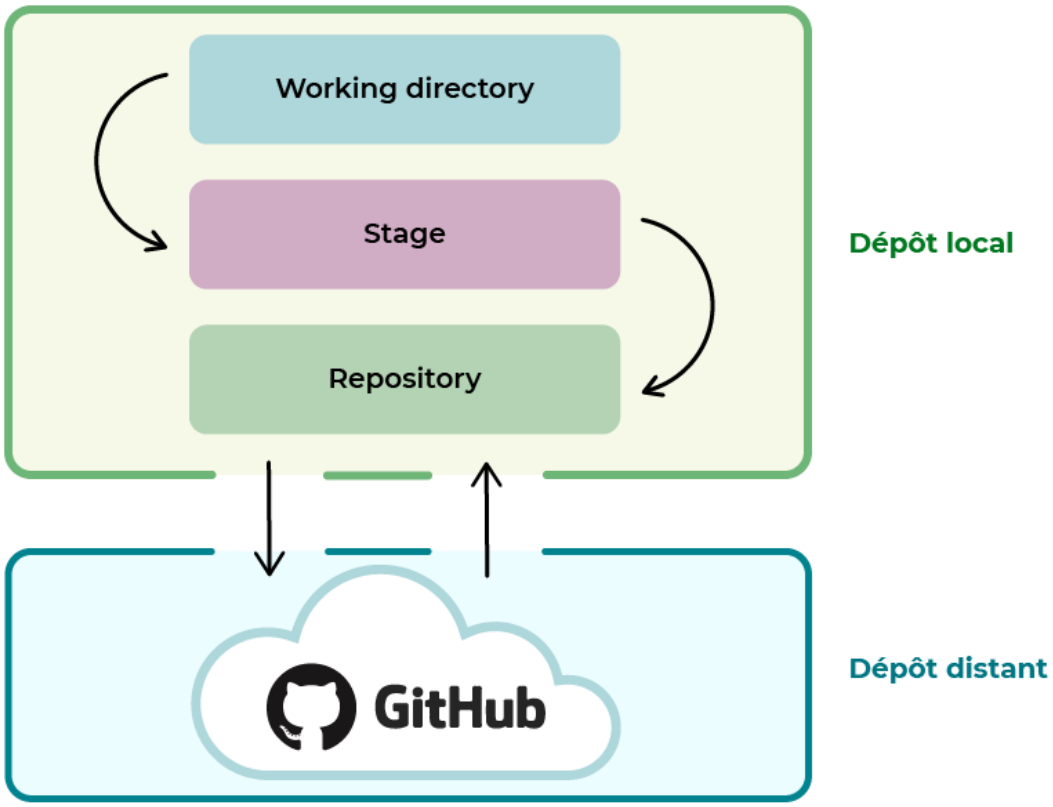
TP2

Travaillez depuis votre dépôt local Git

Le schéma ci-dessous explique le fonctionnement global de Git.



Ce schéma représente le fonctionnement de Git. Il est composé de 3 zones qui forment le dépôt local, et du dépôt distant GitHub.

Regardons plus en détail les différentes zones du dépôt local.

Le Working directory

Cette zone correspond au dossier du projet sur votre ordinateur.

Souvenez-vous, dans la partie précédente nous avons initialisé le dépôt “PremierProjet”. Eh bien ce dépôt, c’est la zone bleue du schéma.

Le Stage ou index

Cette zone est un intermédiaire entre le working directory et le repository. Elle représente tous les fichiers modifiés que vous souhaitez voir apparaître dans votre prochaine version de code.

Le Repository

Lorsque l’on crée de nouvelles versions d’un projet, c’est dans cette zone qu’elles sont stockées.

Ces 3 zones sont donc présentes dans votre ordinateur, en local.

En-dessous, vous trouvez le repository GitHub, c'est-à-dire votre dépôt distant.

Alors comment ça marche ?

Imaginez un projet composé de 3 fichiers : fichier1, fichier2 et fichier3.

Nous faisons une modification sur fichier1, puis une modification sur fichier2, depuis le working directory. Nous avons maintenant une version évoluée de notre projet.

Nous aimerions sauvegarder cette version grâce à Git, c'est-à-dire la stocker dans le repository.

Comment faire ?

Pour commencer, nous allons envoyer les fichiers modifiés (fichier1 et fichier2) du working directory vers l'index. **On dit qu'on va indexer** fichier1 et fichier2. Une fois les fichiers indexés, nous pouvons créer une nouvelle version de notre projet.

Le terme "stage" est aussi beaucoup utilisé par les développeurs à la place du terme "index". On peut dire "indexer un fichier" ou "stage un fichier".

Exemple :
Nous allons mettre en place une base de projet web, avec un fichier HTML et un fichier CSS pour mettre tout cela en pratique.

1) Initialisez un dépôt :

Allez dans votre dépôt Git

Avant tout, vous devez vous situer dans le dépôt Git du projet.

