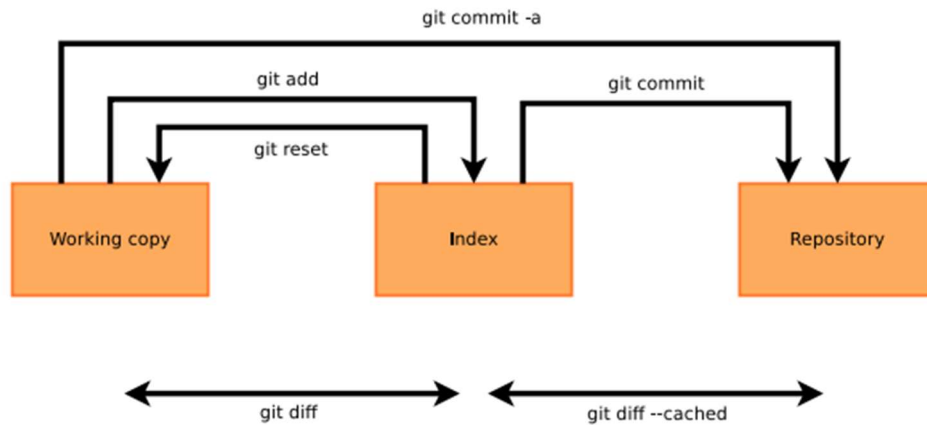


TP : Travailler avec Git



Source : <http://thomas.enix.org/pub/conf/git2011/presentation.pdf>

Pour travailler avec Git il faut comprendre comment le processus marche. Quand on crée ou modifie un fichier, on crée une copie temporaire, alors une *Working Copy*. On prépare ces changements à effectuer par le mettre dans l'*Index* depuis lequel on peut *commit* ces changements en créant et sauvegardant une version du fichier original.

Q2.1 Créer un répertoire, un dépôt et un fichier

Avec la commande `git init sandwich` on peut créer un dépôt Git dans un répertoire *sandwich*.

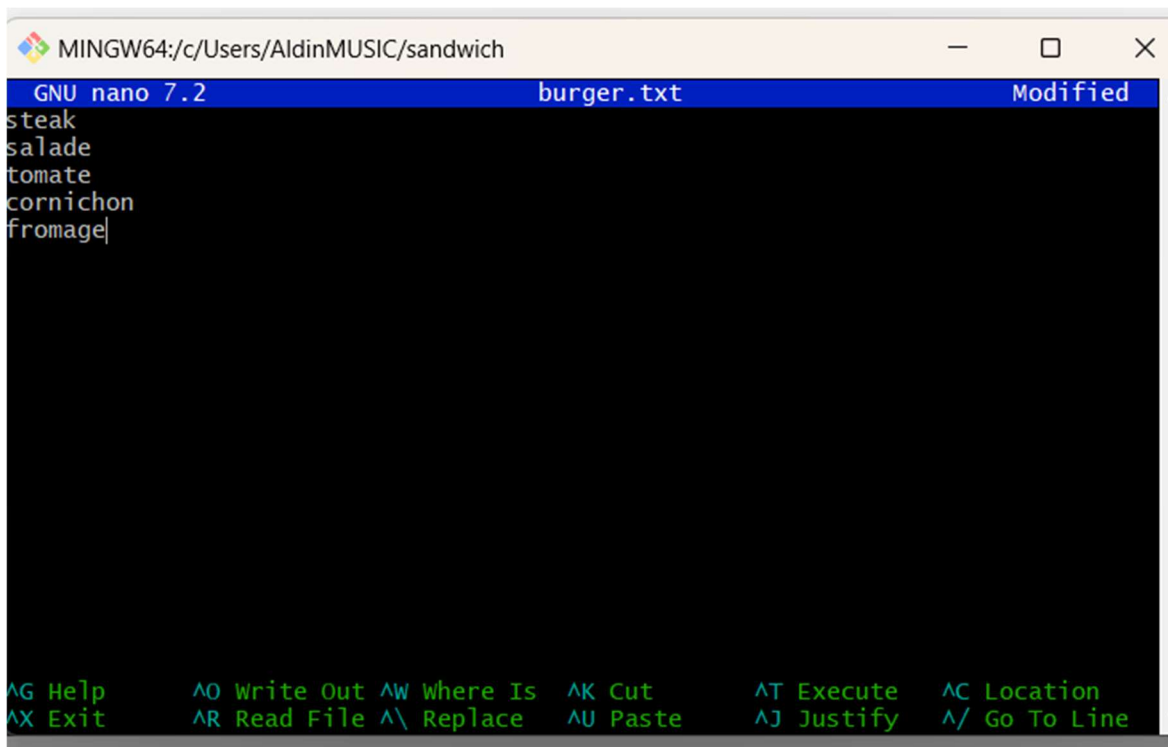
```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~  
$ git init sandwich  
Initialized empty Git repository in C:/Users/AldinMUSIC/sandwich/.git/
```

