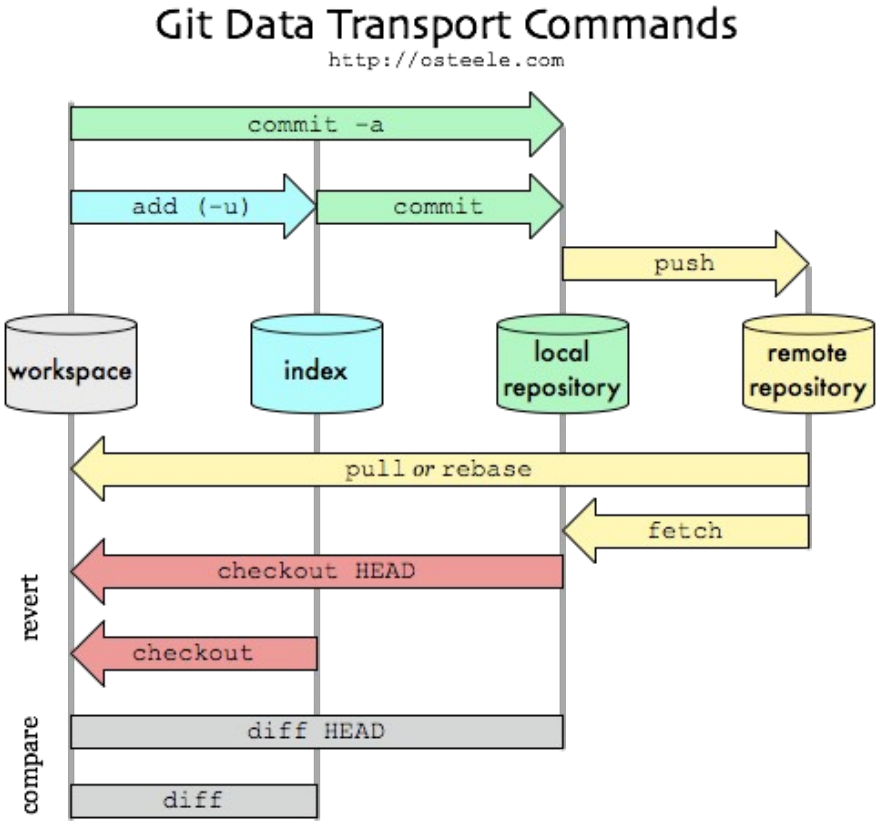
GIT

Dépôts distants



Avantages à utiliser GIT plutôt que le FTP.

Pas de fichiers oubliés

Possibilité de revenir en arrière

Gestion des conflits

Dépot distant sur un serveur dédié

Il est possible d'avoir son dépôt distant sur un serveur dédié, une instance Amazon EC2, etc.

Les plateformes

L'utilisation d'une plateforme pour votre dépôt distant est très fréquent. Elles offrent de nombreux services.

- Bitbucket
- GitHub
- GitLab

Cloner un dépôt distant

Travaux pratiques

1. Créer un répertoire vide
2. et lancer la commande :

```
git clone git@bitbucket.org:Sileax/jeu-de-morpion.git
```

pour cloner le dépôt https://bitbucket.org/Sileax/jeu-de-morpion/src/master/

Visualiser les dépôts distants

Un dépôt local peut être associé à plusieurs dépots distants. Pour avoir la liste:

```
git remote show
```

Le nom du dépôt distant est « origin » par défaut. Pour avoir plus de détails :

```
git remote show origin
```

Création de dépôts distants

Créer un compte sur github ou bitbucket
Créer un dépôt **vide**

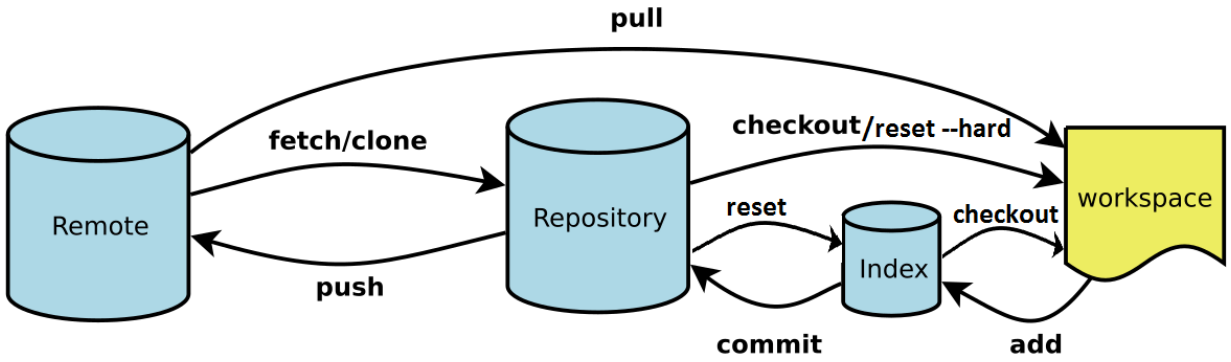
Dans les options : dépôt public. + Pas de fichier dans le dépôt.

