Démystifions Git, Github, Gitlab

Notion d'historique

Travailler à plusieurs

L'écosystème Git

Aaah... l'informatique

Problème

- Mon ordinateur a planté !

Solution ?

Aaah... l'informatique

Problème

- Mon ordinateur a planté !
- Ce document était bien mieux hier !

Solution ?

Aaah... l'informatique

Problème

- Mon ordinateur a planté !
- Ce document était bien mieux hier !
- Vous n'avez pas travaillé sur la bonne version du document.

Solution ?

Aaah... l'informatique

Problème

- Mon ordinateur a planté !
- Ce document était bien mieux hier !
- Vous n'avez pas travaillé sur la bonne version du document.
    - Comment fusionner ?

Solution ?

Aaah... l'informatique

Problème

- Mon ordinateur a planté !
- Ce document était bien mieux hier !
- Vous n'avez pas travaillé sur la bonne version du document.
  - Comment fusionner ?
  - Mais qui a écrit ça ? Et pourquoi ?

Solution ?

Aaah... l'informatique

Problème

- Mon ordinateur a planté !
- Ce document était bien mieux hier !
- Vous n'avez pas travaillé sur la bonne version du document.
  - Comment fusionner ?
  - Mais qui a écrit ça ? Et pourquoi ?

Solution ?

- Sauvegarde régulière avec une notion d'historique

  « `thesis_slides_16x9_Template_Arial.pptx` »

Aaah... l'informatique

Problème

- Mon ordinateur a planté !
- Ce document était bien mieux hier !
- Vous n'avez pas travaillé sur la bonne version du document.
  - Comment fusionner ?
  - Mais qui a écrit ça ? Et pourquoi ?

Solution ?

- Sauvegarde régulière avec une notion d'historique

  « `thesis_slides_16x9_Template_Arial_FINAL.pptx` »

Aaah... l'informatique

Problème

- Mon ordinateur a planté !
- Ce document était bien mieux hier !
- Vous n'avez pas travaillé sur la bonne version du document.
  - Comment fusionner ?
  - Mais qui a écrit ça ? Et pourquoi ?

Solution ?

- Sauvegarde régulière avec une notion d'historique

  « `thesis_slides_16x9_Template_Arial_FINAL_FINAL.pptx` »

Aaah... l'informatique

Problème

- Mon ordinateur a planté !
- Ce document était bien mieux hier !
- Vous n'avez pas travaillé sur la bonne version du document.
  - Comment fusionner ?
  - Mais qui a écrit ça ? Et pourquoi ?

Solution ?

- Sauvegarde régulière avec une notion d'historique

  « `thesis_slides_16x9_Template_Arial_FINAL_FINAL_0925.pptx` »

Aaah... l'informatique

Problème

- Mon ordinateur a planté !
- Ce document était bien mieux hier !
- Vous n'avez pas travaillé sur la bonne version du document.
  - Comment fusionner ?
  - Mais qui a écrit ça ? Et pourquoi ?

Solution ?

- Sauvegarde régulière avec une notion d'historique

  « `thesis_slides_16x9_Template_Arial_FINAL_FINAL_0925.pptx` »
- Passer son temps à (s')envoyer des mails !

Gestionnaire de versions

Les gestionnaires de version datent du début des années 70

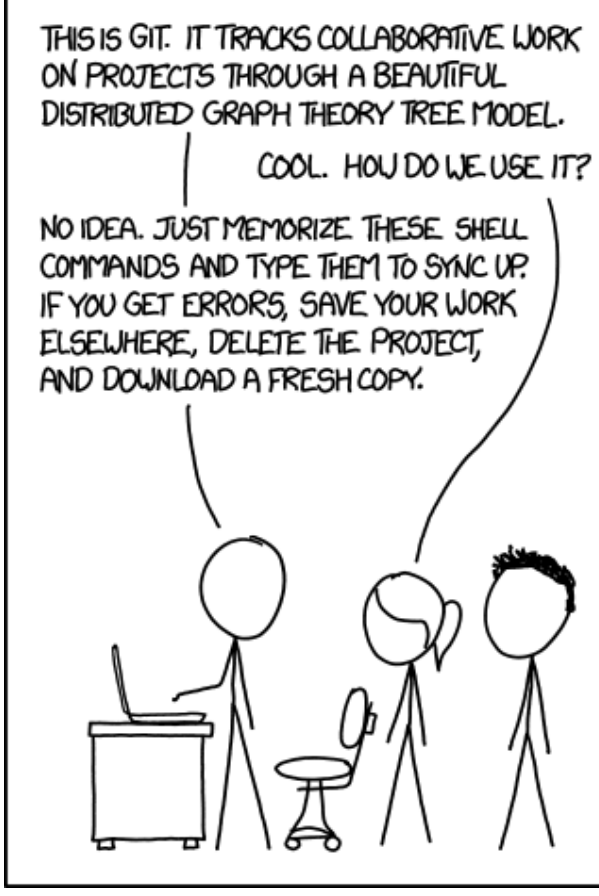
Fonctionnalités essentielles

- Conserve un historique des modifications sans dupliquer les dossiers de sauvegardes
- Sauvegarde les données (avec l'historique) sur des ordinateurs distants
- Fusion "automatique" des fichiers édités par plusieurs personnes

- 2005 : créé pour le développement du noyau Linux par Linus Torvalds
  - $>$ 20 millions de lignes de code
  - $\approx$ 14 000 developeurs

Objectif de cet exposé

- ~~Apprendre à utiliser `git` en ligne de commande~~

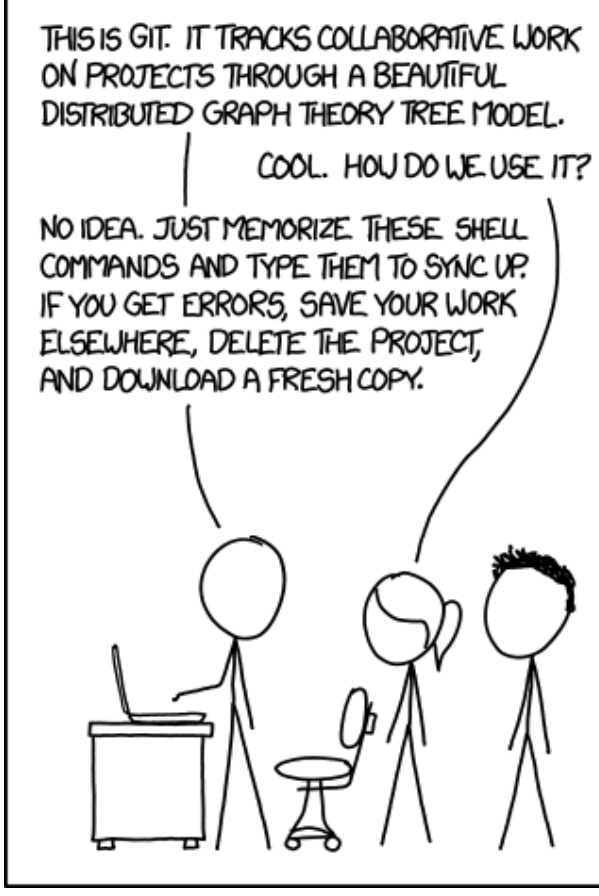


Objectif de cet exposé

- ~~Apprendre à utiliser `git` en ligne de commande~~

  `git` est "explicite" (à la différence de dropbox et autres)

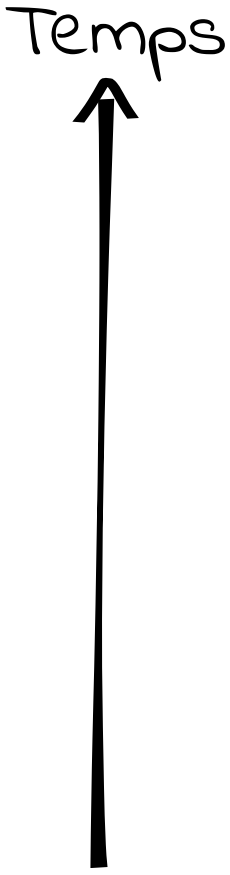
  Heureusement, il y a des interfaces



- Notions essentielles pour utiliser les interfaces à `git`
  - Historique distribué
  - Branche
  - Conflit et fusion
  - L'écosystème git

    GitHub GitLab

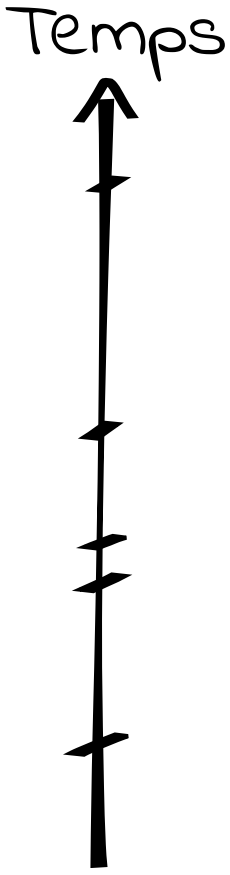
Son propre projet



Accès à un certain nombre de fichiers.

- Temps continu

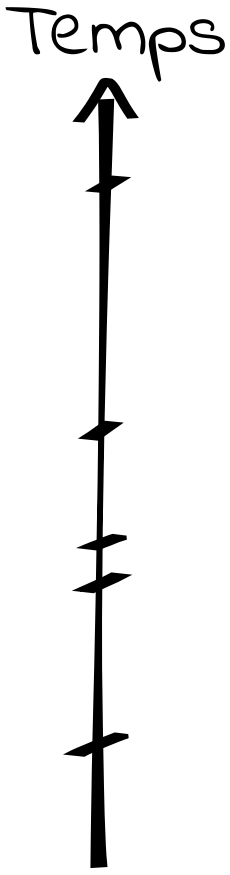
Son propre projet



Accès à un certain nombre de fichiers.

- Temps continu
- Checkpoint/snapshot : conserver les moments clés du projet pour pouvoir y revenir

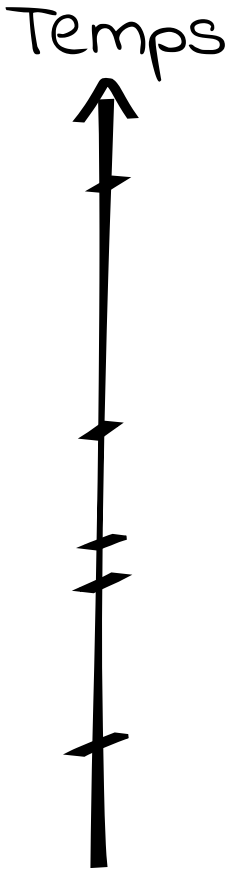
Son propre projet



Accès à un certain nombre de fichiers.

- Temps continu
- Checkpoint/snapshot : conserver les moments clés du projet pour pouvoir y revenir
  - git **add** pour indiquer quelles modifications on souhaite conserver

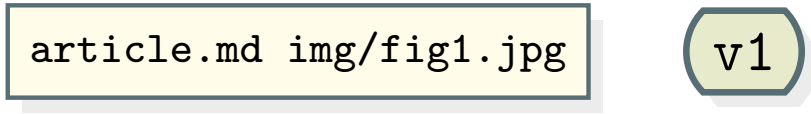
Son propre projet



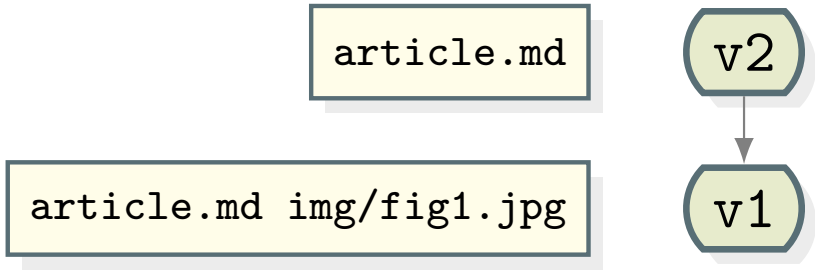
Accès à un certain nombre de fichiers.

