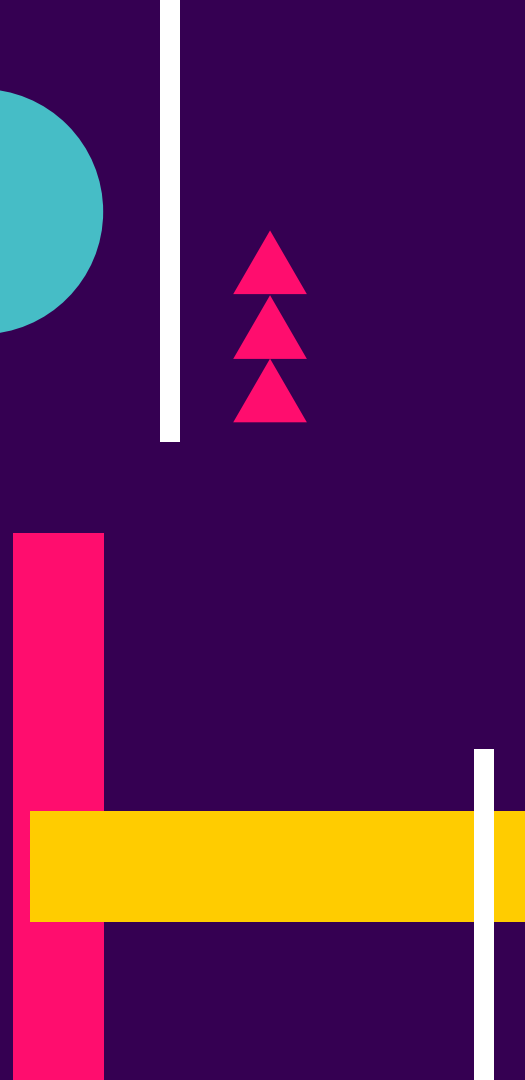
DC4

La Merkanterie

Git & GitHub :
Comment versioner son projet

1.

Objectifs



Maîtriser un logiciel de versionning

- Une séance en physique
- Savoir cloner un projet
- Savoir créer un repository
- Savoir ajouter des fichier à l'indexation
- Rédiger un commit
- Pusher du code sur un repository distant
- Ajouter un collaborateur a un repository
- Comprendre ce qu’est le git flow

2.

Introduction

Git c'est quoi ?

Git

- Permet de sauvegarder du code
- Travailler en collaboration
- Hebergement local du projet
- Gérer plusieures version en même temps
- Traçabilité de l’évolution du projet

GitHub - GitLab - BitBucket etc..

- Interface graphique
- Hebergement distant

Pourquoi utiliser Git ?

GitHub - GitLab - BitBucket etc..

- vous travaillez à plusieurs
- vous souhaitez collaborer à des projets open source
- vous souhaitez conserver un historique de votre projet
- vous voulez pouvoir retrouver par qui a été faite chaque modification
- vous voulez savoir pourquoi chaque modification a eu lieu
- Vous voulez gérer plusieurs version en parallèle

3.

Création d'un compte GitHub

Création et configuration de git

- Créer une compte sur gitHub
- Téléchargement de git
- Configuration:

```
git config --global user.name "John Doe"
git config --global user.email johndoe@example.com
```

```
git config --global color.diff auto
git config --global color.status auto
git config --global color.branch auto
```

4.

Commande de base

Les bases de git

- Toutes les commandes commencent par **git**

Cloner un projet

- Cloner un projet revient à télécharger le projet sur son ordinateur
- git clone http://url-du-repo-git

Sauvegarder une modification

- Ajouter des fichier à l’index du repository git
- git add “nom-du/des-fichiers modifiés”

Commit une modification

- Créer une sous version de son code
- git commit -m “message décrivant les modifications sauvegardées”

Envoyer son code sur le repo distant

- Envoyer les sauvegardes stockées localement sur un serveur distant
- git push

Synchroniser son repos local avec le repos distant

- Récupérer les mises à jours de code
- git pull

Afficher l'état de son repo local

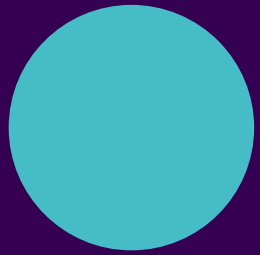
- Donne l'état courant de son repository local
- git status

