Walter.Rudametkin@polytech-lille.fr
https://rudametw.github.io/teaching/

Bureau F011
Polytech Lille

- Je commence à enseigner
  - ce cours est tout nouveau
  - j'accepte des critiques (constructives mais pas que) et surtout des recommandations
  - n'hésitez pas à poser des questions
- Je ne suis pas un expert

Comment gérez-vous vos fichiers ?

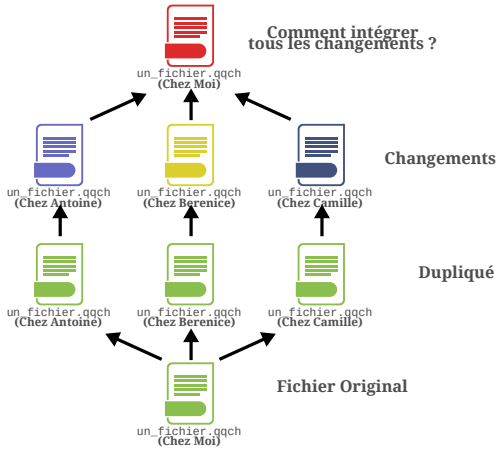
- Garder l'historique
- Partager

Comment gérez-vous vos fichiers ?

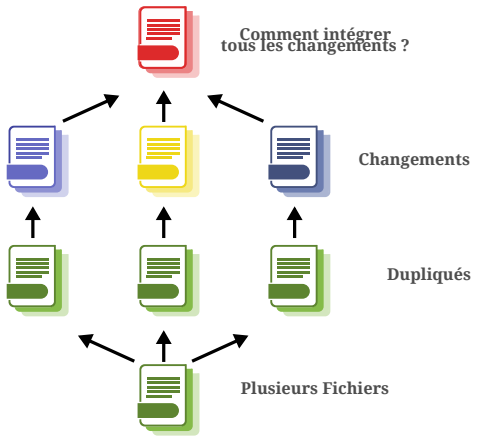
- Garder l'historique
- Partager

Versionnement manuel de fichiers

Comment collaborer sur un fichier ?



Comment collaborer sur plusieurs fichiers ?



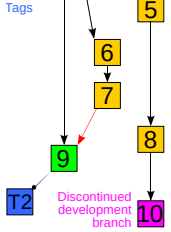
D'autres solutions ?

Dropbox

Google docs

Gestion de versions

La **gestion de versions** (en anglais *version control* ou *revision control*) consiste à maintenir **l'ensemble des versions d'un ou plusieurs fichiers** (généralement en



Par Revision_controlled_project_visualization.svg: *Subversion_project_visualization.svg: Traced by User:Stannered, original by en:User:Sami Keroladerivative work: Moxfyre (talk)derivative work: Echion2 (talk) Revision_controlled_project_visualization.svg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=9562807

- Suivi de responsabilités / propriétaires / coupables
- *Sandboxing* (espace confiné, environnement de test, isolation)
- *Branching and merging*
- Passage à l'échelle (10, 100, 1.000, 10.000 développeurs)

Que mettre dans un Logiciel de Gestion de Versions ?

- Tous les sources du projet
  - code source (`.c .cpp .java .py ...`)
  - scripts de build (`Makefile pom.xml ...`)
  - Documentation (`.txt .tex Readme ...`)
  - Ressources (images ...)
  - Scripts divers (déploiement, `.sql, .sh ...`)

Que mettre dans un Logiciel de Gestion de Versions ?

- Tous les sources du projet
  - code source (`.c .cpp .java .py ...`)
  - scripts de build (`Makefile pom.xml ...`)
  - Documentation (`.txt .tex Readme ...`)
  - Ressources (images ...)
  - Scripts divers (déploiement, `.sql, .sh ...`)

À NE PAS METTRE

- Les fichiers générés
  - Résultat de compilation (`.class .o .exe .jar ...`)
  - Autres fichiers générés (`.ps .dvi .pdf javadoc ...`)

Why the `git` ?

C'est *Ze Standard*

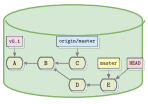
- *git - the stupid content tracker*
- Outil professionnel
- Rapide, multi-plateforme, flexible, puissant

To Share or Not to Share ?

