

INTRODUCTION

A GIT



git

Jour 1



Une histoire vrai



Mon-Projet



Fichier_1.py

1. Je crée le projet
2. J'écris un premier script



Jour 2



On continue le Projet



Mon-Projet



Fichier_1_V2.py



Fichier_2.sh

1. On modifie le premier script
2. On écrit un deuxième script



Jour 3



On continu le Projet



Mon-Projet



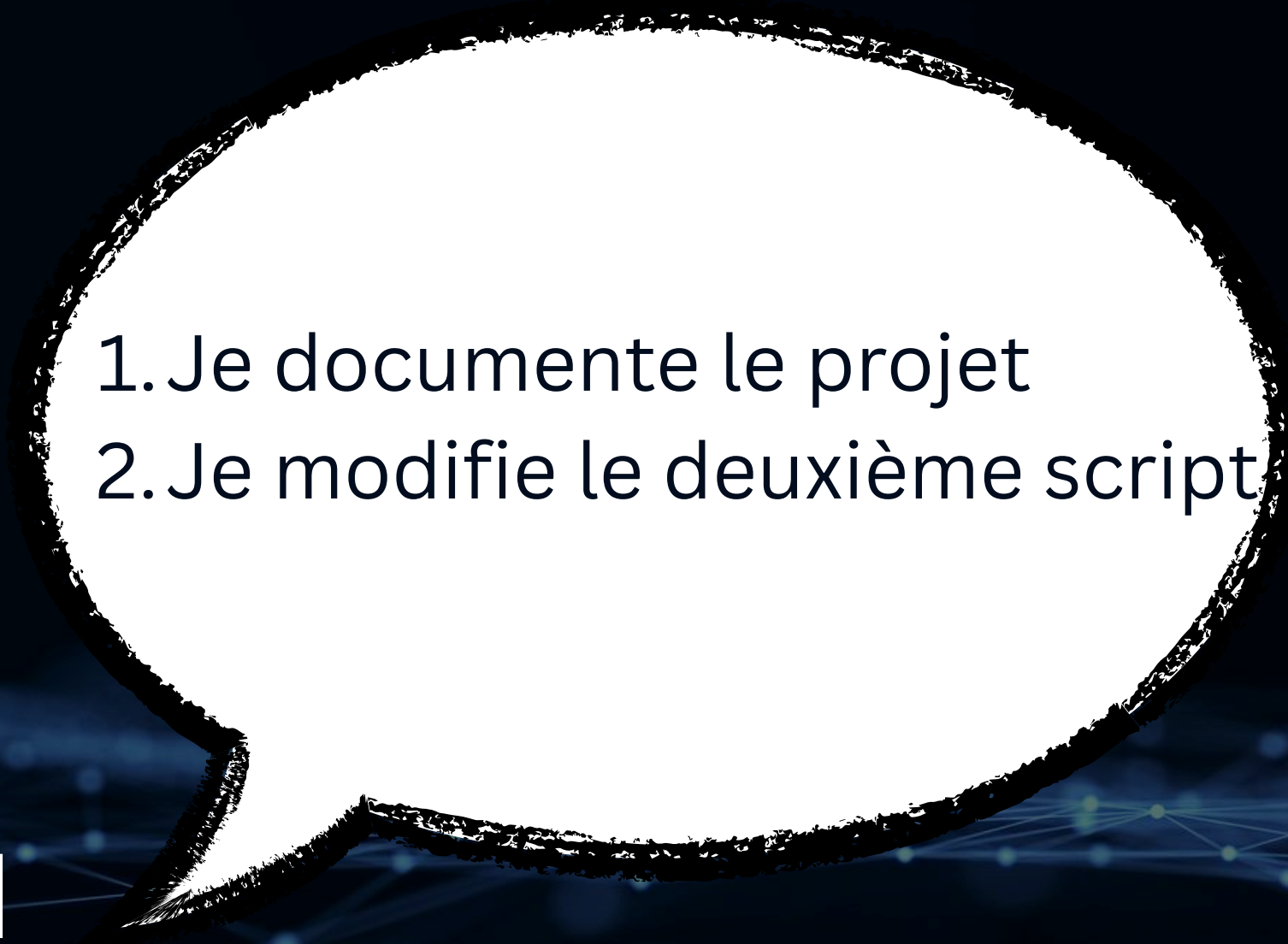
Fichier_1_V2.py



Fichier_2_V4.sh



README.md

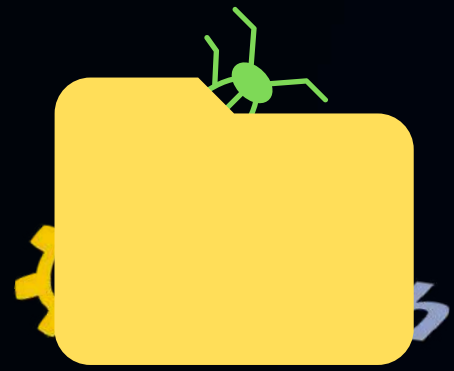
- 
1. Je documente le projet
 2. Je modifie le deuxième script



