Guide Git et GitHub

Git : Système de gestion de versions

- Un outil pour suivre les modifications du code et collaborer avec d'autres développeurs.
- **Caractéristiques** :
    - Open Source
    - Rapide et évolutif

GitHub : Service d'hébergement de dépôts

- Un site web permettant aux développeurs de stocker et gérer leur code avec Git.

Configuration de GitHub

1. **Créer un compte GitHub** :
   Inscrivez-vous sur GitHub.
2. **Ajouter un nouveau dépôt** :
    - Cliquez sur **"New Repository"**.
    - Ajoutez un fichier README lors de la création, si nécessaire.
3. **Mettre à jour les modifications** :
   Suivez un processus en **2 étapes** :
    - **Ajouter** les modifications : `git add <nom du fichier>`
    - **Valider** les modifications : `git commit -m "message"`

Installation de Git

Outils nécessaires :

- **Éditeur de code** : Visual Studio Code (VS Code).
- **Outils en ligne de commande** :
    - Windows : Git Bash
    - Mac : Terminal

Tester Git :

- Vérifiez si Git est installé : `git --version`.
- Si Git n'est pas installé, téléchargez-le depuis git-scm.com.

Configuration de Git

1. **Définir le nom d'utilisateur** :
   ```
   git config --global user.name "votre nom d'utilisateur GitHub"
   ```
2. **Définir l'adresse email** :
   ```
   git config --global user.email "votre email GitHub"
   ```
3. **Vérifier la configuration** :
   ```
   git config --list
   ```

Copier un dépôt GitHub sur votre machine locale

1. **Cloner un dépôt** :
   ```
   git clone <https://github.com/utilisateur/repo.git>
   ```
2. **Vérifier le statut du dépôt** :
   ```
   git status
   ```

États des fichiers

1. **Untracked** : Fichiers non suivis par Git.
2. **Modified** : Fichiers modifiés connus de Git.
3. **Staged** : Fichiers ajoutés avec `git add`.
4. **Unchanged** : Fichiers sans modification ou déjà validés.

Ajouter et valider des modifications

1. **Ajouter des modifications** :
   ```
   git add <nom du fichier>
   ```
2. **Valider des modifications** :
   ```
   git commit -m "votre message"
   ```

Envoyer des modifications sur GitHub

- Envoyez le contenu local sur GitHub :
  `git push origin main`
  (Origin : Nom du dépôt distant, Main : Branche principale par défaut)

Utilisation de Git sur un dépôt local

1. **Initialiser un dépôt local** :
   `git init`
2. **Lier à un dépôt GitHub** :
   - Ajouter un dépôt distant :
     `git remote add origin <lien-du-nouveau-repo>`
   - Vérifier le lien :
     `git remote -v`

Renommer la branche (par défaut, c'est **main**) :
css
Copier le code

```
git branch -M main
git branch
```

3.
4. **Envoyer des modifications sur GitHub** :
   `git push origin main`

Travailler avec des branches Git

1. **Afficher toutes les branches** :
   `git branch`
2. **Renommer une branche** :
   `git branch -M main`
3. **Changer de branche** :
   `git checkout <nom de la branche>`

4. **Créer une nouvelle branche** :
   `git checkout -b <nom de la nouvelle branche>`
5. **Supprimer une branche** :
   `git branch -d <nom de la branche>`

Fusionner des branches

1. **Voir les différences entre les branches** :
   `git diff <nom des branches>`
2. **Résoudre les conflits** :
   - Si des conflits surviennent (codes différents sur une même ligne dans deux branches), résolvez-les manuellement et validez la fusion.
   - Sinon, fusionnez directement sur GitHub.

Annuler des modifications

Cas 1 : Modifications en attente

1. Retirer un fichier de la zone de staging :
   `git reset <nom du fichier>`
2. Retirer tous les fichiers de la zone de staging :
   `git reset`

Cas 2 : Annuler le dernier commit

1. Revenir au commit précédent :
   `git reset HEAD~1`

Cas 3 : Annuler plusieurs commits

1. Afficher l'historique des commits :
   `git log`
2. Revenir à un commit spécifique :
   `git reset <hash du commit>`
   (Le hash du commit est obtenu via `git log`)
3. Réinitialisation forcée (supprime les modifications) :
   `git reset --hard <hash du commit>`

Forker un dépôt

- Forker un dépôt crée une copie d’un dépôt sur votre compte GitHub pour un usage personnel ou une collaboration.