- Enrichissez vos CV
  - `https://github.com/`
- Choisir sa licence
  - Code — GPL, Apache, BSD, MIT, Propriétaire
    `https://choosealicense.com/`
  - Documents/Rapports — Creative commons
    `https://creativecommons.org/`

Concepts et commandes `git`

Copie de travail
(Working Directory)

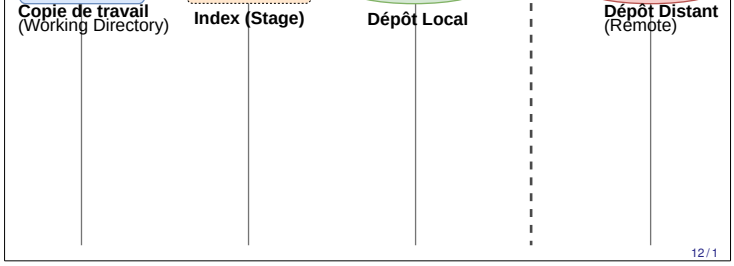


Dépôt

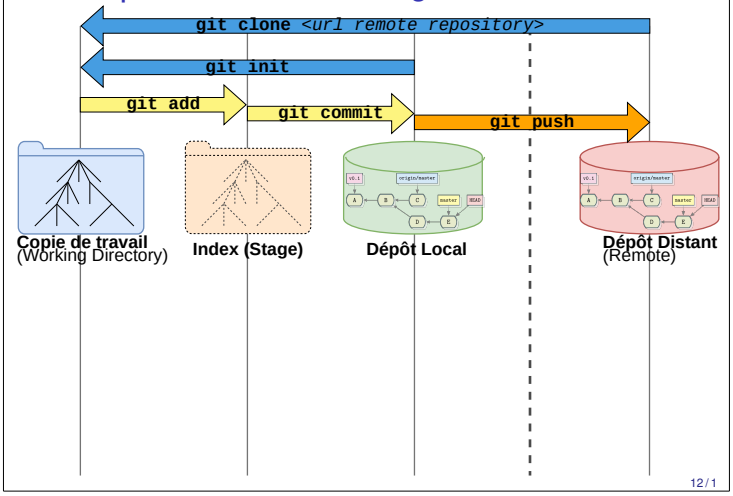
Concepts et commandes `git`

Concepts et commandes `git`

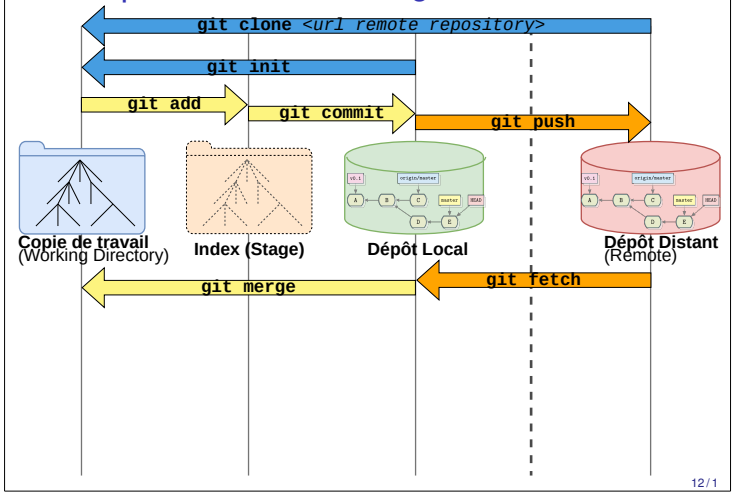
Réseau



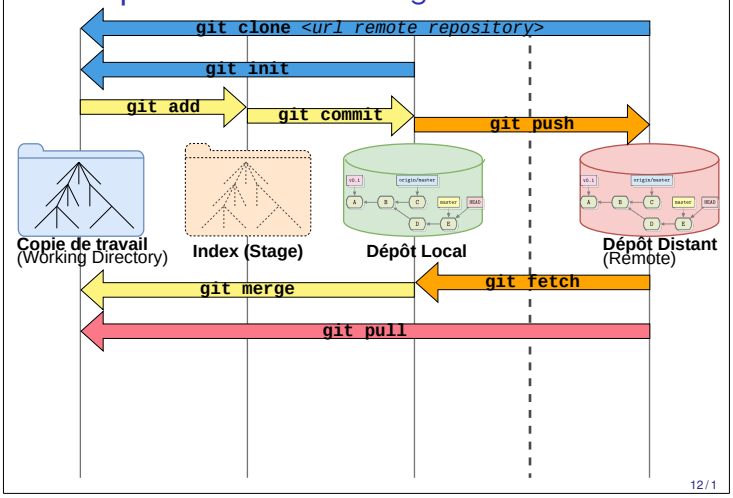
Concepts et commandes `git`



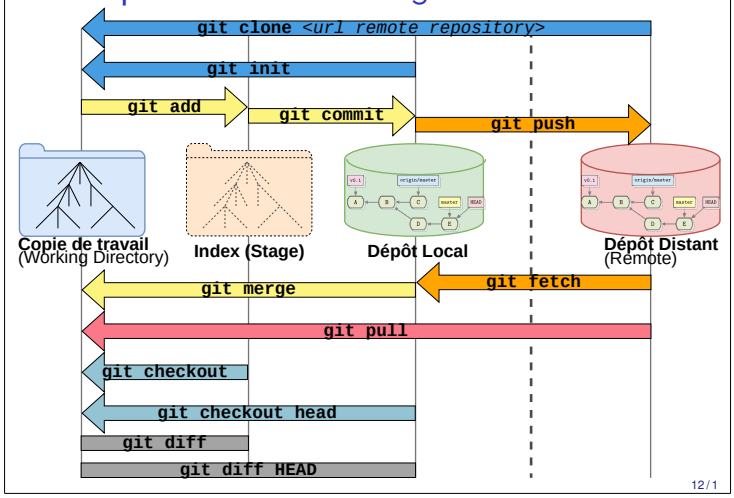
Concepts et commandes `git`



Concepts et commandes `git`



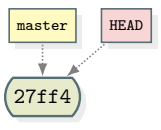
Concepts et commandes `git`



Le *Graphe Orienté Acyclique* de `commits`

(a) Dépôt vide

Le *Graphe Orienté Acyclique* de `commits`



(a) Premier *commit*

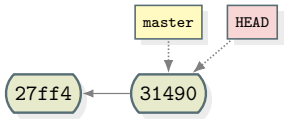
- Le `Commit-ID` est une *empreinte* calculé en utilisant la fonction de hachage `SHA-1` sur
  - **Tout** le contenu du commit + Date + Nom et email du commiteur + Message de log + ID du commit parent + ...