- Temps continu
- Checkpoint/snapshot : conserver les moments clés du projet pour pouvoir y revenir
  - git **add** pour indiquer quelles modifications on souhaite conserver
  - git **commit** pour faire le checkpoint (date, commentaire)

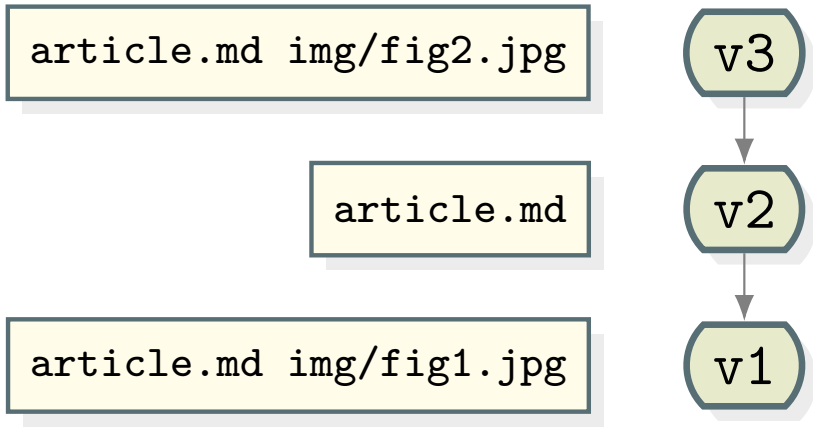
Une histoire simple



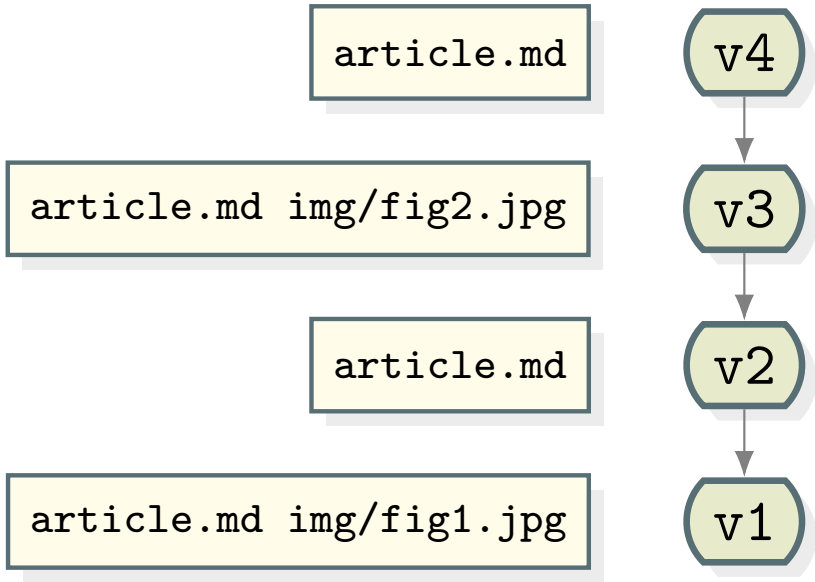
Une histoire simple



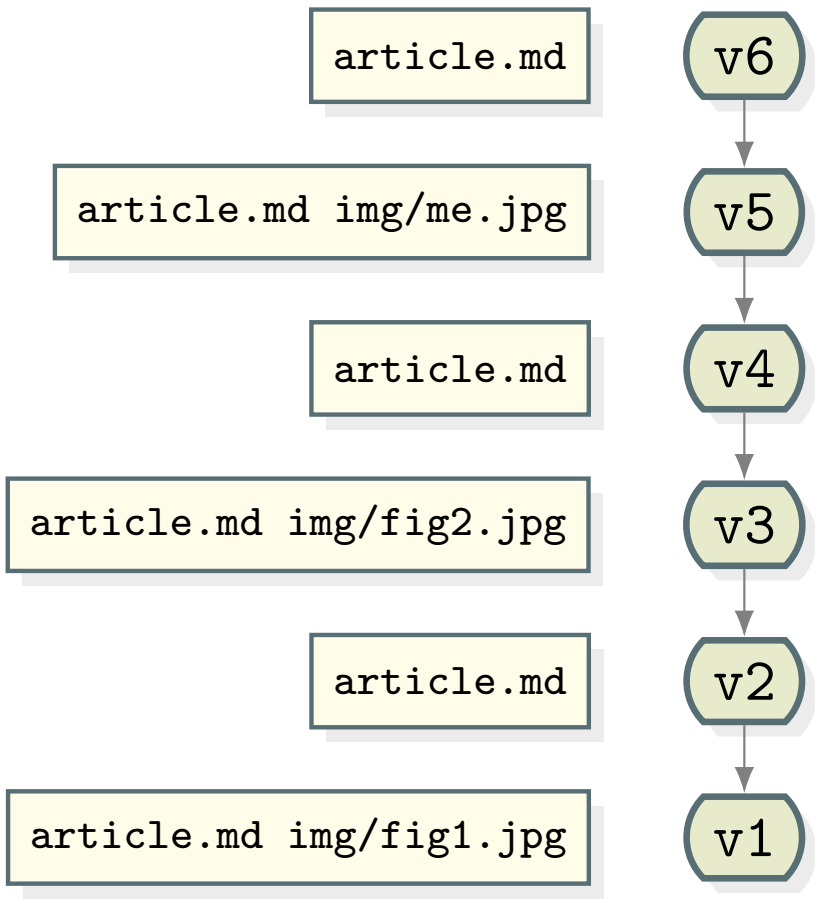
Une histoire simple



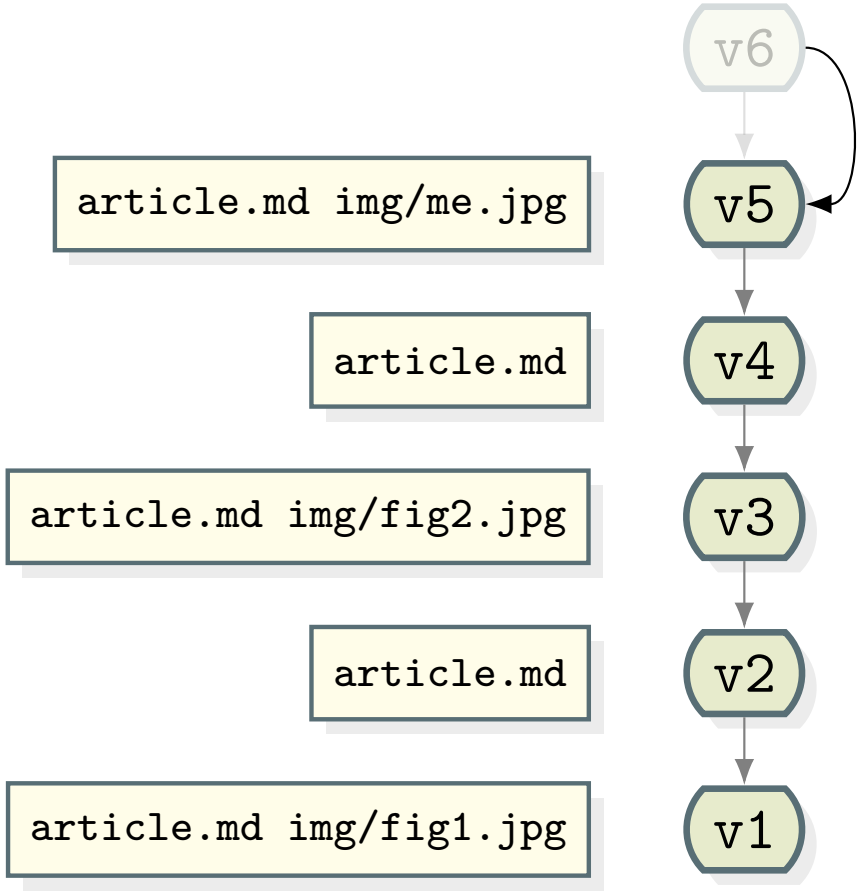
Une histoire simple



