Guastamacchia Livio

Tp GIT :

Introduction :

Lors de ce TP, nous allons utiliser le logiciel GIT. Qui nous allons créer un répertoire de travail et en faire la maintenance de manière ordonnée et structurer.

2.1/ Création d'un nouveau dépôt

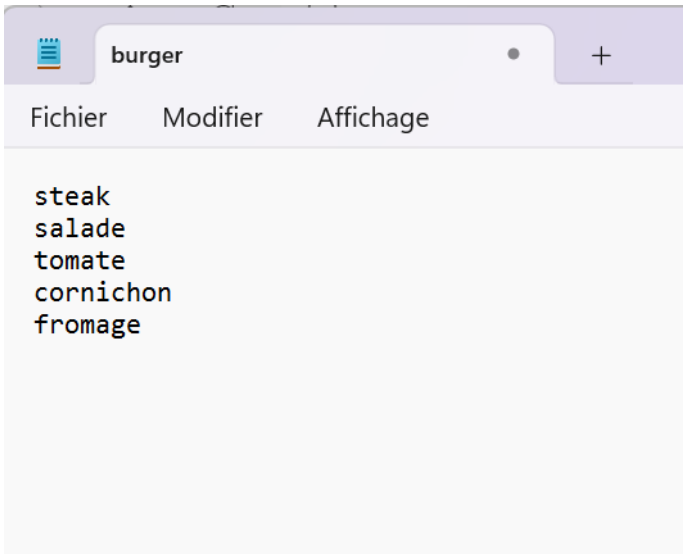
2.1/ Premièrement dans git on va créer une zone de dépôt nommer sandwich grâce à la commande

"git init" + le nom de mon fichier (notamment nommer sandwich dans ce TP)

```
guast@livio MINGW64 ~
$ git init sandwich
Initialized empty Git repository in C:/Users/guast/sandwich/.git/

guast@livio MINGW64 ~
$
```

Dans ce fichier on va créer un document .txt nommer burger

Où dedans on y écrira la liste des ingrédients d'un burger



2.2/ Add et commit

2.2/ vérifions l'état de notre dépôt avec "git status"

```
guast@livio 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)

guast@livio MINGW64 ~/sandwich (master)
```

2.3/ on vas utilisé la commande “git add” pour préparer le fichier burger.txt au commit

```
guast@livio MINGW64 ~/sandwich (master)
$ git add burger.txt

guast@livio MINGW64 ~/sandwich (master)
$
```

Puis utiliser “git status” à nouveau pour pouvoir vérifier que les modifications ont bien été enregistrer dans l’index

```
guast@livio 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
```

En utilisant la commande “gif diff –cached" cela me permet de voir les différences qu’il y a entre l'index actuel et le repository

```
guast@livio MINGW64 ~/sandwich (master)
$ git diff --cached
diff --git a/burger.txt b/burger.txt
new file mode 100644
index 0000000..d106d82
--- /dev/null
+++ b/burger.txt
@@ -0,0 +1,5 @@
+steak
+salade
+tomate
+cornichon
+fromage
\ No newline at end of file

guast@livio MINGW64 ~/sandwich (master)
$
```

2.4/ la commande “git commit –m” permet de sauvegarder la version actuelle du fichier dans le repository.

```
guast@livio MINGW64 ~/sandwich (master)
$ git commit -m "<premiere_version_burger>"
Author identity unknown

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'guast@livio.(none)')
```

Après avoir entré la commande, le logiciel me dit qu'il ne connaît pas l'identité de la personne ayant taper cette commande

Donc j'utilise la commande "git config --global user.email " livio@test.com" pour pouvoir m'authentifier auprès de git

```
guast@livio MINGW64 ~/sandwich (master)
$ git config --global user.email "livio@test.com"
```

