TP 03 - Utilisation de Git

Mies en place de Git

Pour travailler en équipe, il est pratique d'utiliser un serveur Git.

Cette partie vous indique comment faire pour utiliser la plateforme GitHub avec Eclipse.

Création d'un compte GitHub

À faire par **chacun** des membres de l'équipe

Création du compte GitHub

Chaque membre de l'équipe doit avoir créé un compte sur GitHub : https://github.com/

Si vous n'avez pas encore de compte GitHub, suivez les instructions données sur le site. (bouton "Sign up")

Création d'un token

Une fois que le compte est créé et activé, il faut créer un **Personal Access Token**.

En effet, depuis août 2021, ce n'est plus le mot de passe qui doit être utilisé lorsqu'on effectue des transferts de fichiers depuis ou vers un dépôt GitHub. Il faut utiliser un token qui est une sorte de mot de passe spécialisé et à durée limitée.

- Dans GitHub, clic sur l'icône en haut et à droite, puis "Settings" (en bas).
- À gauche, en bas, clic sur "Developer Settings"
- À gauche, tout en bas, clic sur "Personal access tokens"
- À droite, bouton "Generate new token"
- Dans le champ "Note", indiquez : **Projet Java EE**
- Dans la section "Select Scopes", cochez ☑ **repo**
- Bouton "Generate token" (tout en bas)
- Le token est la longue chaîne de caractères qui est affichée. C'est cette chaîne de caractères qui devra être utilisée comme mot de passe dans Eclipse.
  Il faut immédiatement faire une copie du token et l'enregistrer quelque part, car il ne sera plus accessible ensuite. Si on le perd, il faut alors générer un nouveau token.

Création d'un dépôt distant (repository)

À faire par **un seul** membre de l'équipe

Le plus simple est que l'un des membres de l'équipe crée un repository avec son compte GuitHub, puis qu'il invite l'autre membre à collaborer sur le projet.

Création du dépôt

- Dans GitHub, bouton "New" pour créer un nouveau repository
- Indiquer le nom du dépôt. Par exemple : **projet-jee**
- Choisir si vous souhaitez qu'il soit public ou privé
- Bouton "Create Repository"

Invitation de l'autre membre

- Lorsqu'on est sur la page du dépôt **projet-jee**, cliquer sur "Settings" (à droite).
- À gauche, cliquer sur "Manage Access".
- En bas, à droite, bouton "Add people"
- Indiquer le pseudo ou l'adresse e-mail de la personne à inviter.

Acceptation de l'invitation

- La personne invitée reçoit un e-mail qui contient un lien sur lequel elle doit cliquer (bouton "View invitation").
- Sur le site GitHub : bouton "Accept invitation"

Utilisation de Git avec Eclipse

À faire par **chacun** des membres de l'équipe

Configuration d'Eclipse pour Git

- Menu "Window > Preferences > Version Control (Team) > Git"
- À côté du champ "Default Repository Folder", cliquez sur le bouton "Variable...".
- Sélectionnez **workspace_loc** et appuyez sur le bouton "OK".
  Cela permettra à Git de considérer votre workspace comme le lieu de stockage des dépôts locaux.
- Dans la liste de gauche, dépliez l'entrée "Git" et sélectionnez "Configuration".
- Appuyez une première fois sur le bouton "Add Entry...". Puis, indiquez :
  - Dans le champ "Key", la valeur : **`user.name`**
  - Dans le champ "Value" : *votre prénom suivi de votre nom*

  Validez, en appuyant sur le bouton "OK".
- Appuyez une deuxième fois sur le bouton "Add Entry...". Puis, indiquez :
  - Dans le champ "Key", la valeur : **`user.email`**
  - Dans le champ "Value" : *votre adresse e-mail* **xxxxxx@3il.fr**

  Validez, en appuyant sur le bouton "OK".
- Une entrée **user** a dû apparaître dans la partie centrale de la vue.
  Si vous la dépliez, vous devez y voir les 2 paramètres **email** et **name** que vous venez d'enregistrer.
- Appuyez sur le bouton "Apply and Close" pour valider et revenir à l'espace de travail d'Eclipse.

Perspective Git

Pour utiliser Git à partir d'Eclipse, il faut ouvrir la perspective Git.