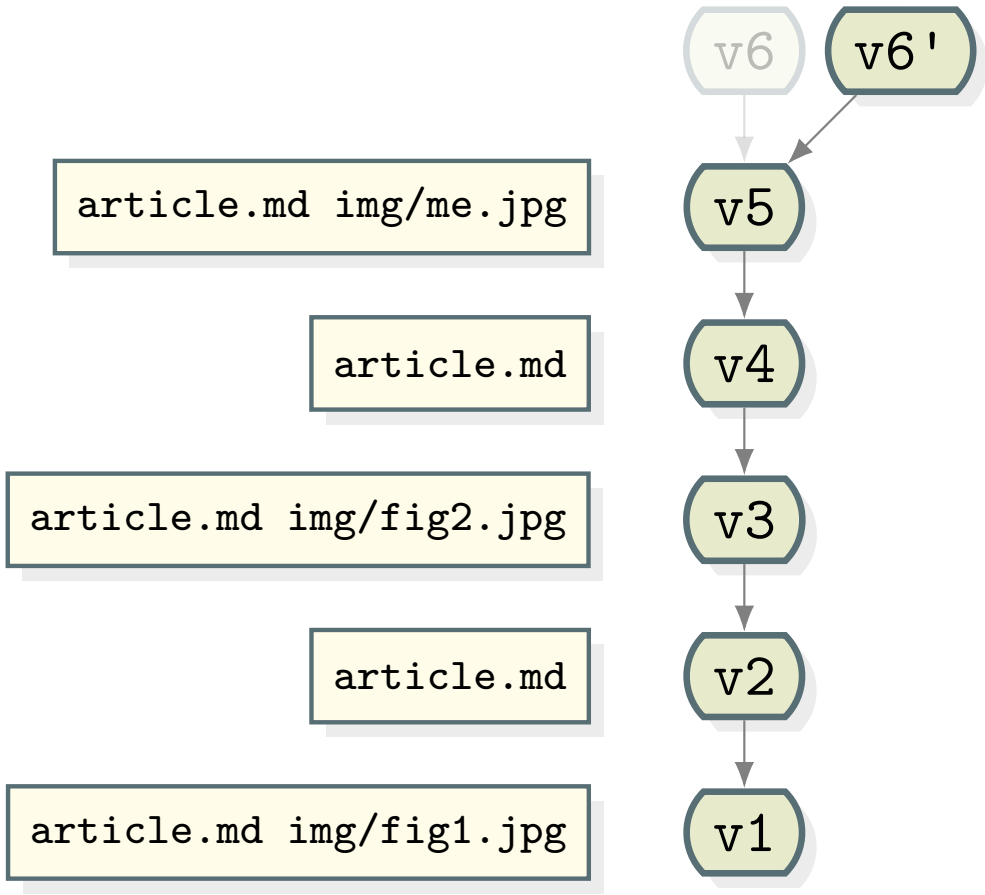
Une histoire simple



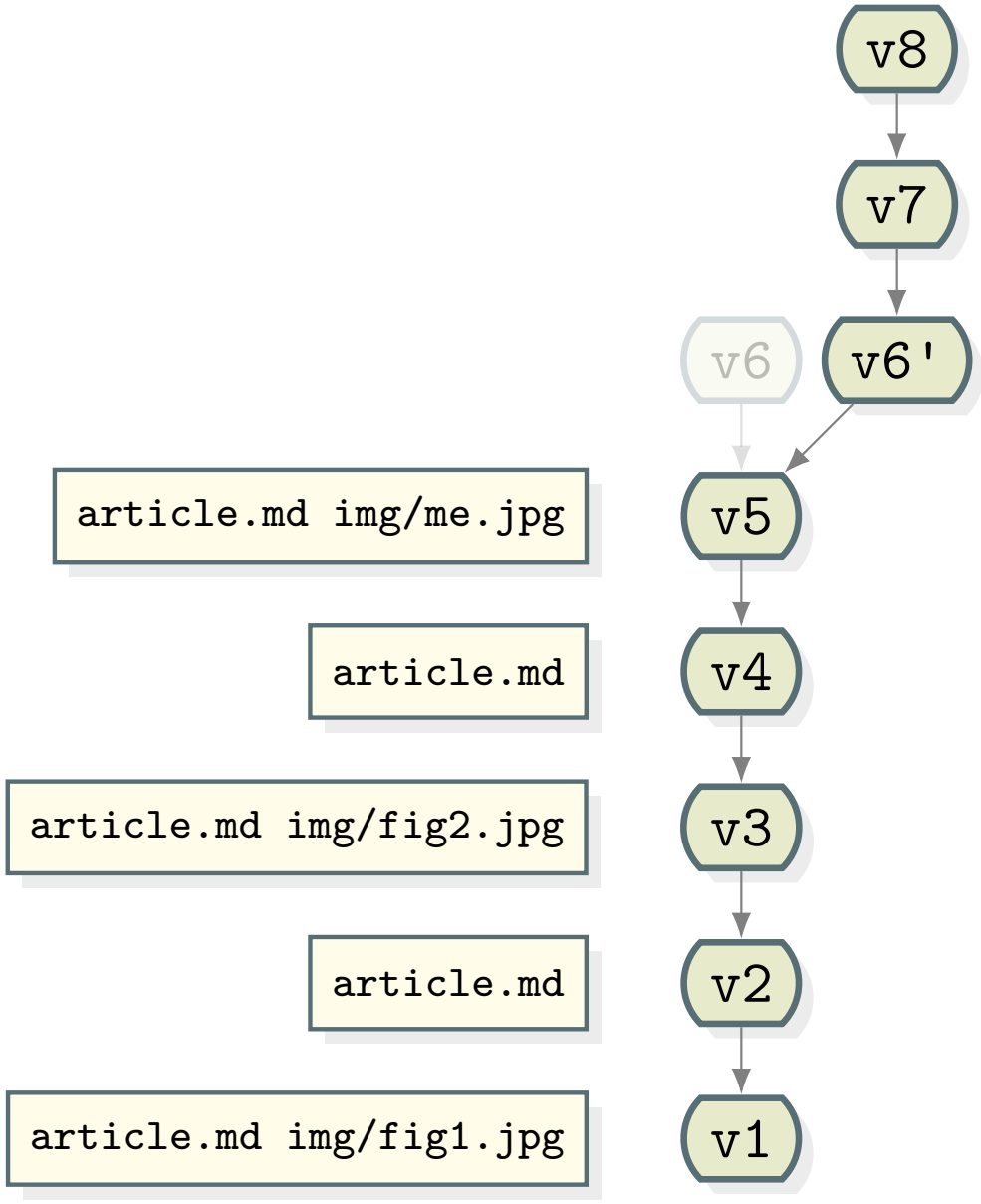
Une histoire simple



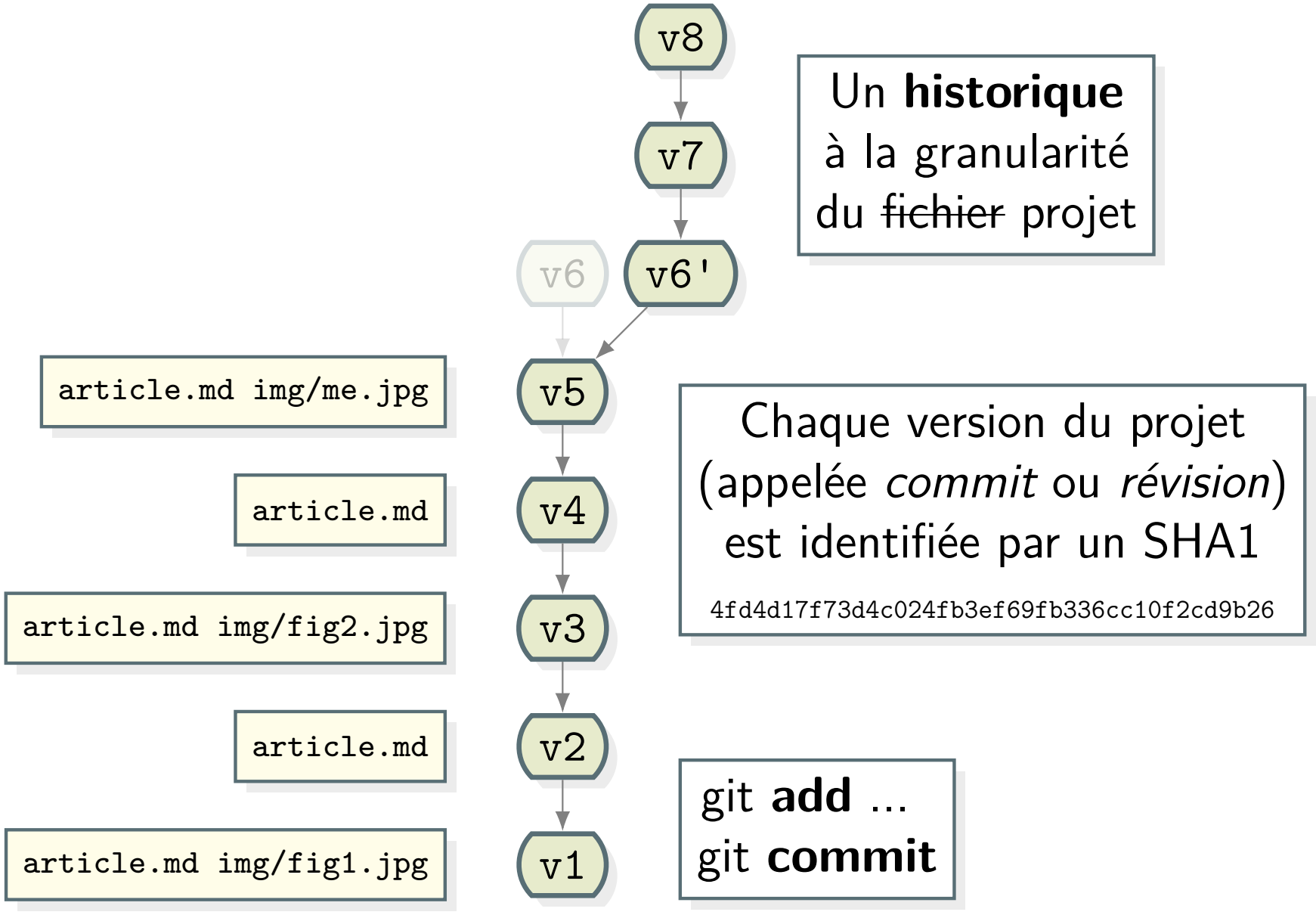
Une histoire simple



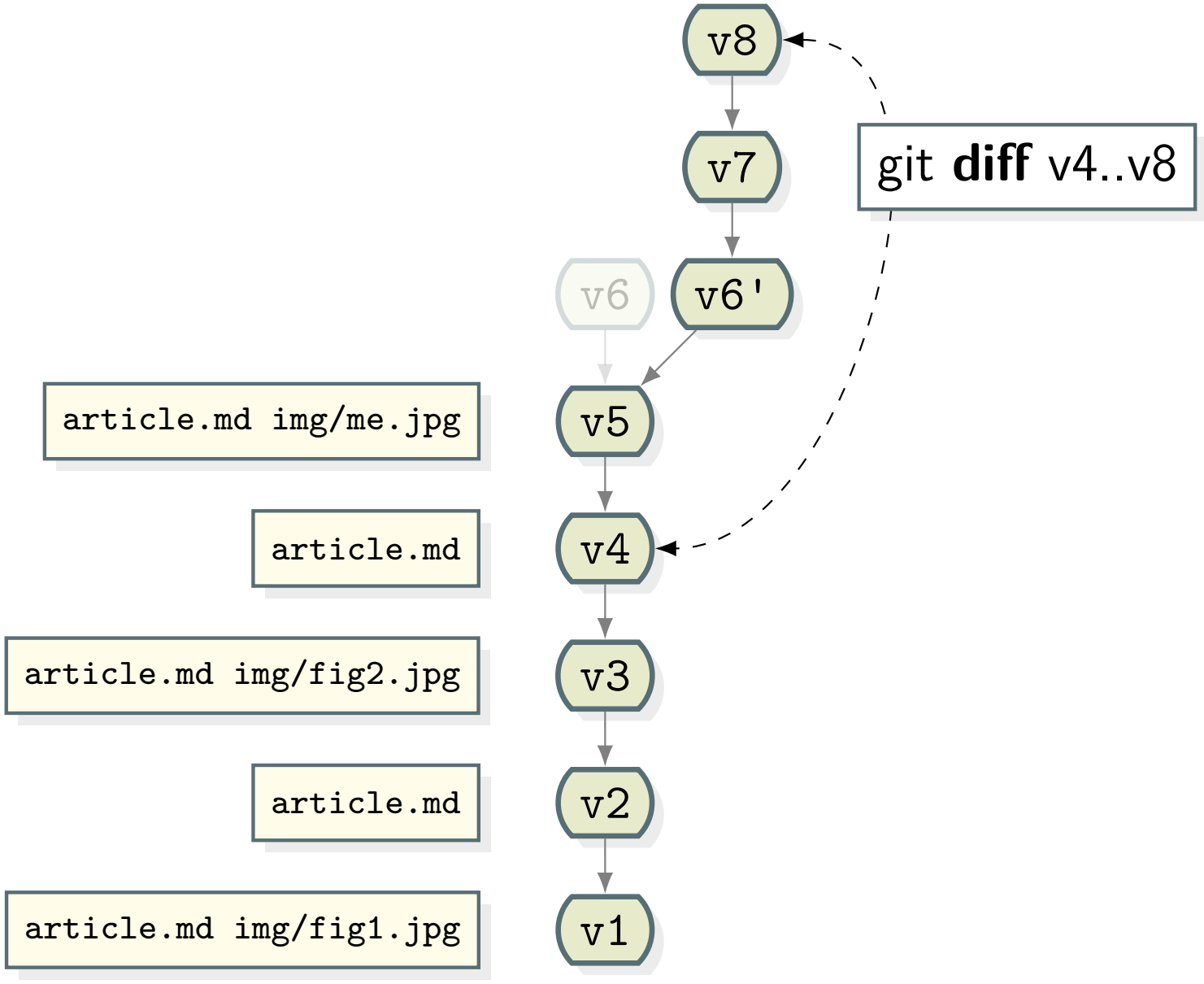
Une histoire simple



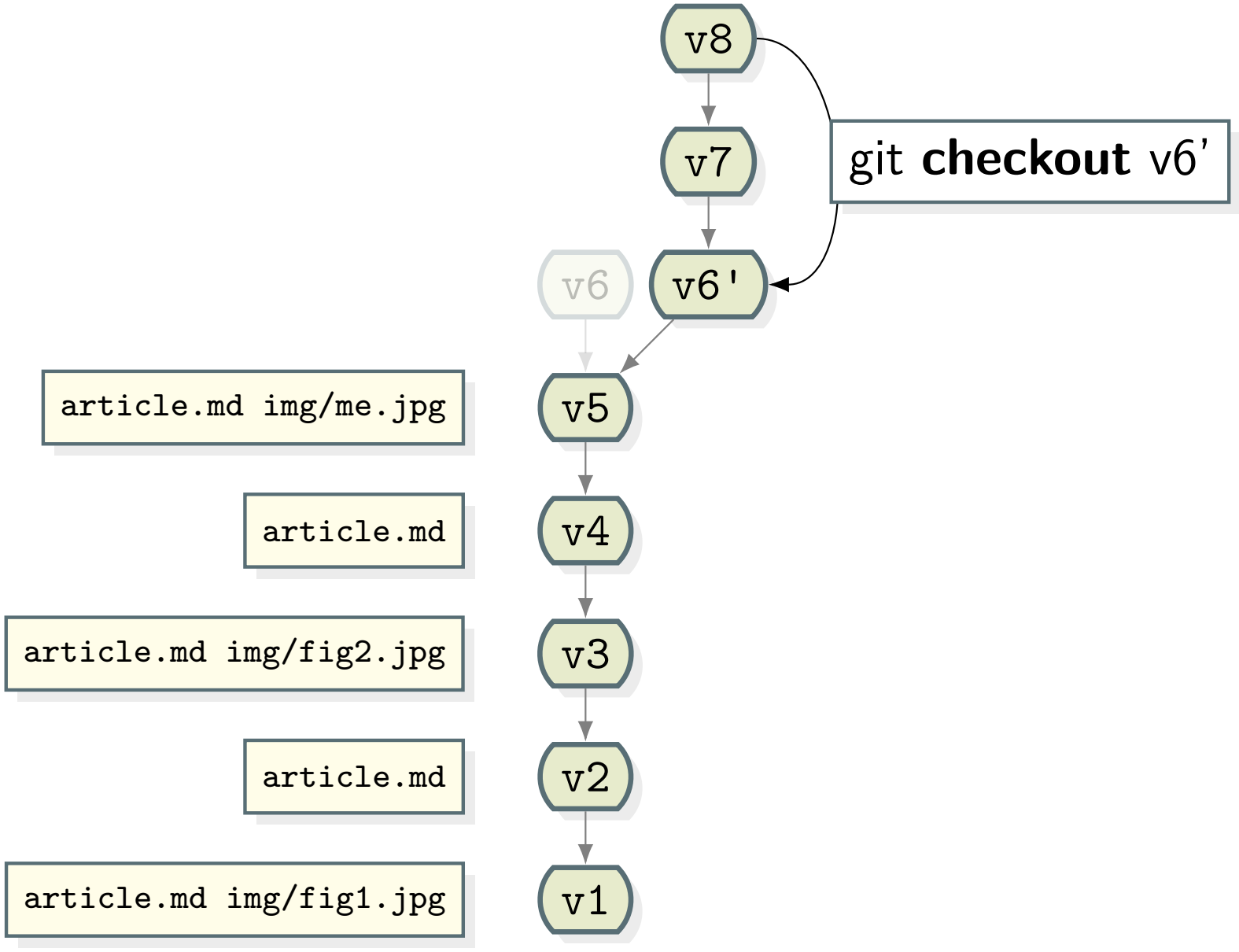
Une histoire simple



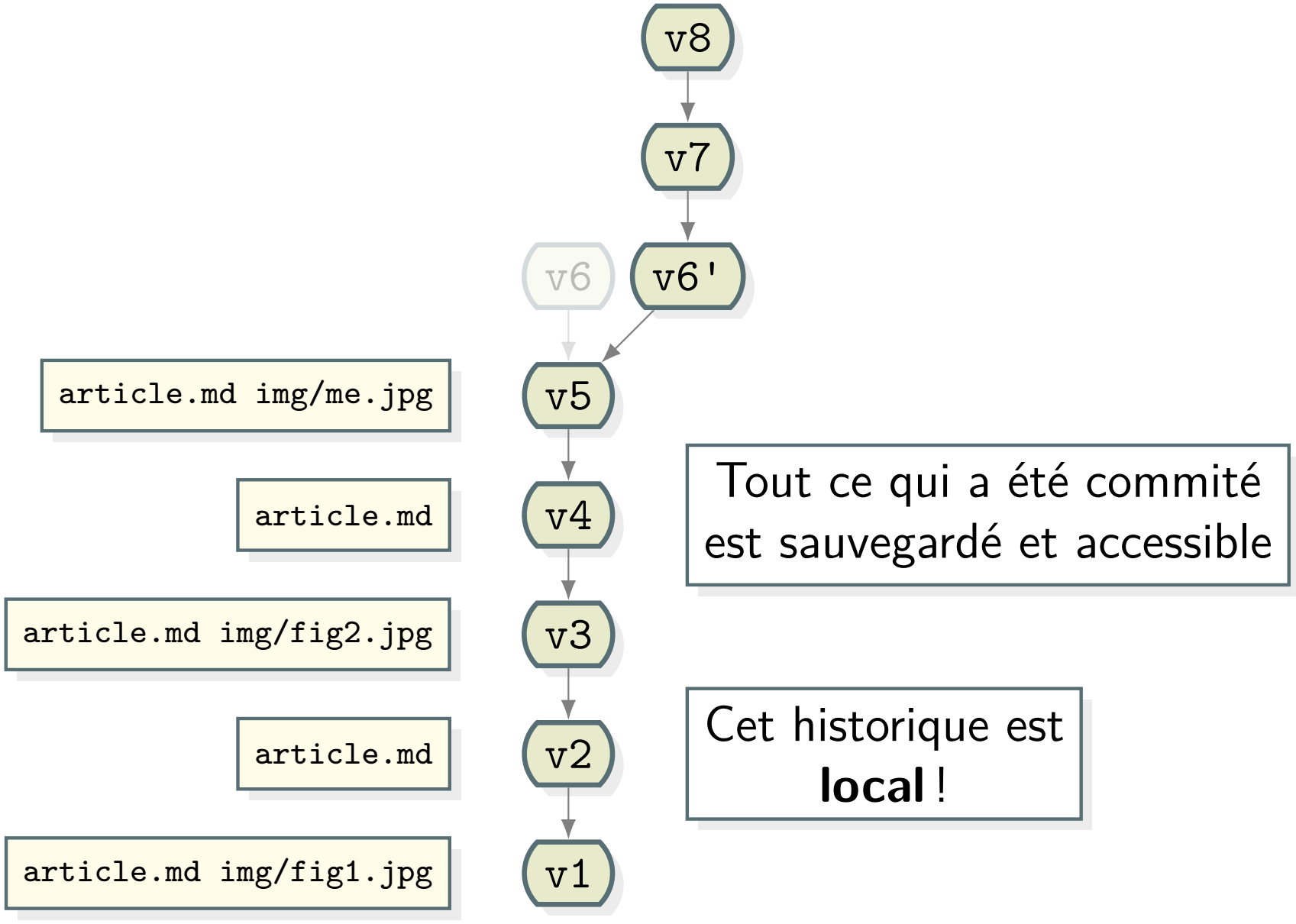
Une histoire simple



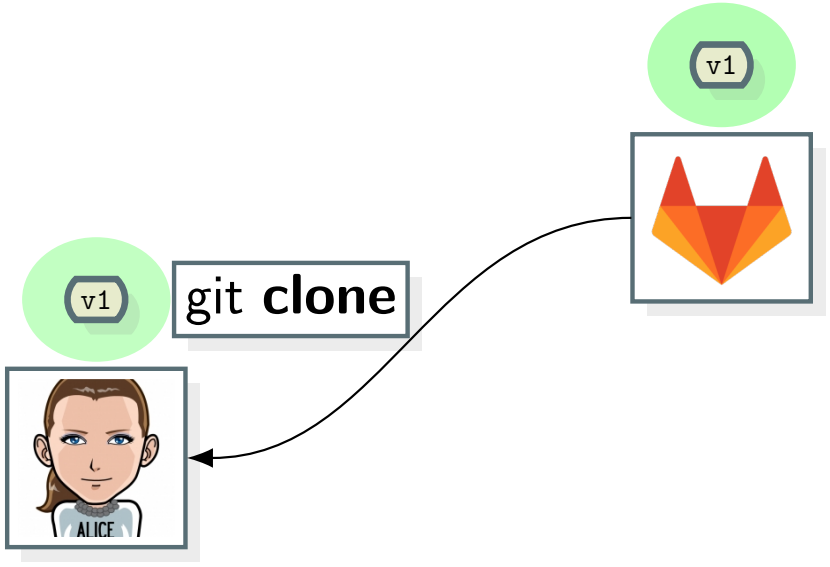
Une histoire simple



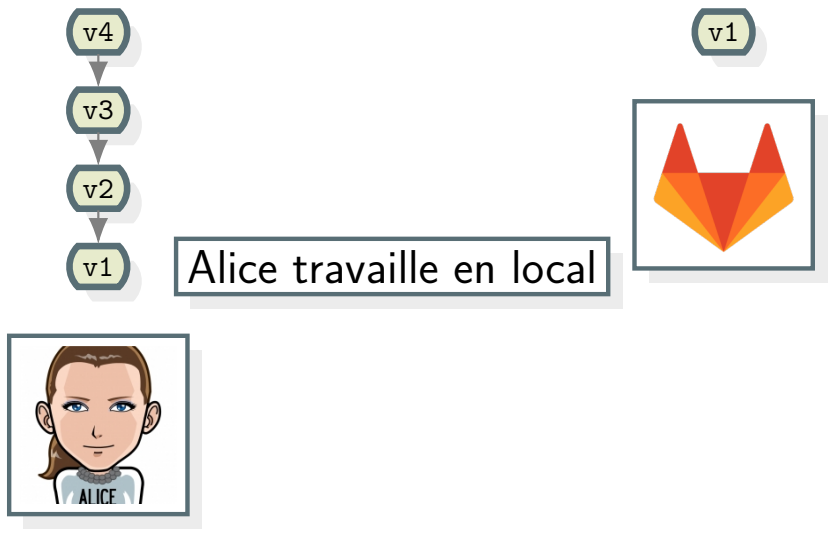
Une histoire simple