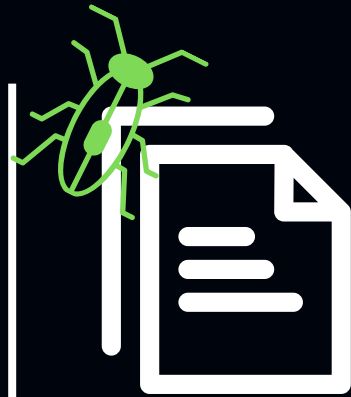
Jour 4



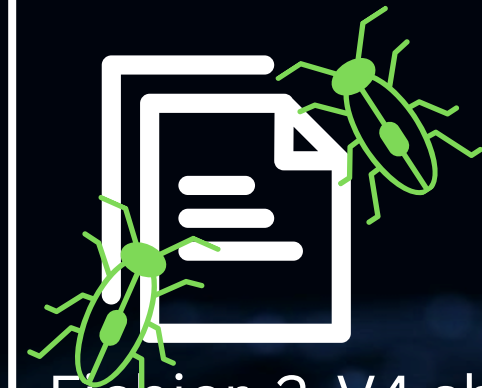
On continue le Projet



Mon-Projet



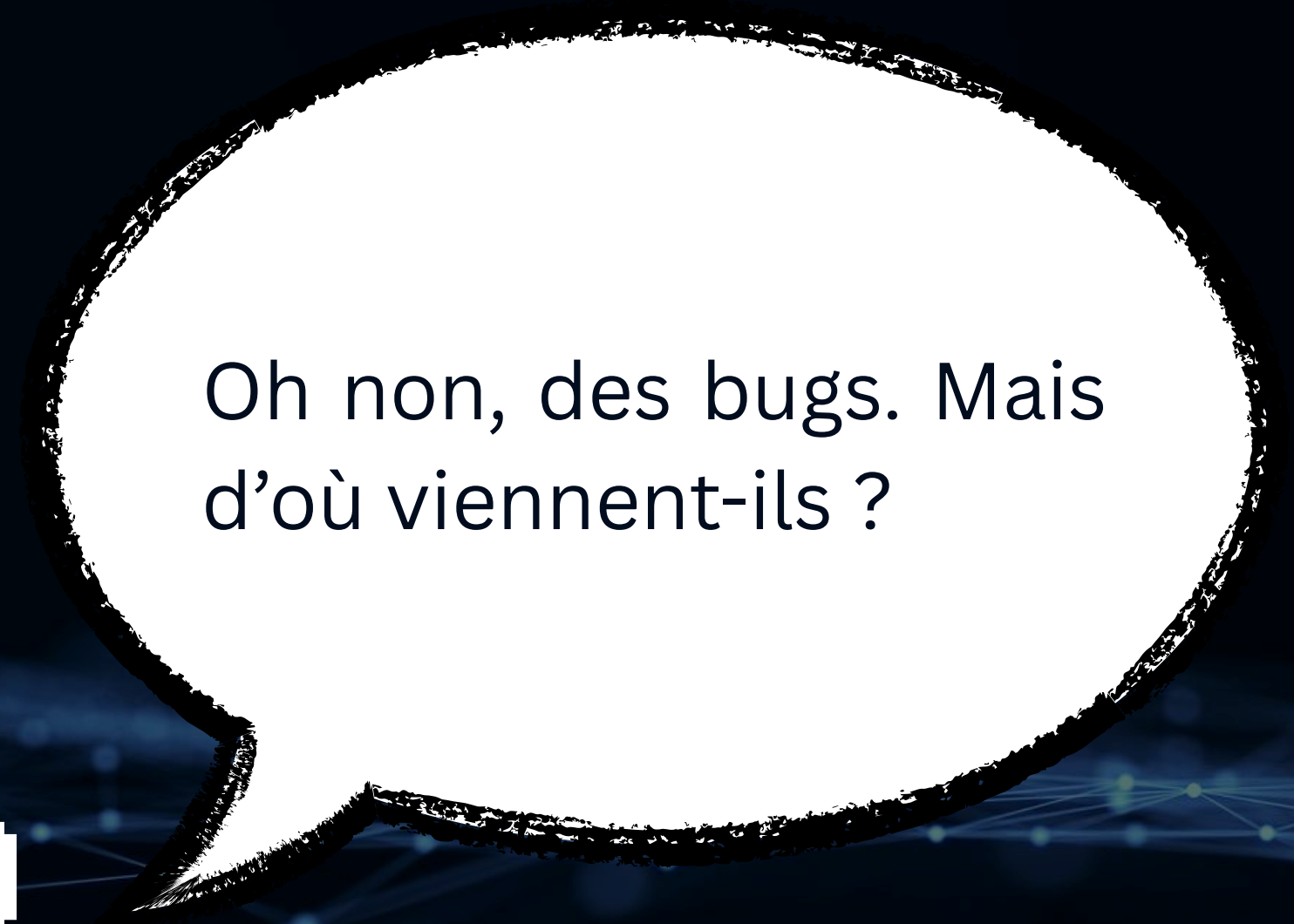
Fichier_1_V2.py



Fichier_2_V4.sh



README.md



Je ne sais pas d'où vient le bug. Je vais devoir tout vérifier. La prochaine fois, j'utiliserai GIT.



GIT permet de versionner son projet.

Je ne sais pas d'où vient le bug. Je vais devoir tout vérifier. La prochaine fois, j'utiliserai GIT.



GIT permet de versionner son projet.

Dans git: 1 version = 1 commit

Avec GIT, je pourrais facilement revenir en arrière et identifier quand le bug a été introduit.



GIT permet de versionner son projet.

Dans git: 1 version = 1 commit



Git en local



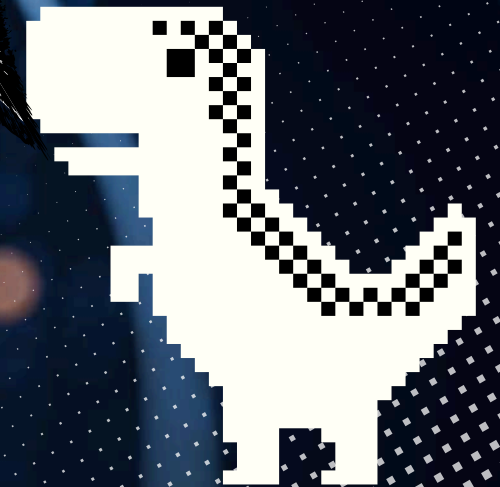
Configuration de Git

```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~$ sudo apt install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.3).
The following packages were automatically installed and are no longer required:
  libdrm-radeon1:i386 libglapi-mesa:i386
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 150 not upgraded.
danura@DESKTOP-VIKI:~$ mkdir mes_premier_pas_git
danura@DESKTOP-VIKI:~$ cd mes_premier_pas_git/
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ ls -la
total 8
drwxrwxr-x  2 danura danura 4096 Sep 29 03:52 .
drwxr-x--- 31 danura danura 4096 Sep 29 03:52 ..
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git init
Initialized empty Git repository in /home/danura/mes_premier_pas_git/.git/
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ ls -la
total 12
drwxrwxr-x  3 danura danura 4096 Sep 29 03:53 .
drwxr-x--- 31 danura danura 4096 Sep 29 03:52 ..
drwxrwxr-x  7 danura danura 4096 Sep 29 03:53 .git
```

Pour commencer :

- j'installe GIT
- je crée un dossier
- j'initialise mon projet git
- ça crée le dossier .git qui stocke tous mes commits.