Maintenant en tapant à nouveau la commande "git commit –m "premiere_version_burger"" on remarque que git ne me demande plus d'authentification car la commande effectuer au préalable à fonctionner

```
guast@livio MINGW64 ~/sandwich (master)
$ git commit -m "<premiere_version_burger>"
[master (root-commit) d51f375] <premiere_version_burger>
 1 file changed, 5 insertions(+)
 create mode 100644 burger.txt
```

2.5/ Je vais revérifier avec la commande "git status" si mes modifications ont bien été comités

```
guast@livio MINGW64 ~/sandwich (master)
$ git status
On branch master
nothing to commit, working tree clean
```

2.6/ En utilisant la commande "git log" je vois la liste des commit effectuer, je vois qu'il y a un changement dans le dépôt.

```
guast@livio MINGW64 ~/sandwich (master)
$ git log
commit d51f375fead85159513eac3343d9edbe8efc24c6 (HEAD -> master)
Author: livio guasta <livio@test.com>
Date:   Thu Jan 4 17:05:38 2024 +0100

    <premiere_version_burger>
```

Le numéro du dernier commit effectuer est :

```
commit d51f375fead85159513eac3343d9edbe8efc24c6
```

2.7/

Première modification je créer un fichier txt « jambon_beurre » :

on ajoute le fichier « jambon_beurre » avec « git add »

```
guast@livio MINGW64 ~/sandwich (master)
$ git add jambon_beurre.txt
```

on fais git status on vois qu'il y a un nouveau fichier dans l'index

```
guast@livio MINGW64 ~/sandwich (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   jambon_beurre.txt
```

on commit le fichier jambon beurre

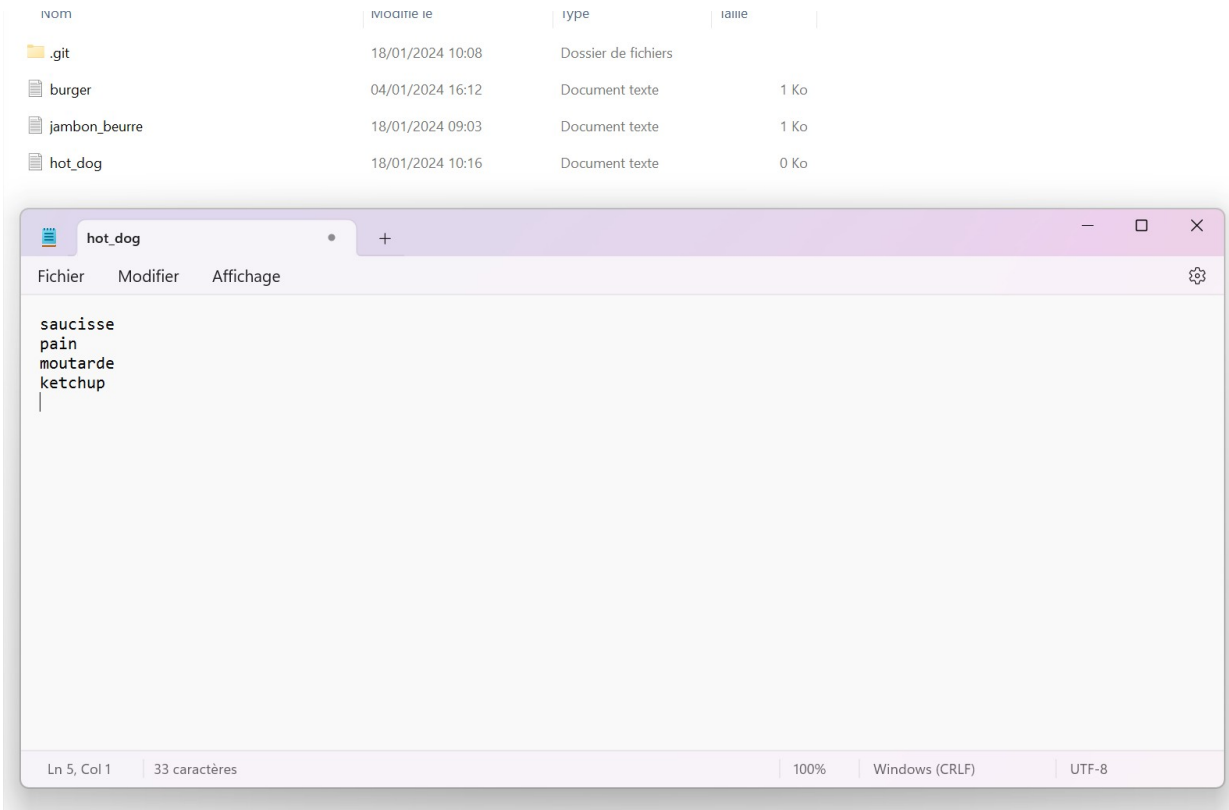
```
guast@livio MINGW64 ~/sandwich (master)
$ git commit -m "ajout_jambon_beurre"
[master 36ea1d9] ajout_jambon_beurre
 1 file changed, 3 insertions(+)
 create mode 100644 jambon_beurre.txt
```