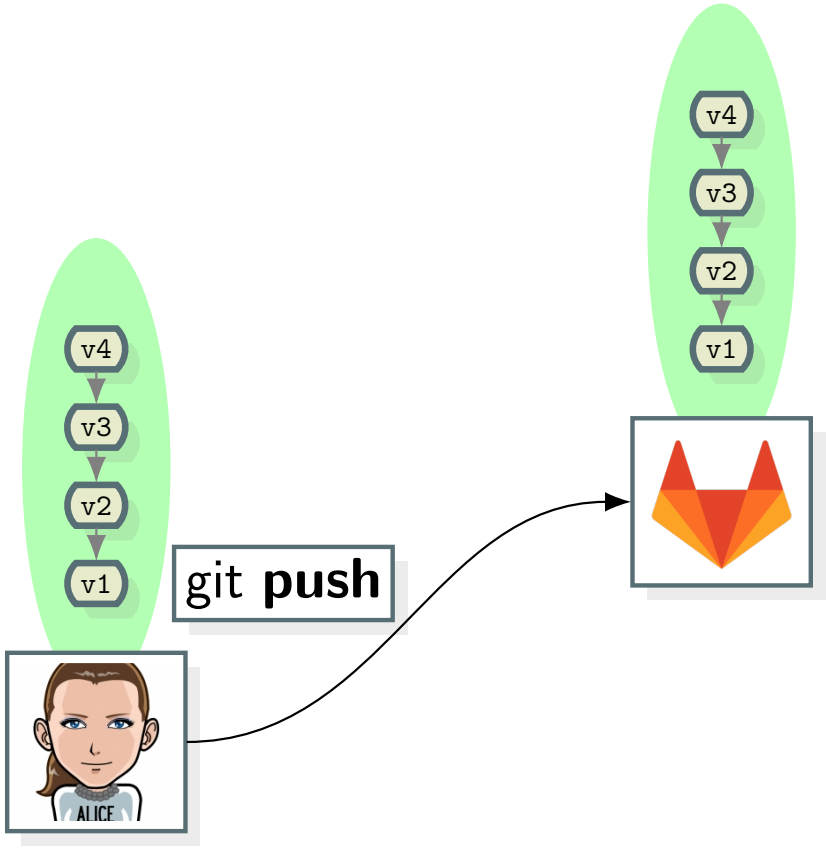
Backup sur un server distant



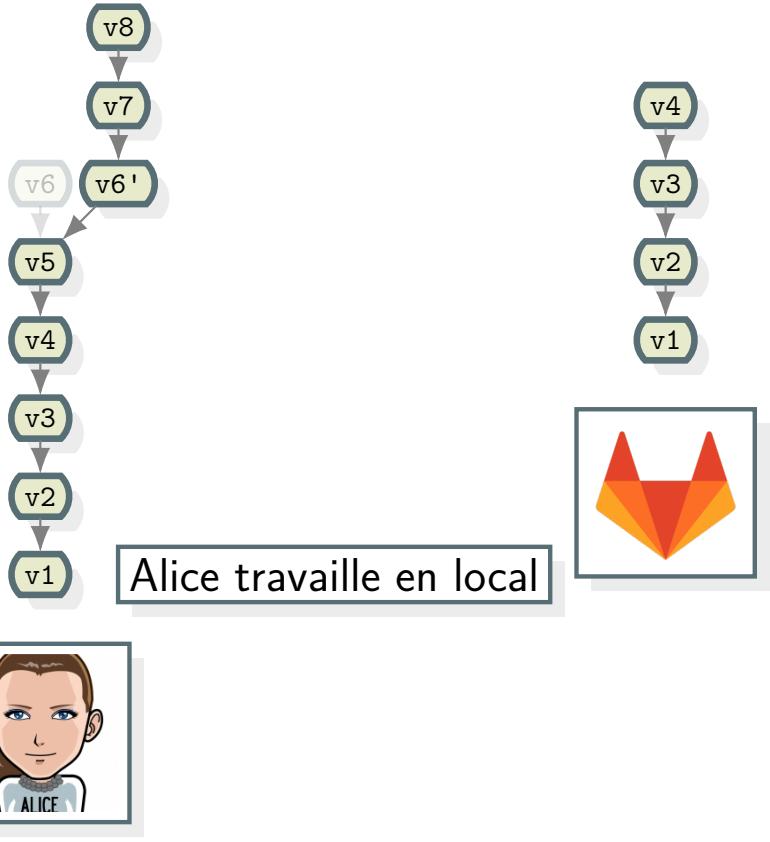
Backup sur un server distant



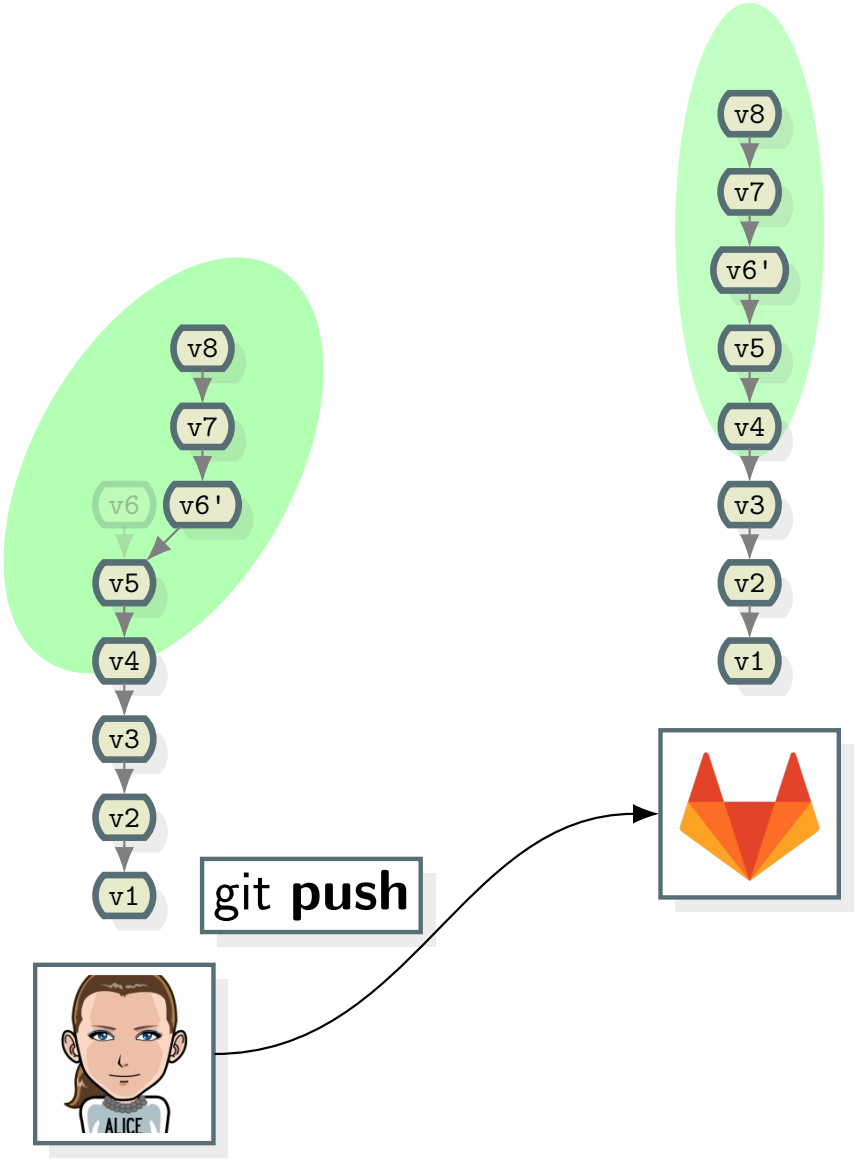
Backup sur un server distant



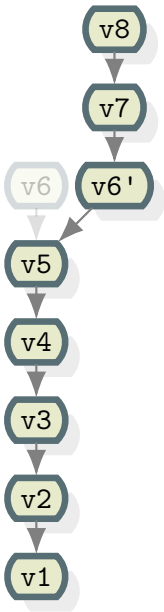
Backup sur un server distant

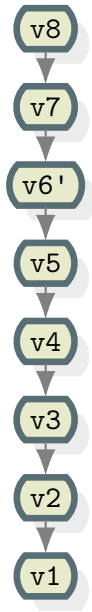


Backup sur un server distant



Backup sur un server distant





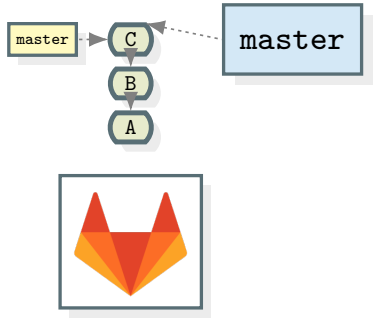
On synchronise des **branches**
en échangeant
des morceaux d'historiques