Git init



Configuration de Git

```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git config user.name
"dino"
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git config user.email
"dino@minet.net"
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ █
```

Astuce : utilisez le flag `--global` pour que cela s'applique partout sur votre ordinateur.

Avant de continuer, il faut que je configure mon email et mon nom afin qu'on puisse identifier mes commits.



Oh look an easter egg

ON REPREND AVEC GIT



Mon-Projet



Ficher_1.py

Allez, c'est reparti
avec de bonnes bases.
Je crée et modifie mes
fichiers.



Les Commits

Je reviens ensuite sur mon terminal et je prépare mon fichier à être commit.



git status

git add <fichiers>

```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        fichier_1.py

nothing added to commit but untracked files present (use "git add" to track)
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git add fichier_1.py
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   fichier_1.py

danura@DESKTOP-VIKI:~/mes_premier_pas_git$
```


Les Commits

Je peut ensuite le
commit avec un
message



```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git commit -m "feat:
added my first python script"
[main (root-commit) 4082d61] feat: added my first python script
1 file changed, 3 insertions(+)
create mode 100644 fichier_1.py
danura@DESKTOP-VIKI:~/mes_premier_pas_git$
```

Un message est ***obligatoire*** pour un commit.
Si vous ne le précisez pas avec -m, alors Git
ouvrira un éditeur pour que vous le fassiez.

git commit

Les Commits

Je peut ensuite le commit avec un message



```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git commit -m "feat:
added my first python script"
[main (root-commit) 4082d61] feat: added my first python script
1 file changed, 3 insertions(+)
create mode 100644 fichier_1.py
danura@DESKTOP-VIKI:~/mes_premier_pas_git$
```

Un message est ***obligatoire*** pour un commit. Si vous ne le précisez pas avec -m, alors Git ouvrira un éditeur pour que vous le fassiez.

Il est ***fortement conseillé*** d'utiliser un préfixe comme ***fix, chore, feat ou docs*** pour rendre le repo plus clair à la lecture.

git commit

Git status

Je peux consulter mes modifs

git status



Mon-Projet



Fichier_1.py



Fichier_2.py



Fichier_3.py

```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   fichier_1.py
    deleted:    fichier_2.py
    new file:   fichier_3.py

danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git log
commit b9c73880a7288fb7e5c9103dc218e3d2ec1a4ed8 (HEAD -> main)
Author: dino <dino@minet.net>
Date:   Mon Sep 29 04:53:03 2025 +0200

    feat: added a second script

commit 4082d6144f6201795166c3bd6e28f6bdd4065013
Author: dino <dino@minet.net>
Date:   Mon Sep 29 04:49:14 2025 +0200

    feat: added my first python script
danura@DESKTOP-VIKI:~/mes_premier_pas_git$
```


Git status

Ou voir l'historique

git log



Mon-Projet



Fichier_1.py



Fichier_2.py



Fichier_3.py

```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   fichier_1.py
        deleted:    fichier_2.py
        new file:   fichier_3.py

danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git log
commit b9c73880a7288fb7e5c9103dc218e3d2ec1a4ed8 (HEAD -> main)
Author: dino <dino@minet.net>
Date:   Mon Sep 29 04:53:03 2025 +0200

    feat: added a second script

commit 4082d6144f6201795166c3bd6e28f6bdd4065013
Author: dino <dino@minet.net>
Date:   Mon Sep 29 04:49:14 2025 +0200

    feat: added my first python script
danura@DESKTOP-VIKI:~/mes_premier_pas_git$
```


Les branches

On peut commit de manière indépendante sur différentes branches.



Git permet de gérer plusieurs branches

Bonne pratique: 1 branche = 1 feature


```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git branch
* main
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git branch my-new-feature
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git branch
* main
  my-new-feature
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git checkout my-new-feature
Switched to branch 'my-new-feature'
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ nano fichier_1.py
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git add fichier_1.py
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git commit
[my-new-feature 99b29aa] feat: caused a merge conflict
1 file changed, 1 insertion(+), 1 deletion(-)
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git branch
  main
* my-new-feature
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ cat fichier_1.py
#!/usr/bin/python3

print("Hello, Earth!")
print("Hello, Dino!")
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git checkout main
Switched to branch 'main'
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ cat fichier_1.py

#!/usr/bin/python3

print("Hello, World!")
print("Hello, Dino!")
```

Les branches

git branch

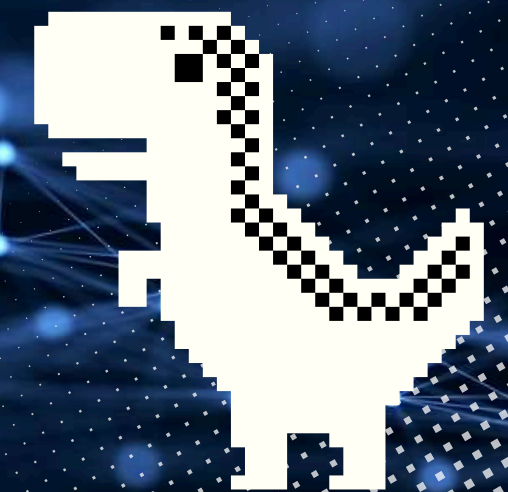
Lister les branches

git branch <nom>

Créer une branche

git checkout <nom>

Se déplacer sur une branche



Les merges

Une fois nos changement fait, on réconcilie les branches entre elle en faisant un merge



GIT permet recombinaison des branches entre elle



Les merges

Si des changements sont faits au même endroit, alors il y a un merge conflict. Il faut le résoudre en allant éditer les fichiers.



git merge <source>

Merge <source> dans la
branche où l'on se trouve

```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git merge my-new-feature
Auto-merging fichier_1.py
CONFLICT (content): Merge conflict in fichier_1.py
Automatic merge failed; fix conflicts and then commit the result
.
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ cat fichier_1.py
#!/usr/bin/python3

<<<<<<< HEAD
print("Hello, Mars!")
=====
print("Hello, Earth!")
>>>>>>> my-new-feature
print("Hello, Dino!")
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ nano fichier_1.py
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git add fichier_1.py
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git commit
[main 191d450] Merge branch 'my-new-feature'
danura@DESKTOP-VIKI:~/mes_premier_pas_git$
```


Gwark!!

