

Cultures numériques avancées

Git : outil de versionnement du code

Ljudmila PETKOVIC

ljudmila.petkovic@sorbonne-nouvelle.fr

```
~/Documents/linux-commands via v3.9.6
> ls -lah
Permissions Size User Date Modified Name
drwxr-xr-x - daniel 8 ago 15:11 .
drwxr-xr-x - daniel 8 ago 00:27 ..
drwxr-xr-x - daniel 8 ago 00:34 commands
drwxr-xr-x - daniel 7 ago 00:45 dir1
drwxr-xr-x - daniel 7 ago 00:45 dir2
drwxr-xr-x - daniel 8 ago 00:10 dir_to_copy
drwxr-xr-x - daniel 8 ago 00:12 new_dir
-rw-r--r-- 0 daniel 8 ago 00:38 BestMoviesOfAllTime
-rw-r--r-- 0 daniel 7 ago 00:44 binarysearch.py
-rw-r--r-- 0 daniel 7 ago 00:43 dummyfile1.txt
-rw-r--r-- 0 daniel 8 ago 00:18 file_to_delete.txt
-rw-r--r-- 0 daniel 7 ago 00:44 get_keys.py
-rw-r--r-- 0 daniel 7 ago 00:44 github_automation.py
-rw-r--r-- 0 daniel 7 ago 00:44 important_file.txt
-rw-r--r-- 0 daniel 8 ago 00:04 new_file.txt
-rw-r--r-- 0 daniel 12 abr 20:45 old_file
```

Source : [Diaz, 2023](#).

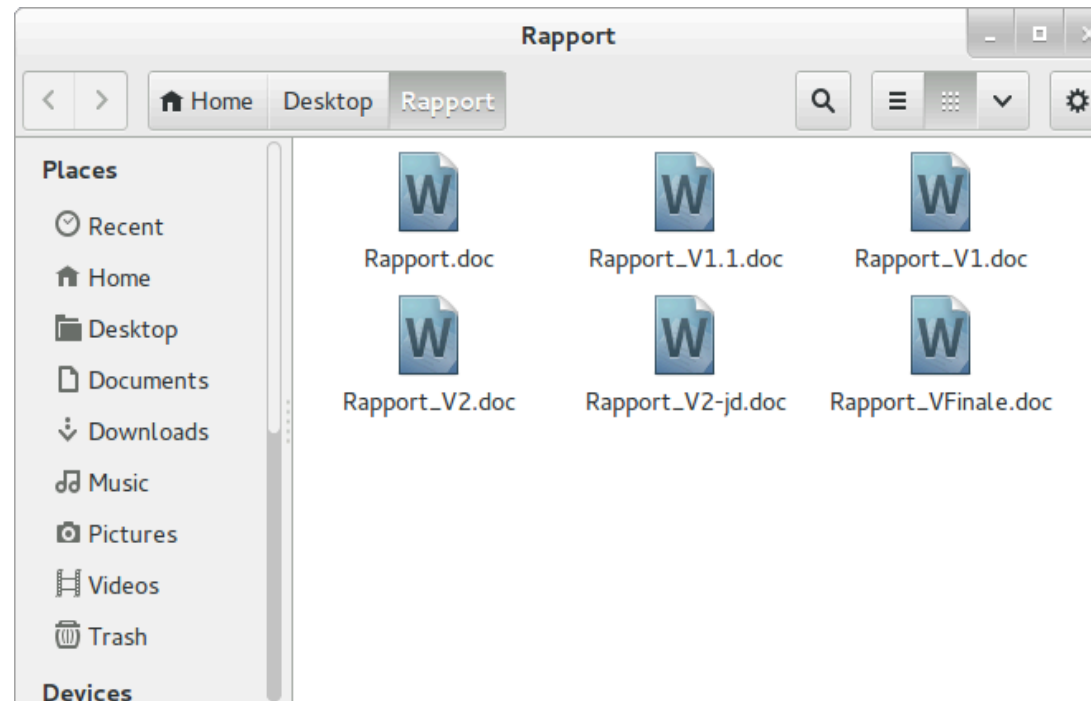
Cultures numériques avancées (L2HN001)

Licence *Sciences du langage*, mineure « Humanités numériques »