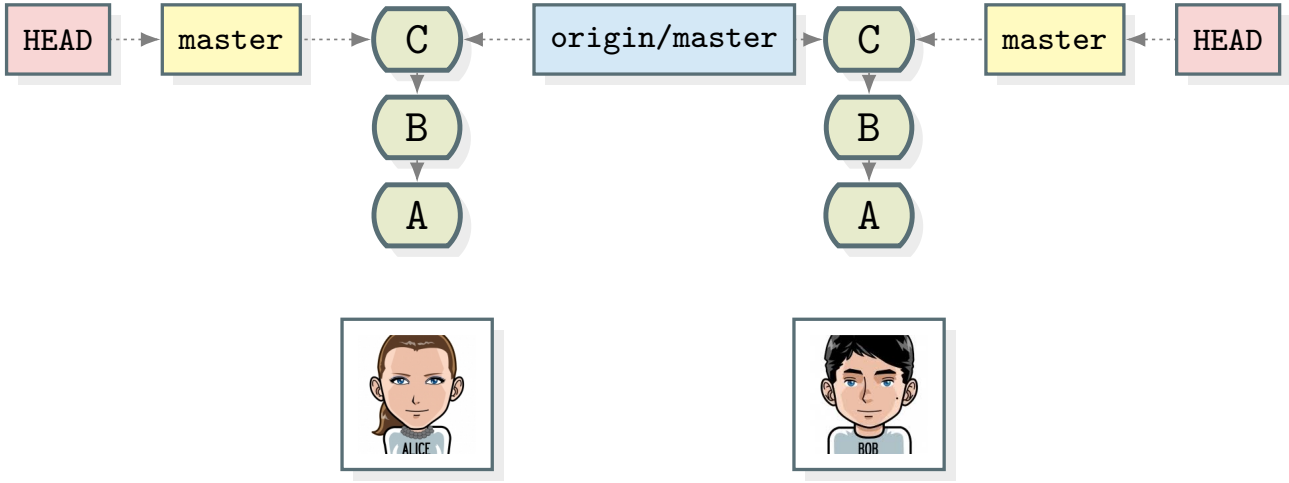
Démystifions Git, Github, Gitlab

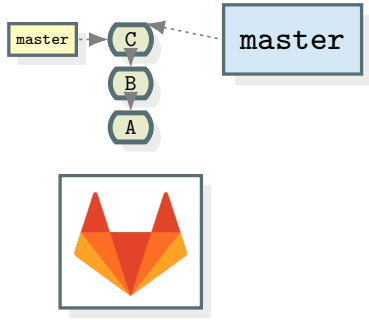
Notion d'historique

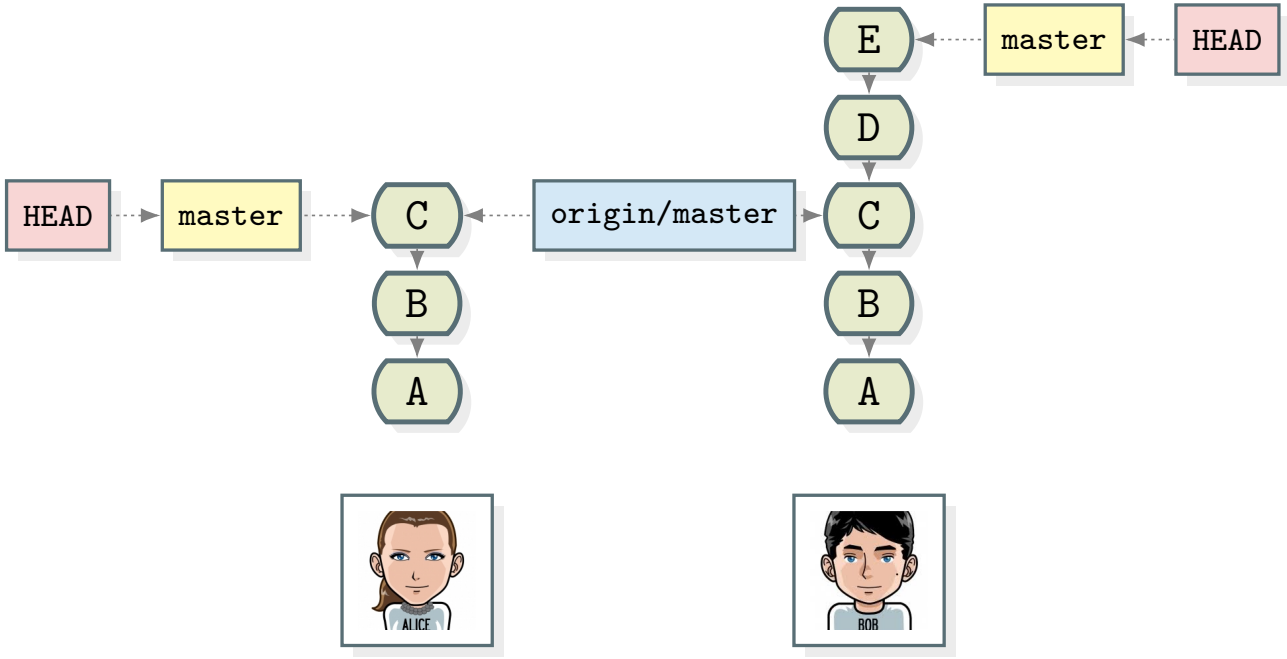
Travailler à plusieurs

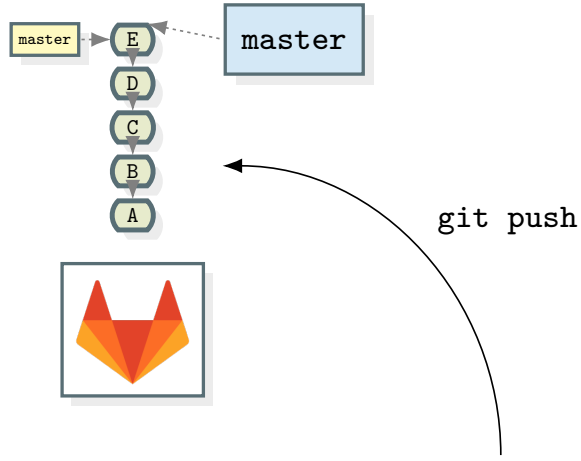
L'écosystème Git

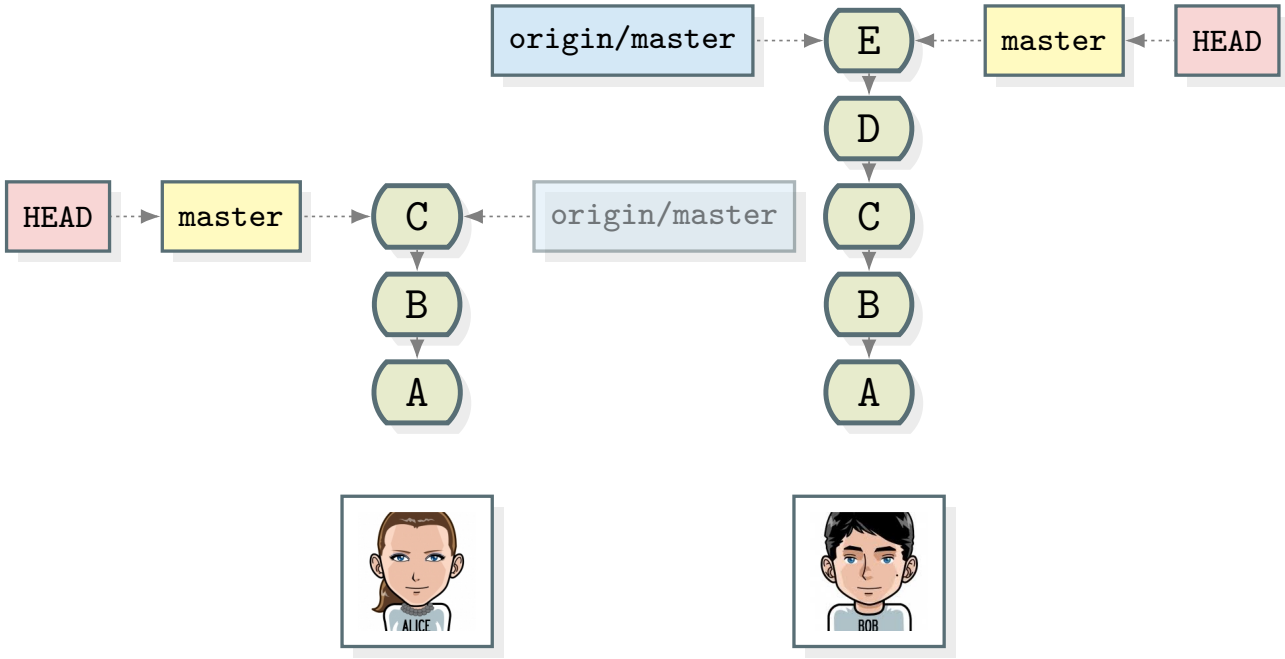

master
C
B
A
master

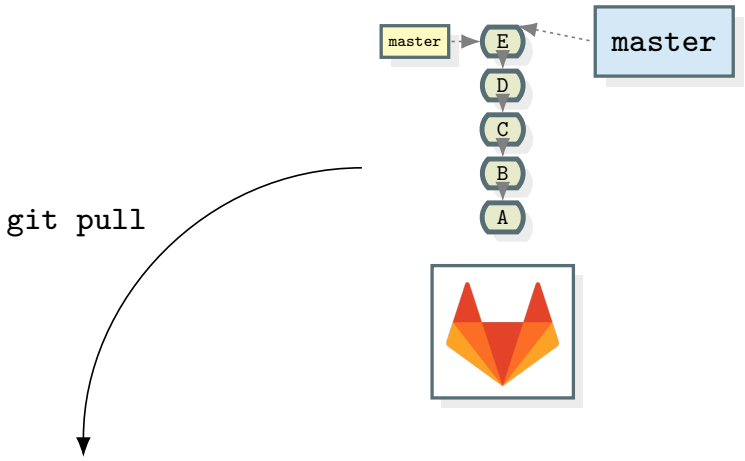

HEAD
master
C
B
A
origin/master
C
B
A
master
HEAD
ALICE
BOB

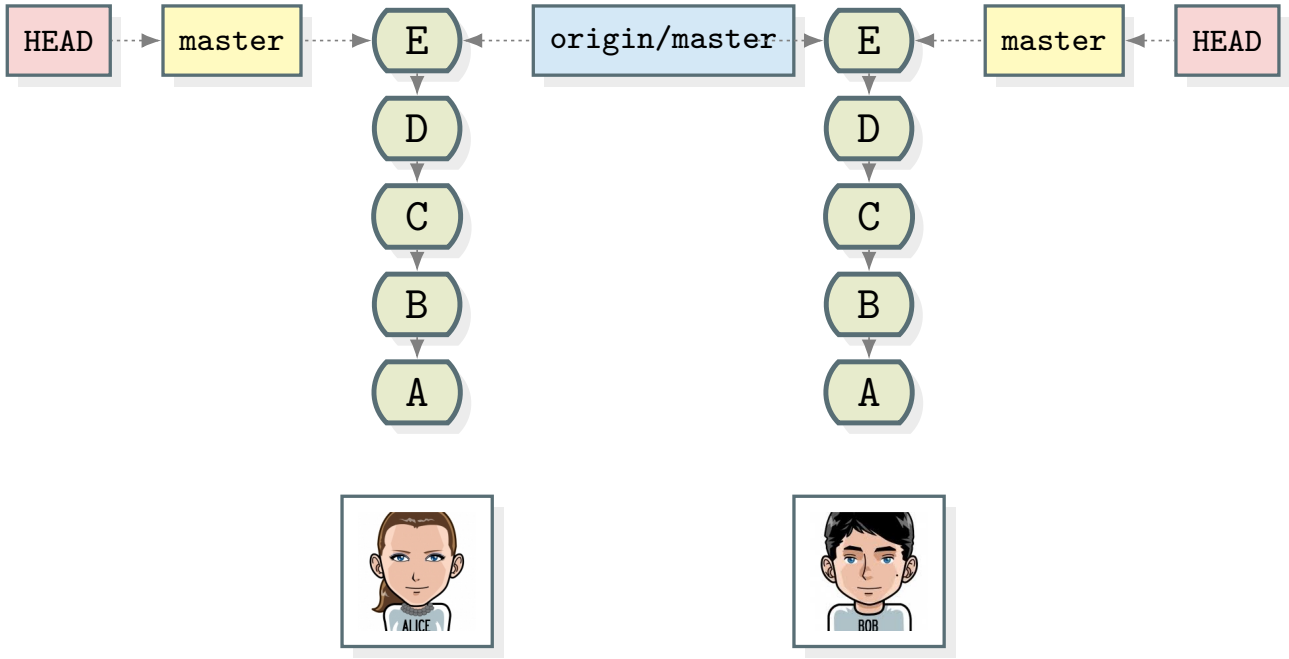

master
C
B
A
master

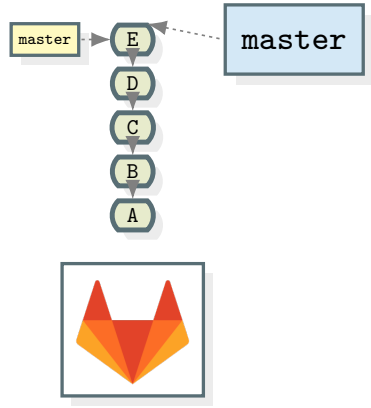

E
master
HEAD
D
HEAD
master
C
origin/master
C
B
B
A
A
ALICE
BOB

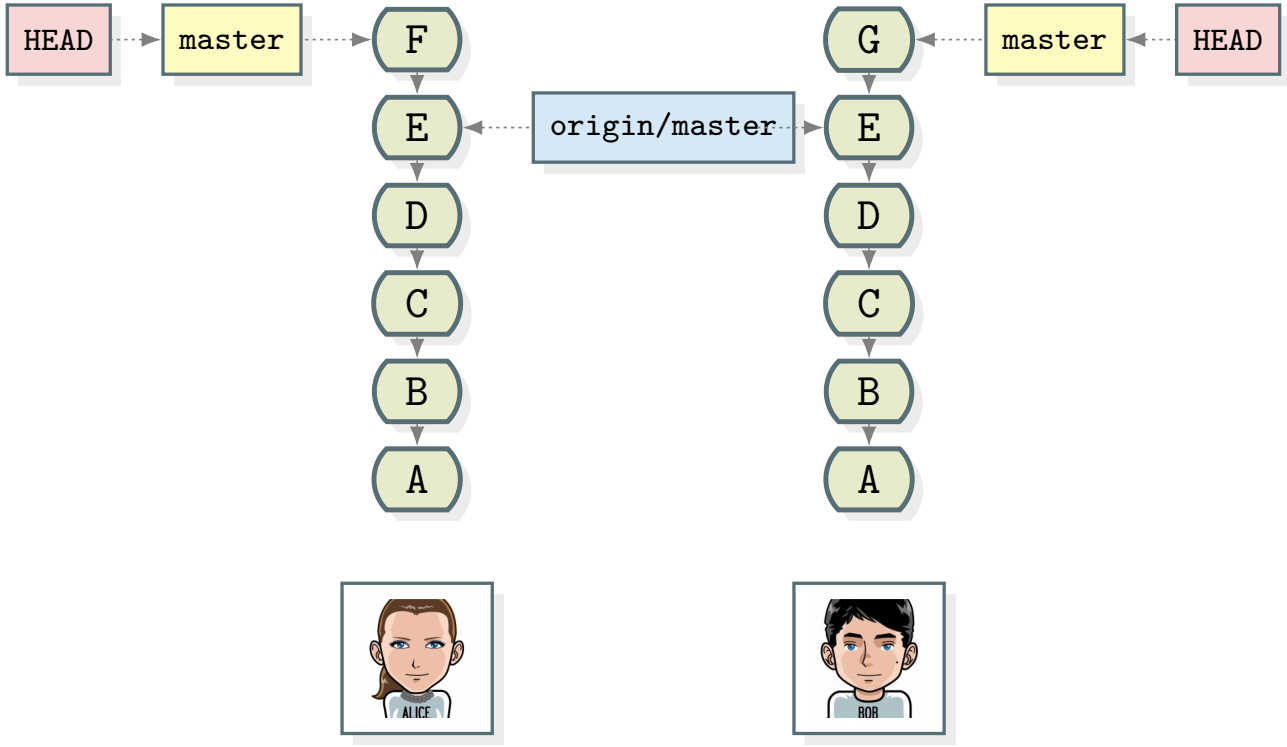

master
E
D
C
B
A
master
git push

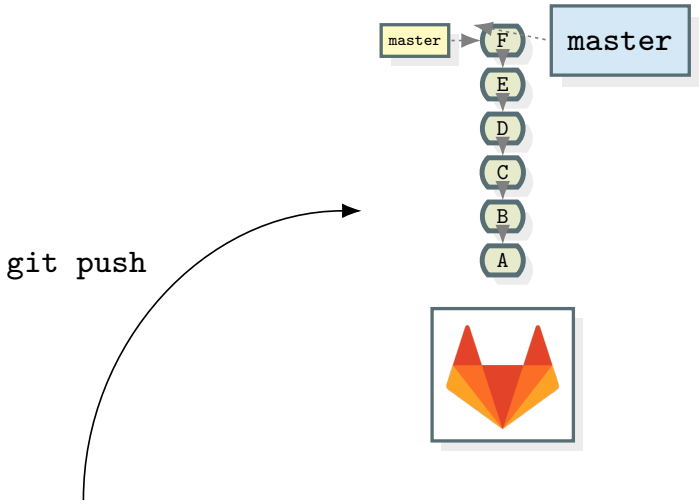

origin/master
E
master
HEAD
D
HEAD
master
C
origin/master
C
B
B
A
A
ALICE
BOB

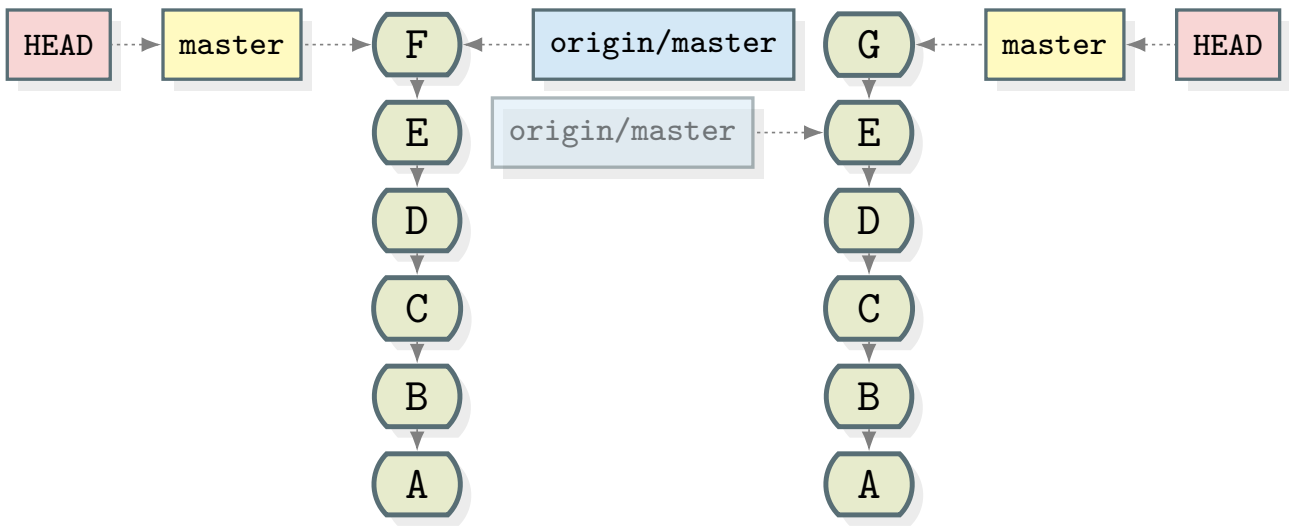

master
E
D
C
B
A
master
git pull


HEAD
master
E
D
C
B
A
origin/master
E
D
C
B
A
master
HEAD
ALICE
BOB


master
E
D
C
B
A
master

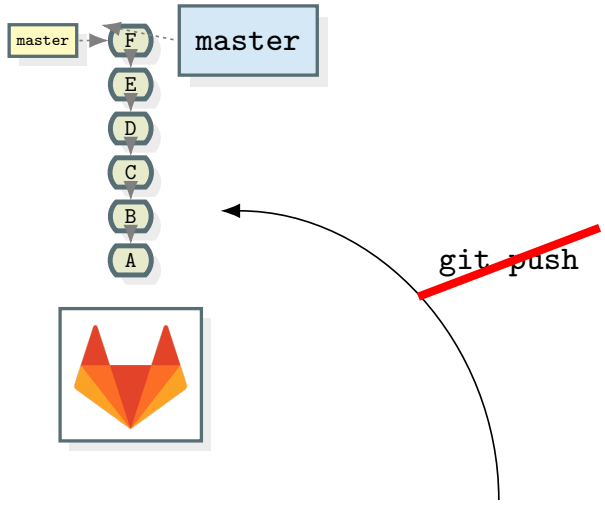

HEAD
master
F
E
D
C
B
A
origin/master
G
E
D
C
B
A
master
HEAD
ALICE
BOB

