

Qualité de développement

TD1 : Utilisation de Git

Objectif : Suivi de votre répertoire **TypeScript** par Git

Remarque : le répertoire **TypeScript** est le répertoire que vous avez créé dans le cours de Développement Objet. Il contient les fichiers de configuration pour vos codes TypeScript et les fichiers sources. Si vous ne l'avez pas appelé **TypeScript**, vous pouvez le renommer.

Le but est de mettre en place un suivi de version de votre répertoire **TypeScript** par Git et de le synchroniser sur un dépôt distant. Ce répertoire sera utilisé pour les TDs de ce semestre de Développement Objet et de Qualité de Développement. La plupart des rendus dans ces matières devront se faire via ce dépôt.

Remarque : si vous n'avez pas encore créé votre répertoire **TypeScript**, vous pouvez le faire en suivant les instructions du cours de Développement Objet.

Configuration globale de git

Configurer votre nom et votre adresse mail pour Git. Pour cela, ouvrez un terminal et tapez les commandes suivantes :

```
git config --global user.name "votre nom"
git config --global user.email "votre adresse mail"
```

Ne pas avoir à saisir son mot de passe à chaque fois

Pour ne pas avoir à saisir votre mot de passe à chaque fois que vous souhaitez synchroniser votre dépôt local avec le dépôt distant, vous pouvez utiliser un token d'accès.

1. Rendez-vous dans votre profil GitLab et cliquez sur **Settings** puis **Access Tokens**.
2. Créez un token d'accès avec les droits : **api**, **read_api**, **read_user**, **read_repository**, **write_repository**.
3. Copiez le token d'accès qui vous est fourni.
4. Lorsqu'on vous demandera vos identifiants, saisissez votre identifiant GitLab et collez le token d'accès dans le champ du mot de passe (vous n'avez à le faire qu'une seule fois).

Vous pouvez ensuite utiliser ce token pour vous connecter au serveur GitLab en utilisant la commande **git config --global credential.helper store**.

Mise en place d'un dépôt local

1. Rendez-vous dans votre répertoire `TypeScript`.
2. Créer un répertoire `qualdev` dans ce répertoire.
3. Créer un fichier `compte_rendu.txt` dans ce répertoire `qualdev`. **Consigne** : vous noterez dans ce fichier l'ensemble des commandes git que vous utiliserez dans ce td. Vous indiquerez aussi pour chaque commande ce qu'elle fait sans paraphraser la commande ainsi que les réponses aux questions posées.
4. Initialisez un dépôt local Git dans le répertoire `TypeScript`.
5. Comment connaître les fichiers / répertoires présents dans le dossier de travail non suivis par Git ?
6. Faites en sorte que le dossier `.vscode` ne soit pas considéré par Git. Comment vérifier qu'il n'est plus considéré par Git ?
7. Indexer les autres fichiers et répertoires pour qu'ils soient suivis par Git.
8. Comment vérifier que les fichiers / répertoires sont bien indexés ?
9. Faites un premier commit avec un message explicite.
10. Comment vérifier que le commit a bien été effectué et que le dépôt local est bien à jour ?

Mise en place d'un dépôt distant

1. Connectez-vous à <https://gitlab.univ-lorraine.fr> avec vos identifiants habituels.

Rappel : possibilité de créer un `Access Token`, décrite page précédente pour ne pas avoir à saisir vos identifiants à chaque fois.

2. Créer un nouveau projet privé nommé `typescript_nom_prénom` (remplacer nom_prénom par votre nom et prénom). Vous créerez ce projet à partir du modèle `Blank project`. Le projet doit être privé. Vous devez aussi décocher la case `Initialiser le répertoire avec un README` (sinon vous pourriez avoir des problèmes lors de la synchronisation avec le dépôt local).
3. Invitez votre enseignant à collaborer à ce projet avec au minimum le rôle `Developer`.
4. Lier votre dépôt local à ce dépôt distant.
5. Comment vérifier que le dépôt local est bien lié au dépôt distant ?
6. Mettre à jour le dépôt distant avec les modifications locales.
7. Comment vérifier que le dépôt distant et le dépôt local sont synchronisés ?

Pour quelques commits de plus

Normalement, vous avez modifié votre fichier `compte_rendu.txt` suite à la question 7 ci-dessus.

1. Vérifiez avec `git status` que cette modification n'est pas prise en compte. Si votre dépôt est à jour, modifiez le fichier `compte_rendu.txt`.
2. Faites un commit du dépôt local avec un message explicite.
3. Si vous avez bien structuré votre code en développement objet, vous devez avoir un répertoire `src`, avec des fichiers et/ou des répertoires correspondant aux deux premiers tds. Ajoutez-y un fichier

`helloworld.ts` et un fichier `io.ts`. Indexez `src`, mettez à jour votre dépôt local et poussez la modification sur le dépôt distant.

4. Faites en sorte d'effacer le fichier `helloworld.ts`, et que cette suppression soit prise en compte par git.
5. Faites en sorte de renommer un `io.ts` en `entrees-sorties.ts` et que cette modification soit prise en compte par git.
6. Faites un commit du dépôt local avec un message explicite.
7. Comment vérifier que le dépôt distant est bien en retard sur le dépôt local ?
8. Synchronisez le dépôt distant avec le dépôt local.

Fin du TD

1. Quand vous avez fini d'écrire votre fichier `compte-rendu.txt` faites un commit du dépôt local avec un message explicite et publiez sur le dépôt distant.
2. Vous avez fait un certain nombre de commits durant le TD et vous allez en faire d'autres dans les cours à venir. Pour aider votre enseignant à retrouver le rendu final de `compte_rendu.txt`, on va associer au commit fait précédemment un **tag**. Un tag est un pointeur vers un commit. Il est utilisé pour marquer des versions importantes. Vous pouvez associer un tag au commit actuel qui est votre rendu de la manière suivant :

```
git tag -a v1.0 -m "Rendu final du compte-rendu"
```

Il faut pousser le tag sur le dépôt distant avec la commande suivante :

```
git push origin --tags
```

Evaluation

Vous serez évalués sur le contenu de votre fichier `compte_rendu.txt` présent sur votre dépôt distant (et associé au tag défini ci-dessus) pour le début de la prochaine séance d'EI (à remettre sur gitlab le 10 mars au plus tard).