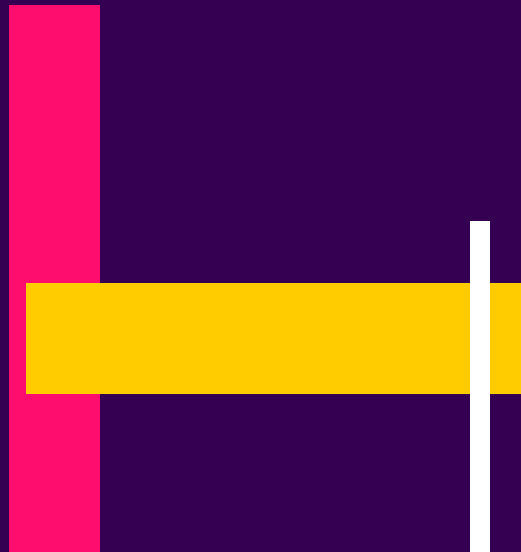
Afficher l'historique du repo

- Affiche l'historique générale des commits
- git log (touche q pour quitter)

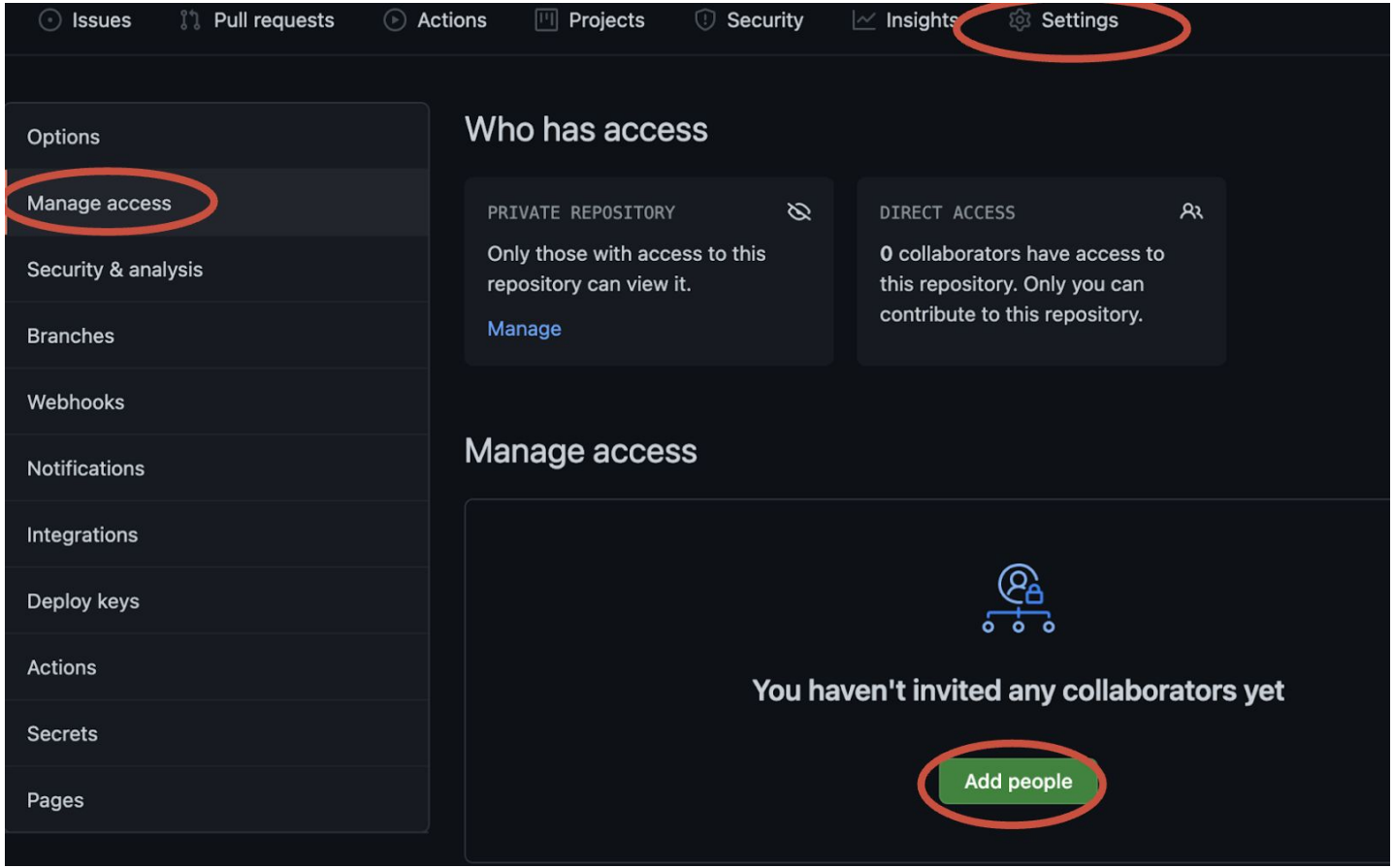
5.