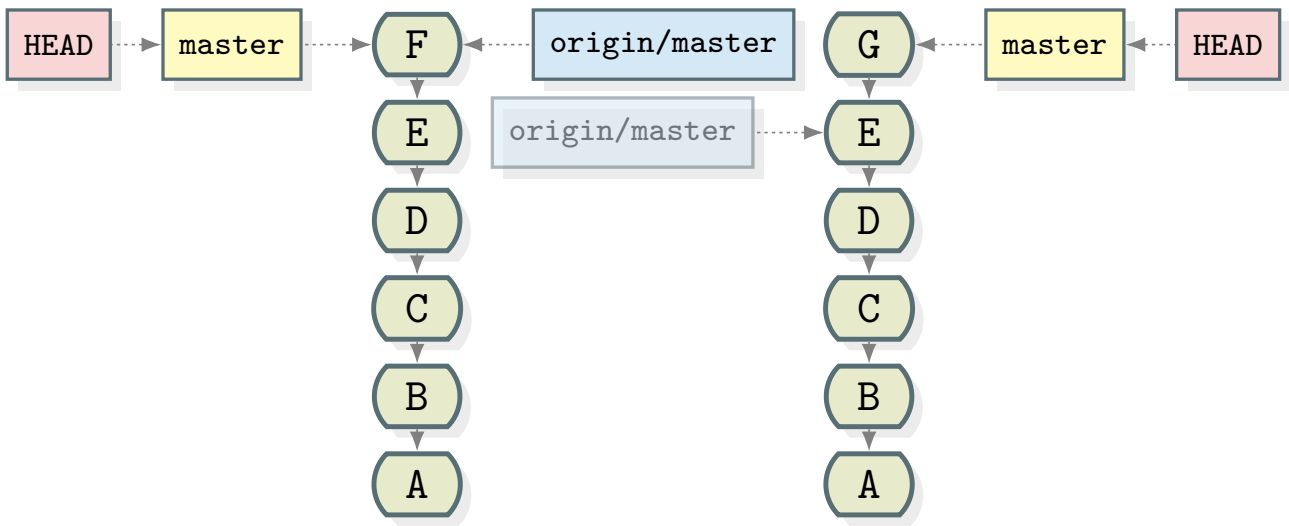

master
F
E
D
C
B
A
master
git push


HEAD
master
F
origin/master
G
master
HEAD
E
origin/master
E
D
D
C
C
B
B
A
A

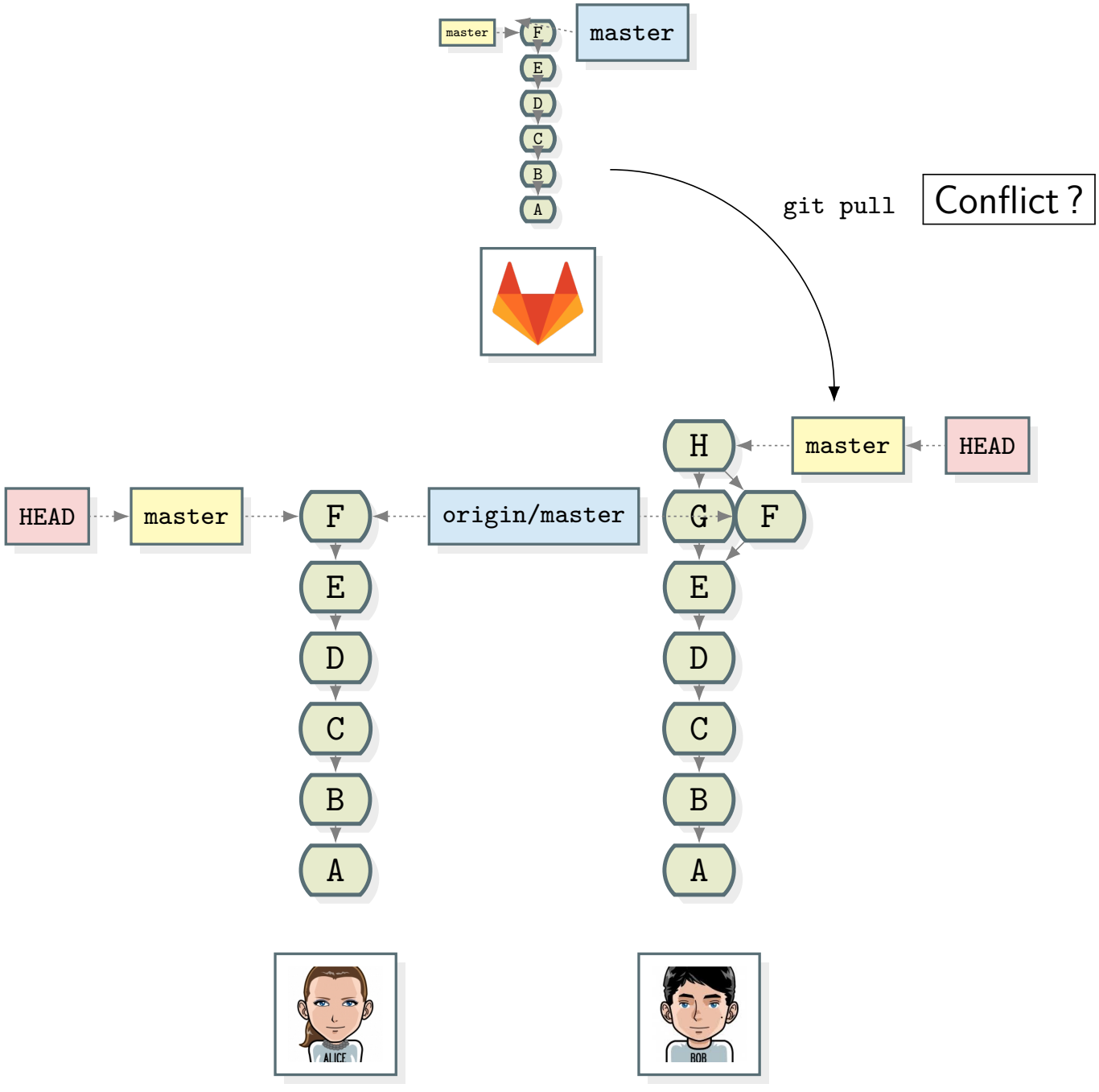

ALICE

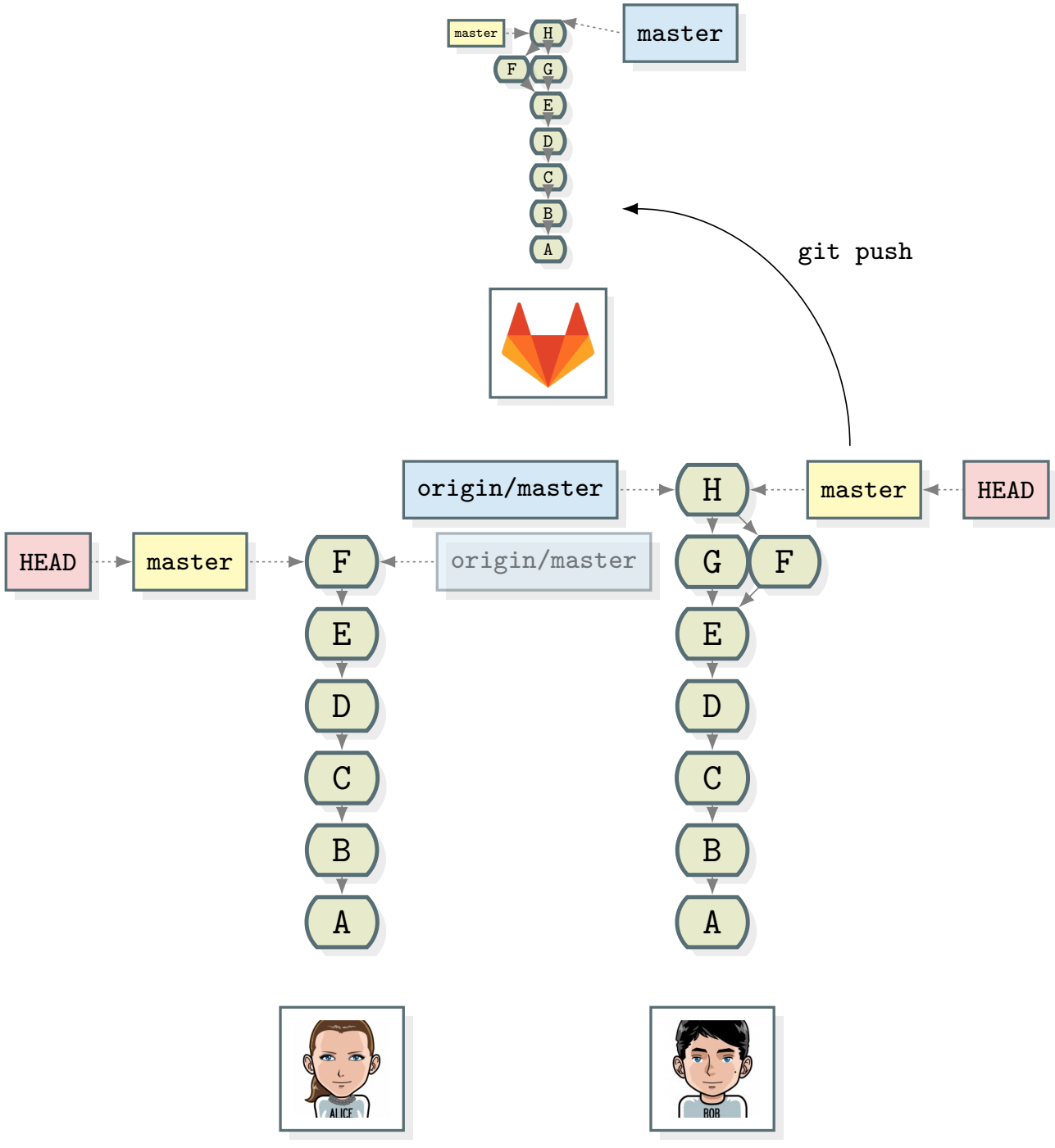

BOB

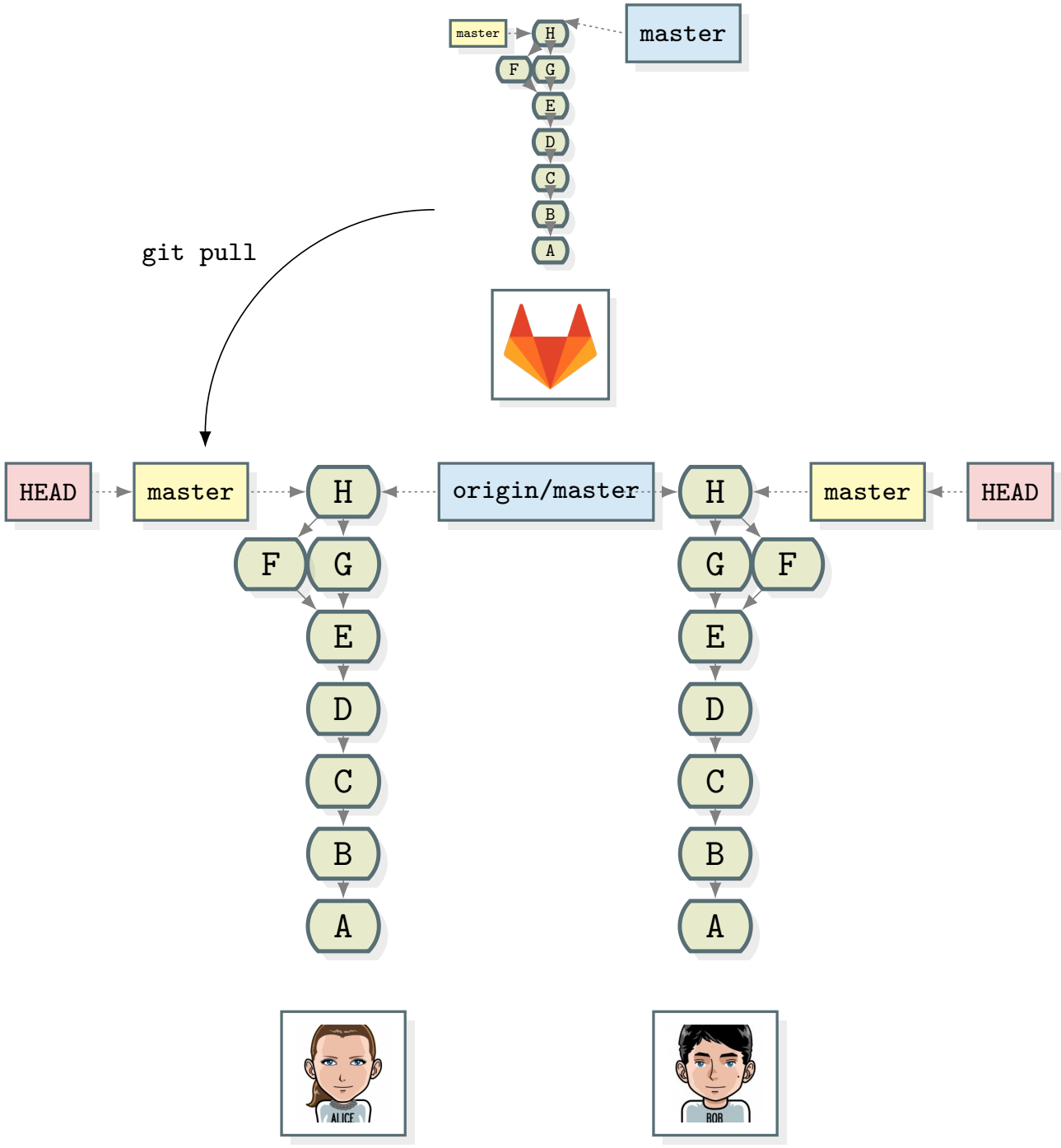

master
F
E
D
C
B
A
master
git push


HEAD
master
F
origin/master
G
master
HEAD
E
origin/master
E
D
D
C
C
B
B
A
A


ALICE
BOB


master
F
E
D
C
B
A
master
git pull
Conflict ?
H
master
HEAD
HEAD
master
F
origin/master
G
F
E
D
C
B
A
E
D
C
B
A
ALICE
BOB


master
H
F
G
E
D
C
B
A
master
git push
origin/master
H
master
HEAD
HEAD
master
F
origin/master
G
F
E
E
D
D
C
C
B
B
A
A
ALICE
BOB


master
H
F
G
E
D
C
B
A
master
git pull
HEAD
master
H
F
G
E
D
C
B
A
origin/master
H
G
F
E
D
C
B
A
master
HEAD
ALICE
BOB

À retenir

- Collaborer = gérer un historique distribué
- Le serveur ne sert à rien : Alice et Bob pourraient échanger directement leurs historiques
- Chacun a tout l'historique $\rightsquigarrow$ très résilient

Le conflit

Au moment du pull, il se peut que le merge ne soit pas possible et qu'un conflit apparaisse. Pas de panique !