maintenant dans les log on voit deux modification

```
guast@livio MINGW64 ~/sandwich (master)
$ git log
commit 36ea1d97b18ec23cddd48f2f96c486db07ee047f (HEAD -> master)
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:08:54 2024 +0100

    ajout_jambon_beurre

commit d51f375fead85159513eac3343d9edbe8efc24c6
Author: livio guasta <livio@test.com>
Date:   Thu Jan 4 17:05:38 2024 +0100

    <premiere_version_burger>
```

Deuxième modification je créer un fichier txt « hot_dog »

on créer dans le dépôt un fichier txt « hot_dog »



on ajoute le fichier hot dog a l'index

```
guast@livio MINGW64 ~/sandwich (master)
$ git add hot_dog.txt
```

on voit bien que le fichier hot dog a été ajouter avec « git status »

```
guast@livio MINGW64 ~/sandwich (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   hot_dog.txt
```

on commit le fichier hot dog

```
guast@livio MINGW64 ~/sandwich (master)
$ git commit -m "ajout_hot_dog"
[master a20b7b0] ajout_hot_dog
 1 file changed, 4 insertions(+)
 create mode 100644 hot_dog.txt
```

on vois grâce a « git log » que le fichier a bien été commit

```
guast@livio MINGW64 ~/sandwich (master)
$ git log
commit a20b7b0a27e689b2b174541f09d10005c1a95751 (HEAD -> master)
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:21:23 2024 +0100

    ajout_hot_dog

commit 36ea1d97b18ec23cddd48f2f96c486db07ee047f
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:08:54 2024 +0100

    ajout_jambon_beurre

commit d51f375fead85159513eac3343d9edbe8efc24c6
Author: livio guasta <livio@test.com>
Date:   Thu Jan 4 17:05:38 2024 +0100

    <premiere_version_burger>
```

avec « git diff » on peux voir que l'index est a jours par rapport a la working copy et grâce a « git diff –cached » on peut voir que le repository est a jours avec l'index.

```
guast@livio MINGW64 ~/sandwich (master)
$ git diff

guast@livio MINGW64 ~/sandwich (master)
$ git diff --cached
```

Troisième modification, modification du fichier burger :

les modifications que je vais effectuer sur le document sont :

- retirer cornichon

- retirer fromage

- ajouter miel

- ajouter fromage de chèvre

- ajouter épice

en faisant « git diff » on voit bien que le fichier a été modifier et n'est pas a jours dans l'index

```
guast@livio MINGW64 ~/sandwich (master)
$ git diff
diff --git a/burger.txt b/burger.txt
index d106d82..6fbc1c2 100644
--- a/burger.txt
+++ b/burger.txt
@@ -1,5 +1,6 @@
 steak
 salade
+fromage de chèvre
+miel
 tomate
-cornichon
-fromage
\ No newline at end of file
+épice
\ No newline at end of file
```