Créer un repository



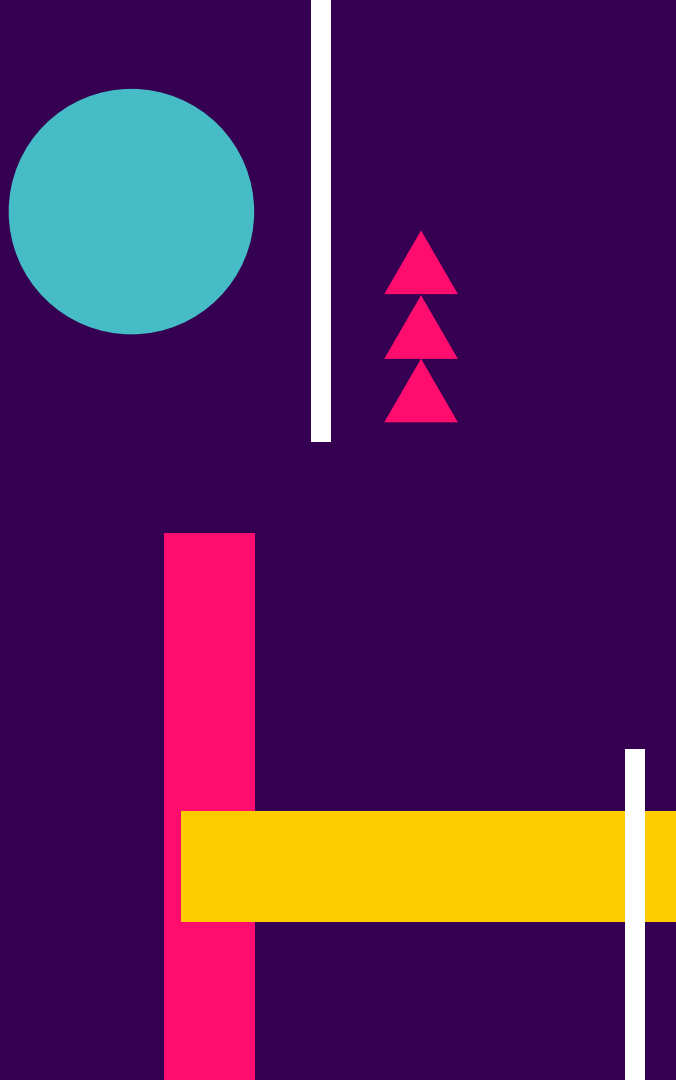


Ajouter un Collaborateur

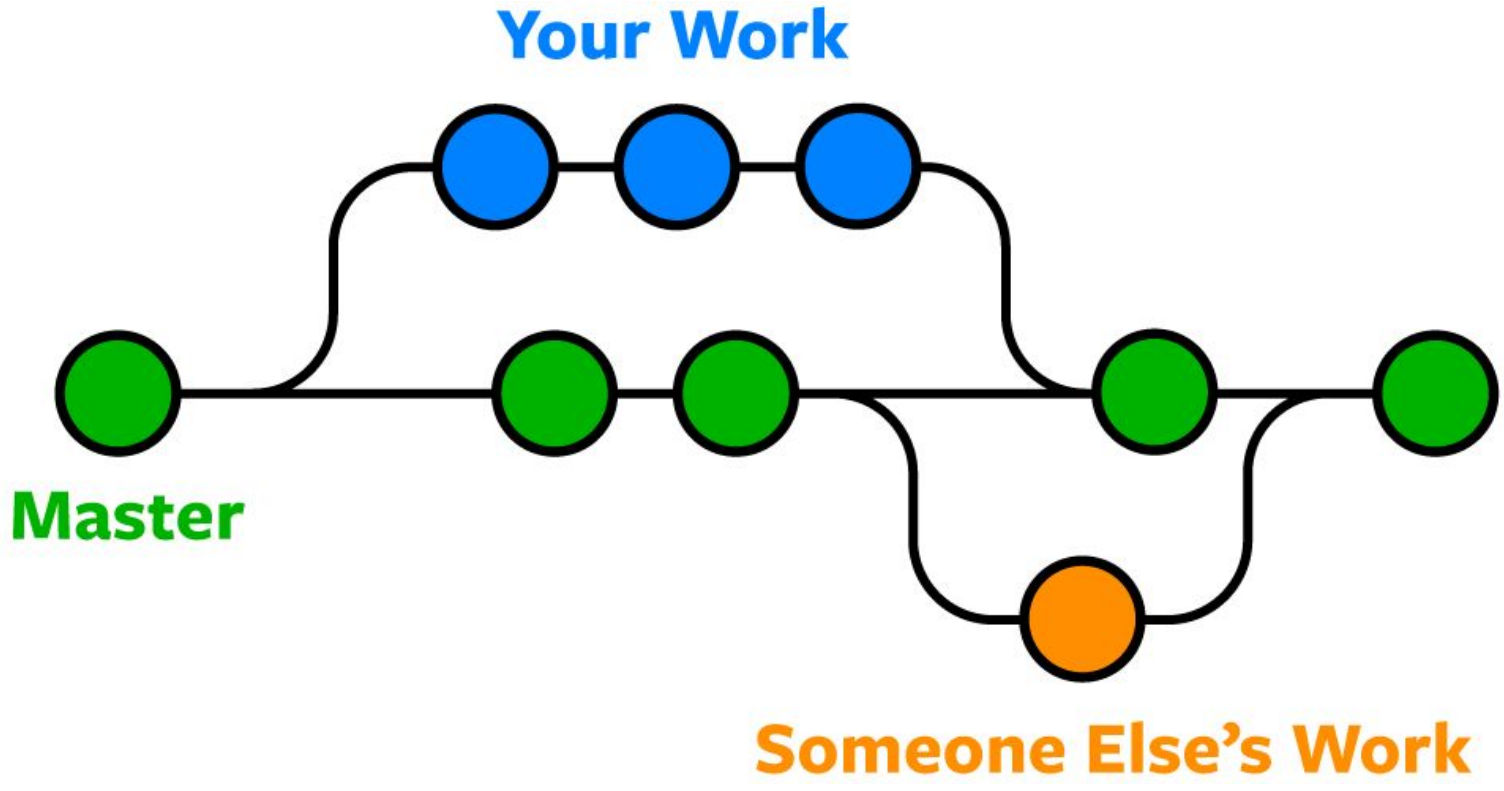

Issues
Pull requests
Actions
Projects
Security
Insights
Settings
Options
Manage access
Security & analysis
Branches
Webhooks
Notifications
Integrations
Deploy keys
Actions
Secrets
Pages
Who has access
PRIVATE REPOSITORY
Only those with access to this repository can view it.
Manage
DIRECT ACCESS