Démonstration !

Le conflit

Au moment du pull, il se peut que le merge ne soit pas possible et qu'un conflit apparaisse. Pas de panique !

Démonstration !

- Git travaille à la granularité du fichier et de la ligne
  - $\Rightarrow$ Modularisez autant que possible

Le conflit

Au moment du pull, il se peut que le merge ne soit pas possible et qu'un conflit apparaisse. Pas de panique !

Démonstration !

- Git travaille à la granularité du fichier et de la ligne
  - ⇒ Modularisez autant que possible
- Il y aura conflit dans un fichier si la même ligne a été modifiée de chaque côté (même un espace)
  - ⇒ Ne réindentez pas votre texte pour rien !
  - ⇒ Faites des commits différents pour les modifications de fonds, de forme (git **status** ; git **diff** ; git **add**)
  - ⇒ Faites de **petits** commits **logiques** (~~`git commit -a`~~)

Le conflit

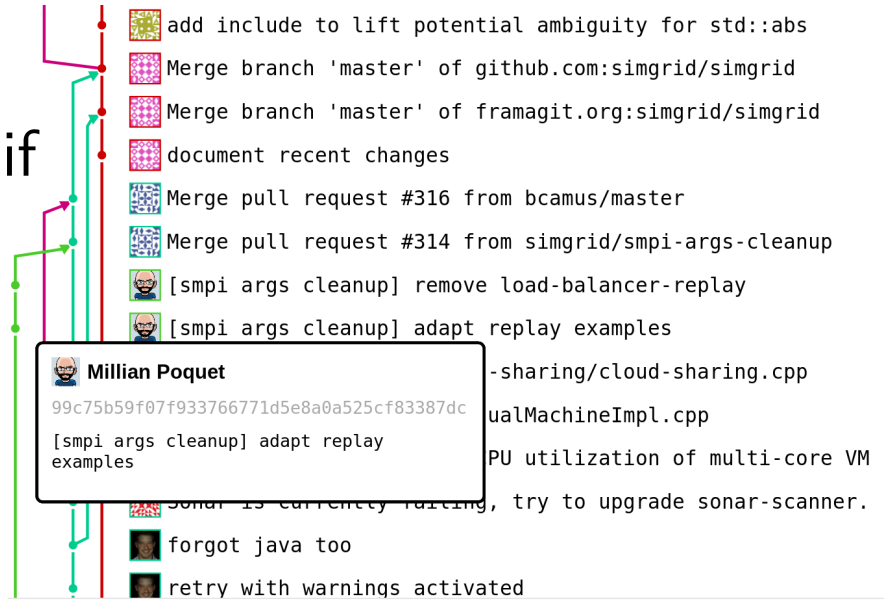
Au moment du pull, il se peut que le merge ne soit pas possible et qu'un conflit apparaisse. Pas de panique !

Démonstration !