- En haut et à droite de l'espace de travail d'Eclipse, se trouvent les icônes qui permettent de changer de perspective.
  Cliquez sur l'icône "Open Perspective" qui se trouve à cet endroit.
- Sélectionnez "Git" et appuyez sur le bouton "Open".
- Pour changer de perspective, il suffit de cliquer sur l'icône de la perspective que l'on veut ouvrir. Vérifiez que vous pouvez facilement revenir à la perspective Java EE.
  Puis, terminez en passant dans la perspective Git.
- On peut aussi avoir 2 fenêtres Eclipse ouvertes simultanément :
  Menu "Window > New Window"
  Il suffit alors d'avoir une fenêtre avec la perspective Java EE et l'autre avec la perspective Git. Ensuite, on peut alterner entre les deux fenêtres.

Lorsque vous serez en train de développer (c'est-à-dire lorsque vous écrirez du code Java ou XHTML), vous utiliserez la perspective Java EE. Et, lorsque vous voudrez communiquer avec GitHub, vous utiliserez la perspective Git.

Création du dépôt local initial avec Eclipse

À faire par **un seul** membre de l'équipe

Un seul des membres va transformer son projet **projet** en dépôt local Git.

Ensuite, il va le transférer sur la plateforme GitHub.

Enfin, l'autre membre va récupérer le projet à partir de GitHub et l'enregistrer dans un dépôt local sur son ordinateur.

Transformation du projet projet en dépôt local

- Dans la perspective Java EE, clic-droit sur le projet **projet**,
  puis "Team > Share Project…"
  - Cocher ☑ **Use or create repository in parent folder of project**
  - Dans la zone centrale, sélectionner le projet **projet**.
  - Bouton "Create Repository", en dessous, à gauche.
  - Bouton "Finish"

Création du fichier .gitignore

- Dans le projet **projet**, à la fin, clic-droit sur le fichier **README.txt**.
  Puis, "Team > Ignore"

Ceci a pour effet de créer un fichier caché nommé **.gitignore**.

Neutralisation des branches target

Les dossiers **target** contiennent des fichiers temporaires. Il est donc préférable de ne pas les transférer sur GitHub.

- Passez dans la perspective "Git".
- À gauche, dans le projet **projet**, dépliez la branche **Working Tree**.

- Double-cliquez sur le fichier **.gitignore**
- Dans la vue centrale, ajoutez la ligne suivante :

  target

  sans / au début.
- Vous pouvez supprimer la ligne :

  /README.txt

  car nous ne souhaitons pas réellement ignorer le fichier **README.txrt**. Nous nous sommes seuelemnt servi de lui pour créer le fichier **.gitignore**.
- Enregistrez la modification et fermez le fichier.
- Ainsi, tous les dossiers nommés **target** seront ignorés lors du transfert vers GitHub.

Transfert vers le dépôt distant GitHub

- Dans la perspective Git, à gauche, sélectionnez le projet **projet**
- À droite, en bas, choisissez l'onglet "Git staging"
  - Bouton "Add all files to the index"
  - Dans "Commit Message" : **`Fichiers initiaux`**
  - Bouton "Commit"
- À gauche, clic-droit sur le projet **projet**, puis "Push Branch Master"
  - Remote name : **GitHub**
  - URL : celle du dépôt distant GitHub
    On la trouve au début de la page du dépôt sur le site GitHub.
    Elle est de la forme :
    `https://github.com/`*`pseudo`*`/projet-jee.git`
  - `Renseigner Username + Password + ☑ Store in Secure Store"`
    Il s'agit des identifiants GitHub.
    C'est le token qu'il faut utiliser en tant que mot de passe
  - Bouton "Preview >"
  - Bouton "Preview >"
  - Bouton "Push"
  - Bouton "Close"

Création du dépôt local par l'autre membre de l'équipe

À faire par **un seul** membre de l'équipe

Si l'autre membre de l'équipe a créé lui aussi un projet **projet**, celui-ci va être redondant avec le projet provenant de GitHub. Il va donc être nécessaire de supprimer ce projet avant de récupérer les fichiers à partir de GitHub.

Configuration d'Eclipse

- Dans Eclipse, supprimez le projet **projet**, si vous l'y avez créé.
  Pensez bien à cocher la case ☑ **Delete project contents on disk (connot be undone)** afin que les fichiers soient supprimés physiquement.

