TP

—

GitHub

KALETA Maxime

BTS SIO

ensemble SAINT-LUC Cambrai
Les chemins de la réussite

Sommaire

- Démarches de base
- Lab 0 staging et commit
- Lab 1 merge

Démarches de base

Installation de GIT

```
#apt install git
```

Création d'un répertoire de travail

```
# mkdir repertoire
```

Initialisation du repository git

```
# git init
```

Vous pouvez vérifiez qu'un répertoire caché est apparu (./git), c'est dans ce
dernier que tout sera organisé.

Configurer git :

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

Pour transfert de fichiers en SSH :

```
#scp /chemin/vers/votre/fichier root@(IP du destinataire):/chemin/
de/destination
```

KALETA Maxime

Démarches de base

Après avoir créé vos fichiers ,vous devez les basculer en staging via la commande :

```
#git add {fichier} (pour un fichier )
ou
#git add . (pour tous les fichiers, le point est obligatoire)
```

Une bonne habitude est d'utiliser la commande status afin de savoir où vous en êtes :

```
#git status
```

Puis il faut commiter en indiquant clairement l'action réalisée

```
#git commit -m "votre commentaire"
```

Contrôler vos commits à l'aide de la commande :

```
#git log
```

Lab 0 staging et commit

- Création du fichier readme.md

Le fichier Readme et une sorte de notice d’utilisation des fichiers présent dans le ripository. Le format .md est un format appelé markdown, celui-ci permet de créer un fichier texte mieux présenté sur son Github.

#nano readme.md

- Ajout du fichier en staging

Une particularité de git est son système de staging qui **permet de sélectionner les fichiers à suivre lors du prochain commit**. Vous pouvez imaginer ça comme une "zone d'attente" où on va lister les fichiers que l'on souhaite voir enregistrés.

#git add readme.mg

KALETA Maxime

Lab 0 staging et commit

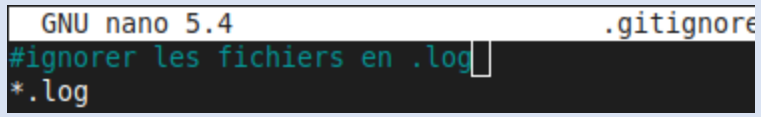
- Faire le commit

Les commits peuvent être considérés comme des instantanés ou des étapes importantes dans la chronologie d'un projet Git.

##git commit -m "voici le fichier readme"

- **Création du .gitignore**
- Le fichier **.gitignore** est un fichier utilisé dans les projets Git pour spécifier quels fichiers ou répertoires ne doivent commités par Git. Cela permet de garder l'historique Git propre en excluant les fichiers qui ne sont pas pertinents pour le contrôle de version, comme les fichiers temporaires, les fichiers de configuration locaux, les fichiers compilés, ou les dépendances externes.

#nano .gitignore



#git add .gitignore

KALETA Maxime

Lab 0 staging et commit

- Création du fichier control.log

#nano control.log

#git add control.log

Nous faisons le commit et le control.log n'est pas traquer car il n’apparait pas, il y a 1 file changé mais 2 insertions

```
root@debian:~/tpgit# git commit -m " voici le gitignore"
[main c1ecbe6]  voici le gitignore
 1 file changed, 2 insertions(+)
 create mode 100644 tpgit/.gitignore
```

- Création de tags

Celui-ci permet d’annoncer les versions

#git tag v1,0,0

Voir le tag avec :

#git log

```
root@debian:~/tpgit# git tag v1.0.0
root@debian:~/tpgit# git tag
v1.0.0
root@debian:~/tpgit# git log
commit c1ecbe6c8e702df9f7b22995a4b429bd43cea9e5 (HEAD -> main, tag: v1.0.0)
Author: d4rtox <maximk815@gmail.com>
Date:   Wed Nov 6 10:46:24 2024 +0100

    voici le gitignore
```

KALETA Maxime

Lab 1 merge

- Création d'une branche feature 1

#git branch feature 1

- Basculement sur celle-ci

#git checkout feature1

- Modification du html

#git add index.html

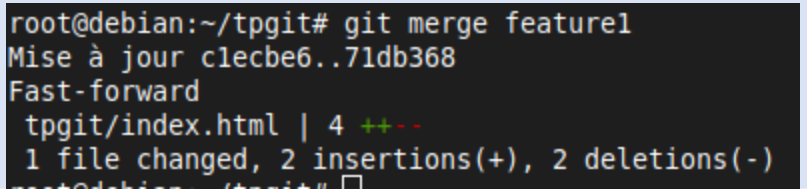
#git commit -m "voici mon html dans ma feature1"git commit -m "voici mon html dans ma feature1"

- Commande merge :

La commande git merge vous permet de sélectionner les lignes de développement indépendantes créées avec git branch et de les intégrer à une seule branche

#git merge feature1

Nous revenons sur la branche main avec un checkout et faisons cette commande :



KALETA Maxime

Lab 1 merge

- **Suppression de la branche feature1 :**

```
root@debian:~/tpgit# git branch -D feature1
Branche feature1 supprimée (précédemment 71db368).
```

- **Branch fix5 :**

Modification du script.sh sur la branch fix :

```
 GNU nano 5.4                          script.sh *
#!/bin/bash
# Script to add a user to Linux system
if [ $(id -u) -eq 0 ]; then
        read -p "Enter username : " maxime
        read -s -p "Enter password : " kaleta
        egrep "^$username" /etc/passwd >/dev/null
        if [ $? -eq 0 ]; then
                echo "$username exists!"
                exit 1
        else
                pass=$(perl -e 'print crypt($ARGV[0], "password")' $password)
                useradd -m -p $pass $username
                [ $? -eq 0 ] && echo "User has been added to system!" || echo "
        fi
else
        echo "Only root may add a user to the system"
        exit 2
fi
```

Modification du script mais sur la branch main

```
 GNU nano 5.4                          script.sh *
#!/bin/bash
# Script to add a user to Linux system
if [ $(id -u) -eq 0 ]; then
        read -p "Enter username : " username1
        read -s -p "Enter password : " password1
        egrep "^$username" /etc/passwd >/dev/null
        if [ $? -eq 0 ]; then
                echo "$username exists!"
                exit 1
        else
                pass=$(perl -e 'print crypt($ARGV[0], "password")' $password)
                useradd -m -p $pass $username
                [ $? -eq 0 ] && echo "User has been added to system!" || echo "
        fi
else
        echo "Only root may add a user to the system"
        exit 2
fi
```

KALETA Maxime

Lab 1 merge

- **Après le merge** nous nous retrouvons avec une erreur car les deux fichiers identique ont un contenu différent

```
root@debian:~/tpgit# git merge fix5
Mise à jour 71db368..407aefd
error: Vos modifications locales aux fichiers suivants seraient écrasées par la
fusion :
        tpgit/script.sh
Veuillez valider ou remiser vos modifications avant la fusion.
Abandon
```

- **On se rend dans le fichier concerné :**

```
<<<<<<< HEAD
        read -p "Enter username : " username1
        read -s -p "Enter password : " password1
=======
        read -p "Enter username : " maxime
        read -s -p "Enter password : " kaleta
>>>>>>> fix5
```

Suppression de ce qu'il y a dans le head et nous laissons uniquement ce qui est souhaité et il n'y as plus de problèmes

- **Ajout et diffusion des fichiers :**

#git add script.sh

#git commit

KALETA Maxime