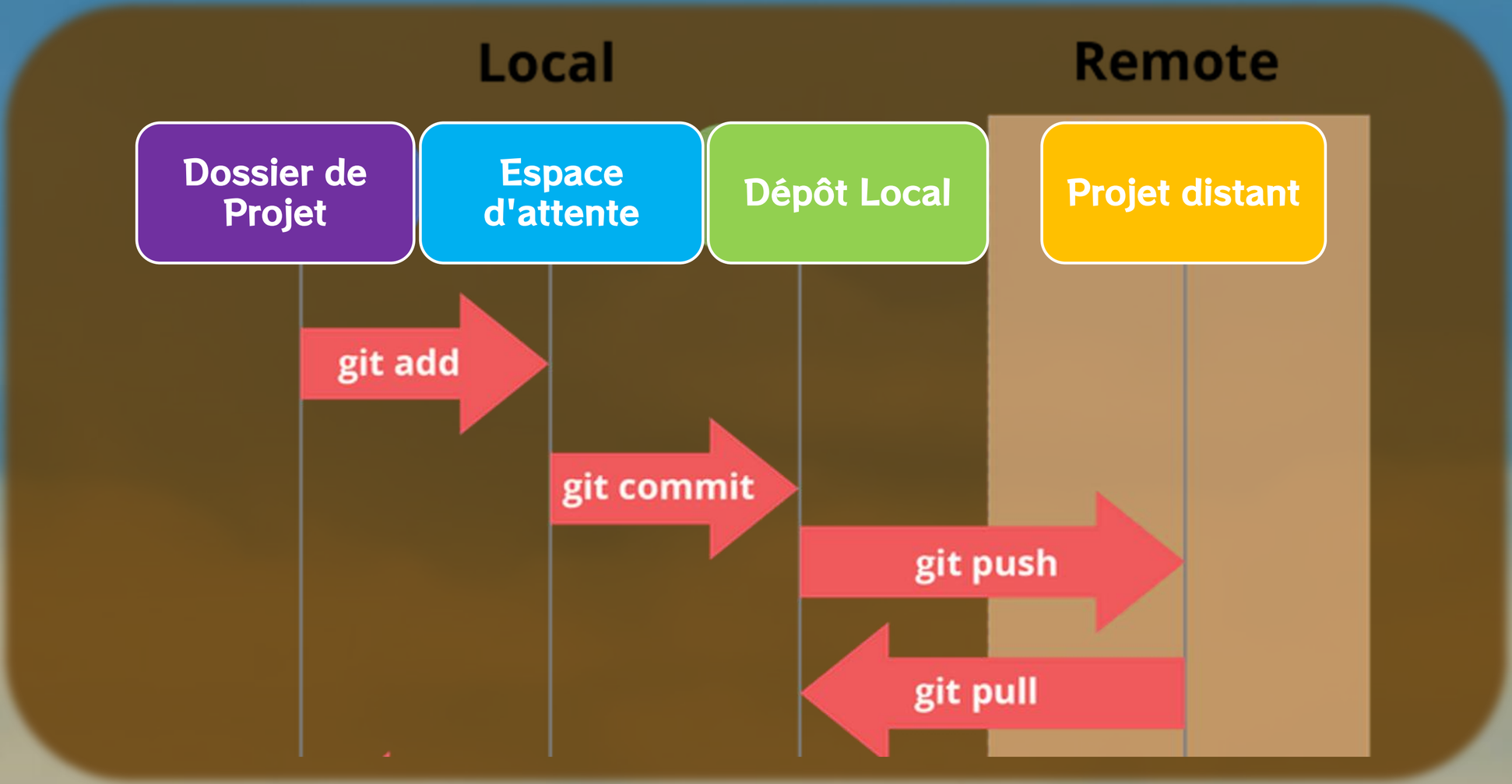
```
# Ajouter un fichier
git add fichier.txt
# Ajouter tous les fichiers modifiés
git add .
```

# GIT ADD .

## Student

Dans un des dossier créé un fichier text réalisez un git add sur ce fichier
Vous pouvez faire un git status pour voir l'état de votre projet

# GIT
# COMMIT PULL PUSH



# GIT
# COMMIT PULL PUSH

Lors d'un projet toujours récupérer les éléments avant de préparer et d'envoyer nos documents

# GIT COMMIT -M 'MESSAGE'

## Student

Sauvegarde des changements ajoute en zone de préparation (git add .)
Nécessite un message clair