on ajoute le fichier a l'index

```
guast@livio MINGW64 ~/sandwich (master)
$ git add burger.txt
```

on voit bien que le fichier burger a été modifier et est prêt a être commit

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

on commit le fichier burger

```
guast@livio MINGW64 ~/sandwich (master)
$ git commit -m "modif_burger"
[master 1b1ed93] modif_burger
 1 file changed, 3 insertions(+), 2 deletions(-)
```

on voit bien dans les log que le fichier a été commit

```
guast@livio MINGW64 ~/sandwich (master)
$ git log
commit 1b1ed937a14f5736dd66dc2401fdd88c5fe6b4d7 (HEAD -> master)
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:38:37 2024 +0100

    modif_burger
```

l'index et le repository sont bien a jours

```
guast@livio MINGW64 ~/sandwich (master)
$ git diff

guast@livio MINGW64 ~/sandwich (master)
$ git diff --cached
```

quatrième modification, modification fichier « hot_dog » :

les modifications que je vais effectuer sur le fichier « hot_dog » sont :

- retirer saucisse

- retirer moutarde

- retirer ketchup

- ajouter saucisse blanche

- ajouter mayonnaise

avec « git dif » on voit bien les modification effectuer

```
guast@livio MINGW64 ~/sandwich (master)
$ git diff
diff --git a/hot_dog.txt b/hot_dog.txt
index 96a587c..e455051 100644
--- a/hot_dog.txt
+++ b/hot_dog.txt
@@ -1,4 +1,3 @@
-saucisse
+saucisse blanche
 pain
-moutarde
-ketchup
+mayonnaise
```

on ajoute hot dog pour actualiser l'index

```
guast@livio MINGW64 ~/sandwich (master)
$ git add hot_dog.txt
```

on commit le fichier

```
guast@livio MINGW64 ~/sandwich (master)
$ git log
commit 18f3fffbbecece6d3cad580c68046798ad5bcae9 (HEAD -> master)
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:54:15 2024 +0100

    modif_hot_dog
```

l index et le repository sont tous deux a jours

```
guast@livio MINGW64 ~/sandwich (master)
$ git diff

guast@livio MINGW64 ~/sandwich (master)
$ git diff --cached
```

Cinquième modification, modifcation fichier « jambon_beurre » :

les modifcations que je vais effectuer sur ce fichier sont :

- retirer jambon

- ajouter jambon fumée

- ajouter salade

on voit bien que mon fichier a été modifier

```
guast@livio MINGW64 ~/sandwich (master)
$ git diff
diff --git a/jambon_beurre.txt b/jambon_beurre.txt
index dda64e5..8bf123f 100644
--- a/jambon_beurre.txt
+++ b/jambon_beurre.txt
@@ -1,3 +1,4 @@
-jambon
+jambon fumée
 beurre
-pain
\ No newline at end of file
+pain
+salade
\ No newline at end of file
```

on ajoute jambon beurre a l'index pour l'actualiser

```
guast@livio MINGW64 ~/sandwich (master)
$ git add jambon_beurre.txt
```

on voit bien que le fichier a été ajouter a l'index et est prêt a être commit

```
guast@livio MINGW64 ~/sandwich (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   jambon_beurre.txt
```

on commit

```
guast@livio MINGW64 ~/sandwich (master)
$ git commit -m "modif_jambon_beurre"
[master 3507897] modif_jambon_beurre
 1 file changed, 3 insertions(+), 2 deletions(-)
```

le commit a bien été effectuer

```
guast@livio MINGW64 ~/sandwich (master)
$ git log
commit 3507897ef99d57dee64f11624ebc7ed73c8077f4 (HEAD -> master)
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 11:05:00 2024 +0100

    modif_jambon_beurre
```

l'index et le repository sont bien a jours

```
guast@livio MINGW64 ~/sandwich (master)
$ git diff

guast@livio MINGW64 ~/sandwich (master)
$ git diff --cached
```