**Tu veut travailler sur
le meme git que
moi? Je suppose
que je peut te
partager le dossier**



Clack clack

**Tiens mon disque
fait un bruit bizzare**

Click



... silence ...



Oh non!





Mon GIT!



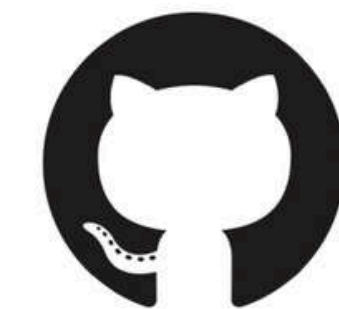


Git en réseau: Les forges





**Heureusement,
tout mon Git est sur
une forge.**



Github



Gitlab

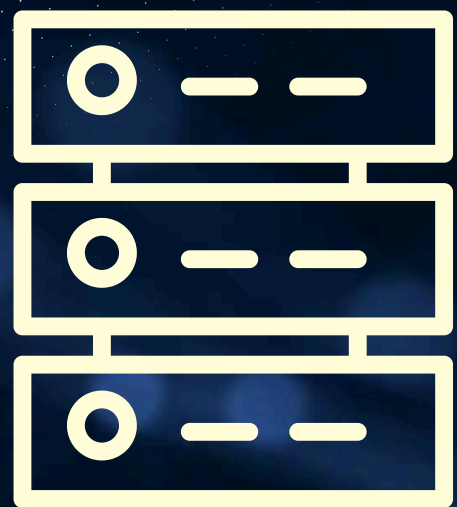
**Un forge Git est une plateforme en ligne qui héberge
des dépôts Git et fournit des outils de collaboration,
comme le suivi des issues, les demandes de fusion
(pull requests) et la gestion de projets.**

Les deux plus connu sont Gitlab et Github

**Je commence par
créer mon compte
sur gitlab.minet.net**

**Puis je crée mon
dépôt Git**





—



Pour l'instant le serveur n'a rien

Commit 1

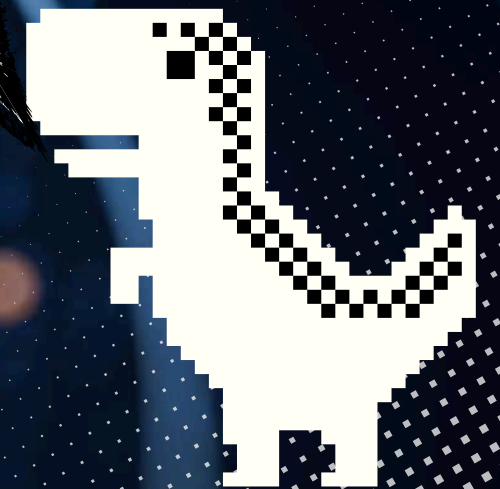
Commit 2

Commit 3

Commit Z


```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git remote
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git remote add origin
https://gitlab.minet.net/dino/mes-premier-pas-git.git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git remote
origin
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git push --set-upstream
origin --all
Username for 'https://gitlab.minet.net': dino
Password for 'https://dino@gitlab.minet.net':
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 16 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (17/17), 2.06 KiB | 2.06 MiB/s, done.
Total 17 (delta 5), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for my-new-feature, visit:
remote:   https://gitlab.minet.net/dino/mes-premier-pas-git/-/me
rge_requests/new?merge_request%5Bsource_branch%5D=my-new-feature
remote:
To https://gitlab.minet.net/dino/mes-premier-pas-git.git
* [new branch]      main -> main
* [new branch]      my-new-feature -> my-new-feature
branch 'main' set up to track 'origin/main'.
branch 'my-new-feature' set up to track 'origin/my-new-feature'.
```

Dans mon Git local,
je peux définir le
serveur distant et
ensuite envoyer
tout mon code sur le
serveur.

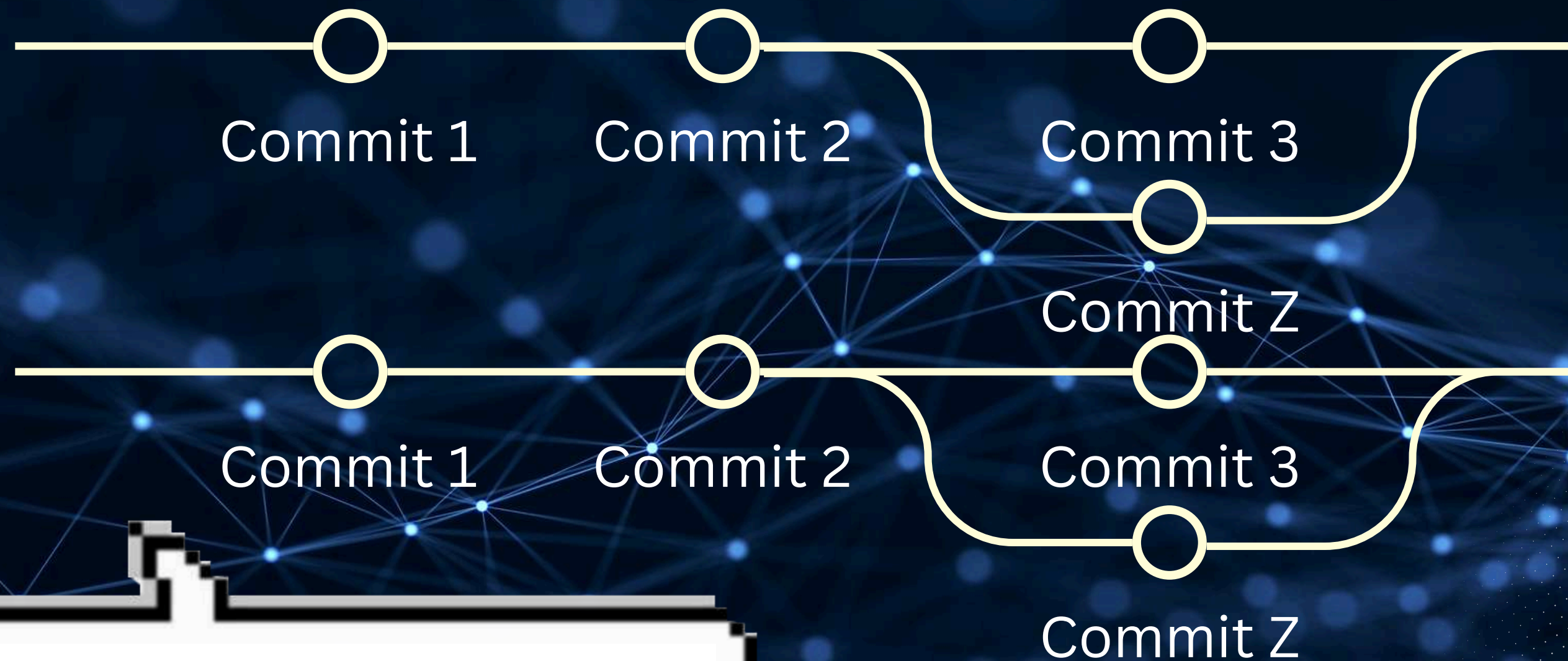
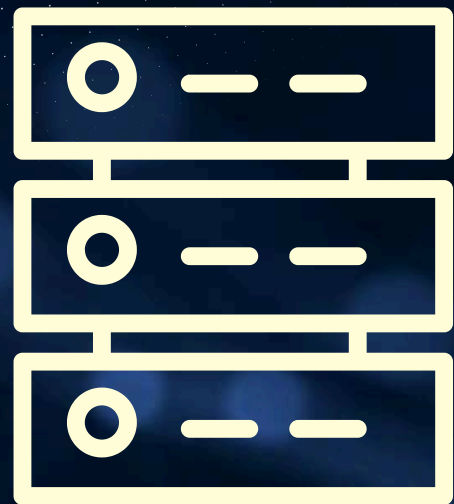