0 collaborators have access to this repository. Only you can contribute to this repository.
Manage access
You haven't invited any collaborators yet
Add people

6.

Les branches



Les Branches



Lister les branches

- Liste toutes les branches ouvertes
- git branch

Créer une branche

- Créer une branche en lui donnant un nom
- git branch “nom-de-la-branche”

Changer de branche

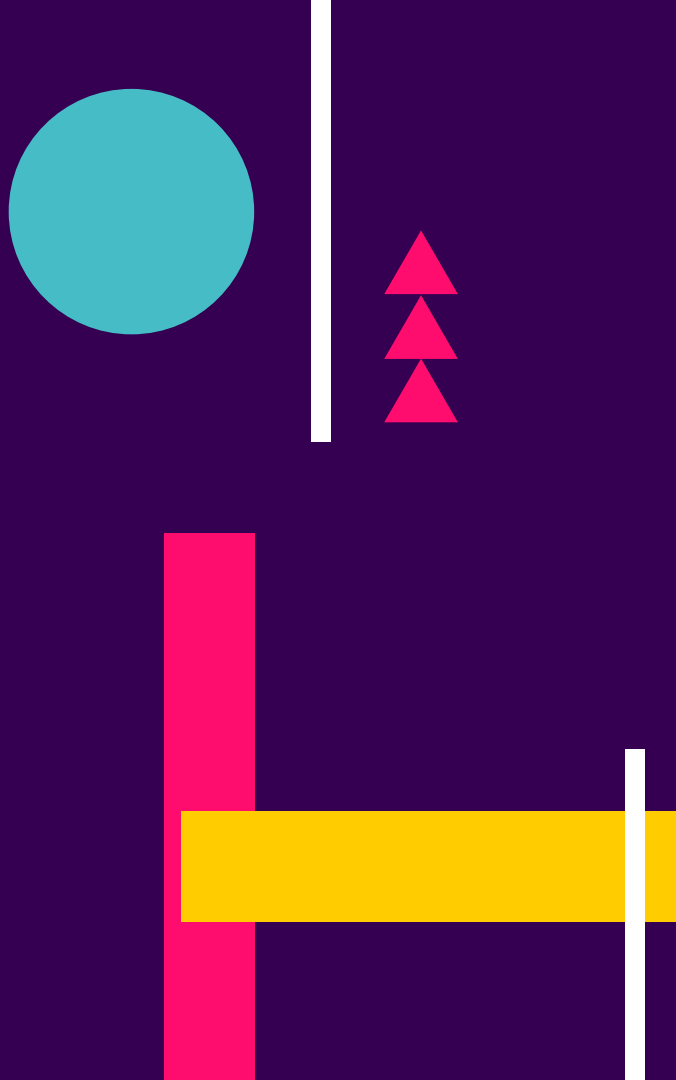
- Passer d’une branche à une autre
- git checkout “nom-de-la-branche”

