

# Git 101:

## Tout ce qu'il faut connaitre pour etre un bon dev.

### Atelier Git(hub)

Kevin VANDERVAEREN

ULB – URLAB

November 12, 2024



Faculté  
des  
Sciences

# Agenda



- ▶ donné et partage de fichiers



- ▶ donné et partage de fichiers
- ▶ Introduction à Git
  - ▶ comment ça marche ?
  - ▶ quel sont les avantages ?
  - ▶ comment l'installer ?
  - ▶ comment l'utiliser ?
  - ▶ exercices pratiques



- ▶ donné et partage de fichiers
- ▶ Introduction à Git
  - ▶ comment ça marche ?
  - ▶ quel sont les avantages ?
  - ▶ comment l'installer ?
  - ▶ comment l'utiliser ?
  - ▶ exercices pratiques
- ▶ Introduction à GitHub (ou autre plateforme)
  - ▶ En quoi est-ce différent de Git ?
  - ▶ Comment l'utiliser ?
  - ▶ avantages et inconvénients
  - ▶ exercices pratiques



- ▶ donné et partage de fichiers
- ▶ Introduction à Git
  - ▶ comment ça marche ?
  - ▶ quel sont les avantages ?
  - ▶ comment l'installer ?
  - ▶ comment l'utiliser ?
  - ▶ exercices pratiques
- ▶ Introduction à GitHub (ou autre plateforme)
  - ▶ En quoi est-ce différent de Git ?
  - ▶ Comment l'utiliser ?
  - ▶ avantages et inconvénients
  - ▶ exercices pratiques
- ▶ Conclusion et synthèse



- ▶ donné et partage de fichiers
- ▶ Introduction à Git
  - ▶ comment ça marche ?
  - ▶ quel sont les avantages ?
  - ▶ comment l'installer ?
  - ▶ comment l'utiliser ?
  - ▶ exercices pratiques
- ▶ Introduction à GitHub (ou autre plateforme)
  - ▶ En quoi est-ce différent de Git ?
  - ▶ Comment l'utiliser ?
  - ▶ avantages et inconvénients
  - ▶ exercices pratiques
- ▶ Conclusion et synthèse
- ▶ Et pour aller plus loin



comment partagée des fichiers (code) entre plusieurs personnes ou plusieurs machines ?



- ▶ photo de l'écran



# donné et partage de fichiers



- ▶ photo de l'écran
- ▶ une clé USB



- ▶ photo de l'écran
- ▶ une clé USB
- ▶ fichier sur discord



- ▶ photo de l'écran
- ▶ une clé USB
- ▶ fichier sur discord
- ▶ Cloud (google drive, dropbox, . . . )



- ▶ photo de l'écran
- ▶ une clé USB
- ▶ fichier sur discord
- ▶ Cloud (google drive, dropbox, ...)

## Attention

Toutes les méthodes ne se valent pas !



# donné et partage de fichiers



Méthode	Historique	Sauvegarde	Collaboration	Travail en parallèle	Commentaires	Automatisation
Prendre une photo	X	X	X	X	X	X
Clé USB	X	✓	X	X	X	X
Serveur Discord	X	✓	✓	X	X	X
Cloud	✓	✓	✓	X	✓	X



Méthode	Historique	Sauvegarde	Collaboration	Travail en parallèle	Commentaires	Automatisation
Prendre une photo	✗	✗	✗	✗	✗	✗
Clé USB	✗	✓	✗	✗	✗	✗
Serveur Discord	✗	✓	✓	✗	✗	✗
Cloud	✓	✓	✓	✗	✓	✗
Git (VCS)	✓	✓	✓	✓	✓	✓



- ▶ Comment ça marche ?
- ▶ Quels sont les avantages ?
- ▶ Comment l'installer ?
- ▶ Comment l'utiliser ?
- ▶ Exercices pratiques



Git est un système de contrôle de version (VCS) distribué, un VCS est une base de données qui utilise un système de fichiers pour enregistrer les modifications apportées à un ensemble de fichiers utilisent la théorie des graphes... BREF !