git remote add origin <url>


```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git branch -a
* main
my-new-feature
remotes/origin/main
remotes/origin/my-new-feature
danura@DESKTOP-VIKI:~/mes_premier_pas_git$
```

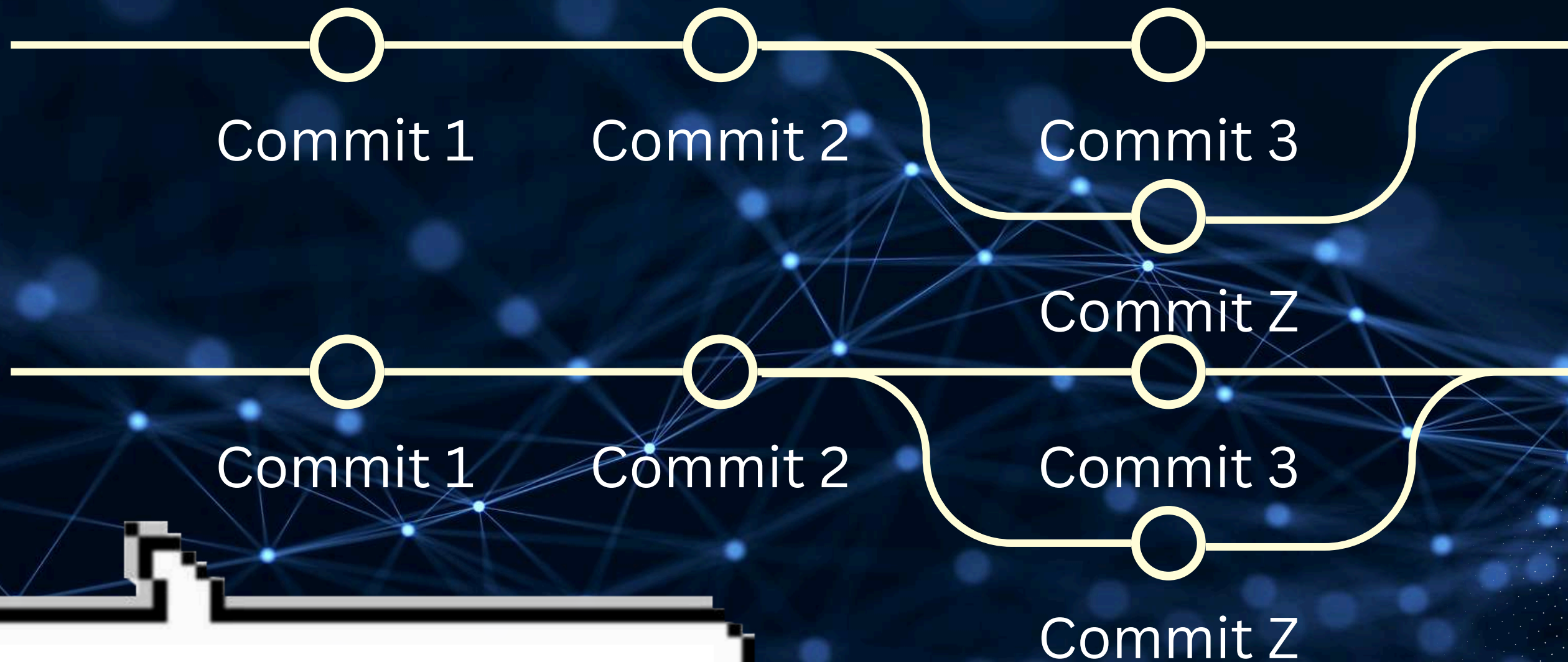
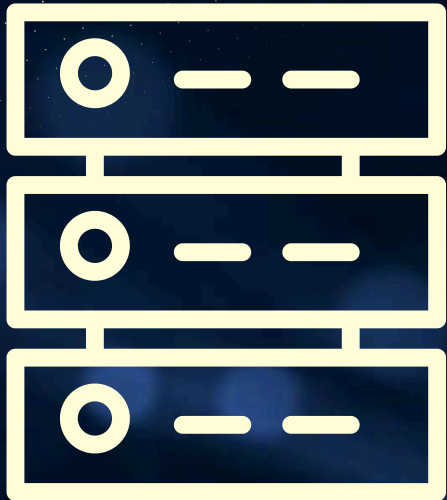
Je peut maintenant
voir les branches du
serveur