Pour aller dans ce répertoire on utilise la commande `cd C:/Users/AldinMUSIC/sandwich/`. À la place d'*AldinMUSIC* vous mettez votre utilisateur.

Pour créer un fichier `burger.txt` (ou le modifier si le nom donné existe déjà) on met `nano burger.txt`.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)  
$ cd C:/Users/AldinMUSIC/sandwich/  
  
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)  
$ nano burger.txt|
```

Cela nous ouvre un éditeur de texte permettant de modifier le texte du fichier.



```
MINGW64:/c/Users/AldinMUSIC/sandwich
GNU nano 7.2 burger.txt Modified
steak
salade
tomate
cornichon
fromage|

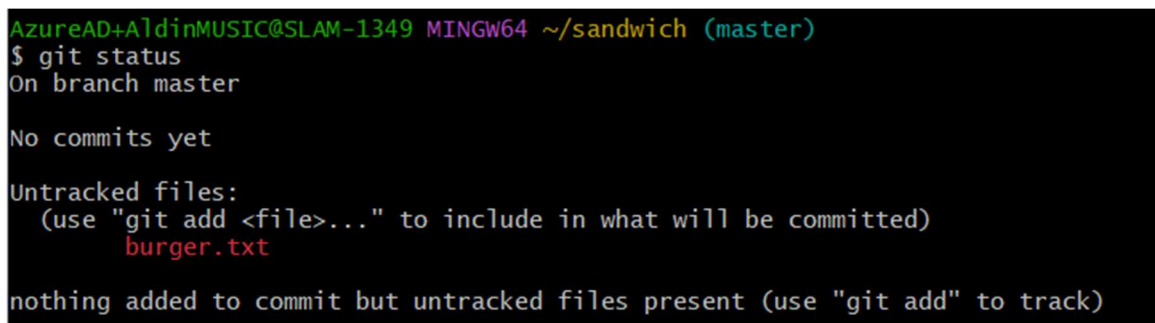
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

J'ai mis une liste des ingrédients dans le fichier. Pour sauvegarder on fait *Ctrl + O* et puis *Ctrl + X*.

Q2.2/2.3 Mettre les changements dans l'Index

Les fichiers créés sont des fichiers temporaires, alors des *Working copies*.

Pour voir le status d'un fichier on met la commande *git status*.



```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    burger.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Ici on voit que rien n'était *committed* alors les copies n'étaient pas sauvegardées.

On voit aussi que notre fichier burger.txt existe mais n'était toujours pas préparé pour le *commit*.

Pour faire cela, alors pour le mettre dans l'*Index* on met la commande `git add burger.txt`.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git add burger.txt
warning: in the working copy of 'burger.txt', LF will be replaced by CRLF the ne
xt time Git touches it
```

Si on fait encore un `git status`.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   burger.txt
```

On voit que le fichier est maintenant dans les changements préparés à être *committed*.

Pour voir les changements exacts qui sont en train d'être enregistrés on met `git diff --cached`.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git diff --cached
diff --git a/burger.txt b/burger.txt
new file mode 100644
index 0000000..3788869
--- /dev/null
+++ b/burger.txt
@@ -0,0 +1,5 @@
+steak
+salade
+tomate
+cornichon
+fromage
```

Ici on voit ce qu'est en train d'être ajouté en vert et avec un +.