Paris, le 22 mars 2024, année 2023-2024

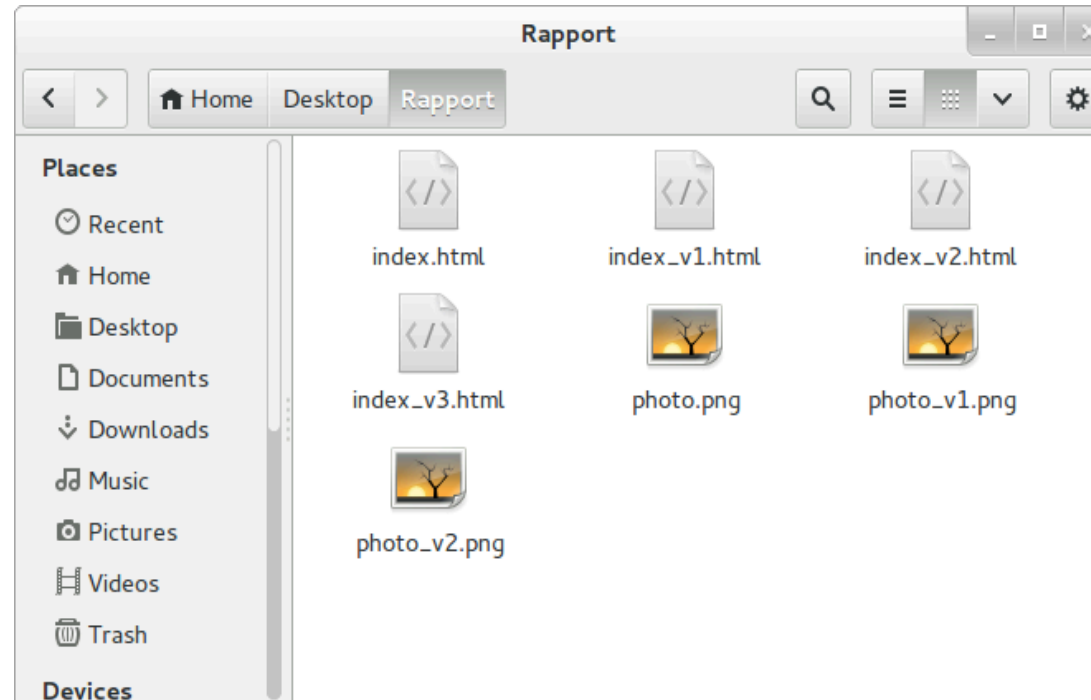
Diapositives adaptées de l'*IUT Lyon 1* et de *Thibault Clérice*

Problème 1



- la version la plus à jour est-elle Rapport.doc ou Rapport_VFinale.doc ?
- et si on avait aussi Rapport_VFinale1.doc et Rapport_VFinale2.doc ?
- les versions n'apparaissent pas dans l'ordre (1.1 , 1 , 2)
- la version 2-jd vient elle avant ou après la version 2 ?

Problème 1.1



- les versions de l'image sont-elles numérotées indépendamment, ou par rapport aux versions de la page ?
- nécessité de renommer les fichiers pour visualiser une ancienne version (pour que les liens fonctionnent)

Besoins des utilisateur·trice·s

Les utilisateur·trice·s ont donc besoin de :

- comprendre les versions
- pouvoir revenir en arrière, avoir une « trace » des documents
- pouvoir avoir une collaboration simple

Gestion des versions

- la gestion des versions est un travail fastidieux et méthodique
- les humains ne sont pas doués pour ce genre de travail
- laissons cela à l'ordinateur et concentrons-nous sur la partie du travail où nous sommes meilleurs que l'ordinateur

→ VCS (*Version Control System*)

Git : un outil de versionnage

- plusieurs applications :
 - gestion de code source pour les projets logiciels
 - rédaction de la documentation
 - création d'un site web
- permet un travail collaboratif, grâce à la :
 - facilité d'échange
 - traçabilité
 - gestion des conflits

Notions de base

Dépôt

- (angl. *repository*, abbr. *repo*)
- répertoire et entrepôt virtuel d'un projet qui permet d'enregistrer les versions des documents et d'y accéder au besoin
- il contient un répertoire (caché) `.git` avec toutes les données dont Git a besoin pour gérer l'historique des données

Vous pouvez démarrer un dépôt Git de deux manières :

1. transformer le répertoire existant en dépôt Git (initialiser le dépôt) : `git init`
2. *cloner* un dépôt Git existant depuis un serveur distant sur votre machine, p. ex. :

```
git clone https://github.com/ljpetkovic/L2HN001.git
```