# GIT PULL

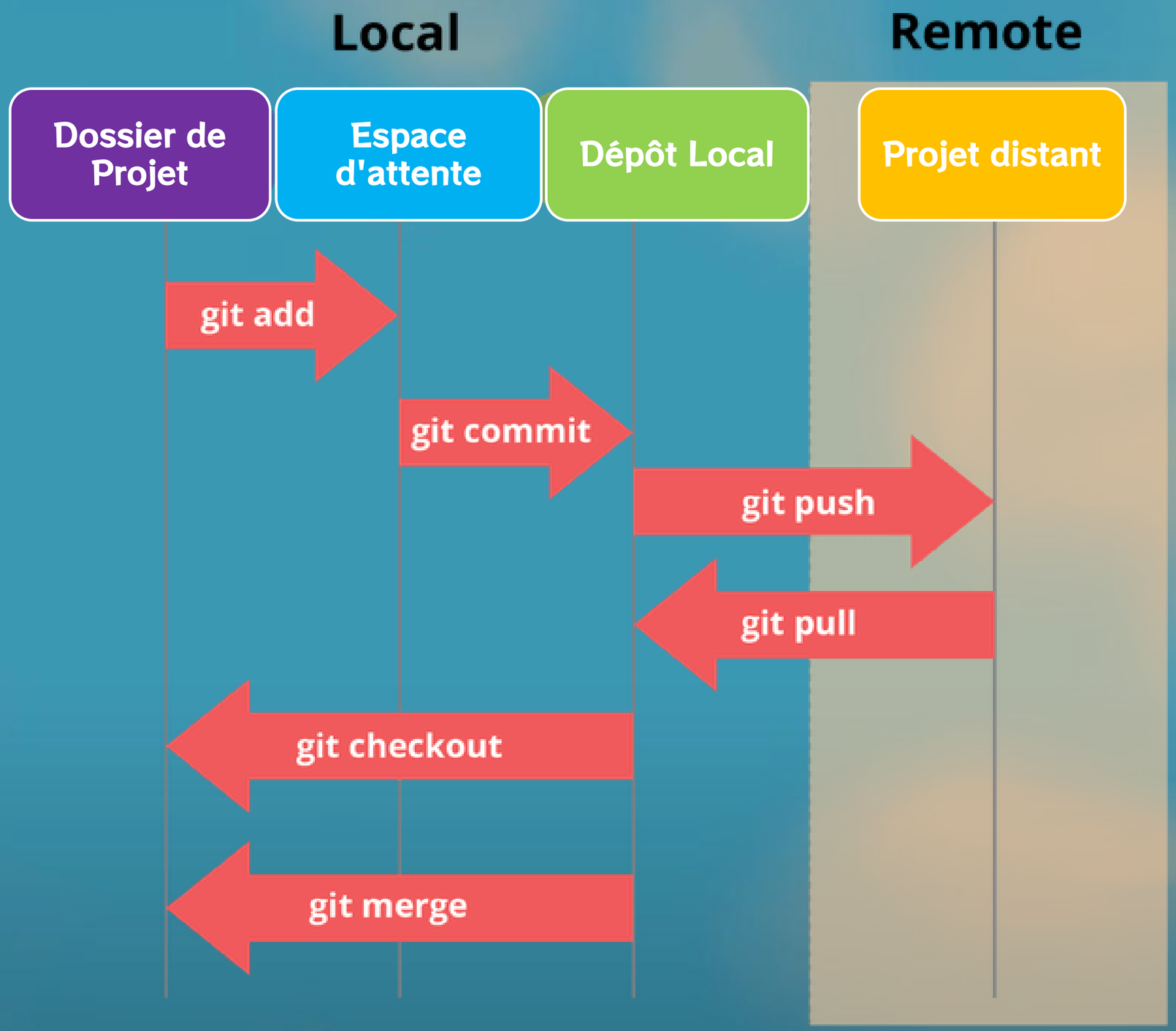
## Student

Récupérer les fichiers modifiés depuis le dépôt distant
mettre à jour la branche locale

# GIT PUSH

## Student

Envoie les fichiers modifiés vers le dépôt distant


Local
Remote
Dossier de Projet
Espace d'attente
Dépôt Local
Projet distant
git add
git commit
git push
git pull
git checkout
git merge

# Exercice

## Student

Au sein de vos projets, ajouter un élément dans le premier dossier et partager le avec le second dossier

Git add fichier.txt
Git pull
Git commit –M
Git push

# GIT LOG

**Student**

Afficher l’historique des commit

# GIT CHECKOUT -B <BRANCHE>

**Student**

Changer de branche ou revenir à une version anterieur
Naviger dans historique ou dans brancher

# GIT CHECKOUT -B <BRANCHE>

**Student**

```
# Changer de branche
git checkout ma-branche

# Créer et se déplacer sur une nouvelle branche
git checkout -b nouvelle-branche
```

# CONFLIT

Apparait quand plusieurs personnes modifient le même fichier / la même ligne
==> Répartition des tâches ...