- Git travaille à la granularité du fichier et de la ligne
  - ⇒ Modularisez autant que possible
- Il y aura conflit dans un fichier si la même ligne a été modifiée de chaque côté (même un espace)
  - ⇒ Ne réindentez pas votre texte pour rien !
  - ⇒ Faites des commits différents pour les modifications de fonds, de forme (git **status** ; git **diff** ; git **add**)
  - ⇒ Faites de **petits** commits **logiques** (~~`git commit -a`~~)
- Les fichier binaires n'ont pas de lignes... le diff et la fusion sont souvent impossibles
  - ⇒ Préférer le format texte

En conclusion, git :

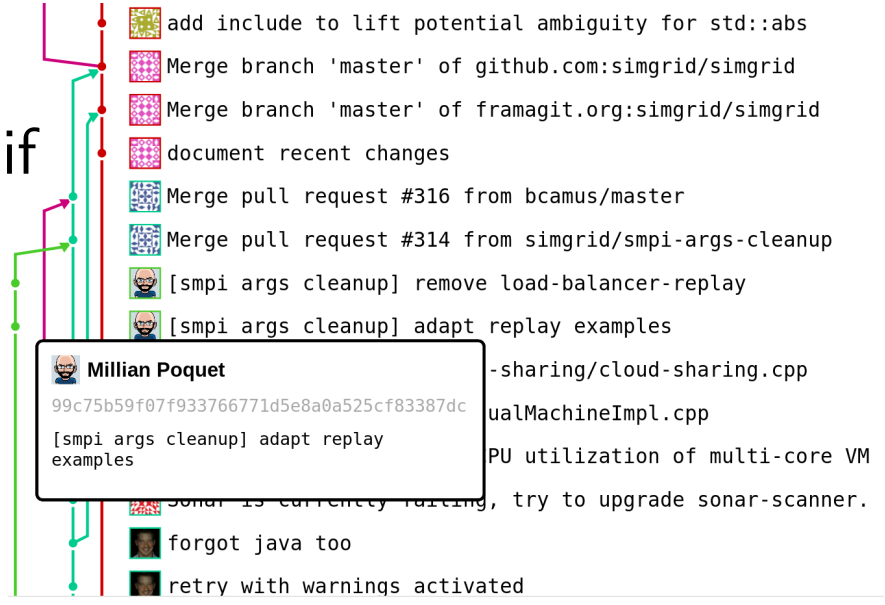
- Ça versionne ! (git **hist**)
- Ça facilite le travail collaboratif (git **merge**) !
- Ça permet de stocker ses modifications localement (git **commit**) !



- Ça permet de partager ses modifications quand on le souhaite (git **pull/push**) !

En conclusion, git :

- Ça versionne ! (git **hist**)
- Ça facilite le travail collaboratif (git **merge**) !
- Ça permet de stocker ses modifications localement (git **commit**) !



- Ça permet de partager ses modifications quand on le souhaite (git **pull/push**) !

Mais comment mettre ça en place ?

- Comment avoir un serveur de backup ?
- Comment gérer qui peut lire/écrire ?

Démystifions Git, Github, Gitlab

Notion d'historique

Travailler à plusieurs

L'écosystème Git

Les plates-formes de développement collaboratif (1/2)

GitHub, GitLab, et les autres !

- Hébergement gratuit de projets publics
- Interfaces web (navigation, preview, édition en ligne)
- Gestion des permissions (lecture/écriture, public/privé) et des utilisateurs

Les plates-formes de développement collaboratif (1/2)

GitHub, GitLab, et les autres !

- Hébergement gratuit de projets publics
- Interfaces web (navigation, preview, édition en ligne)
- Gestion des permissions (lecture/écriture, public/privé) et des utilisateurs

Un réseau social des développeurs

- Statistiques, émulation, gestion de communauté
- *Issues*, correction de bugs, revue de code

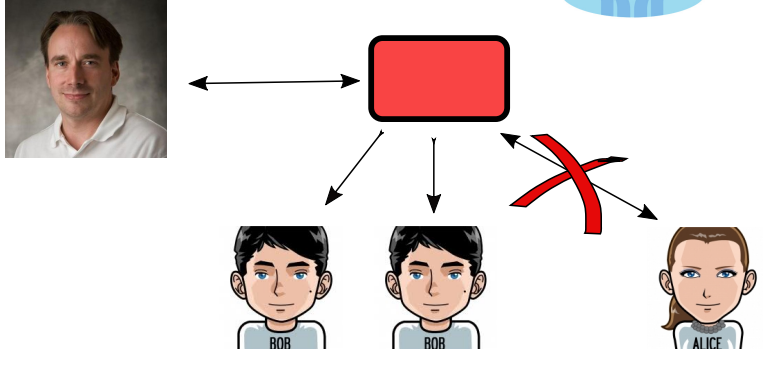
Les plates-formes de développement collaboratif (1/2)

GitHub, GitLab, et les autres !

- Hébergement gratuit de projets publics
- Interfaces web (navigation, preview, édition en ligne)
- Gestion des permissions (lecture/écriture, public/privé) et des utilisateurs

Un réseau social des développeurs

- Statistiques, émulation, gestion de communauté
- *Issues*, correction de bugs, revue de code
- Le *fork* et la *pull request*



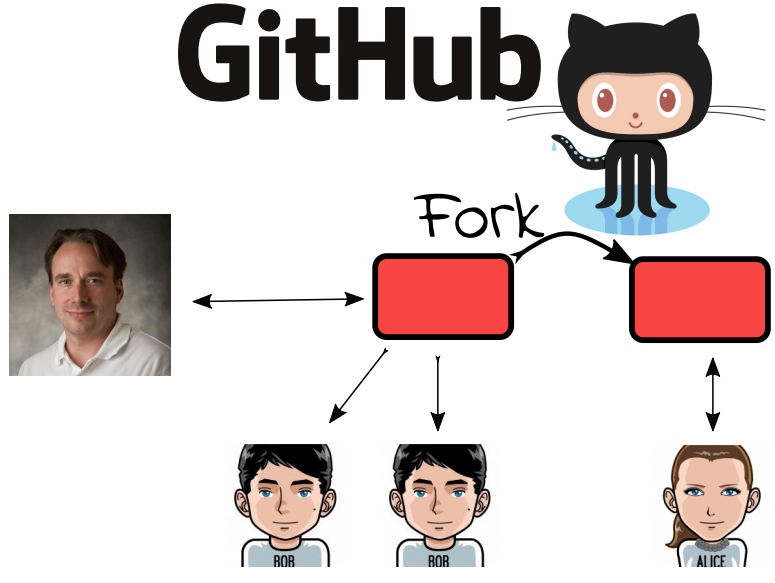
Les plates-formes de développement collaboratif (1/2)

GitHub, GitLab, et les autres !

- Hébergement gratuit de projets publics
- Interfaces web (navigation, preview, édition en ligne)
- Gestion des permissions (lecture/écriture, public/privé) et des utilisateurs

Un réseau social des développeurs

- Statistiques, émulation, gestion de communauté
- *Issues*, correction de bugs, revue de code
- Le *fork* et la *pull request*



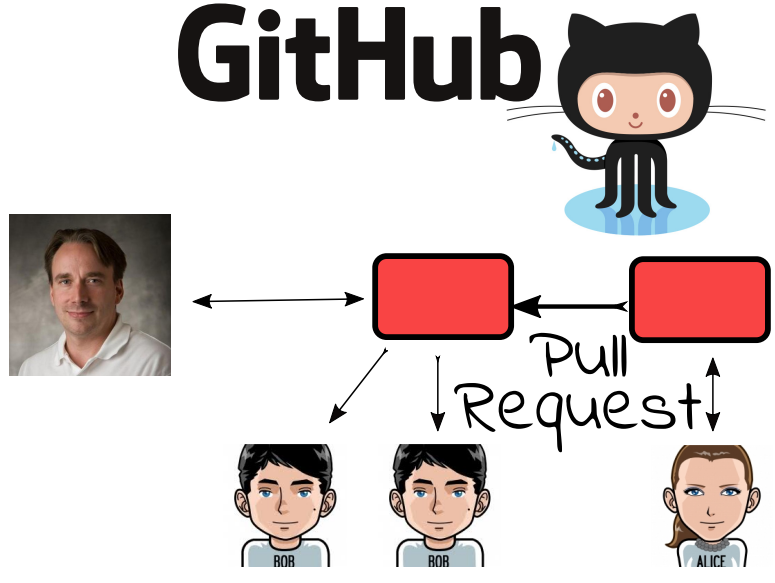
Les plates-formes de développement collaboratif (1/2)

GitHub, GitLab, et les autres !

- Hébergement gratuit de projets publics
- Interfaces web (navigation, preview, édition en ligne)
- Gestion des permissions (lecture/écriture, public/privé) et des utilisateurs

Un réseau social des développeurs

- Statistiques, émulation, gestion de communauté
- *Issues*, correction de bugs, revue de code
- Le *fork* et la *pull request*



Les plates-formes de développement collaboratif (1/2)

GitHub, GitLab, et les autres !

- Hébergement gratuit de projets publics
- Interfaces web (navigation, preview, édition en ligne)
- Gestion des permissions (lecture/écriture, public/privé) et des utilisateurs

Un réseau social des développeurs

- Statistiques, émulation, gestion de communauté
- *Issues*, correction de bugs, revue de code
- Le *fork* et la *pull request*

Des tas de fonctionnalités incroyables [→MOOC RR avancé]

- Intégration continue
- Déversement vers des archives (Zenodo, SWH)

Les plates-formes de développement collaboratif (2/2)

GitHub, GitLab, et les autres !

- Hébergement **gratuit** de projets publics ?!?

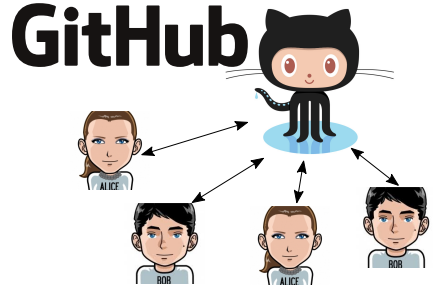
Les plates-formes de développement collaboratif (2/2)

GitHub, GitLab, et les autres !

- Hébergement **gratuit** de projets publics ?!?

Github Leaders du marché mais logiciel propriétaire

- Projets privés :
  - gratuit pour étudiants/académiques
  - payant pour les autres



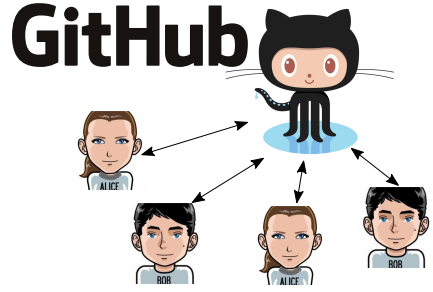
Les plates-formes de développement collaboratif (2/2)

GitHub, GitLab, et les autres !

- Hébergement **gratuit** de projets publics ?!?

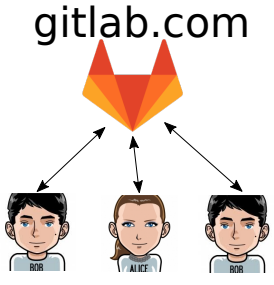
Github Leaders du marché mais logiciel propriétaire

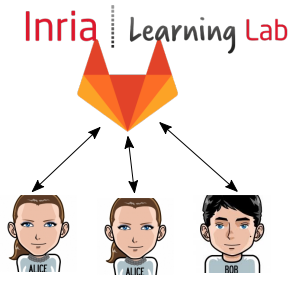
- Projets privés :
  - gratuit pour étudiants/académiques
  - payant pour les autres

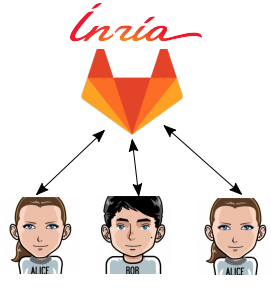


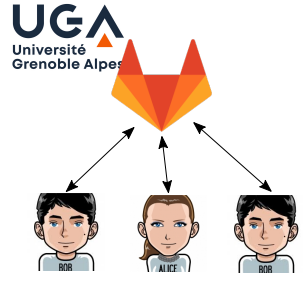
GitLab Logiciel libre, plusieurs instances

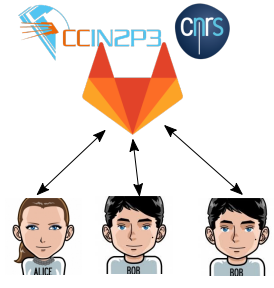
- Instance `gitlab.com` similaire à GitHub
- Compagnies, institutions : support avancé











Les plates-formes de développement collaboratif (2/2)

GitHub, GitLab, et les autres !

- Hébergement **gratuit** de projets publics ?!?

Github Leaders du marché mais logiciel propriétaire

- Projets privés :
  - gratuit pour étudiants/académiques
  - payant pour les autres

GitLab Logiciel libre, plusieurs instances

- Instance `gitlab.com` similaire à GitHub
- Compagnies, institutions : support avancé

Lequel choisir ? (+ FramaGit, BitBucket, SourceForge, ...)

- Visibilité/pérénité
- Projets privés ou publiques
- Confidentialité des données

Interfaces graphiques

- JupyterLab-git
- Rstudio
- Emacs Magit

- GitHub Desktop
- GitKraken
- SmartGit