Télécharger vs. *cloner* le dépôt Git

- lorsque vous **téléchargez** le dépôt, vous ne récupérez que les fichiers-source sans le répertoire `.git`, donc vous n'avez pas le dépôt Git fonctionnel
 - toutes vos modifications ne resteront que dans votre répertoire local
 - impossible de synchroniser le dépôt local avec le dépôt distant
- lorsque vous **clonez**, vous obtenez une copie de l'historique des modifications et c'est un dépôt Git fonctionnel
 - vous pouvez synchroniser le dépôt local avec le dépôt distant

Copie de travail

(angl. *working copy*)

- les fichiers effectivement présents dans le répertoire géré par Git (en local)
 - **NB** : leur état peut être différent de celui enregistré dans le *dépôt distant* (cf. slide 24).

Nom	Taille	Dernière modification
1_Ligne_de_commande	23 éléments	11:52
2_Git	3 éléments	13:28
3_Regex	0 élément	30 janv.
4_Voyant_Tools	0 élément	30 janv.
extra	6 éléments	12 mars
README.md	3,3 ko	21 févr.
test.sh	246 octets	7 févr.

Figure 1. Copie de travail.

ljpeticovic renommer le dossier a419ecd 1 hour ago 41 commits		
1_Ligne_de_commande	Git Bash	last month
2_Git	renommer le dossier	1 hour ago
.gitignore	déplacer .gitignore	last month
README.md	autoriser l'exécution des scripts PowerShell sous Windows	last month

Figure 2. Dépôt distant (sur [GitHub](#), serveur centralisé pour Git).

Commit

Pour qu'une modification soit enregistrée, il faut que cela soit explicité (≠ p. ex. Google Drive, où les modifications sont automatiquement enregistrées).

- ces modifications archivées sont appelées *commits*
 - « commiter » : enregistrer la nouvelle version d'un fichier sur un système de gestion de configuration
- les commits possèdent une date, un auteur, une description textuelle et un identifiant SHA-1 (valeur *hash*)
- les commits peuvent comporter plusieurs fichiers
- les fichiers qui ont subi des modifications doivent y être ajoutés explicitement
 - pour commiter les modifications indexées

```
git commit -m "Ceci est mon premier commit"
```

Commit

- l'historique d'un projet est une séquence de « photos » (commits), contenant l'état de tous les fichiers du projet
- pour afficher la liste des commits en ligne de commande (historique) : `git log`

```
(base) ljudmila@ljudmila-Latitude-5580:~/Bureau/mineure_HN/COURS/L2HN001/2_Git$ git log
commit a419ecd7f09ea003ee3a631d9551e88f7e5ee793 (HEAD -> main, origin/main)
Author: ljpetkovic <ljuca94@gmail.com>
Date:   Wed Mar 22 11:53:39 2023 +0100

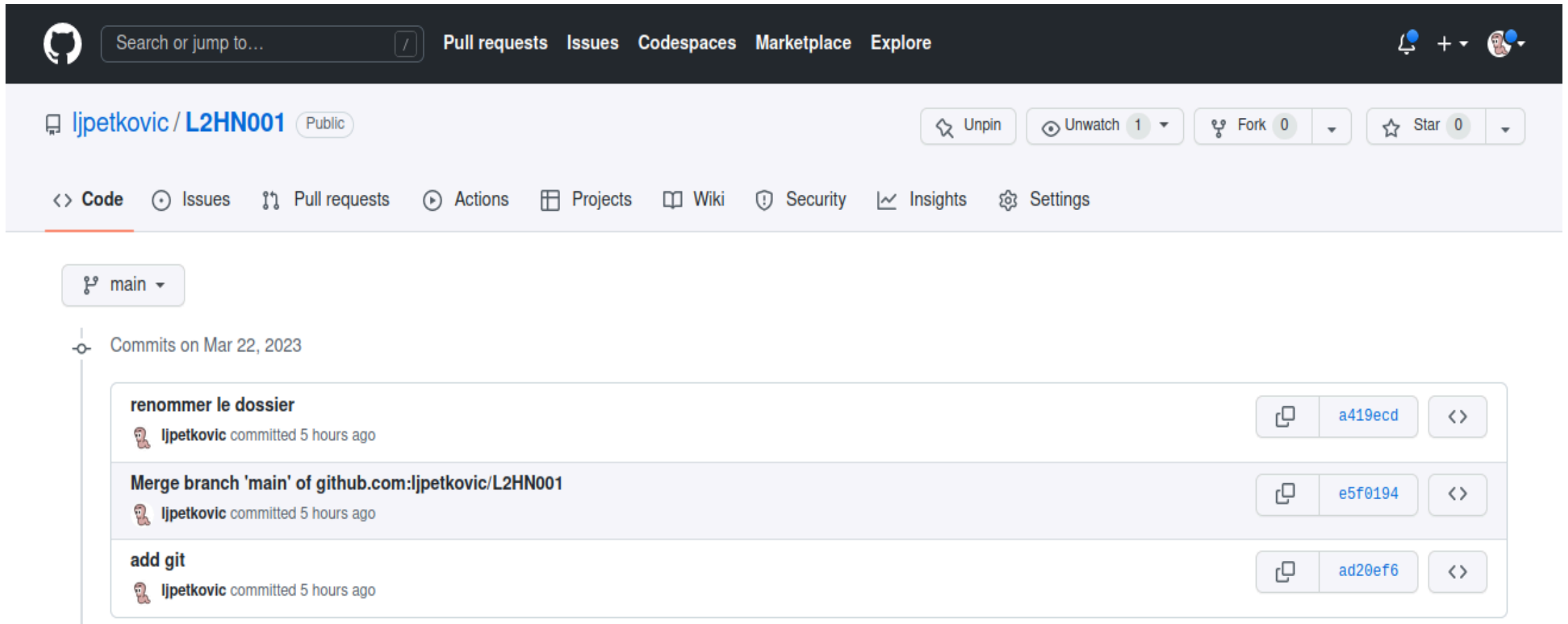
    renommer le dossier

commit e5f0194cb8065a08c7d690721640c04b450aa86b
Merge: ad20ef6 4a3a98f
Author: ljpetkovic <ljuca94@gmail.com>
Date:   Wed Mar 22 11:52:48 2023 +0100

    Merge branch 'main' of github.com:ljpetkovic/L2HN001
...
```

