RAPPORT DE PROJET SUR GIT & MAKEFILE

October 9, 2023

Auteur: ALIOUNE BADARA IBN DIENE M1.CHPS

0.1 I_ Git

0.2 I.1 Vérification de la version de git

Une fois qu'on a téléchargé git sur notre pc, la première chose à faire est de vérifier si git est bien installée. Pour cela, on utilise la commande:

0.3 git –version

Si nous voyons git version et le numéro de version, on peut s'assurer que tout c'est bien passé.

`I.2 Définir l'identité de l'utilisateur`

Le paramétrage nous permet d'éviter après chaque modification de préciser notre identité, pour cela on utile cette commande suivante:

0.4 git config –global user.name

0.5 git config –global user.email

exemple:

git config –global user.Alioune_Badara

git config –global user.aliounebadaraibnabutalibdiene@gmail.com

0.6 I.3 Création dun projet avec les commandes de base

On va commencer par créer un répertoire avec la commande:

mkdir

for exempl : mkdir WOLFRAM1

Et puis saisir la commande:

cd

pour entrer dans WOLFRAM1.

Une fois que c'est fait, on utilise maintenant la commande:

git init

pour initialiser notre projet.

0.7 Initialized empty Git repository in /Users/cash/WOLFRAM1/.git/

git est initialisé !

Maintenant, on aimerait avoir un statut global de nos fichiers pour savoir l'évolution du projet. C'est très simple, om utilise la commande:

0.8 git status

On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

- no commits yet, on n'a pas encore créé un fichier.

0.9 Création dun fichier makefile.txt

Une fois que ce fichier est créé, on peut prendre une photo du projet par ces deux commandes suivantes:

0.10 git add "file name"

et

0.11 git commit -d "first commit"

la première commande est une sélection du fichier makefile.txt et par la deuxième commande on prend une photo de ce fichier

0.12 exemple

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git add makefile.txt

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git commit -m "project init"

[master (root-commit) 0372d32] project init

1 file changed, 1 insertion(+)

create mode 100644 makefile.txt

1 Notre project a été sauvegardé !

Maintenant avec git c'est possible d'apporter des modifications nécessaires à notre projet et puis après de prendre une nouvelle prise de photo !

pour ça, on va modifier le fichier makefile.txt en le renommant MaKEFILE.TXT et puis sauvegarder ce dernier.

Puis sur notre terminal, demandé à git de faire une autre capture à travers les commandes:

1.1 git add “file name”

et

1.2 git commit -d “first commit”

Avant cela, en utilisant la commande git status, on peut savoir l'état actuel de notre projet.

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git status On branch master Untracked files: (use “git add …” to include in what will be committed)

MAKEFILE.TXT

nothing added to commit but untracked files present (use “git add” to track)

1.3 (base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git add MAKEFILE.TXT

1.4 (base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git commit -m ” modif de la chaine de carac en maj”

[master cb76a69] modif de la chaine de carac en maj 1 file changed, 5 insertions(+) create mode 100644 MAIN.C

```
project init
```

1.5 TIME LINE DES COMMITS

la commande git log

```
(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git log
```

commit cb76a69c686b5012e70200d04ec2b28d61e2671f (HEAD -> master)

```
Author: Alioune-Badara <aliounebadaraibnabutalibdiene@gmail.com>
```

Date: Thu Oct 5 22:46:03 2023 +0200

```
modif de la chaine de carac en maj
```

commit 0372d32cc8024b9bce8aca28ef8b7a6a4cefad4e Author: Alioune-Badara aliounebadaraibnabutalibdiene@gmail.com Date: Thu Oct 5 22:42:16 2023 +0200

Nous voyons que les deux commits sont présentes avec la date, l'explication et le nom de la personne qui a fait la modification.

2 I.4 Création d'une branche

avec la commande git branch

```
(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ # notion de branche

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git branch mchps
```

```
(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git branch
* master

mchps
```

On peut voir qu'on a deux branches: master et mchps on constate que la branche master précédée par une êtoile, est en couleur verte. Cela signifie qu'on se trouve sur la branche master.

Mais c'est possible de changer de branche, il suffit de saisir la commande

git checkout 'mchps'

```
(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git checkout 'mchps'

M       makefile.txt

Switched to branch 'mchps'
```

Bravo! la bascule a été faite correctement !

3 On peut rappatrier notre travail dans la branche principale. Pour cela, on doit réaliser trois étapes.

1- C'est la sélection de la branche principale

avec la commande **git checkout master**

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git checkout master

M makefile.txt

Switched to branch 'master'

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git branch

- master

  mchps

2- ensuite on fusionne les commits

avec la commande **git merge** NomBranche

```
(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git merge mchps

Already up to date.
```

On vérifie avec un git log si tout s'est bien passé:

```
(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git log
```

commit cb76a69c686b5012e70200d04ec2b28d61e2671f (HEAD -> master, mchps)

Author: Alioune-Badara aliounebadaraibnabutalibdiene@gmail.com

Date: Thu Oct 5 22:46:03 2023 +0200

`modif de la chaine de carac en maj`

commit 0372d32cc8024b9bce8aca28ef8b7a6a4cefad4e

Author: Alioune-Badara aliounebadaraibnabutalibdiene@gmail.com

Date: Thu Oct 5 22:42:16 2023 +0200

`project init`

3- et enfin la Suppresion de la branche

avec la commande **git branch -d** NomBranche

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git branch -d mchps

Deleted branch mchps (was cb76a69).