# CONFLIT

Marque les conflits avec :

**<<<<<< / ====== / >>>>>>**

# Conflit

Choisir la version que l’on veut garder,
Supprimer l’autre et faire un :
git add . – git commit – git push

GitHub

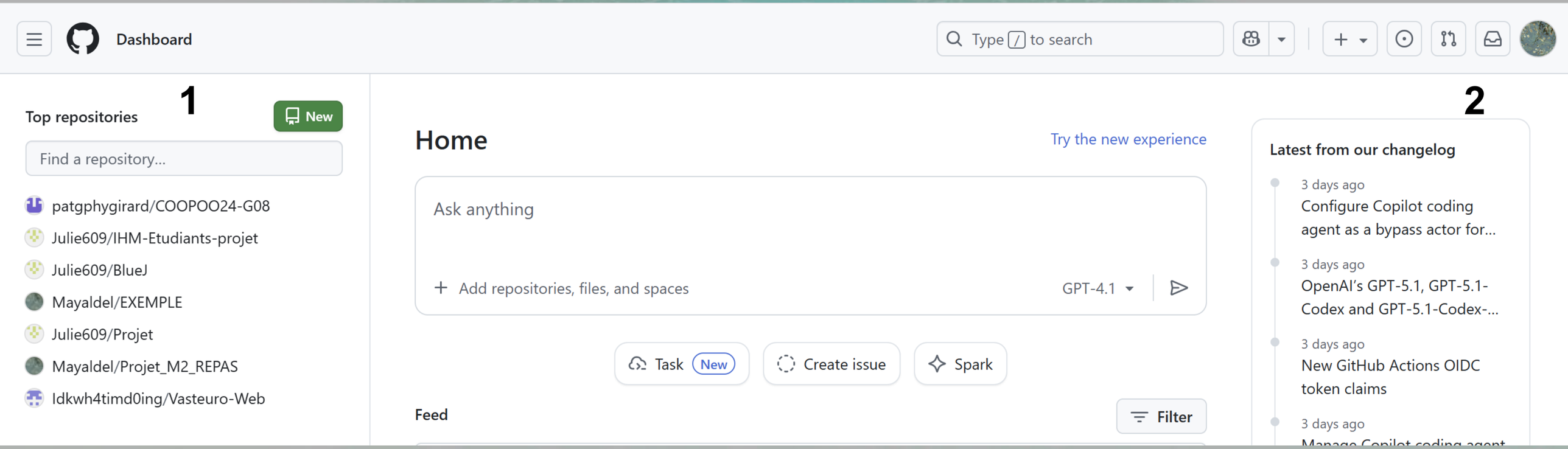
# GIT HUB

Git Hub = Git en ligne + pas de ligne de code

# GIT HUB

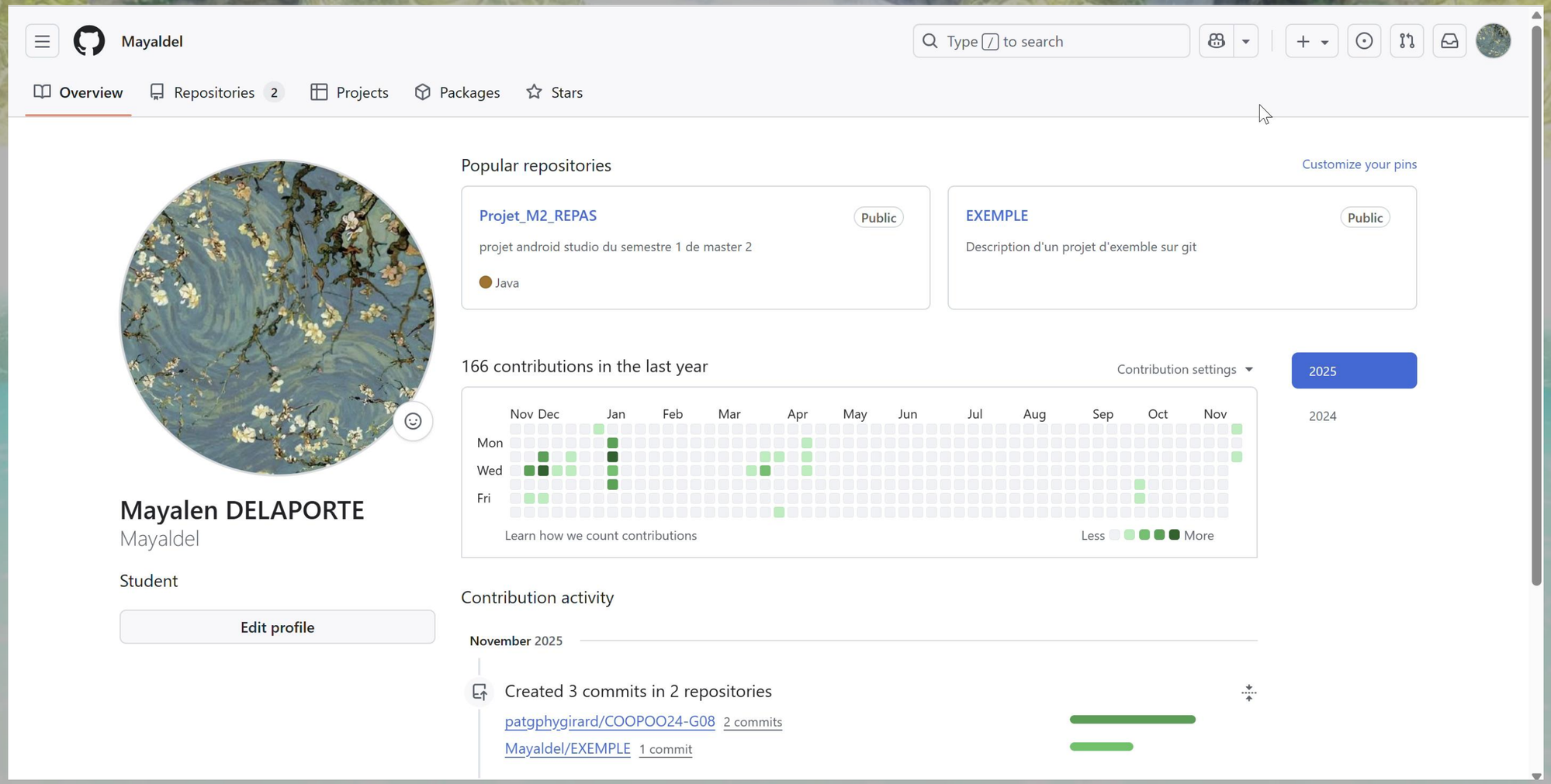
Echange à distance facilité
Possibilité d'utiliser les interfaces
Partage de projets au sein de diverses applications