Commit

Liste des commits sur GitHub :



The screenshot displays the GitHub interface for the repository `ljpetkovic / L2HN001`. The top navigation bar includes links for Pull requests, Issues, Codespaces, Marketplace, and Explore. Below the repository name, there are buttons for Unpin, Unwatch (1), Fork (0), and Star (0). The main navigation bar shows links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The commit history is shown for the `main` branch, with a filter for commits on Mar 22, 2023. Three commits are listed:

Commit Message	Author	Time	SHA-1	View Code
renommer le dossier	ljpetkovic	committed 5 hours ago	a419ecd	<>
Merge branch 'main' of github.com:ljpetkovic/L2HN001	ljpetkovic	committed 5 hours ago	e5f0194	<>
add git	ljpetkovic	committed 5 hours ago	ad20ef6	<>

Commit

- afficher le détail d'un commit particulier : `git show <identifiant_du_commit>`

```
(base) ljudmila@ljudmila-Latitude-5580:~/Bureau/mineure_HN/COURS/L2HN001/2_Git$ git show a419ecd7f09e  
a003ee3a631d9551e88f7e5ee793  
commit a419ecd7f09ea003ee3a631d9551e88f7e5ee793 (HEAD -> dev, origin/main, main)  
Author: ljetkovic <ljuca94@gmail.com>  
Date:   Wed Mar 22 11:53:39 2023 +0100
```

renommer le dossier

```
diff --git a/3_Git/Git.md b/2_Git/Git.md  
similarity index 100%  
rename from 3_Git/Git.md
```

Index

- espace temporaire contenant les modifications (création, modification ou suppression des fichiers) prêtes à être « commitées »
- fichier binaire (stocké dans le répertoire `.git`) contenant une liste triée de noms de chemin, chacun avec des autorisations et l'identifiant SHA1 d'un objet *blob* (angl. *binary large object*) ;

La commande `git ls-files --stage` montre le contenu de l'index :

```
(base) liudmila@liudmila-Latitude-5580:~/Bureau/mineure HN/COURS/L2HN001/2_Git$ git ls-files --stage
100644 7c9b106c72c98d58d057ed71a5b113310f217f15 0 Git.md
100644 ed8b5d00f671294eb650bc898382a7809fa6a75c 0 Git.pdf
100644 37632c24cb278be02344b623da4f3a2a49e51b58 0 img/commit.png
```

Permission (en l'occurrence le fichier régulier non exécutable)

Identifiant SHA-1 d'un objet blob

Nom de chemin

Index

- ajouter un fichier dans l'index (ajouter des modifications) : `git add test.txt`
 - ajouter tous les fichiers modifiés : `git add -A`
- retirer un fichier de l'index : `git reset test.txt`
- voir la liste des modifications réalisées (différence entre l'état archivé et actuel)
 - résumé : `git status`
 - détails de changements : `git diff`

Résumé des modifications

```
(base) ljudmila@ljudmila-Latitude-5580:~/Bureau/mineure_HN/COURS/L2HN001/2_Git$ git status
```

Sur la branche main

Votre branche est à jour avec 'origin/main'.