`git nous dit que la branche mchps est bien supprimée !`

On peut le vérifier en saisisant git branch

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git branch * master

Bravo !

Maintenant si l'on veut collaborer avec les autres sur notre projet on est obligé de faire un dépot distant. C'est ce que nous allons voir.

4 I.5 Depot distant

Cela consiste à déposer le projet sur un endroit accessible aux autres membres du groupe. Ces dernies pourront faire des modifications et puis les partagées.

étape 1: create a new repository sur github: **DEBUTANT**

étape 2: associer ce dépot distant à notre dépot local

et c'est grace à la commande, **git remote add + origin** de l'url de notre dépot, qu'on va pouvoir créer ce pont pour l'envoie.

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git remote add origin https://github.com/Alioune-Badara/DEBUTANT.git

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git remote

origin

GIT CONNAIT UN DEPOT DISTANT: ORIGIN

Il nous reste qu'à déposer le projet avec la commande:

git push

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ git push origin master

Enumerating objects: 6, done. Counting objects: 100% (6/6), done. Delta compression using up to 4 threads Compressing objects: 100% (4/4), done. Writing objects: 100% (6/6), 540 bytes | 540.00 KiB/s, done. Total 6 (delta 1), reused 0 (delta 0) remote: Resolving deltas: 100% (1/1), done. remote: remote: Create a pull request for 'master' on GitHub by visiting: remote: https://github.com/Alioune-Badara/DEBUTANT/pull/new/master remote: To https://github.com/Alioune-Badara/DEBUTANT.git * [new branch] master -> master

(base) Macbooks-MacBook-Air:WOLFRAM1 cash$ #DIENE ALIOUNE

5 II- MAKEFILE

Dans cette partie, nous allons voir la syntaxe des règles sur un makefile.

Tout d'abord, rappelons qu'un makefile est un fichier qui va être utiliser pare MAKE.

all:file1

```
cat file1 > file2
```

file1:

```
echo "hello" > file1
```

quant on fait make, il va regarder d'abord la première règle **all**

Il va regarder qu'on a besoin du fichier file1 et puis il se rendra compte qu'il existe une règle du nom de file1 ensuite il executera sa recette echo "hello" > file1

testons pour voir!

(base) Macbooks-MacBook-Air:DAKAR cash$ make

cat file1 > file2

(base) Macbooks-MacBook-Air:DAKAR cash$

maintenant utilisons la commande **cat makefile**

DAKAR cash$ cat makefile

all:file1 cat file1 > file2

file1: echo "hello" > file1

```
(base) Macbooks-MacBDAKAR cash$
```

Visiblement la première commande dans le makefile est executée avant la deuxième. Car c'est ce dont on a besoin pour faire le file1 et générer le file2.

On peut vérifier l'existance de ces deux fichiers par la commande **ls**

DAKAR cash$ ls

build.sh file1 file2 main.c main.d makefile obj queens.h rdtsc.h src

(base) Macbooks-MacBook-Air:DAKAR cash$

Si on utilise la commande **cat** fike1 file2 le message **hello** s'affiche !

DAKAR cash$ cat file1 file2

hello

hello

(base) Macbooks-MacBook-Air:DAKAR cash$

Voyons maintenant la règle **clean** qui sert à supprimer des fichiers.

DAKAR cash$ make clean

rm file1 file2

(base) Macbooks-MacBook-Air:DAKAR cash$

On vérifie avec la commande ls

DAKAR cash$ ls

build.sh main.c main.d makefile obj queens.h rdtsc.h src

(base) Macbooks-MacBook-Air:DAKAR cash$

Et voila, nos deux fichiers sont supprimés !

C'est aussi possible d'utiliser le makefile pour compiler un programme .c en un executable qui est mon prénom: alioune.

6 Exemple:

DAKAR cash$ make

gcc *.c -o alioune

DAKAR cash$ cat makefile

all: gcc *.c -o alioune

```
(base) Macbooks-MacBDAKAR cash$
```

DAKAR cash$ ls

build.sh main.c main.d makefile obj alioune queens.h rdtsc.h src

(base) Macbooks-MacBook-Air:DAKAR cash$

On peut utiliser la commande **zip** pour créer une archive.

Avec un projet de plusieurs fichiers, 1.000 par exemple, on aura du mal à nommer tous ces fichiers objets. Et donc pour rester dans le principe du makefile on peut gérer le temps en utilisant des **variables**. Illustrons un exemple.

7 Exemple

CC = gcc

EXEC = prog

all: $(EXEC)

$(EXEC): main.o player.o

```
$(CC) -o $(EXEC) main.o  player.o
```

main.o: main.c

```
$(CC) -o main.o -c main.c
```

player.o.o: player.c

```
$(CC) -o player.o -c player.c
```

8 En utilisant les variables on rend notre travail un peu plus dynamique.

Et si l'on utilise la commande make ça fonctionne !

DAKAR cash$ make

gcc -o prog main.o player.o

```
 DAKAR cash$ .\prog.exe

  Unknown : Bonjour Ali
```

9 SOMMAIRE