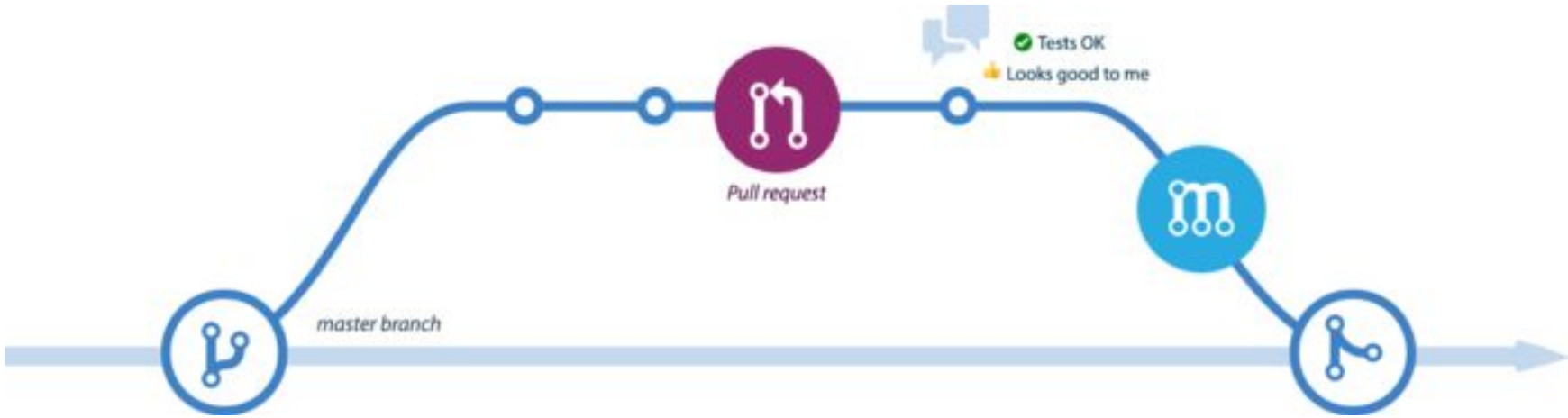
Suppression d'une branche

- git branch -d “nom-de-la-branche”

7.

Les pull/merge request



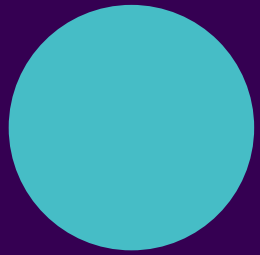

Tests OK
Looks good to me
Pull request
master branch

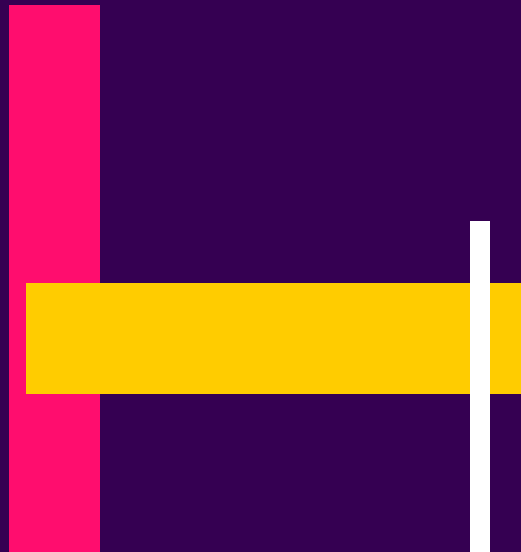
Pourquoi faire une pull request ?

- Code review
- Approbation de l'équipes du code fourni
- Fusion du code dans la branche principale (master/main)
- Fermeture de la branche

8.

Ignorer des fichiers





.gitignore

- Ignore la détection de fichier/dossier dans l'indexation
  - Fichiers contenant des informations sensibles
  - Fichiers volumineux n'apportant rien à la partie technique d'un projet

```
.DS_Store
node_modules
/dist


# local env files
.env.local
.env.*.local
```

Cas pratique

Ressources

Git basics

Cours

Git advanced

Cours

Utiliser un terminal

Installer git