# Compte


Dashboard
Type / to search
1
Top repositories
New
Find a repository...
patgphygirard/COOPOO24-G08
Julie609/IHM-Etudiants-projet
Julie609/BlueJ
Mayaldel/EXEMPLE
Julie609/Projet
Mayaldel/Projet_M2_REPAS
Idkwh4timd0ing/Vasteuro-Web
Home
Try the new experience
Ask anything
Add repositories, files, and spaces
GPT-4.1
Task
New
Create issue
Spark
Feed
Filter
2
Latest from our changelog
3 days ago
Configure Copilot coding agent as a bypass actor for...
3 days ago
OpenAI's GPT-5.1, GPT-5.1-Codex and GPT-5.1-Codex-...
3 days ago
New GitHub Actions OIDC token claims
3 days ago

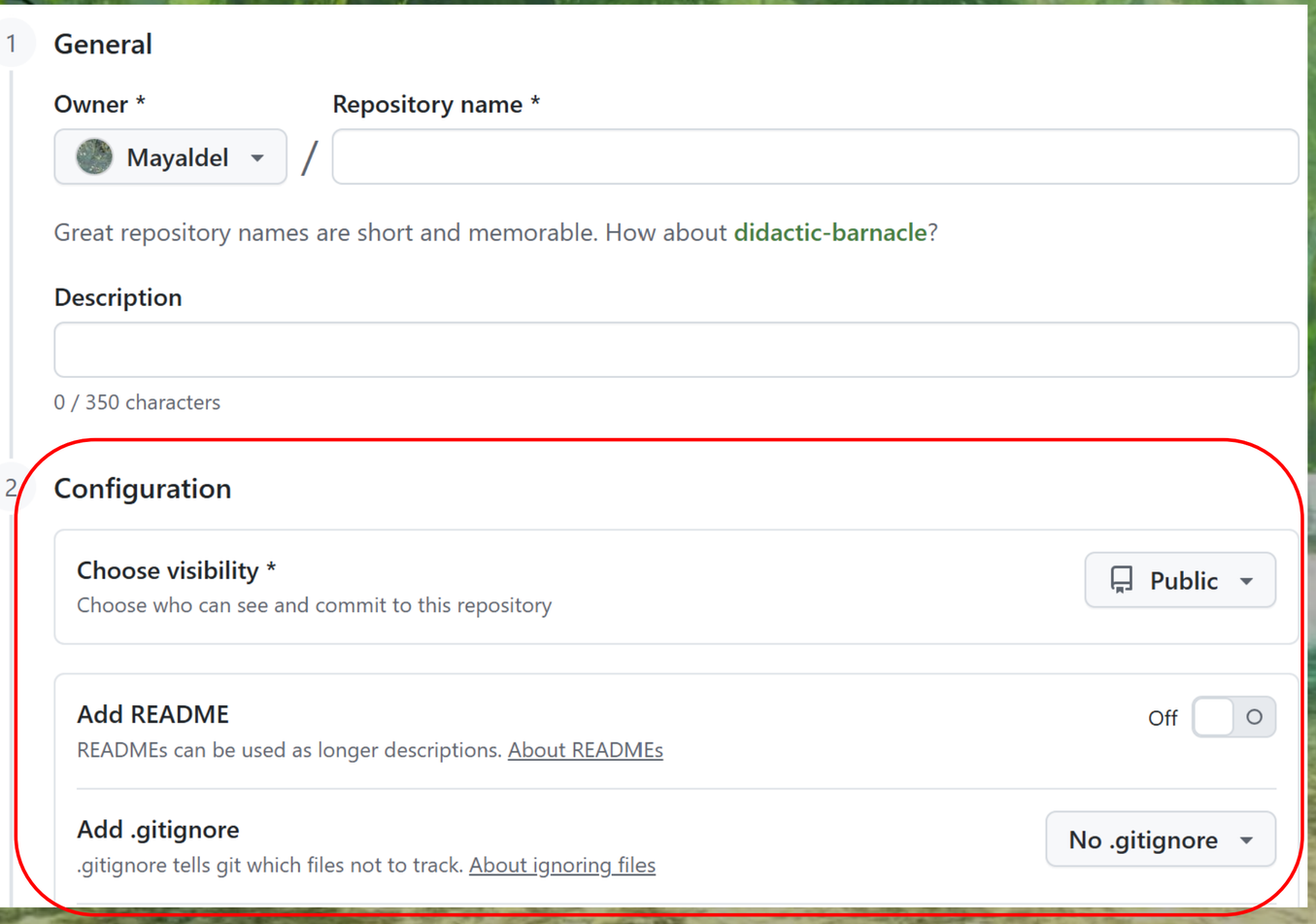
# GIT HUB

Création d'un compte selon les instructions du site
Sur la page "Dashboard" possibilité de voir et crée des projet "repository"
Mais aussi d'accéder au profil utilisateur




Sauvegardez votre token quelque part !

# CREER UN PROJET




**1.Public / Privé**

Permettre un partage sécurisé
Permette un libre accès
Ajout de membre

**2.README**

Presque toujours présent
Documentation de votre projet

**3.Git.ignore**

Permet de sélectionner des fichiers qui ne seront pas transmit

# Creer un projet

## STUDENT

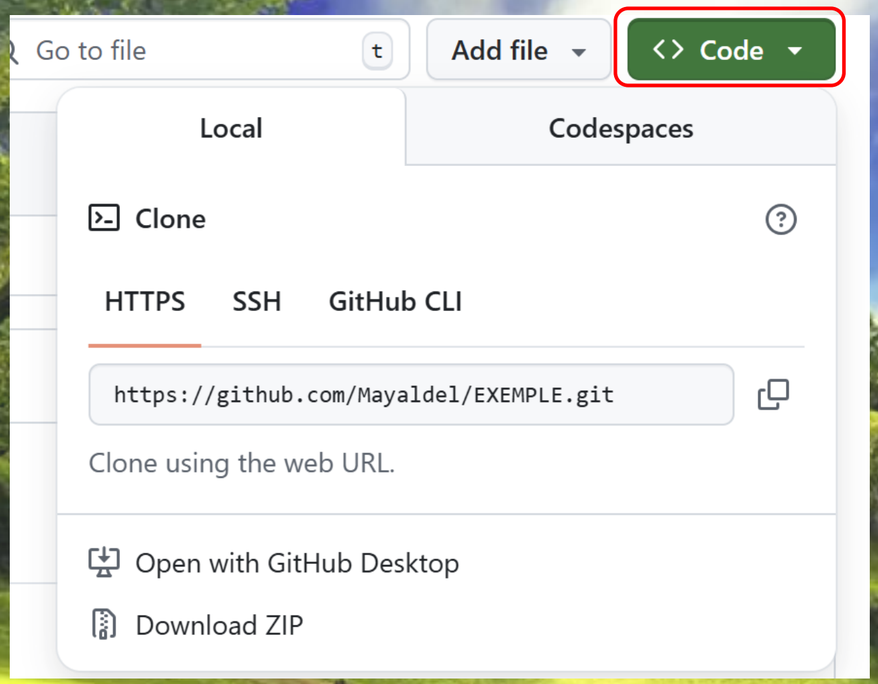
Une fois le dossier créer :

Utilisation des lignes de commandes git (init=> add . => commit=> push) pour avoir tous les éléments

