
Introduction à Git et GitHub

les versions control
comme git permettent le
“control Z sur les
stéroïdes” et la
collaboration accrue avec
les branches

Soutien aux équipes
distribuées : Git et
GitHub permettent aux
équipes de contribuer
depuis divers lieux sans
conflits majeurs.

Favorise
l'expérimentation : les
branches permettent de
développer des
fonctionnalités sans
perturber le code en
production.

Pourquoi Git?



Architecture basée sur les snapshots : chaque commit capture l'état complet du projet.

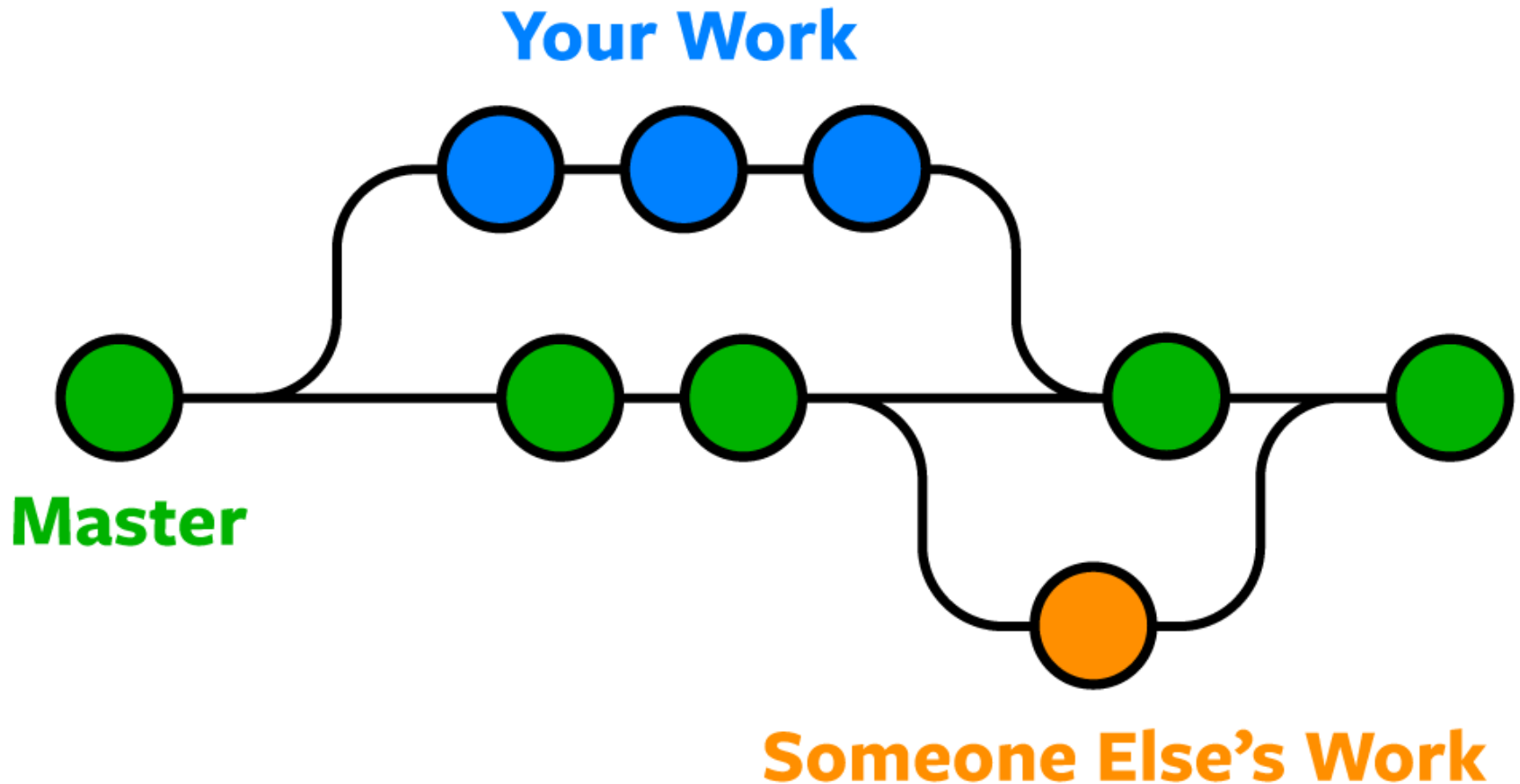


Efficacité des branches : branches légères et efficaces pour un développement parallèle.

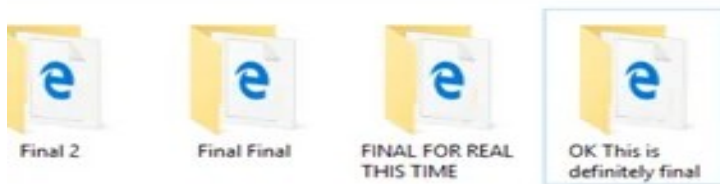


Adoption généralisée : plus de 87% du marché mondial utilise Git.