# Comment ça marche ?

comme xkcd le dit si bien...



# Comment ça marche ?

git est magic!



il peut faire tout ce que vous voulez, mais il faut savoir comment lui demander



# Quels sont les avantages ?



en quoi git est-il mieux que les autres méthodes ?

- ▶ Historique des modifications



# Quels sont les avantages ?



en quoi git est-il mieux que les autres méthodes ?

- ▶ Historique des modifications
- ▶ Sauvegarde



# Quels sont les avantages ?



en quoi git est-il mieux que les autres méthodes ?

- ▶ Historique des modifications
- ▶ Sauvegarde
- ▶ Commentaires



# Quels sont les avantages ?



en quoi git est-il mieux que les autres méthodes ?

- ▶ Historique des modifications
- ▶ Sauvegarde
- ▶ Commentaires
- ▶ Automatisation



les commandes de base:

- ▶ git init
- ▶ git add
- ▶ git commit

les commandes un peu plus avancées (surtout utile en collaboration)

- ▶ git branch
- ▶ git checkout et git switch
- ▶ git merge

### Note

Nous ne verrons que les commandes en ligne de commande, mais il existe des interfaces graphiques pour git (voir ??)



► git init

### git init

git init permet de créer un nouveau dépôt git dans le répertoire courant sous la forme d'un dossier caché .git

### note

.git n'a pas besoin d'être modifié directement, il est géré par git lui-même. mais si vous voulez automatiser des tâches, vous pouvez ajouter des scripts dans le dossier .git/hooks

commande à taper: git init

# les commandes de base



- ▶ git init
- ▶ git status

## git status

git status permet de voir l'état de votre dépôt git, c'est-à-dire les fichiers qui ont été modifiés, ajoutés, supprimés, ...

commande à taper: git status



- ▶ git init
- ▶ git status
- ▶ git add

## git add

git add permet d'ajouter des fichiers à l'index de git, c'est-à-dire de dire à git que vous voulez suivre les modifications de ce fichier et quand vous ferez un commit, **Uniquement** les modifications de ces fichiers seront prises en compte.

commande à taper: git add <fichier>

<fichier>: peut être un fichier ou un dossier



- ▶ git init
- ▶ git status
- ▶ git add
- ▶ git commit

### git commit

git commit permet de sauvegarder les modifications de vos fichiers dans le dépôt git. Il est important de mettre un message clair et concis pour expliquer les modifications apportées.

commande à taper: git commit -m "message"

Le -m est obligatoire et le message doit être entre guillemets



# les commandes de base



Bravo vous avez fait votre premier commit !



Bravo vous avez fait votre premier commit !



bon maintenant comment voir ce que vous avez fait ?

## git log

```
commit a126e921efaf7a12ee2146368de5963fab8a6e77
Author: LinSfa <kevin.vandervaeren@ulb.be>
Date:   Thu Oct 31 16:22:35 2024 +0100
```

update teh pdf

```
commit 34f08423e8cb79719f020550bbf57528323933ca
Author: LinSfa <kevin.vandervaeren@ulb.be>
Date:   Thu Oct 31 15:40:25 2024 +0100
```

change the default exemple1

```
commit 23ace16e5f9dc884e996db91f3aba022f097e164
Author: LinSfa <kevin.vandervaeren@ulb.be>
Date:   Thu Oct 31 15:32:03 2024 +0100
```

exemple1 for modification tracking

```
commit dc5378ff1fcfe40b48bd1778ccdf70d5d1699e4eec
Author: LinSfa <kevin.vandervaeren@ulb.be>
Date:   Mon Oct 28 20:59:11 2024 +0100
```

update the image

```
commit 93a9709f6d28538b85e2d309ac7153a052182219
Author: LinSfa <kevin.vandervaeren@ulb.be>
Date:   Sun Oct 27 22:18:52 2024 +0100
```

first few pdf

```
commit b5eb1cbc8cd363fbb909a851e950d5eb431963db
Author: LinSfa <kevin.vandervaeren@ulb.be>
Date:   Tue Oct 22 15:44:01 2024 +0200
```

init based on template

```
commit e67c6bc6e711942853a29236247c61f0dacffe32
Author: Linsfa <92955808+Kvan0005@users.github.com>
Date:   Tue Oct 22 14:32:42 2024 +0200
:]
```