Q2.4/2.5/2.6 Finaliser les changements

Maintenant on va finalement créer la copie du fichier avec tous les changements.

On peut *commit* avec la commande `git commit -m "<votre message>"`.

Le `-m` et ce qui vient après ne sont pas nécessaires mais ils nous permettent d'attacher un message pour cette version.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git commit -m "Commande 66"
[master (root-commit) e56e9c4] Commande 66
Committer: Aldin MUSIC <aldin.music@mewo-campus.fr>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 5 insertions(+)
create mode 100644 burger.txt
```

Maintenant si on fait encore un `git status`.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git status
On branch master
nothing to commit, working tree clean
```

On voit qu'il n'y a plus des fichiers temporaires.

Pour voir toutes les *commit* qu'on a fait on utilise la commande `git log`.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git log
commit e56e9c4c5ee19a69668e44afadbd913f0e1075b8 (HEAD -> master)
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 11:20:19 2024 +0100

    Commande 66
```

On a fait seulement un *commit* donc on voit seulement un *commit* dans le *log*.

Dans le *log* on voit plusieurs informations :

- commit <numéro identifiant haché en SHA1>
- Auteur <Nom et adresse mail>
- Date
- Message (le message qu'on a mit après le `git commit -m`).

Q2.7 Plusieurs changements

Maintenant on crée encore deux sandwiches *burrito.txt* et *hot_dog.txt*

```
AzureAD+AlidinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ nano hot_dog.txt

AzureAD+AlidinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ nano burrito.txt

AzureAD+AlidinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        burrito.txt
        hot_dog.txt

nothing added to commit but untracked files present (use "git add" to track)
```

On a vu qu'avec *git diff* on peut voir les changements qui sont en train d'être effectués.

```
AzureAD+AlidinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git diff

AzureAD+AlidinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
```

Mais ici cette commande n'affiche rien parce que cela marche seulement pour ce qui est dans l'*Index*.

Maintenant on fait un *git add* et puis encore un *git diff*.

```
AzureAD+AlidinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git add burrito.txt
warning: in the working copy of 'burrito.txt', LF will be replaced by CRLF the
next time Git touches it

AzureAD+AlidinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git add hot_dog.txt
warning: in the working copy of 'hot_dog.txt', LF will be replaced by CRLF the
next time Git touches it
```

```
AzureAD+AlidinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git diff --cached
diff --git a/burrito.txt b/burrito.txt
new file mode 100644
index 0000000..356f18e
--- /dev/null
+++ b/burrito.txt
@@ -0,0 +1,4 @@
+minced meat
+sweet pepper
+cheese
+corn
diff --git a/hot_dog.txt b/hot_dog.txt
new file mode 100644
index 0000000..3d40b55
--- /dev/null
+++ b/hot_dog.txt
@@ -0,0 +1,4 @@
+mustard
+ketchup
+fried onions
+sausage
```

Cela marchait parce qu'après le *git add* les fichiers sont mis dans l'*Index*.

Maintenant on a deux fichiers dans l'*Index* mais si on veut seulement faire le *commit* avec un fichier spécifique on met la commande `git commit -o <fichier>`.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git commit -o hot_dog.txt
```

L'extension `-o` nous permet de choisir les fichiers pour le *commit*.

Après taper la commande on a un affichage montrant quels fichiers vont être *committed* et quels fichiers vont rester dans l'*Index*.

Apparemment il est nécessaire d'ajouter un message, cela est fait en tapant le message toute en haut entre guillemets. (Ici : "*Commande 69*")

```
MINGW64: C:/Users/AldinMUSIC/sandwich
GNU nano 7.2 C:/Users/AldinMUSIC/sandwich/.git/COMMIT_EDITMSG Mod
'Commande 69'|
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Committer: Aldin MUSIC <aldin.music@mewo-campus.fr>
#
# On branch master
# Changes to be committed:
#   modified:   hot_dog.txt
#
# Changes not staged for commit:
#   modified:   burrito.txt
#
```

Pour effectuer les *commit* on fait `ctrl + O` et puis `ctrl + X`.