Propriété : **Unicité** quasi-universelle de l'`ID`

Dans un terminal ...

```
echo banane >> fruits.txt
git add fruits.txt
git commit -m "Ajouté banane à fruits.txt"
  ⇒ ID = 31490
```


Le Graphe : Commit 2

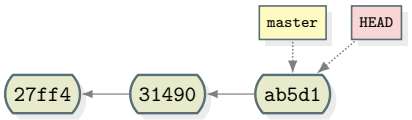


(a) Deuxième *commit*

Dans un terminal ...

```
echo banane >> fruits.txt
git add fruits.txt
git commit -m "Ajouté banane à fruits.txt"
  ⇒ ID = 31490
```


Le Graphe : Commit 3

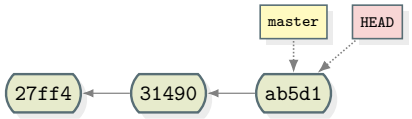


(a) Troisième *commit*

Dans un terminal ...

```
1  echo orange >> fruits.txt
2  git add fruits.txt
3  git commit -m "Ajouté orange à fruits.txt"
4    ⇒ ID = ab5d1
```

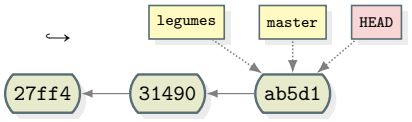

Le Graphe : Branche `legumes`



(a) Avant branche

```
git branch legumes ; git checkout legumes
```


Le Graphe : Branche `legumes`

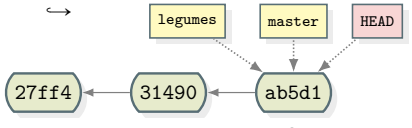


(a) Après branche

⇒ une nouvelle *étiquette* (`legumes`) apparait, elle pointe vers le même `commit` que `HEAD`

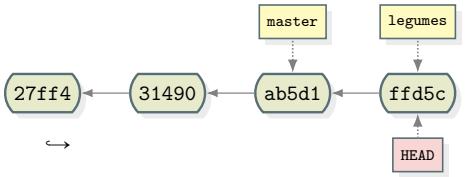
```
git branch legumes ; git checkout legumes
```