## git graph sur vscode

Graph	Description	Date	Author	Commit
○	Uncommitted Changes (2)	4 Nov 2024 2...	*	*
○ ↗ main ↗ origin ↗ origin/HEAD	add base of commit	1 Nov 2024 1...	LinSfa	fec2edc5
update teh pdf		31 Oct 2024 ...	LinSfa	a126e921
change the default exemple1		31 Oct 2024 ...	LinSfa	34f08423
exemple1 for modification tracking		31 Oct 2024 ...	LinSfa	23ace16e
update the image		28 Oct 2024 ...	LinSfa	dc5378ff
first few pdf		27 Oct 2024 ...	LinSfa	93a9709f
init based on template		22 Oct 2024 ...	LinSfa	b5eb1cbc
Initial commit		22 Oct 2024 ...	LinSfa	e67c6bc6

Avant de continuer, avez-vous des questions ?



Avant de continuer, avez-vous des questions ? Avant les

commandes un peu plus avancées, nous allons voir comment utiliser GitHub



- ▶ En quoi est-ce différent de Git ?
- ▶ Comment l'utiliser ?
- ▶ avantages et inconvénients



simple analogie:



The screenshot shows a Reddit post in the subreddit r/learnprogramming. The post is titled "Difference between GitHub and Git" and was made by u/Amantulsyan35 12 hours ago. It has 308 upvotes and 76 comments. The post text asks: "If you were to explain a 10 year old what git and github are and the differences between them what would your answer be?". Below the post are the upvote, downvote, comment, share, and award buttons. A "BEST COMMENTS" section is visible, showing a comment by user raalag 10 hours ago with 8 awards. The comment text is "Git is porn GitHub is pornhub". At the bottom are reply and upvote/downvote buttons.



# En quoi est-ce différent de Git ?



Si git est un livre, alors GitHub est une bibliothèque où vous pouvez stocker vos livres et les partager avec d'autres personnes.  
Git sont vos données, Git est un cloud pour les stocker et les

partager



# En quoi est-ce différent de Git ?



vs



Git	GitHub
programme	service
gestion de version décentralisé	plateforme de collaboration
local	en ligne
open source	propriétaire (Microsoft)





# Gitea



# GitLab



# Bitbucket

et bien d'autres...



# comment l'utiliser ?



- ▶ créer un compte
- ▶ clé d'accès ?
- ▶ premier dépôt (aka repo)



avez vous vraiment besoin d'une explication ?

### conseil

apres avoir créé votre compte, vous pouvez utiliser le GitHub student pack pour avoir des avantages supplémentaires



► pourquoi ?

pourquoi ?

utiliser par github pour vous identifier, elle vous permet de vous authentifier quand vous fait des actions sur votre compte (clone, push, pull, . . . )



- ▶ pourquoi ?
- ▶ comment ?

## comment ?

suivant les étapes suivantes:

- ▶ aller dans les paramètres de votre compte se trouvant en haut à droite
- ▶ settings de developer (ou alors SSH and GPG key mais c'est hores sujet)
- ▶ clé d'accès personnel (personal access token)
- ▶ Tokens (classique)
- ▶ générer un nouveau token et donner toutes les permissions (par simplicité)
- ▶ **copier le token et le garder en lieu sûr**



crée un nouveau dépôt:

- ▶ aller sur la page d'accueil de votre compte
- ▶ cliquer sur le bouton "new"
- ▶ donner un nom à votre dépôt
- ▶ ajouter une description
- ▶ choisir si vous voulez un dépôt public ou privé
- ▶ cliquer sur "create repository"

simple non ?



comment interagir avec un dépôt GitHub ?