Pour vous en assurer, tapez la commande "pwd" dans Git Bash.

```
c:/users/JohnDoe/Documents/PremierProjet/
```

c'est que vous êtes bien situé ! Sinon, il suffit d'utiliser la commande "cd" pour vous replacer dans le dossier/dépôt Git du projet.

Ensuite, initialisez un dépôt en vous basant sur l'exemple "PremierProjet" du chapitre précédent.

2) Créer les premiers fichiers

- Dans notre dossier "PremierProjet", créez un fichier "index.html" avec ce contenu :

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <link rel="stylesheet" type="text/css" href="styles.css">
</head>
```

```
<body>
  <h1>Un super titre</h1>
</body>
</html>
```

- Puis, créez un fichier “styles.css” avec ce contenu :

```
h1 {
  color: red;
}
```

3) Maintenant que vous avez une première base pour travailler sur votre projet web, créez une version du projet dans son état actuel :

Indexez vos fichiers avec la commande git add

Vous avez créé 2 fichiers, index.html et styles.css. Ces fichiers sont situés dans le dépôt Git. Pour créer une nouvelle version de votre projet, vous allez maintenant indexer les fichiers. Pour cela, retournez dans l’outil “Git Bash”.

Vous pouvez maintenant faire passer les fichiers index.html et styles.css vers l’index en utilisant la commande “git add” suivie du nom du fichier :

```
$ git add index.html styles.css
```

Voilà, vous avez indexé vos deux fichiers !

Créez une nouvelle version avec la commande git commit

Maintenant que vos fichiers modifiés sont indexés, vous pouvez créer une version, c'est-à-dire archiver le projet en l’état. Pour ce faire, utilisez la commande "git commit" :

```
git commit -m “Ajout des fichiers html et css de base”
```

-m (comme message) est ce qu'on appelle un **argument**, qui est ajouté à la commande principale. Ici "-m" permet de définir un message particulier rattaché au commit effectué. Si vous n’utilisez pas cet argument, la commande “git commit” ouvrira un éditeur de texte dans lequel vous pourrez saisir le message de commit.
“Ajout des fichiers html et css de base” est le message rattaché au commit grâce à l'argument "-m".

Vos modifications sont maintenant enregistrées avec la description “Ajout des fichiers html et css de base”.

La description est très importante pour retrouver le fil de vos commits, et revenir sur un commit en particulier. Ne la négligez pas ! Il ne serait pas pratique de nommer des commits du type "version 1", "version 2", "version 2.1", car vous seriez alors obligé de lire le fichier pour connaître les modifications réalisées. Avec un message clair, vous pouvez tout de suite identifier les modifications effectuées.

Et voilà! Vous venez de créer une version de votre projet !

Envoyez votre commit sur le dépôt distant avec la commande git push

Créer une version de votre projet, c'est super, mais il faut désormais passer votre commit du repository à votre dépôt distant. On dit qu'il faut "pusher" votre commit.

Votre premier push va vous demander un peu de configuration.

Pour commencer, vous allez devoir "reliez" votre dépôt local au dépôt distant que vous avez créé sur GitHub précédemment. Pour cela :

- Allez sur GitHub.
- Cliquez sur la petite image en haut à droite.
- Cliquez sur "your repositories".
- Cliquez sur le repository créé dans la première partie du cours, "Project1".

Il va falloir copier le lien du repository.

Maintenant, retournez sur Git Bash et tapez la commande suivante :

```
$ git remote add origin https://github.com/Project1.git
```

Vous l'aurez compris, le lien à saisir est le lien que vous venez de copier.

Ensuite, tapez la commande :

```
git branch -M main
```

Ça y est ! Vous avez relié le dépôt local au dépôt distant. Vous pouvez donc envoyer des commits du repository vers le dépôt distant GitHub en utilisant la commande suivante :

```
git push -u origin main
```

La version du projet est maintenant stockée dans le cloud !

Reprenez le fichier HTML créé. Disons que vous allez modifier le titre "Un super titre" en "Un super titre !".

Pour créer une version du projet avec le fichier HTML à jour et l'envoyer sur le dépôt distant GitHub, vous allez :

- Indexer le fichier HTML modifié grâce à la commande
  ```
  git add index.html
  ```
- Créer une nouvelle version grâce à la commande :
  ```
  git commit -m "Modification du titre H1"
  ```
- Envoyer la nouvelle version sur le dépôt distant grâce à la commande :
  ```
  git push origin main
  ```

En résumé

- git add permet d'ajouter des fichiers dans l'index, qui est une zone intermédiaire dans laquelle stocker les fichiers modifiés.
- git commit permet de créer une nouvelle version avec les fichiers situés dans l'index.
- git commit -m permet de créer une nouvelle version et de préciser le message rattaché au commit.
- git push permet d'envoyer les modifications faites en local vers un dépôt distant.