- Il faut que Eclipse ait été configuré pour Git et que la perspective Git ait été mise en place (voir les instructions déjà données plus haut)

Récupération du contenu du dépôt GitHub

- Dans la perspective Git, à gauche, cliquez sur le lien "Clone a Git repository".
  - URL : celle du dépôt distant GitHub
    On la trouve sur GitHub, à la page du dépôt, bouton "Code".
    Elle est de la forme :
    `https://github.com/`*`pseudo`*`/projet-jee.git`
  - `Renseigner Username + Password + ☑ Strore in Secure Store"`
    Il s'agit des identifiants GitHub.
    NB. C'est le token qu'il faut utiliser en tant que mot de passe
  - Bouton "Next >".
- Vérifiez que la branche **master** est cochée, puis bouton "Next >".
- Sur le panneau "Local Destination" :
  - Important ! Dans le champ "Directory", vérifiez que le chemin correspond bien à celui du projet **projet** au sein de votre workspace. Si ce n'est pas le cas, c'est que la configuration d'Eclipse n'a pas été faite correctement.
  - Dans le champ "Directory", il est probable que le chemin se termine par **projet-jee**. Si c'est le cas, supprimez les 4 derniers caractères pour que le chemin se termine par **projet**.
    Cela permettra d'avoir le même nom de dossier que dans le workspace d'origine.
  - Remote name : **github**
  - Cochez ☑ **Import all existing Eclipse projects after clone finishes**
  - Bouton "Finish".

  L'opération dure quelques instants et, à la fin, le volet de gauche de la perspective Git contient une entrée correspondant à la copie locale du dépôt **projet-jee**.
- Passez dans la perspective Java EE pour constater que le projet **projet-jee** et ses sous projets y ont bien été importés.
  Si des erreurs sont signalées, faites les disparaître en faisant **[Alt]+[F5]**.
- S'il y a toujours des erreurs, redémarrez Eclipse (menu "File > Restart").
  Puis, faites de nouveau **Alt**+**F5**.

Opérations de base avec Git

À faire tout au long du projet

Cette partie décrit les différentes opérations qu vous aurez à faire tout au long de la réalisation du projet;

Mise à jour du dépôt local à partir du dépôt distant (PULL)

- Commencez par faire un **COMMIT**. Pour cela :
  - Dans la perspective Git, onglet "Git Staging"
  - Cliquer sur l'icône pour faire passer tous les fichiers dans la zone "Staged Changes".
  - Écrire quelque chose dans la zone "Commit Message".
  - Bouton "Commit".
- Ensuite, on peut faire le **PULL** :
  - À gauche, clic-droit sur le projet **projet**, puis 'Pull".

Remarque : Pour pouvoir faire **PULL**, il faut que la vue "Git Staging" soit complètement vide. Tant qu'i reste des fichiers qui n'ont pas participé à un **COMMIT**, il n'est pas possible de faire **PULL**.

Misse à jour du dépôt distant à partir du dépôt local (PUSH)

- Commencer par faire **COMMIT** et **PULL** (voir ci-dessus) sinon, le **PUSH** risque d'échouer..
- Ensuite, on peut faire le **PUSH** :
  - À gauche, clic-droit sur le projet **projet**, puis 'Push to GitHub".

Résolution de conflit apparu lors d'un PULL

- Dans la vue "Git Staging", double-cliquez sur un fichier marqué en conflit (pictogramme rouge). On peut aussi faire un clic-droit, puis "Merge Tool".
  Cela fait apparaître une vue permettant d'opérer la fusion.
- La vue de fusion présente :
  - à gauche votre code,
  - à droite celui qui a été modifié par l'autre personne.
- Dans la partie gauche, faites les modifications nécessaires pour obtenir le résultat souhaité.
  S'il n'y a rein à modifier, ajoutez au moins un espace quelque part pour qu'Eclipse considère que vous avez modifié le fichier.
- Puis, enregistrez les modifications.
- Fermez la vue de fusion.

- Une fois que les conflits ont tous été résolus, faites **COMMIT** pour valider l'état de votre projet.
- Vous pouvez ensuite faire **PUSH**, si vous le souhaitez.