- ▶ git clone
- ▶ git push
- ▶ git pull



► git clone

### git clone

git clone permet de copier un dépôt git sur votre machine. cela vous permet de travailler sur le code sans affecter le code original.

commande à taper: `git clone <url_repo>`



- ▶ git clone
- ▶ git push

## git push

git push permet d'envoyer vos modifications sur le dépôt distant. cela permet de partager vos modifications avec les autres collaborateurs.

commande à taper: git push [ origin <branche>]



- ▶ git clone
- ▶ git push
- ▶ git pull

## git pull

git pull permet de récupérer les modifications du dépôt distant sur votre machine. cela permet de travailler sur la dernière version du code.

commande à taper: git pull [ origin <branche>]



pour résumer: si vous voulez commencer un nouveau projet sur GitHub: a faire 1 fois:

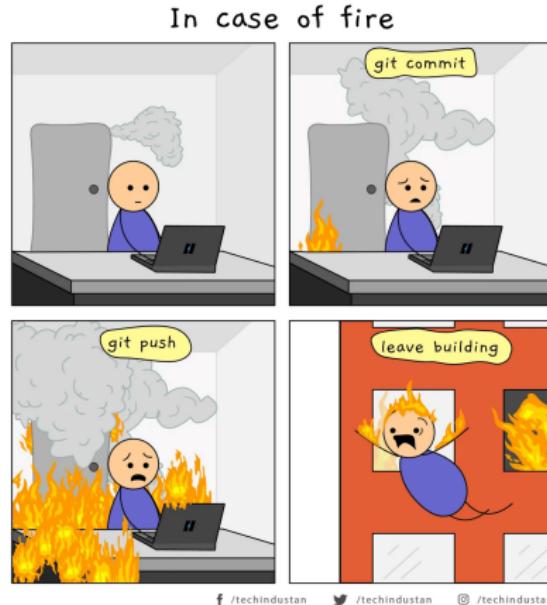
- ▶ cree un nouveau dépôt sur GitHub
- ▶ clone sur votre machine avec git clone <url\_repo>

a faire à chaque fois que vous voulez partager vos modifications:

- ▶ mettre à jour votre dépôt local avec git pull
- ▶ coder vos modifications
- ▶ ajouter vos modifications avec git add <fichier>
- ▶ faire un commit avec git commit –m "message"
- ▶ envoyer vos modifications sur GitHub avec git push



vous savez maintenant comment utiliser git et GitHub pour gérer vos projets solo sur vos machines !



# les commandes un peu plus avancées



- ▶ git branch
- ▶ git checkout et git switch
- ▶ git merge



# les commandes un peu plus avancées pour git



les commandes un peu plus avancées qui sont surtout utiles en collaboration:

- ▶ git branch

## git branch

git branch permet de créer une nouvelle branche dans votre dépôt git. une branche est une copie de votre code à un moment donné, vous pouvez travailler sur cette branche sans affecter le code de la branche principale (master).

commande à taper: `git branch <nom_branche>`

# les commandes un peu plus avancées pour git



les commandes un peu plus avancées qui sont surtout utiles en collaboration:

- ▶ git branch
- ▶ git checkout et git switch

## git checkout et git switch

git checkout et git switch permettent de changer de branche.  
git checkout est la commande historique, mais git switch est plus intuitive.

commande à taper: git checkout <nom\_branche> ou  
git switch <nom\_branche>



# les commandes un peu plus avancées pour git



les commandes un peu plus avancées qui sont surtout utiles en collaboration:

- ▶ git branch
- ▶ git checkout et git switch
- ▶ git merge

## git merge

git merge permet de fusionner une branche avec une autre. il est important de faire attention à ce que vous fusionnez, car si vous avez des modifications sur les mêmes fichiers, git ne saura pas quoi faire.

commande à taper: `git merge <nom_branche>`



comment résoudre un conflit ?

### résoudre un conflit

- ▶ ouvrir le fichier en conflit
- ▶ chercher les balises de conflit
- ▶ résoudre le conflit
- ▶ ajouter les fichiers modifiés
- ▶ faire un commit