Le Graphe : Branche `legumes`



(a) Après branche

Le Graphe : Branche `legumes`



(a) Après un premier commit dans la branche `legumes`

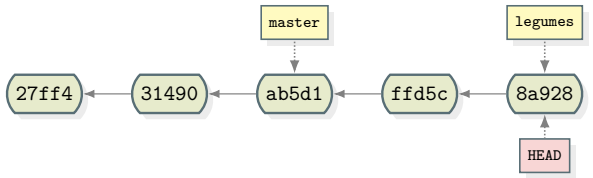
```
git branch legumes ; git checkout legumes
echo aubergine >> legumes.txt ; git add legumes.txt
git commit -m "Ajout aubergine à legumes"
  ⇒ ID = ffd5c
echo courgette >> legumes.txt ; git add legumes.txt
git commit -m "Ajout courgette à legumes"
  ⇒ ID = 8a928
```



```
git branch legumes ; git checkout legumes
echo aubergine >> legumes.txt ; git add legumes.txt
git commit -m "Ajout aubergine à legumes"
  ⇒ ID = ffd5c
echo courgette >> legumes.txt ; git add legumes.txt
git commit -m "Ajout courgette à legumes"
  ⇒ ID = 8a928
```

17/1

Le Graphe : Branche `master`

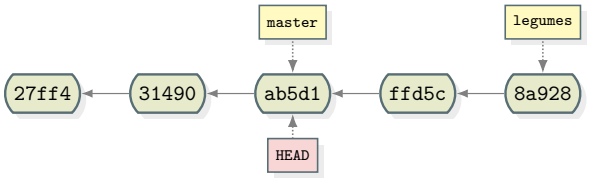


(a) Travaillons sur `master`

↪

```
git checkout master
```


Le Graphe : Branche `master`

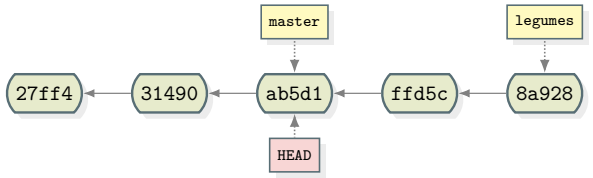


(a) Travaillons sur `master`

⇒ `legumes.txt` n'existe plus dans la Copie de Travail (*Working Directory*)

```
git checkout master
```

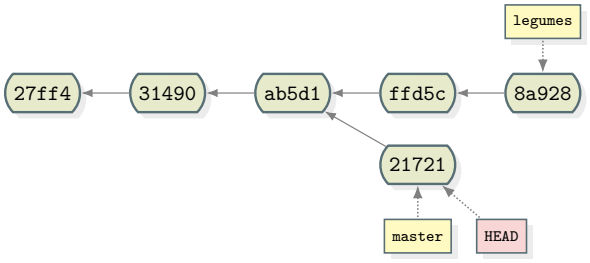

Le Graphe : Branche `master`



(a) Travaillons sur `master`

```
git checkout master
↪
echo poire >> fruits.txt ; git add fruits.txt
git commit -m "Ajouté poire à fruits.txt"
    ⇒ ID = 21721
```

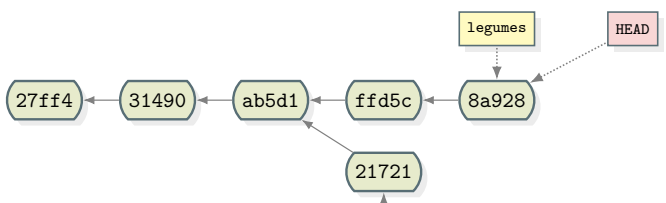

Le Graphe : Branche `master`



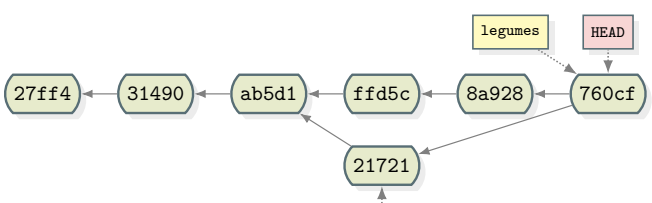
(a) Après nouveau *commit* sur `master`

```
git checkout master
echo poire >> fruits.txt ; git add fruits.txt
git commit -m "Ajouté poire à fruits.txt"
↪  ⇒ ID = 21721
```