OU

Se rendre dans le projet git hub et importer les fichiers

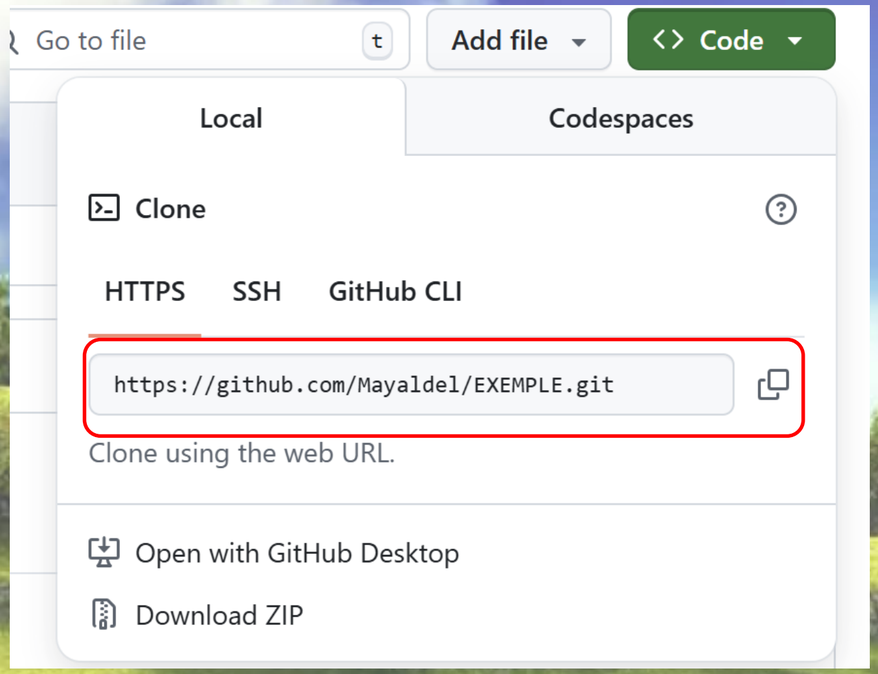
# Recuperer un projet




**1.<>Code**

Permet de récupérer le projet sous différents formats

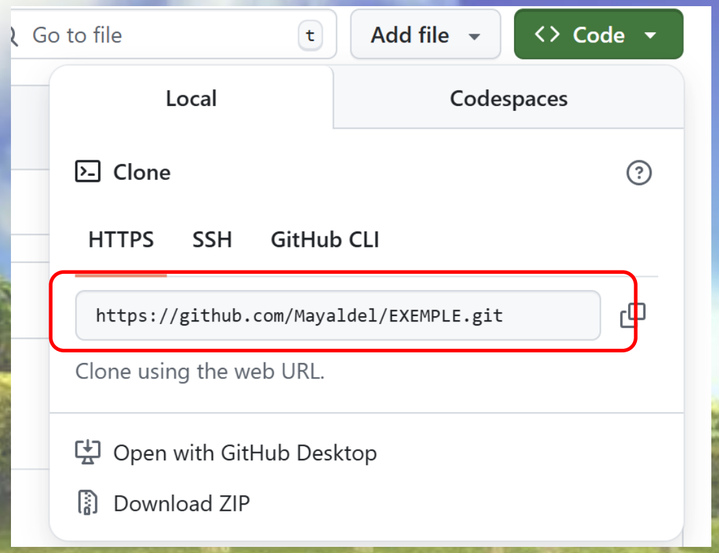
# Recuperer un projet




## 2.URL

Utiliser pour réaliser le Git clone URL afin de récupérer le projet sur votre ordi (depuis terminal ordinateur ou terminal application)

# Recuperer un projet




## 3.Visual studio code

```
PS C:\Users\mayal> git clone https://github.com/Mayaldel/EXEMPLE.git
Cloning into 'EXEMPLE'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\mayal>
```

# STUDENT

En Binôme :

Créer un projet partager et le récupérer sur votre ordinateur

# Exercice Guide

**A: Ajouter un nouvel élément au projet.**

Ouvrir avec VSCode ou dans votre dossier
Ajouter un nouveau fichier avec un nom adapté

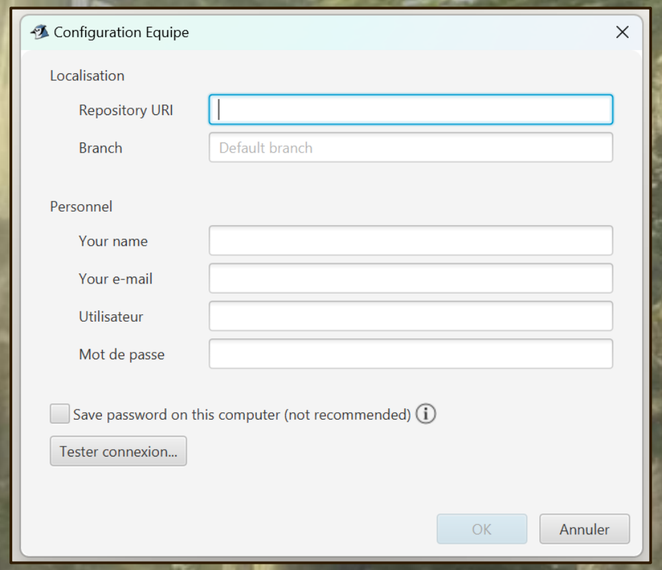
**B:enregistrement et envoie à partir du teminal**

```
git add NOM
git pull
git commit –m MESSAGE
git push
```

