GIT

...

Kelig Martin

WIZARDS
TECHNOLOGIES

Qui suis-je ?

Kelig Martin, développeur fullstack à Wizards Technologies.

Parcours : BAC S → BTS SIO → Bachelor Architecture des logiciels → Master Architecture des logiciels

Sommaire du cours

1. Présentation de Git
2. Mise en pratique + TP noté
3. QCM
4. TP noté

La notation

- Un QCM
- Un TP noté

Présentation de GIT

(la partie chiante)

Qu’est-ce que GIT ?

- Créé en 2005
- VCS : Version Control System (comme SVN ou Mercurial)
- Logiciel permettant de sauvegarder un ensemble de fichiers et leur modification de manière chronologique.

Pourquoi utiliser git ?

- Commandes clair, intuitives, faciles à utiliser, difficiles à maîtriser
- Travail collaboratif
- Historique des modifications (rollback plus simples)
- Gestion par branches

Il reste toutefois compliqué de travailler à plusieurs sur un même fichier, les conflits seront monnaie courante, mais facile à résoudre.

Installer git

https://git-scm.com/download

\+ git bash si vous êtes sur windows

\+ se créer un compte sur github

Comment se présente un projet git ?

- Appeler “repo” ou “dépôt” il s’agit d’un dossier comportant d’autres fichiers.
- Contient l’historique des “commit” par fichier, ainsi que son ou ses contributeurs.
- Est consultable sur un hub (Github, gitlab, bitbucket etc...)
- Il peut interagir avec différents outils (Jenkins, Docker, TravisCI etc...)
- Il peut être privé ou public (open source)
- Se présente sous la forme d'un arbre à branches.

Représentation visuelle

nombre de branches du projet

branche actuelle

dernier commit et ses infos

Liste des fichiers présents sur la branche actuelle

Derniers commits effectués sur les fichiers correspondants

Date du dernier commit

Pré requis

Lancez cette commande pour générer une clef ssh qui vous servira à sécuriser vos commandes git et entrez là sur https://github.com/settings/ssh/new .

```
ssh-keygen -t rsa -b 2048
```

Mise en pratique

Notre premier projet git

```
kelig@kelig:~/Bureau/4-SRC$ git init
Dépôt Git vide initialisé dans /home/kelig/Bureau/4-SRC/.git/
kelig@kelig:~/Bureau/4-SRC$
```

La commande “**git init**” permet de signifier à git que le dossier courant doit être suivi.

A partir de là il comprendra les modifications, ajouts et suppressions de fichiers qui s’y effectueront.

Suivi de notre projet

```
kelig@kelig:~/Bureau/4-SRC$ touch script.sh
kelig@kelig:~/Bureau/4-SRC$ git status
Sur la branche master

Aucun commit

Fichiers non suivis:
  (utilisez "git add <fichier>..." pour inclure dans ce qui sera validé)

        script.sh

aucune modification ajoutée à la validation mais des fichiers non suivis sont pr
ésents (utilisez "git add" pour les suivre)
```

Nous avons créé un fichier “**script.sh**” dans notre projet git. Celui-ci est reconnu par git mais n’est pas encore suivi, comme nous l’indique le message lors de la commande “**git status**”. Cette commande nous permet de connaître le statut de notre projet depuis sa dernière “mise à jour”.

Ajouter un fichier

```
kelig@kelig:~/Bureau/4-SRC$ git add script.sh
kelig@kelig:~/Bureau/4-SRC$ git status
Sur la branche master

Aucun commit

Modifications qui seront validées :
  (utilisez "git rm --cached <fichier>..." pour désindexer)

        nouveau fichier : script.sh
```

Notre fichier a été ajouté et est maintenant suivi. Plus que 2 étapes avant de pouvoir l'envoyer vers notre dépôt distant.

Commit du fichier

```
kelig@kelig:~/Bureau/4-SRC$ git commit -m "first commit"
[master (commit racine) 9d413ca] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 script.sh
kelig@kelig:~/Bureau/4-SRC$ git status
Sur la branche master
rien à valider, la copie de travail est propre
kelig@kelig:~/Bureau/4-SRC$
```

Maintenant que nous avons ajouté notre fichier, il faut le “commit” avec la commande git du même nom.
Grâce à l’argument -m on peut mettre un message de commit, si on n’en met pas, un éditeur de texte par défaut sera ouvert.

Maintenant on push c'est ça ?

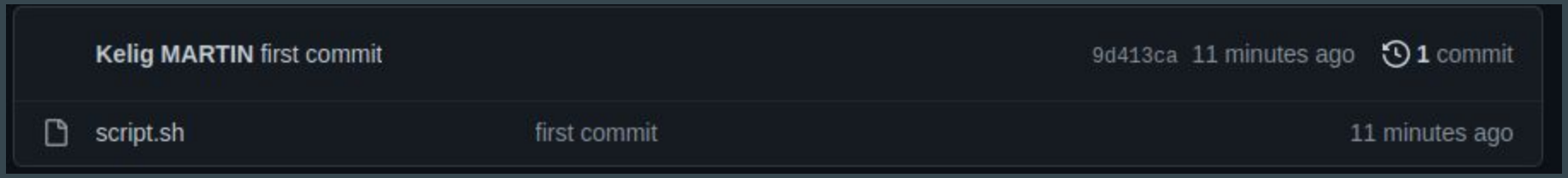
Avant de pouvoir pousser notre travail, il faut dire où on l'envoie. Pour cela, nous lançons la commande suivante :

```
kelig@kelig:~/Bureau/4-SRC$ git remote add origin git@github.com:KeligMartin/4-SRC.git
kelig@kelig:~/Bureau/4-SRC$ git push -u origin master
Décompte des objets: 3, fait.
Écriture des objets: 100% (3/3), 239 bytes | 239.00 KiB/s, fait.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:KeligMartin/4-SRC.git
 * [new branch]      master -> master
La branche 'master' est paramétrée pour suivre la branche distante 'master' depuis 'origin
'.
```

Les commandes que nous venons d'exécuter sont affichées à chaque répo créé.

Quel est le résultat ?

Si je me rend sur github, je vois maintenant que mon dépôt contient le fichier que j'ai créé.



Informations utiles :

Git cheat sheet : https://education.github.com/git-cheat-sheet-education.pdf

Git Karma : http://karma-runner.github.io/6.4/dev/git-commit-msg.html

GitMoji : https://gitmoji.dev/

TP Utilisation de GIT