Le Graphe : Merge `master`⇒`legumes`

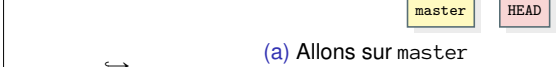


Le Graphe : Merge `master`⇒`legumes`



```
* ab5d1c0 [2017-12-01] Ajouté orange à fruits.txt [rudametw]
* 3149017 [2017-12-01] Ajouté banane à fruits.txt [rudametw]
* 27ff4c1 [2017-11-30] Pomme ajouté à la liste de fruits [rudametw]
```

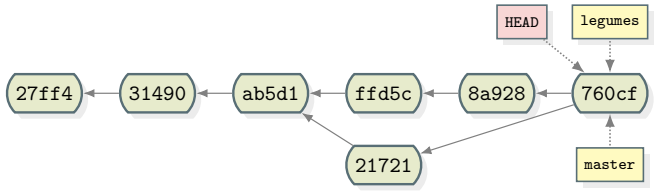
```
git log --all --graph --oneline --date=short
```

(a) Allons sur `master`

```
git checkout master
```

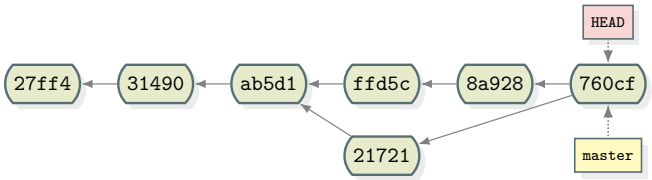

Le Graphe : Merge `legumes`⇒`master`



(a) Merger `légumes` dans `master` : pas de nouveau commit

```
git checkout master
git diff legumes
git merge legumes
```

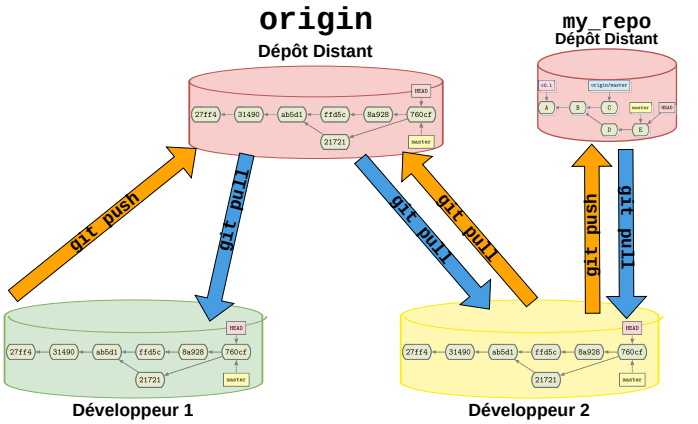

Le Graphe : Merge `legumes`⇒`master`



(a) Effacer la branche `légumes`

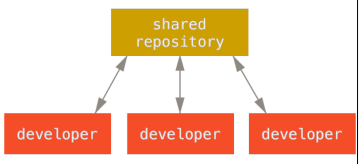
```
git checkout master
git diff legumes
git merge legumes
git branch -d legumes
```


Partager : dépôts distants



Dépôt Centralisée : initialisation

Premier commit
(dépôt central doit être créé et vide)

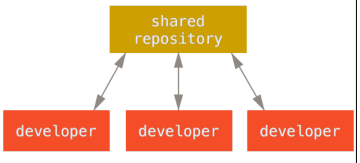



```
git init .
git add .
git commit -m "first commit"

git remote add origin
  ↪ git@github.com:rudametw/Learning-Git-Test-Repo.git
git push -u origin master
```


Dépôt Centralisée : initialisation

Premier commit
(dépôt central doit être créé et vide)




```
git init .
git add .
git commit -m "first commit"

git remote add origin
```

Dépôt Centralisée : travail

Chacun travaille sur une branche `fonctX`. Une fois la fonctionnalité fini, on merge `foncX` dans `master`.

```
git pull ; git status //update & check work
git branch fonctionnalitéX
git checkout fonctionnalitéX
while (je travaille = vrai) {
```

. . . comment faire pour les résoudre ?