Il est possible de créer des dépôts privé. Dans ce cas, il faudra configurer des accès utilisateur le cas échéant.

Dépôt publique / privé

ajouter des collaborateurs sur un dépôt privé

Interagir avec un dépôt distant



Pousser

1. On dispose d'un dépôt vide sur GitHub ou Bitbucket et d'un code versionné en local
2. Lier ce dépôt :

Pour lier le dépôt local au dépôt distant, on a besoin de l'adresse du dépôt distant. Celle-ci est fourni dans 2 protocoles https ou ssh

```
git remote add origin <url>
git remote show
```

3. pousser des modifications :

```
    git push origin
```

Constater que votre dépôt contient bien votre code

Tirer

Objectif : Travail en binôme pour récupérer le code du dépôt du User 1

Prérequis :

- Chacun doit avoir pousser dans son dépôt le code permettant de faire l'upload de fichier.
- Vérifier que :
  - le dossier uploads est bien présent et vide.
  - Il n'y a pas de fichier/dossier en trop.

USER 2:

Créer un répertoire local vide et l'initialiser avec git init

Cloner le dépôt du USER 1

USER 1 :

Pousser des modifications sur le dépôt

USER 2 :

Mettre à jour le répertoire de travail :

```
git pull origin master
```

Inverser les rôles

Gestion des conflits

Historique du dépôt distant

La commande git log indique l'historique du dépôt local. Elle peut indiquer l'historique d'un dépôt distant :

```
git fetch
git log origin/master
```

Pour aller plus loin :

https://rakhesh.com/coding/git-view-the-commit-log-of-a-remote-branch/

Résolution de conflits

Dépôt local en avance sur le distant = Pas de conflits
Dépôt distant en avance sur le local = Conflits potentiels

Exercice en binôme

Prérequis :

Chacun doit avoir pousser dans un nouveau dépôt un code php permettant d'afficher un message simple.

User 1 :

Pousse une modification sur un fichier

User 2 :

Essaye de pousser une modification sur le même fichier

Constate qu'il est en retard :

```
git fetch :
git log origin/master
```

Tire les modifications et prend connaissance des fichiers concernés

```
git pull
```

Il résout les conflits entre :

- la version locale HEAD
- et la version distante représentée par le hash du commit

Puis ajoute au stage et commit

User 1 :

Tire les modifications

Cas particuliers

Pousser des tags

Les tags ne sont pas poussés implicitement.

```
git push origin tags
```

Fichiers ignorés

Rappel de la syntaxe

Comment supprimer des fichiers déjà commités

Dossier Vendor

Le dossier vendor doit être ignoré. Il sera créé par la commande composer install. La commande composer install va lire le fichier composer.lock pour alimenter le dossier vendor.

Exercice en binome

Un des membre du binome initialise un projet avec composer et installe la librairie Monolog. Il pousse l'application dans un depôt distant. L'autre membre récupère le dépôt et lancer la commande

```
composer install
```

Mots de passes de connexion à la Base de données

Les mots de passe sont confidentiels et ne doivent pas être protégés.

Exercice à terminer pour la séance suivante

A partir d'un code précédent (exemple : ajout d'email dans une BDD), trouver une solution pour ne pas versionner les paramètres de connexion à la base de données.
Pousser le code dans un dépôt

Résolution des problèmes de connexion

Il y a deux méthodes de connexion à votre dépôt distant :

- en renseignant manuellement votre login et votre mot de passe
- automatiquement, en utilisant un jeu de clef ssh

On privilégie la seconde méthode.
Cette méthode nécessite de

1. créer un jeu de clef publique/privée
2. déposer la clef publique sur Github/Gitlab/etc…
3. configurer votre machine pour utiliser la clef privé pour vous connecter au dépôt

Ressources

Github : https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

GitLab :

https://docs.gitlab.com/ee/user/ssh.html

BitBucket

https://support.atlassian.com/bitbucket-cloud/docs/set-up-personal-ssh-keys-on-windows/

Un peu de théorie

Les clefs asymétriques est un système qui offre 2 possibilités d'utilisation.

- Lorsque le message est chiffré avec une clef publique, seul le détenteur de la clef privée pourra le déchiffrer => CONFIDENTIALITE

- Lorsque le message est chiffré avec une clef privée, les détenteurs de la clef publique sauront que l'émetteur est bien celui qu'il prétent être => AUTHENTIFICATION

Pour aller plus loin :

https://blog.devensys.com/notions-sur-le-chiffrement/

https://openclassrooms.com/fr/courses/1757741-securisez-vos-donnees-avec-la-cryptographie/6031872-utilisez-le-chiffrement-asymetrique