Le serveur a la meme chose

gwark gwark :(



mais pas Dino Temu



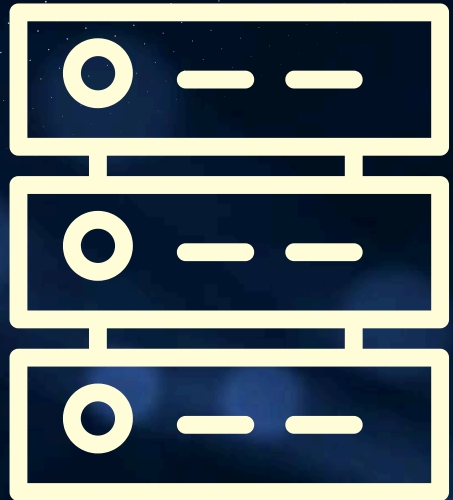
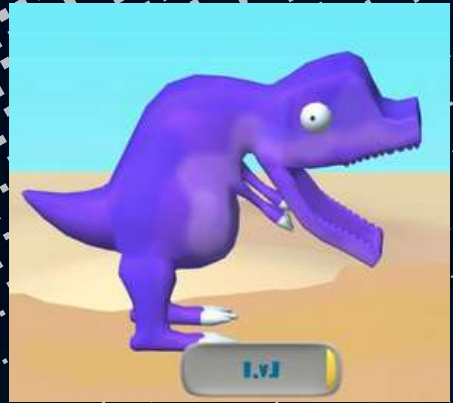
gwark



```
danura@DESKTOP-VIKI: /tmp/mes-premier-pas-git
danura@DESKTOP-VIKI:/tmp$ git clone https://gitlab.minet.net/dino/mes-premier-pas-git
Cloning into 'mes-premier-pas-git'...
remote: Enumerating objects: 17, done.
remote: Counting objects: 100% (17/17), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 17 (delta 5), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (17/17), done.
Resolving deltas: 100% (5/5), done.
danura@DESKTOP-VIKI:/tmp$ cd mes-premier-pas-git/
danura@DESKTOP-VIKI:/tmp/mes-premier-pas-git$ ls
fichier_1.py  fichier_3.py
danura@DESKTOP-VIKI:/tmp/mes-premier-pas-git$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
danura@DESKTOP-VIKI:/tmp/mes-premier-pas-git$ git log
commit 191d4500c419458ee554fafdc18d68397575d46b (HEAD -> main, origin/main, origin/HEAD)
Merge: 84696aa 99b29aa
Author: dino <dino@minet.net>
Date: Mon Sep 29 05:56:14 2025 +0200
```

git clone <url>



Dino Temu est maintenant a jour



Si je fais un
commit, je vois que
le local est en
avance sur origin.



```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git log
commit 68cdc6d6030af71fec3c13498735ac293f89e1ca (HEAD -> main)
Author: dino <dino@minet.net>
Date:   Mon Sep 29 07:46:51 2025 +0200

    feat: local is now ahead of remote until push

commit 191d4500c419458ee554fafdc18d68397575d46b (origin/main)
Merge: 84696aa 99b29aa
Author: dino <dino@minet.net>
Date:   Mon Sep 29 05:56:14 2025 +0200

    Merge branch 'my-new-feature'

commit 84696aaf0d8d755d85981f5ecd8e70e50173d8e5
Author: dino <dino@minet.net>
Date:   Mon Sep 29 05:50:29 2025 +0200
```


gwark!!!



Commit 1

Commit 2

Commit 3

Commit Z

Commit 1

Commit 2

Commit 3

Commit Z

Commit 1

Commit 2

Commit 3

Commit 4

Commit Z

Je suis en avance



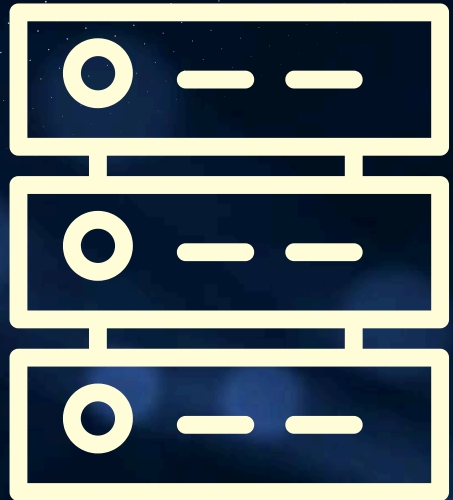
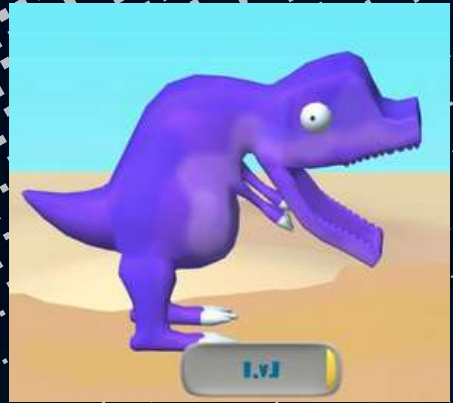
En faisant push, je mets à jour le serveur.