Avec `git status` on peut voir que seulement *burrito.txt* reste dans l'*Index* et que *hot_dog.txt* n'est plus là.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   burrito.txt
```


En dehors j'ai fait plusieurs changements moi-même donnant l'image suivante quand on fait `git log`.

```
MINGW64:~/sandwich (master)
$ git log
commit a20eaa89e86f56a60aac80b8959e75e0e69cfa3a (HEAD -> master)
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:03:51 2024 +0100

    burrito.txt

commit 80d4a702b457d972ead347b3c9236a901b85585f
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:02:43 2024 +0100

    "Commande 69"

commit bb3deff16459c1cfc32d86646d8962c8861c50b4
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:00:24 2024 +0100

    "Commande 68"

commit 59477c13b664a1d541ff1c5638e89a713da87588
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 11:53:26 2024 +0100
```

On remarque que le dernier *commit* est marqué avec (*HEAD -> master*).

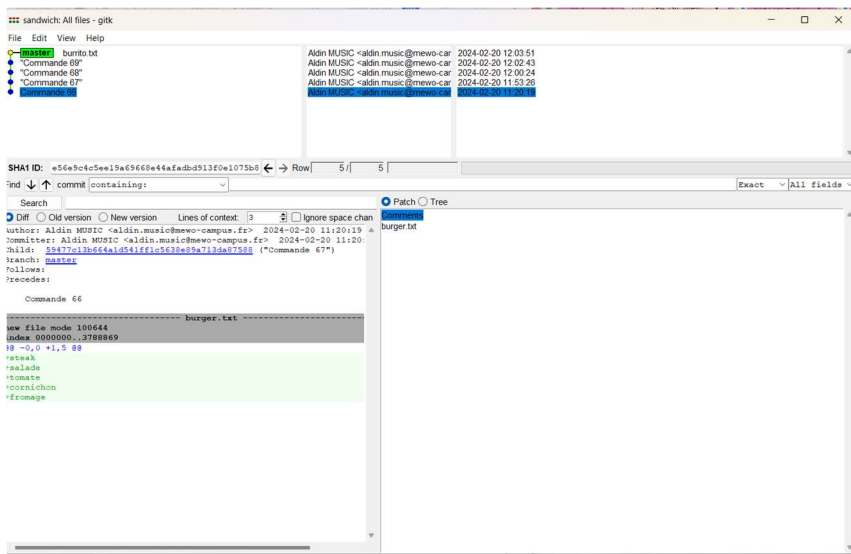
Q2.8 Des différentes *logs*

On a vu le `git log` mais il existe plusieurs affichages du *log*.

Avec `git log --graph --pretty=short` on peut voir le même log mais sans dates, alors plus court.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git log --graph --pretty=short
* commit a20eaa89e86f56a60aac80b8959e75e0e69cfa3a (HEAD -> master)
  | Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
  |
  | burrito.txt
  |
  | * commit 80d4a702b457d972ead347b3c9236a901b85585f
  |   | Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
  |   |
  |   | "Commande 69"
  |   |
  |   | * commit bb3deff16459c1cfc32d86646d8962c8861c50b4
  |   |   | Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
  |   |   |
  |   |   | "Commande 68"
  |   |   |
  |   |   | * commit 59477c13b664a1d541ff1c5638e89a713da87588
  |   |   |   | Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
  |   |   |   |
  |   |   |   | "Commande 67"
```

Pour avoir un log plus détaillé on met la commande *gitk*.



Cela nous donne beaucoup des informations sur les fichiers et ses versions.

Ils existent beaucoup des affichages différents qu'on télécharger sur l'internet.