**C:Vérification**

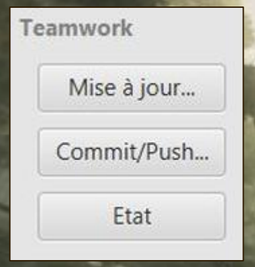
Local ou Git hub

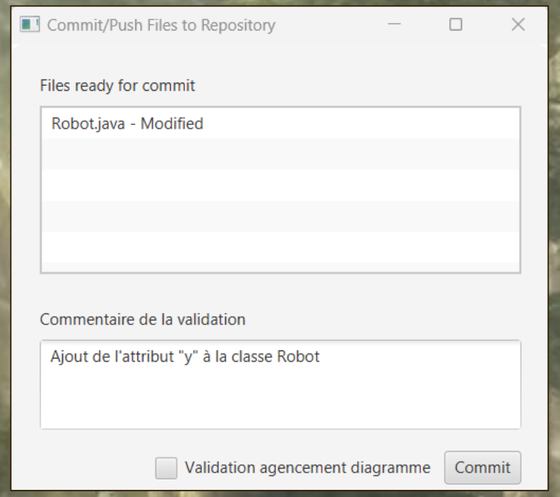
# Blue J

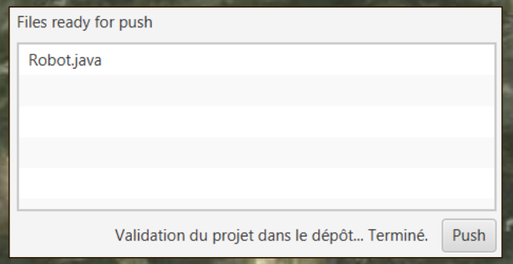



Coller l'URL HTTPS dans la barre Repository URI
Remplir :
nom (Your name),
e-mail (Your e-mail),
pseudo (Utilisateur)
token (Mot de passe)