2.8/

voici le test de la commande « git log » :

```
guast@livio MINGW64 ~/sandwich (master)
$ git log
commit 3507897ef99d57dee64f11624ebc7ed73c8077f4 (HEAD -> master)
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 11:05:00 2024 +0100

    modif_jambon_beurre

commit 18f3fffbbecece6d3cad580c68046798ad5bcae9
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:54:15 2024 +0100

    modif_hot_dog

commit 1b1ed937a14f5736dd66dc2401fdd88c5fe6b4d7
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:38:37 2024 +0100

    modif_burger

commit a20b7b0a27e689b2b174541f09d10005c1a95751
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:21:23 2024 +0100

    ajout_hot_dog

commit 36ea1d97b18ec23cddd48f2f96c486db07ee047f
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:08:54 2024 +0100

    ajout_jambon_beurre

commit d51f375fead85159513eac3343d9edbe8efc24c6
Author: livio guasta <livio@test.com>
Date:   Thu Jan 4 17:05:38 2024 +0100

    <premiere_version_burger>
```

la commande nous permet de voir notre historique de commit.

voici le test de la commande « git log –graph –pretty=short » :

```
guast@livio MINGW64 ~/sandwich (master)
$ git log --graph --pretty=short
* commit 3507897ef99d57dee64f11624ebc7ed73c8077f4 (HEAD -> master)
| Author: livio guasta <livio@test.com>
|
|     modif_jambon_beurre
|
* commit 18f3fffbbecece6d3cad580c68046798ad5bcae9
| Author: livio guasta <livio@test.com>
|
|     modif_hot_dog
|
* commit 1b1ed937a14f5736dd66dc2401fdd88c5fe6b4d7
| Author: livio guasta <livio@test.com>
|
|     modif_burger
|
* commit a20b7b0a27e689b2b174541f09d10005c1a95751
| Author: livio guasta <livio@test.com>
|
|     ajout_hot_dog
|
* commit 36ea1d97b18ec23cddd48f2f96c486db07ee047f
| Author: livio guasta <livio@test.com>
|
|     ajout_jambon_beurre
|
* commit d51f375fead85159513eac3343d9edbe8efc24c6
  Author: livio guasta <livio@test.com>

      <premiere_version_burger>
```

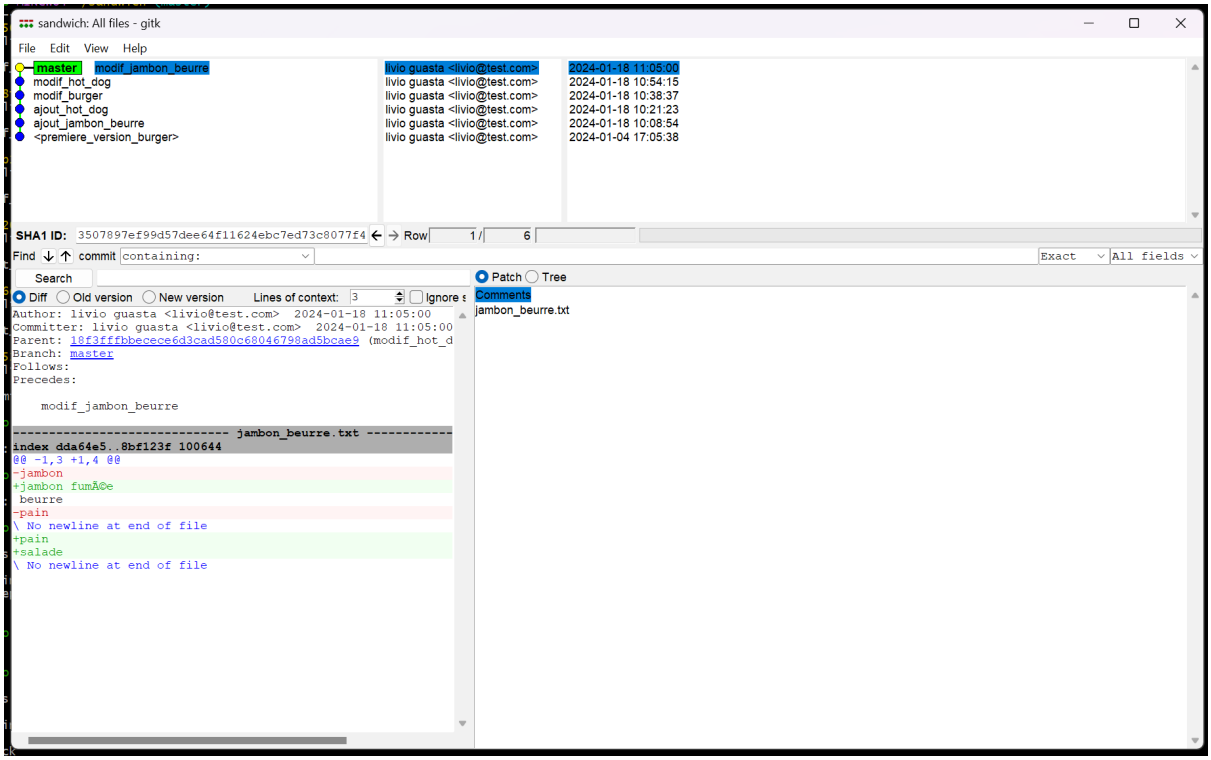
la commande nous permet de voir notre historique de commit mais en nous indiquant ou commence et termine le commit comparer a « git log ».