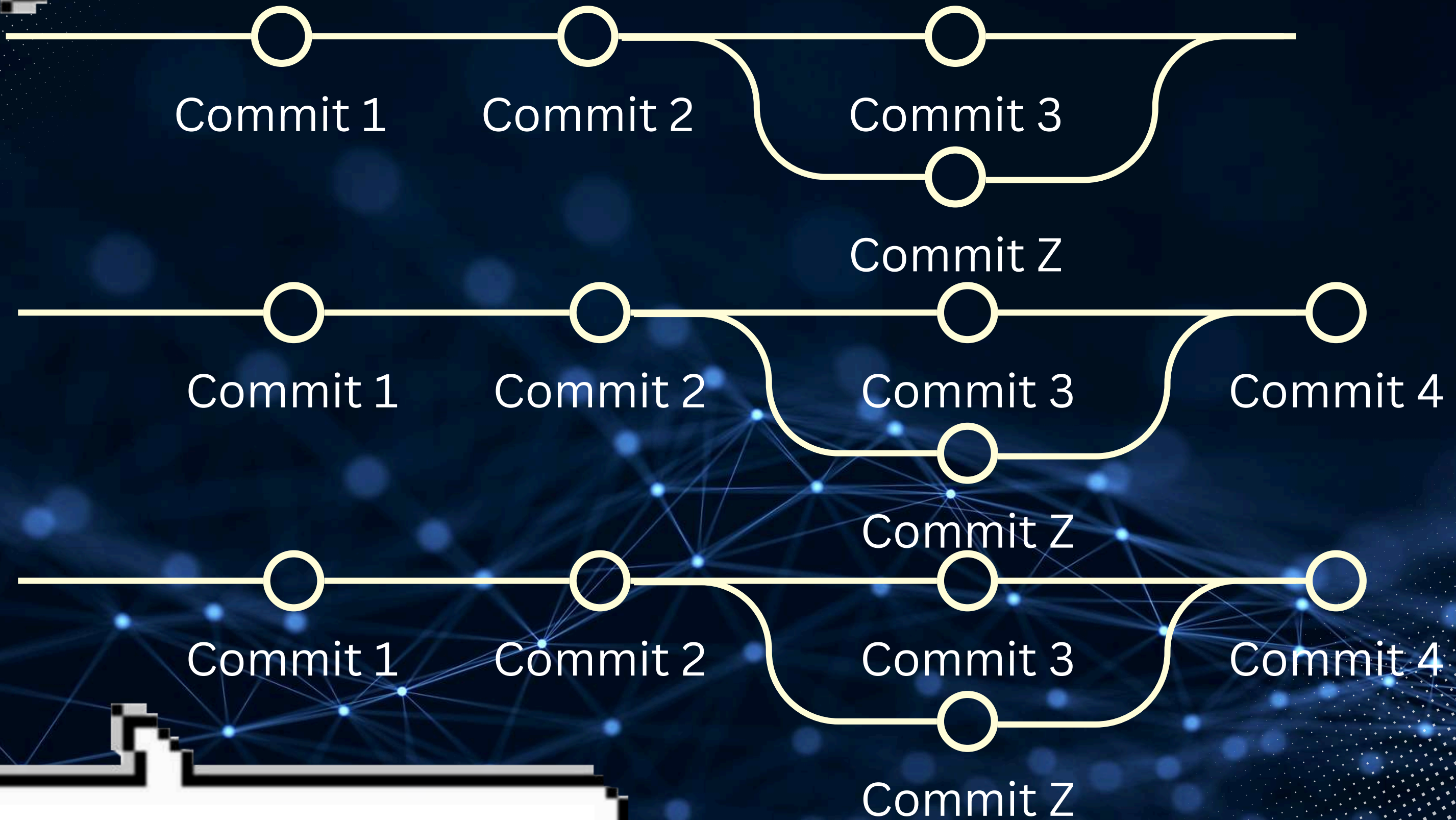
```
danura@DESKTOP-VIKI: ~/mes_premier_pas_git
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git push
Username for 'https://gitlab.minet.net': dino
Password for 'https://dino@gitlab.minet.net':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 566 bytes | 566.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://gitlab.minet.net/dino/mes-premier-pas-git.git
    191d450..68cdc6d  main -> main
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
danura@DESKTOP-VIKI:~/mes_premier_pas_git$ git log
commit 68cdc6d6030af71fec3c13498735ac293f89e1ca (HEAD -> main, origin/main)
Author: dino <dino@minet.net>
Date:   Mon Sep 29 07:46:51 2025 +0200

    feat: local is now ahead of remote until push
```