Q2.9/2.10/2.11/2.12 Annuler les changements

J'ai effectué des changements sur *burger.txt* et j'ai fait un *git add* pour ajouter le fichier dans l'*Index*.

Pour voir cela on fait un *git status*.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   burger.txt
```


Mais cette fois à la place de faire un *commit* on met la commande `git reset burger.txt` et puis `git status`.

```
AzureAD+AladinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git reset burger.txt
Unstaged changes after reset:
M    burger.txt

AzureAD+AladinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   burger.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

On voit que le fichier n'est plus préparé pour un *commit*, il est alors plus dans l'*Index*. Il est revenu dans la *Working copy*, alors dans la phase précédente.

Pour retirer un fichier dans la *Working copy* on met la commande `git checkout burger.txt` et puis `git status` pour voir que le fichier n'est plus là. Ces changements ont été alors supprimés.

```
AzureAD+AladinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git checkout burger.txt
Updated 1 path from the index

AzureAD+AladinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git status
On branch master
nothing to commit, working tree clean
```

Cette même commande peut aussi être utilisée pour un fichier qui est déjà *committed*.

Pour faire cela `git checkout <numéro identifiant>`.

(Ici : `git checkout 80d4a702b457d972ead347b3c9236a901b85585f`)

```
AzureAD+AladinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git checkout 80d4a702b457d972ead347b3c9236a901b85585f
Note: switching to '80d4a702b457d972ead347b3c9236a901b85585f'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

    git switch -c <new-branch-name>

Or undo this operation with:

    git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 80d4a70 "Commande 69"
```

Après un `git status` on voit que ce fichier est la version depuis laquelle on va travailler, il est donc devenu notre fichier initial.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich ((80d4a70...))
$ git status
HEAD detached at 80d4a70
nothing to commit, working tree clean
```

Après un `git log` on remarque que tous les fichiers avant le fichier on vient de retirer manquent. Ces fichiers sont toujours là mais comme ce fichier est maintenant le nouveau fichier initial, il est devenu le `HEAD` causant qu'il est toute en haut dans le log affiché.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich ((80d4a70...))
$ git log
commit 80d4a702b457d972ead347b3c9236a901b85585f (HEAD)
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:02:43 2024 +0100

    "Commande 69"

commit bb3deff16459c1cfc32d86646d8962c8861c50b4
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:00:24 2024 +0100

    "Commande 68"

commit 59477c13b664a1d541ff1c5638e89a713da87588
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 11:53:26 2024 +0100

    "Commande 67"
```

Pour afficher tout le `log` on utilise la commande `git log --all`.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich ((80d4a70...))
$ git log --all
commit a20eaa89e86f56a60aac80b8959e75e0e69cfa3a (master)
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:03:51 2024 +0100

    burrito.txt

commit 80d4a702b457d972ead347b3c9236a901b85585f (HEAD)
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:02:43 2024 +0100

    "Commande 69"

commit bb3deff16459c1cfc32d86646d8962c8861c50b4
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
```

Si on veut retourner sur le fichier *master*, alors le *commit* plus récent, on peut utiliser la commande *git checkout master*.

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich ((80d4a70...))
$ git checkout master
Previous HEAD position was 80d4a70 "Commande 69"
Switched to branch 'master'

AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git log
commit a20eaa89e86f56a60aac80b8959e75e0e69cfa3a (HEAD -> master)
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:03:51 2024 +0100

    burrito.txt

commit 80d4a702b457d972ead347b3c9236a901b85585f
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:02:43 2024 +0100

    "Commande 69"

commit bb3deff16459c1cfc32d86646d8962c8861c50b4
Author: Aldin MUSIC <aldin.music@mewo-campus.fr>
Date: Tue Feb 20 12:00:24 2024 +0100
```

```
AzureAD+AldinMUSIC@SLAM-1349 MINGW64 ~/sandwich (master)
$ git status
On branch master
nothing to commit, working tree clean
```

On voit que le fichier *master* est redevenu le *HEAD* et après faire un *git status* on voit aussi que le message que le *HEAD* est détaché n'est plus là.