# Blue J

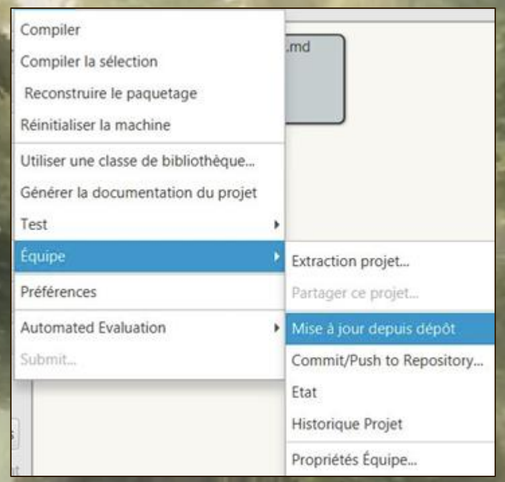


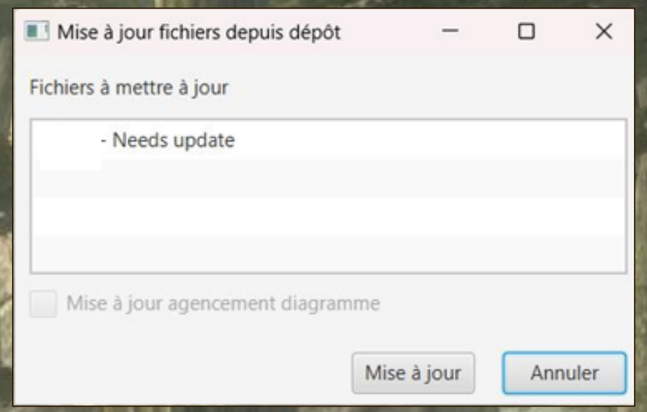




Ecrire un commentaire (= git commit –m)
Appuyer sur Commit
Appuyer sur Push

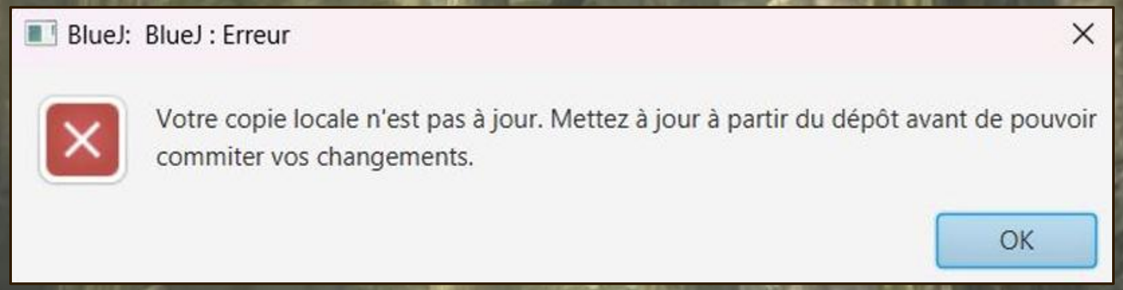
# Blue J



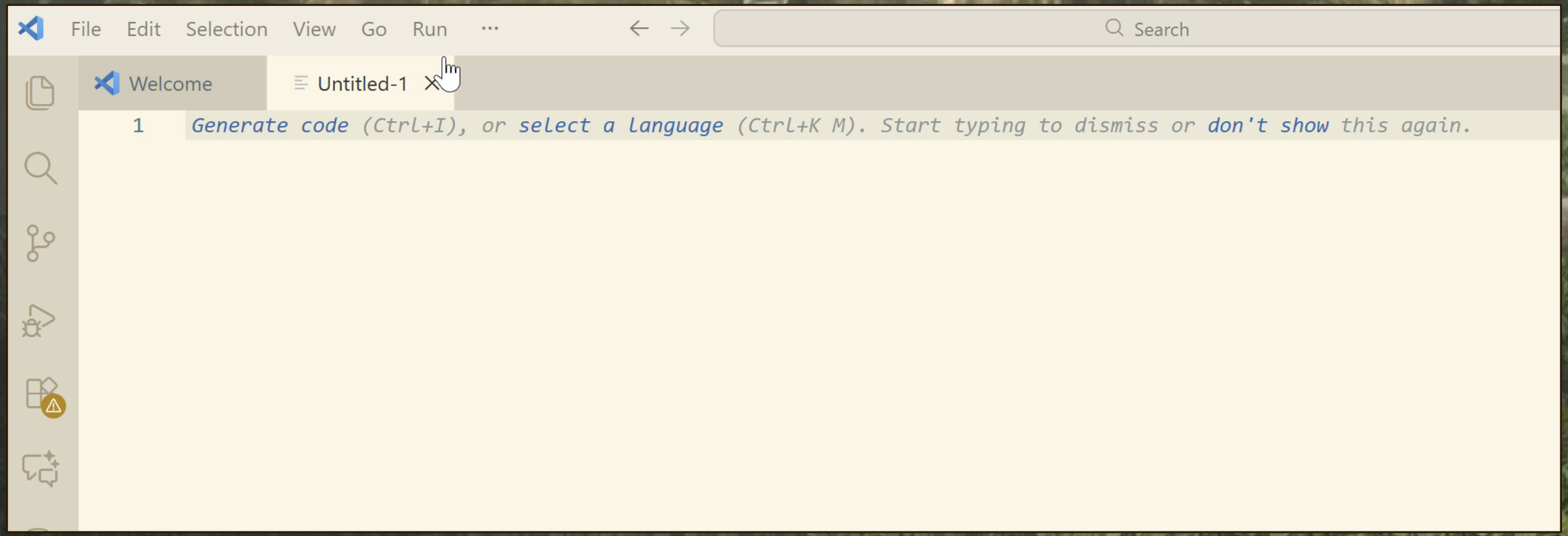

Cliquer sur Mise à jour pour récupérer la classe

# Blue J




Corriger les erreurs (conserver une des deux variables)
Supprimer les marqueurs :
Marqueur de début de conflit : <<<<<<< HEAD,
Séparateur de conflit : =======,
Marqueur de fin de conflit : >>>>>>>
nom_branche

# VS Code


File
Edit
Selection
View
Go
Run
Search
Welcome
Untitled-1
1
Generate code (Ctrl+I), or select a language (Ctrl+K M). Start typing to dismiss or don't show this again.

# VS Code

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL HISTORY TASK MONITOR

```
PS C:\Users\mayal> git clone https://github.com/Mayaldel/Projet_M2_REPAS.git
Cloning into 'Projet_M2_REPAS'...
remote: Enumerating objects: 84, done.
remote: Counting objects: 100% (84/84), done.
remote: Compressing objects: 100% (66/66), done.
remote: Total 84 (delta 3), reused 84 (delta 3), pack-reused 0 (from 0)
Receiving objects: 100% (84/84), 236.06 KiB | 2.17 MiB/s, done.