voici le test de la commande « gitg » :

```
guast@livio MINGW64 ~/sandwich (master)
$ gitg
bash: gitg: command not found
```

sur windows « gitg » n'est pas installer par défaut

voici le test de la commande « gitk » :



la commande permet de voir les anciennes version, les nouvelles et toute les modification de chaque fichier que l'on a effectuer sur le dépôt sandwich.

2.3/ Voyage dans le temps

2.9/ on va modifier tous les fichiers, les modifications effectuer sur chaque fichier sont :

Pour jambon beurre :

- retirer jambon fumée
- ajouter tomate

Pour hot dog :

- retirer saucisse blanche
- retirer mayonnaise
- ajouter merguez
- ajouter ketchup
- ajouter moutarde

Pour burger :

- retirer steak
- retirer fromage de chèvre
- retirer miel
- retirer épice
- ajouter steak de thon
- ajouter ketchup
- ajouter fromage

on voit bien que tous les fichier son modifier et que l'index n'est pas a jours

```
guast@livio MINGW64 ~/sandwich (master)
$ git diff
diff --git a/burger.txt b/burger.txt
index 6fbc1c2..414bc3c 100644
--- a/burger.txt
+++ b/burger.txt
@@ -1,6 +1,5 @@
-steak
+steak de thon
 salade
-fromage de chèvre
-miel
+fromage
 tomate
-épice
\ No newline at end of file
+ketchup
\ No newline at end of file
diff --git a/hot_dog.txt b/hot_dog.txt
index e455051..9154fbb 100644
--- a/hot_dog.txt
+++ b/hot_dog.txt
@@ -1,3 +1,4 @@
-saucisse blanche
+merguez
 pain
-mayonnaise
+ketchup
+moutarde
\ No newline at end of file
diff --git a/jambon_beurre.txt b/jambon_beurre.txt
index 8bf123f..2fe38f2 100644
--- a/jambon_beurre.txt
+++ b/jambon_beurre.txt
@@ -1,4 +1,5 @@
-jambon fumée
+jambon
 beurre
 pain
-salade
\ No newline at end of file
+salade
+tomate
\ No newline at end of file
```

on va ajouter tous les fichier a l'index pour qu'il s'actualise avec « git add --all »

```
guast@livio MINGW64 ~/sandwich (master)
$ git add --all
```

grâce a « git status » on vois bien que les 3 fichiers on été actualiser dans l'index.

```
guast@livio MINGW64 ~/sandwich (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   burger.txt
        modified:   hot_dog.txt
        modified:   jambon_beurre.txt
```