Modifications qui ne seront pas validées :

(utilisez "git add/rm <fichier>..." pour mettre à jour ce qui sera validé)

(utilisez "git restore <fichier>..." pour annuler les modifications dans le répertoire de travail)

modifié : Git.pdf

supprimé : img/commit.png

Fichiers non suivis:

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

../1_Ligne_de_commande/Bash_recap.html

...

aucune modification n'a été ajoutée à la validation (utilisez "git add" ou "git commit -a")

Détails des changements

```
(base) ljudmila@ljudmila-Latitude-5580:~/Bureau/mineure_HN/COURS/L2HN001/2_Git$ git diff
diff --git a/1_Ligne_de_commande/commandes/coucou_2.sh b/1_Ligne_de_commande/commandes/coucou_2.sh
deleted file mode 100644
index 36532ad..0000000
--- a/1_Ligne_de_commande/commandes/coucou_2.sh
+++ /dev/null
@@ -1,3 +0,0 @@
-#!/bin/bash
-nom="Michel"
-echo "Coucou $nom"
diff --git a/2_Git/Git.md b/2_Git/Git.md
index 7c9b106..27795a3 100644
--- a/2_Git/Git.md
+++ b/2_Git/Git.md
@@ -15,7 +15,6 @@ section {
     color: black;
 }

-
h1 {
    color: DarkBlue;
}
@@ -98,10 +97,6 @@ Paris, le 21 mars 2023

---
:
```

Branches

Dans certaines situations, on peut souhaiter faire cohabiter et évoluer *plusieurs* versions divergentes du même projet.

Ces versions peuvent parfois converger à nouveau (mais pas forcément).

Une branche est la *lignée* (généalogique) de commits, à laquelle on a donné un nom.

- par commodité, la version-maître est nommée `main` ou `master`, et les variantes `dev`, `test` ou autre.

Branches

Exemple : CV

Pour un CV, on souhaite avoir :

- une version « maître » que l'on maintient à jour,
- des variantes pour chaque demande d'emploi, adaptées en fonction de l'employeur visé.

Branches



NB : L'historique n'a plus une structure linéaire, mais *arborescente* (ce qui justifiera la métaphore de la « branche »).

Branches

- créer une nouvelle branche : `git branch [nom de la branche]` (ex. *dev*)
- obtenir la liste des branches courantes : `git branch`
 - l'étoile indique sur quelle branche nous travaillons (en l'occurrence *main*)

