2. Mise en Route

2.1. Création de compte sur GitHub

Pour commencer à utiliser GitHub, la première étape est de créer un compte. GitHub est une plateforme en ligne qui héberge des dépôts Git, permettant la collaboration et la gestion de version pour les projets de développement logiciel. Voici comment créer un compte :

1. Allez sur le site web de GitHub (https://github.com).
2. Cliquez sur le bouton "Sign up" pour commencer le processus d'inscription.
3. Remplissez le formulaire d'inscription avec votre nom d'utilisateur, votre adresse email et un mot de passe sécurisé.
4. Vous recevrez un email de vérification. Suivez le lien fourni pour vérifier votre adresse email.
5. Une fois votre adresse email vérifiée, vous pouvez compléter votre profil et commencer à utiliser GitHub.

2.2. Installation de Git et Git Bash sur Windows

Git est un système de contrôle de version essentiel pour gérer vos projets de développement. Git Bash, quant à lui, est un terminal qui permet d'utiliser Git sur Windows avec une interface de ligne de commande. Pour installer Git et Git Bash :

1. Téléchargez le dernier installateur de Git pour Windows sur le site officiel (https://git-scm.com/download/win).
2. Lancez l'installateur téléchargé et suivez les instructions. Pendant l'installation, vous pouvez laisser les options par défaut, mais faites attention à l'écran "Adjusting your PATH environment", où sélectionner "Git from the command line and also from 3rd-party software" est recommandé pour les débutants.
3. Après l'installation, ouvrez Git Bash à partir du menu de démarrage. Vous avez maintenant accès à l'interface de ligne de commande Git sur votre Windows.

2.3. Configuration de la clé SSH avec GitHub

La configuration d'une clé SSH (Secure Shell) est un élément crucial pour établir une connexion sécurisée entre votre ordinateur et GitHub sans nécessiter de saisir votre nom d'utilisateur et votre mot de passe à chaque interaction. Suivez ces étapes détaillées pour créer et ajouter une clé SSH à votre compte GitHub.

2.3.1. Génération de votre clé SSH

1. Ouvrez Git Bash. C'est une application qui simule un environnement de ligne de commande similaire à ceux trouvés sur les systèmes Linux et Unix.

2. Générez une nouvelle clé SSH en utilisant votre adresse email associée à votre compte GitHub. Tapez la commande suivante, en remplaçant "your_email@example.com" par votre adresse email réelle :

   ```
   ssh-keygen -t ed25519 -C "your_email@example.com"
   ```

   Pourquoi `ed25519` ? C'est un algorithme de cryptographie moderne recommandé pour sa sécurité.
3. Lorsque le système vous demande un emplacement pour sauvegarder la nouvelle clé, appuyez sur `Enter` pour accepter l'emplacement par défaut proposé. Il créera un fichier dans votre dossier utilisateur, dans un sous-dossier `.ssh`.
4. Si vous le souhaitez, vous pouvez également créer un mot de passe (passphrase) pour votre clé SSH pour une sécurité additionnelle. Sinon, appuyez simplement sur `Enter` pour passer cette étape.

2.3.2. Ajout de votre clé SSH à l'agent SSH

Pour que votre système utilise automatiquement votre clé SSH, vous devez l'ajouter à l'agent SSH, un programme qui gère les clés SSH pour vous.

5. Assurez-vous que l'agent SSH est en cours d'exécution avec la commande :

   ```
   eval $(ssh-agent -s)
   ```

6. Ajoutez votre clé SSH à l'agent SSH :

   ```
   ssh-add ~/.ssh/id_ed25519
   ```

2.3.3. Ajout de votre clé SSH à votre compte GitHub

Pour finaliser la connexion sécurisée entre votre ordinateur et GitHub, vous devez ajouter votre clé SSH publique à votre compte GitHub.

7. Ouvrez le fichier contenant votre clé publique (`id_ed25519.pub`) avec un éditeur de texte et copiez tout son contenu.
8. Connectez-vous à votre compte GitHub, naviguez vers les paramètres de votre compte (Settings), trouvez la section "SSH and GPG keys", et cliquez sur "New SSH key".
9. Dans le champ "Title", donnez un nom à votre clé qui vous aidera à la reconnaître, par exemple, "Mon PC personnel".
10. Collez votre clé SSH publique copiée dans le champ "Key".
11. Cliquez sur "Add SSH key" pour terminer.

Avec ces étapes, vous avez établi une méthode sécurisée et pratique pour interagir avec GitHub depuis votre ordinateur local, sans avoir besoin de saisir vos informations d'identification à chaque fois.

2.4. Configuration de Git

Lorsque vous commencez à utiliser Git sur votre machine locale, il est essentiel de configurer votre nom d'utilisateur et votre adresse e-mail. Ces informations seront utilisées pour identifier vos contributions aux dépôts Git. Voici comment procéder :

2.4.1. Configuration du nom d'utilisateur

Pour configurer votre nom d'utilisateur dans Git, utilisez la commande suivante :

```
git config --global user.name "Votre Nom"
```

Remplacez "Votre Nom" par votre nom réel ou le nom que vous souhaitez utiliser pour vos contributions. Par exemple :

```
git config --global user.name "John Doe"
```

2.4.2. Configuration de l'adresse e-mail

Pour configurer votre adresse e-mail dans Git, utilisez la commande suivante :

```
git config --global user.email "votre@email.com"
```

Remplacez "votre@email.com" par votre adresse e-mail. Par exemple :

```
git config --global user.email "johndoe@email.com"
```

Il est important d'utiliser la même adresse e-mail que celle associée à votre compte GitHub si vous prévoyez de contribuer à des projets sur cette plateforme.

2.4.3. Autres configurations

Outre la configuration du nom d'utilisateur et de l'adresse e-mail, vous pouvez également personnaliser d'autres aspects de votre configuration Git.

— **push.default**
Cette option définit le comportement par défaut de Git lors du push de branches. Par exemple, vous pouvez définir cette option sur "simple" pour pousser uniquement la branche actuelle sur sa branche distante avec le même nom. Pour configurer cela, utilisez la commande suivante :

```
git config --global push.default simple
```

— **init.defaultBranch**
Cette option définit la branche par défaut pour la commande git init. À partir de Git 2.28, la branche principale est souvent appelée "main" au lieu de "master". Vous pouvez configurer cette option pour utiliser "main" comme branche par défaut. Par exemple :

```
git config --global init.defaultBranch main
```

2.4.4. Affichage des configurations

Pour afficher l'ensemble de vos configurations Git, vous pouvez utiliser la commande suivante :

```
git config --list
```

Cette commande affichera toutes les configurations définies sur votre système, y compris les configurations globales (utilisées pour tous les projets) et les configurations locales (spécifiques à un projet).