Resolving deltas: 100% (3/3), done.
PS C:\Users\mayal>
```

File Edit Selection View Go Run ···

COOPOO24-G08

EXPLORER

OPEN EDITORS 1 unsaved

TESTGIT U

COOPOO24-G08

pieces-Photoroom.png
Pieces.java
PiecesTest.java
placeholder.jpg
Pnj.java
PnjTest.java
Question.java
README.TXT
Red-bulle (1)-Photoroom.png
Red-bulle.PNG
RedBulle.java
RedBulleTest.java
Robot-Disco.png
RobotDanse.java
Room.java
RoomTest.java
Shoe.java
Shoes.png
ShoeTest.java
Shop.java
ShopTest.java
small-world.wav
Stand_nourriture.png
Stuff.java
StuffTest.java
TESTGIT U
TimeMachine.java
TimeMachineTest.java
vision.pnj
Voyage_dans_letemps.png

OUTLINE

TIMELINE

TESTGIT U

TESTGIT

```
1   exemple de ficier pour un git sur vscode
```

CHAT

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

SUGGESTED ACTIONS

Build Workspace Show Config

TESTGIT

Describe what to build next

Agent Pick Model

master* 0 0 Ln 1, Col 41 Spaces: 4 UTF-8 CRLF { } Plain Text Go Live

# Questions

Noëmie UGOLIN :
ugolin.noemie@gmail.com

Mayalen DELAPORTE :
delaporte.mayalen@gmail.com

## Bonus

Dépôt git : https://github.com/Noemie-U/Formation_git_support.git

- Diaporama
- Support Rédigé
- Memento des lignes de commandes