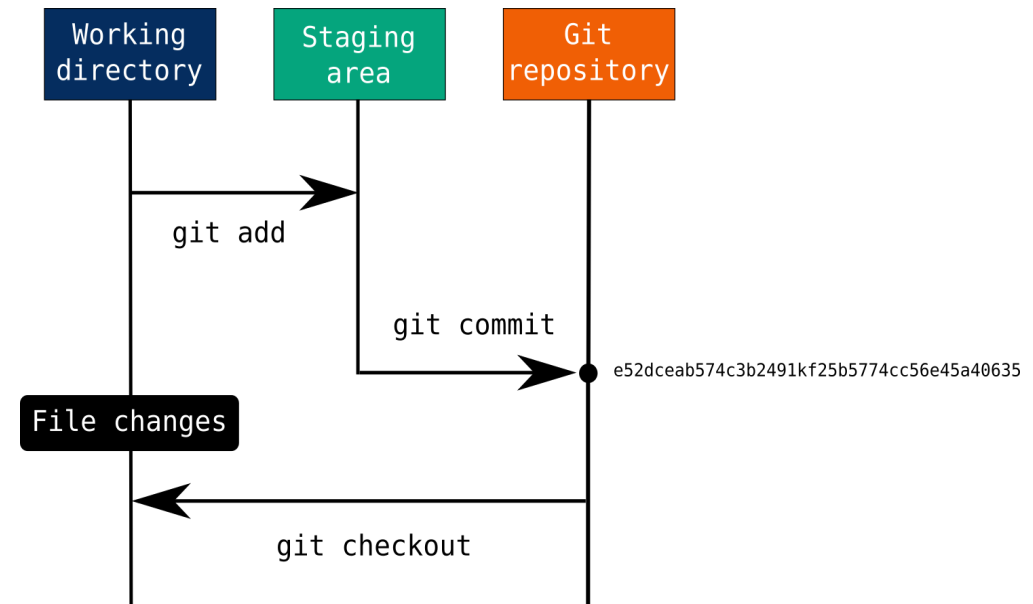
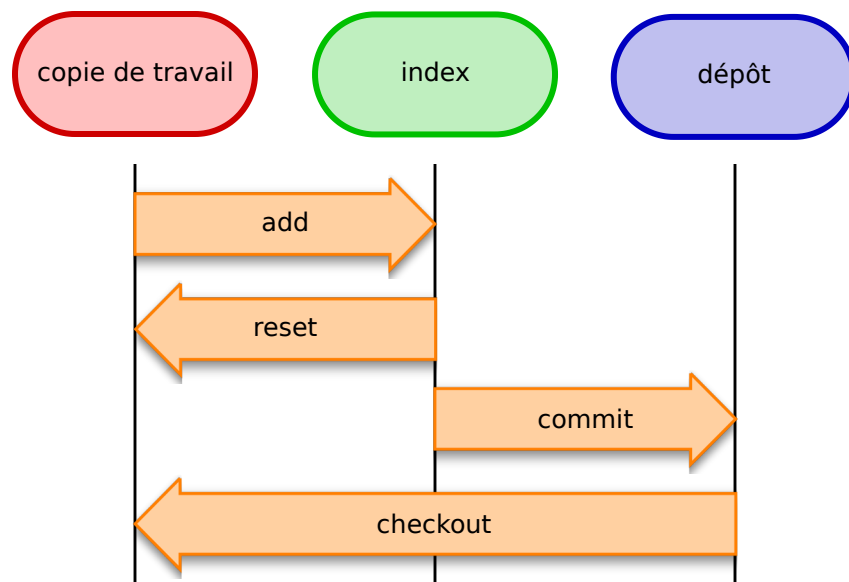
```
(base) ljudmila@ljudmila-Latitude-5580:~/Bureau/mineure_HN/COURS/L2HN001/2_Git$ git branch dev
(base) ljudmila@ljudmila-Latitude-5580:~/Bureau/mineure_HN/COURS/L2HN001/2_Git$ git branch
dev
* main
```

- pour basculer sur une autre branche : `git checkout [nom de la branche]`

Si vous relancez `git branch`, vous verrez que la branche courante a changé.

Trois états de fichiers

1. **modifié** : le fichier a été modifié, mais il n'a pas encore été ajouté à un commit
2. **indexé** : le fichier a été ajouté à un commit, mais le commit n'a pas été finalisé avec un message → *staging area, stage* ou *index*
3. **commité** : les modifications du fichier ont été enregistrées



Dépôt distant

Un dépôt distant (angl. *remote repository*) est un dépôt Git, similaire à un dépôt local, mais accessible à distance via une URL.

- exemple : <https://github.com/ljpetkovic/L2HN001>

Un dépôt local peut être lié à un dépôt distant ; Git offre des fonctionnalités pour copier des commits de l'un à l'autre.

Avantages de l'utilisation d'un dépôt distant

- sauvegarder le projet (fichiers + historique)
- conservation de l'historique (nominatif) des fichiers (qui a fait quoi ?)
- synchroniser le projet avec un serveur distant
- travailler sur plusieurs machines
- rendre le projet accessible à d'autres personnes
- travailler sur un projet publié par quelqu'un d'autre
- collaborer à plusieurs sur un projet
- possibilité de retour en arrière
- fusion des modifications lors du travail collaboratif
- visualiser les changements au cours du temps

Push

Publier les commits sur un serveur donné : `git push`

Prérequis pour la mise en œuvre

1. créer un compte sur GitHub
2. installer GitHub Desktop (interface graphique)
 - [tutoriel](#)

Références

- **Champin, P.-A. & Cordier, A.** Introduction à GIT. <https://perso.liris.cnrs.fr/pierre-antoine.champin/enseignement/intro-git/>
- **Clérice, T.** (2021). Cours pour apprendre Git et ses bonnes pratiques [*dépôt GitHub*]. <https://github.com/Pontelneptique/cours-git>