Le serveur est a jour





...

```
danura@DESKTOP-VIKI: /tmp/mes-premier-pas-git$ git log
commit 191d4500c419458ee554fafdc18d68397575d46b (HEAD -> main, origin/main, origin/HEAD)
Merge: 84696aa 99b29aa
Author: dino <dino@minet.net>
Date: Mon Sep 29 05:56:14 2025 +0200

    Merge branch 'my-new-feature'

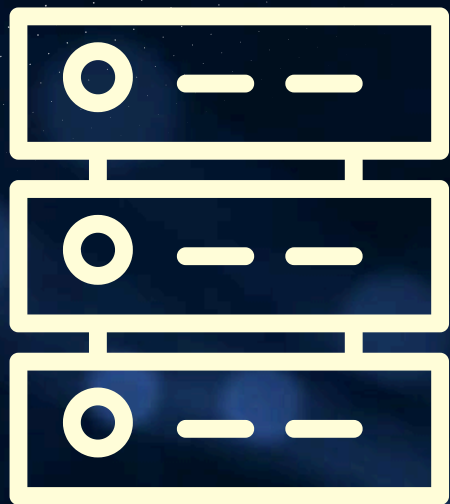
commit 84696aaf0d8d755d85981f5ecd8e70e50173d8e5
Author: dino <dino@minet.net>
Date: Mon Sep 29 05:50:29 2025 +0200

    feat: caused an merge conflict on main

commit 99b29aa626839105b9d600c610cb36af8ab862e1 (origin/my-new-feature)
Author: dino <dino@minet.net>
Date: Mon Sep 29 05:27:25 2025 +0200

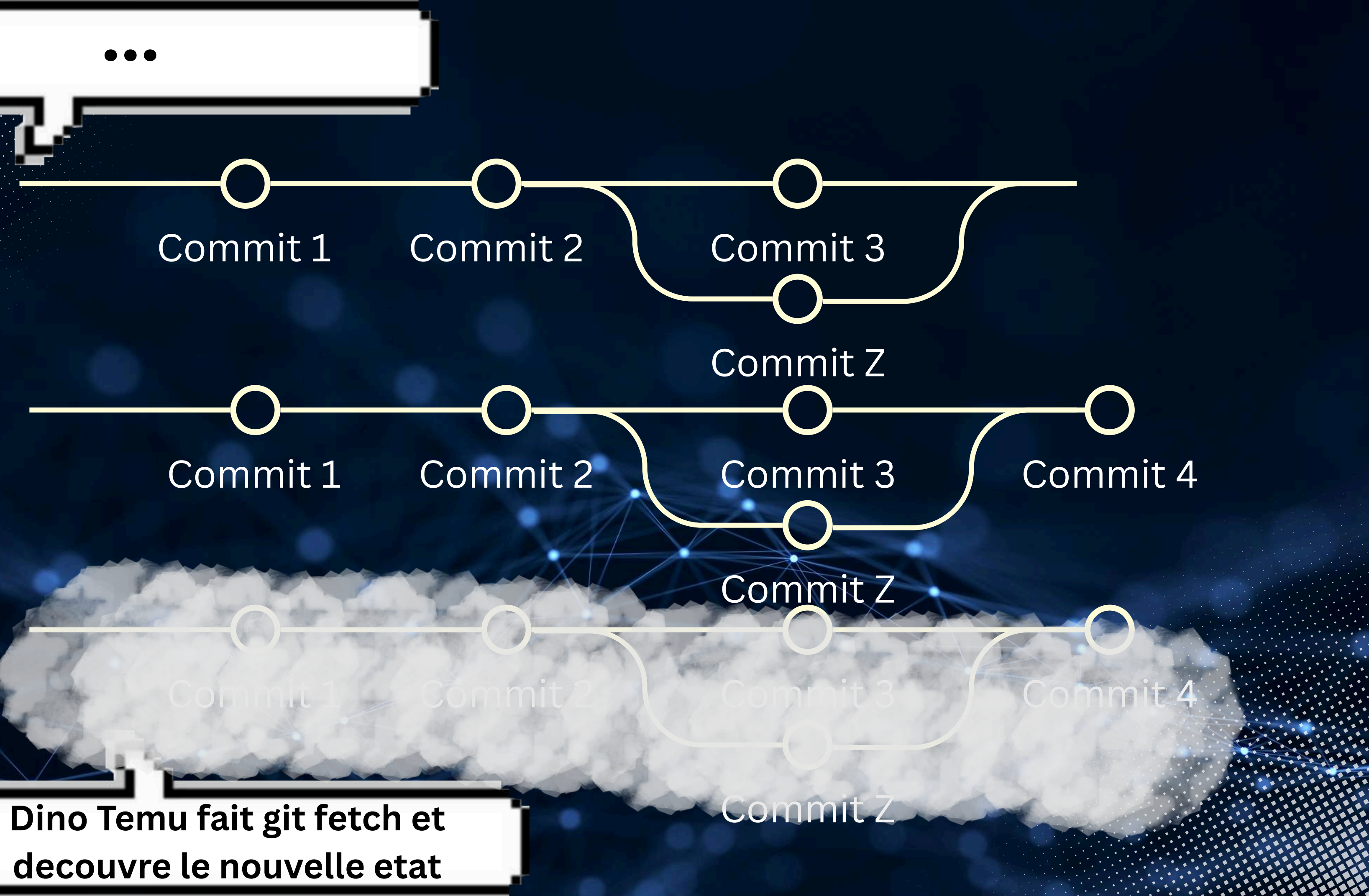
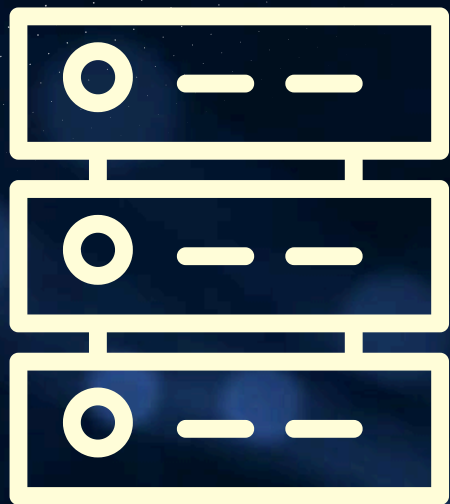
    feat: caused a merge conflict

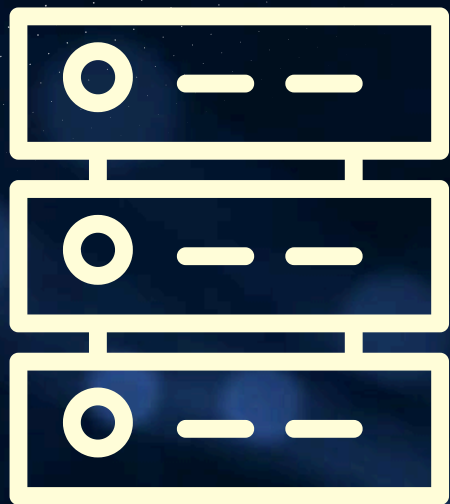
commit 1df0cdc5706292c9f74970ef188b58db0e3212d9
Author: dino <dino@minet.net>
```

**Dino Temu ne sait pas que le
serveur a été mis à jour.**







**Dino Temu fait git merge
origin/main et se met a jour**

GwArk



```
danura@DESKTOP-VIKI: /tmp/mes-premier-pas-git$ git fetch
danura@DESKTOP-VIKI: /tmp/mes-premier-pas-git$ git status
On branch main
Your branch is behind 'origin/main' by 1 commit, and can be fast
-forwarded.
  (use "git pull" to update your local branch)

nothing to commit, working tree clean
danura@DESKTOP-VIKI: /tmp/mes-premier-pas-git$ git merge origin/
origin/HEAD
origin/main
danura@DESKTOP-VIKI: /tmp/mes-premier-pas-git$ git merge origin/m
ain
Updating 191d450..68cdc6d
Fast-forward
 fichier_1.py | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
danura@DESKTOP-VIKI: /tmp/mes-premier-pas-git$
```


**Donc git fetch et git
merge permet de ce
mettre a jour**

**Mais on peut faire
git pull pour faire
les deux commande
en un**



**Un dernier concept
pour la route:
Le PR/MR**



Un dernier concept
pour la route:
Le PR/MR

Dans le GUI de
Github ou de Gitlab,
on peut créer un
pull request.

Cela permet de résoudre les
conflits et permet aux autres
de vérifier votre travail avant
un merge.

New merge request

From `my-new-feature` into `main` [Change branches](#)

Title (required)

☒ Mark as draft
Drafts cannot be merged until marked ready.

Description

Normal text | **B** *I* U |

Describe the goal of the changes and what reviewers should be aware of.

Add description templates to help your contributors to communicate effectively!

Assignee

Reviewer

Milestone

Labels

Merge can start

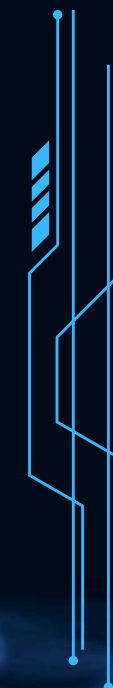
Requires that merge checks pass.

Merge options

☒ Delete source branch when merge request is accepted.

☐ Squash commits when merge request is accepted.

BLOOKET



git

TP



gitlab.minet.net/dino/format-git

Suivre les instructions du
README



git

POUR ALLEZ PLUS LOIN

Git, vscode, and co (Merci π)
man git-la command qui interesse
Convention de nommage de commit