Ressource – Git et GitHub

Objectif :

- Utiliser l'application de versioning git sur le service en ligne GitHub

Essentiel : les dépôts

- Un *dépôt* est comme un dossier qui conserve un historique des versions et des modifications d'un projet. Il est essentiel pour travailler en équipe ou collaborer à un projet open source.
- Un *dépôt local* est l'endroit où l'on stocke, sur sa machine, une copie d'un projet, ses différentes versions et l'historique des modifications.
- Un *dépôt distant* est une version dématérialisée du dépôt local, que ce soit sur Internet ou sur un réseau. Il permet de centraliser le travail des développeurs dans un projet collectif.
- Il existe plusieurs services en ligne pour héberger un dépôt distant, GitHub, GitLab, Bitbucket.

On peut initialiser un *dépôt* (*repository*) soit en local, soit via l'interface de GitHub.

Essentiel : les pull requests

Les *pull requests* (ou *demandes de pull*), vous permettent d'informer les autres utilisateurs des modifications que vous avez appliquées à une branche d'un *repository* sur GitHub, et que vous voulez fusionner avec le code principal.

Prise en main de Git

- Installation (sous Windows cliquez sur launch git bash pour avoir l'invite de commande git ad hoc);
- Initialisation :

Ouvrir git bash (Windows) ou le Terminal (Mac) ;

Se rendre dans le dossier abritant votre projet avec la commande cd ;

git config --global user.name "misterned" (configure git pour tous les futurs projets sauf si vous retirez –global)
git config --global user.email nedelec.stephane@yahoo.fr

git config --global color.diff auto

git config --global color.status auto

git config --global color.branch auto (configuration des couleurs suivant les contextes)

git config --global core.editor vim

git config --global merge.tool vimdiff(configuration des éditeurs suivant les contextes)

git init (création d'un fichier caché .git)

- Ajouter des fichiers (modifiés ou créés) aux dépôts

git add Data/birds.csv flask1.py (on indexe les fichiers)

git commit -m "Ajout du csv de base et la première version Flask" (création d'une nouvelle version, la première ici , -m permettant l'ajout du message)

Créer un repository sur GitHub.

git remote add origin https://github.com/misterned/birdsforlife.git (Configurer l'ajout au dépôt distant)

git branch -M main (Relier au dépôt distant)

git push -u origin main (Ajout à la branche main)

Si votre mot de passe n'est pas reconnu, utilisez un token

En cas d'erreur, pour réinitialiser le dépôt : rm -rf .git
Si ce message ! [rejected] master -> master (fetch first) apparait : git push --force origin main

- Sauvegarde d'un fichier dans les dépôts après sa modification

git add flask1.py

git commit -m "début de la base oiseaux"

git push origin main

Collaborer avec Git

- Le propriétaire du dépôt distant invite ses collaborateurs

Dans settings > collaborators

Les collaborateurs reçoivent un mail et acceptent l'invitation via Git Credentiel Manager

- Les collaborateurs récupèrent le dépôt distant

Dans le git bash, avec cd, se rendre dans le dossier qui accueillera le repo

git clone https://github.com/misterned/birdsforlife.git

- Après modification du dépôt local, mise à jour du dépôt distant :

Ouvrir git bash (Windows) ou le Terminal (Mac), se rendre dans le dossier qui accueille le repo ;

git push origin main

- Après modification du dépôt distant par un autre collaborateur ou le propriétaire, mise à jour du dépôt local :

Ouvrir git bash (Windows) ou le Terminal (Mac)), se rendre dans le dossier qui accueille le repo ;

git pull origin main

Cette méthode est dangereuse : il vaut mieux privilégier les branches

Protéger la branche main

- Le propriétaire du dépôt bloque cette branche, c'est bien plus prudent

Dans settings > branches > Branch name pattern (main) et vous cochez Require a pull request before merging

- Désormais si on souhaite

git pull origin main

alors on a

```
! [remote rejected] main -> main (protected branch hook declined)
```

Utiliser une autre branche

- Création d'une branche

git branch (on affiche la liste des branches créées. Celle qui est active est précédée d'une étoile)

git branch css (on crée une branche dédiée à une fonctionnalité. Si on utilise SCRUM qui coïncide plutôt avec un sprint)

git checkout css (on change de branche active)

et on recommence

git add…

- Pull request

Mais au lieu de « pusher » la branche main on push la branche css.

Et alors une Pull Request apparaît sur GitHub validée ou pas par le propriétaire du repo en cliquant sur le bouton Create Pull Request.

Voici le repo ayant permis d'illustrer cette ressource : https://github.com/misterned/birdsforlife