Provoquer un conflit dans `fruits.txt`

Branche ananas

```
git checkout master
git branch ananas
git checkout ananas
awk 'NR==3\{print
↪  "ananas"\}1' fruits.txt >
↪  fruits.txt
git add fruits.txt
git commit -m "+ananas"
```

Branche kaki

```
git checkout master
git branch kaki
git checkout kaki
awk 'NR==3\{print kaki\}1'
↪  fruits.txt | grep -v
↪  orange > fruits.txt
git add fruits.txt
git commit -m "+kaki -orange"
```

Les merges

```
git checkout master
git merge ananas
```

Sorties console

```
Updating 760cf0e..1711864
Fast-forward
fruits.txt | 1 +
1 file changed, 1 insertion(+)
```

```
git merge kaki
```

```
Auto-merging fruits.txt
CONFLICT (content): Merge conflict in fruits.txt
Automatic merge failed; fix conflicts and then
↪  commit the result.
```


`diff` entre `ananas` et `kaki` avant de merger

```
wrudamet@beaner[merge_fruits L|✔] ~/COURS/Git/mon_depot $ git diff 1711864 34dabb6
diff --git a/fruits.txt b/fruits.txt
index e3922ba..5dbddd0 100644
--- a/fruits.txt
+++ b/fruits.txt
@@ -1,5 +1,4 @@
 pomme
 banane
-ananas
-orange
+kaki
 poire
```

Différences entre les *commits* réalisés sur les branches `kaki` et `ananas` qui avaient pour objectif de produire un conflit. En rouge, les lignes qui existent sur la branche `ananas` et pas `kaki`. En vert les lignes qui éxistent sur la branche `kaki` et pas `ananas`.

Résoudre un conflit dans `fruits.txt`

immédiatement après la commande `git merge kaki`

Conflit dans `fruits.txt`

`git` ajoute des guides pour s'y retrouver

```
pomme
banane
<<<<<<< HEAD
ananas
orange
||||||| merged common ancestors
orange
=======
kaki
>>>>>>>
poire
```


Résoudre un conflit dans `fruits.txt`

immédiatement après la commande `git merge kaki`

Conflit dans `fruits.txt`

`git` ajoute des guides pour s'y retrouver

```
pomme
banane
<<<<<<< HEAD
ananas
orange
||||||| merged common ancestors
orange
=======
kaki
>>>>>>>
poire
```

Solution *(édité à la main)*

```
pomme
banane
ananas
kaki
poire
```


Résoudre un conflit dans `fruits.txt`

immédiatement après la commande `git merge kaki`

Conflit dans `fruits.txt`

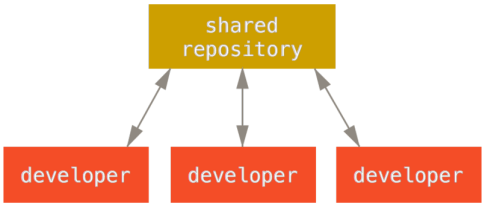
`git` ajoute des guides pour s'y retrouver

```
pomme
banane
```

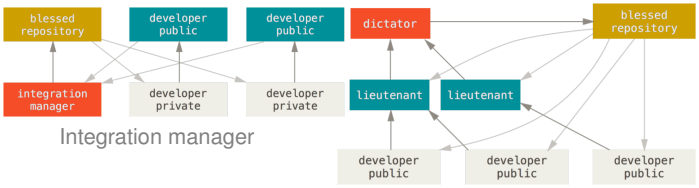
Solution *(édité à la main)*

```
pomme
banane
ananas
kaki
```

Git distribué : Développements distribués



Centralized



Integration manager

Benevolent Dictator

//fr.slideshare.net/PierreSudron/diapo-git

- Où stocker vos projets
  - https://archives.plil.fr/
  - https://github.com/
  - https://bitbucket.org/
  - Votre serveur perso

Liens, aides et outils (2/2)

- Tutoriels
  - http://www.cristal.univ-lille.fr/TPGIT/
  - https://learngitbranching.js.org/
  - https://try.github.io/
  - https://www.miximum.fr/blog/enfin-comprendre-git/
- Vidéos
  - https://www.youtube.com/watch?v=OqmSzXDrJBk
  - https://www.youtube.com/watch?v=uR6G2v_WsRA
  - https://www.youtube.com/watch?v=3a2x1iJFJWc
  - https://www.youtube.com/watch?v=1ffBJ4sVUb4
  - https://www.youtube.com/watch?v=duqBHik7nRo