On va reset tous les fichiers maintenant avec la commande « git reset --hard »

on voit que l'on est retourner au dernier commit donc le commit « modif_jambon_beurre »

```
guast@livio MINGW64 ~/sandwich (master)
$ git reset --hard
HEAD is now at 3507897 modif_jambon_beurre
```

on vois bien avec « git status » que il n'y a plus de fichier modifier qui sont prêt a être commit

```
guast@livio MINGW64 ~/sandwich (master)
$ git status
On branch master
nothing to commit, working tree clean
```

2.10/ avec la commande « git checkout » on peut supprimer les modifications que l'on avait fait

```
guast@livio MINGW64 ~/sandwich (master)
$ git checkout jambon_beurre.txt
Updated 0 paths from the index

guast@livio MINGW64 ~/sandwich (master)
$ git checkout burger.txt
Updated 0 paths from the index

guast@livio MINGW64 ~/sandwich (master)
$ git checkout hot_dog.txt
Updated 0 paths from the index
```

on voit que il n'y a rien a commit

```
guast@livio MINGW64 ~/sandwich (master)
$ git status
On branch master
nothing to commit, working tree clean
```

2.11/La commande « git checkout commitid » nous permet de retourner au commit sélectionner, j'ai sélectionner le commitid de modif burger.

```
guast@livio MINGW64 ~/sandwich (master)
$ git checkout 1b1ed937a14f5736dd66dc2401fdd88c5fe6b4d7
Note: switching to '1b1ed937a14f5736dd66dc2401fdd88c5fe6b4d7'.

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 1b1ed93 modif_burger
```

On voit bien que dans les log on s’arrête au commit de « modif_burger »

```
guast@livio MINGW64 ~/sandwich ((1b1ed93...))
$ git log
commit 1b1ed937a14f5736dd66dc2401fdd88c5fe6b4d7 (HEAD)
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:38:37 2024 +0100

    modif_burger

commit a20b7b0a27e689b2b174541f09d10005c1a95751
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:21:23 2024 +0100

    ajout_hot_dog

commit 36ea1d97b18ec23cddd48f2f96c486db07ee047f
Author: livio guasta <livio@test.com>
Date:   Thu Jan 18 10:08:54 2024 +0100

    ajout_jambon_beurre

commit d51f375fead85159513eac3343d9edbe8efc24c6
Author: livio guasta <livio@test.com>
Date:   Thu Jan 4 17:05:38 2024 +0100

    <premiere_version_burger>
```

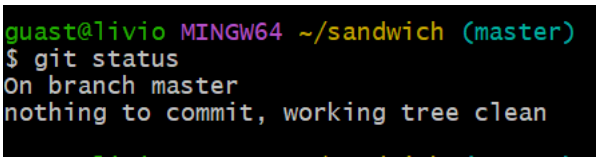
« git status » nous affiche maintenant que l’on continue a partir du commit « modif_burger »

```
guast@livio MINGW64 ~/sandwich ((1b1ed93...))
$ git status
HEAD detached at 1b1ed93
nothing to commit, working tree clean
```

2.12/ la commande « git checkout master » nous permet de retourner au dernier commit que nous avons effectuer.

```
guast@livio MINGW64 ~/sandwich ((1b1ed93...))
$ git checkout master
Previous HEAD position was 1b1ed93 modif_burger
Switched to branch 'master'
```

« git status » nous affiche maintenant que nous somme revenu au dernier commit



Conclusion :

Lors de ce TP j'ai rencontré quelque problème :

Le fait de ne pas connaître la commande pour continuer de travailler sur le répertoire que j'avais créé, la commande est « cd + le nom du répertoire ». Je n'avais pas bien compris aussi comment fonctionnait GIT avec la working copy, l'index et le repository après avoir compris cela le TP est devenu beaucoup plus facile à réaliser.