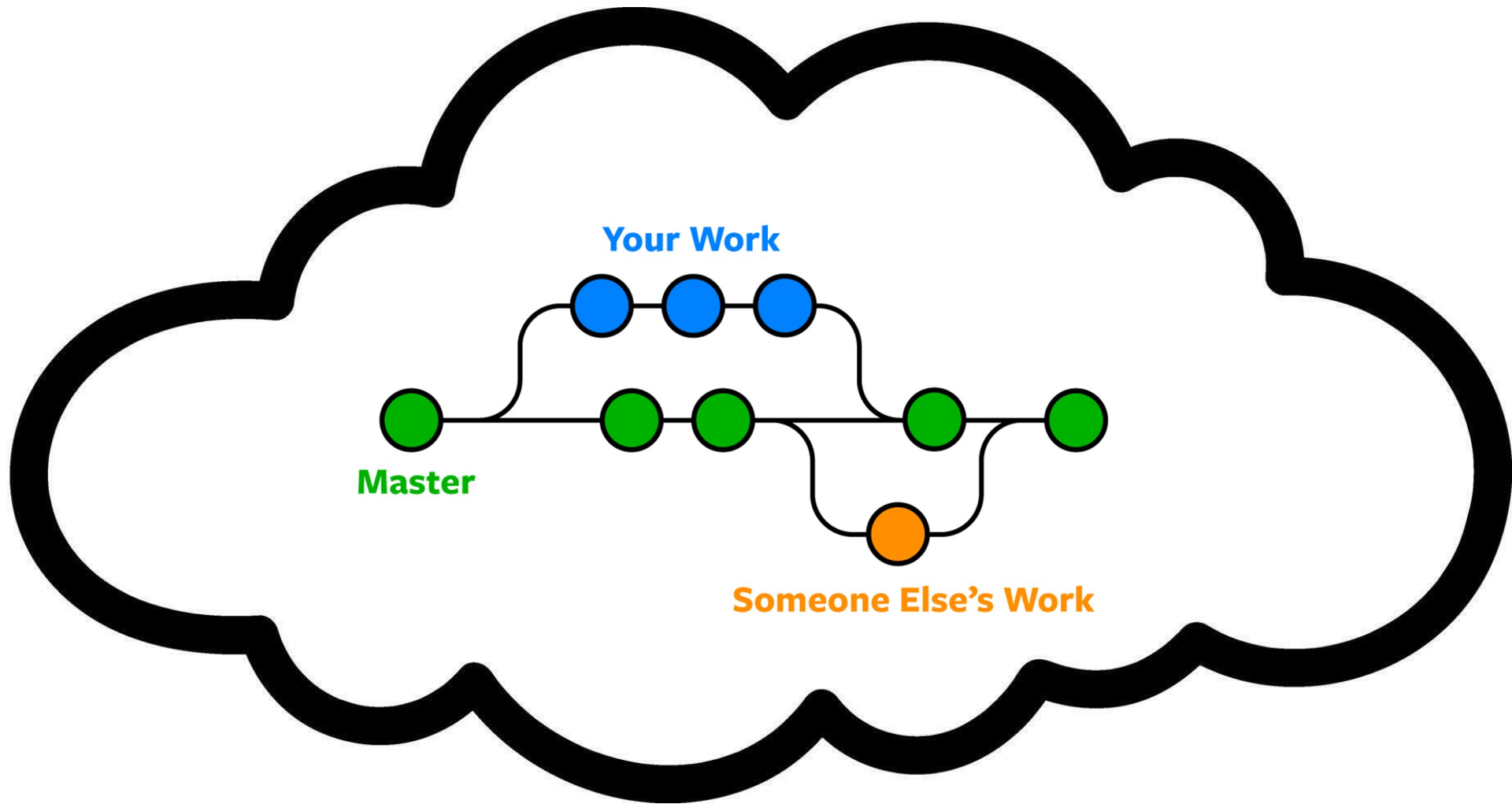
Comment le système de branche fonctionne



Git en bref



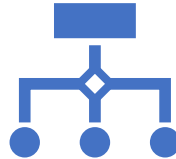
Git hub en bref



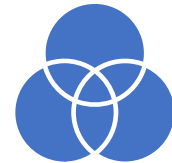
Bonnes pratiques Git & GitHub



écrire des messages clairs à chaque commits pour pouvoir te retrouver dans tes sauvegardes



Utiliser des branches : permet un développement modulaire.



Ne jamais push directement dans main, pusher dans sa propre branche et utiliser les pull requests .

Conclusion & Questions



Git : Contrôle de version basé sur des snapshots



GitHub : Outil pour sauvegarder git dans le cloud



Bonne pratique : JAMAIS PUSH DIRECT DANS MAIN.



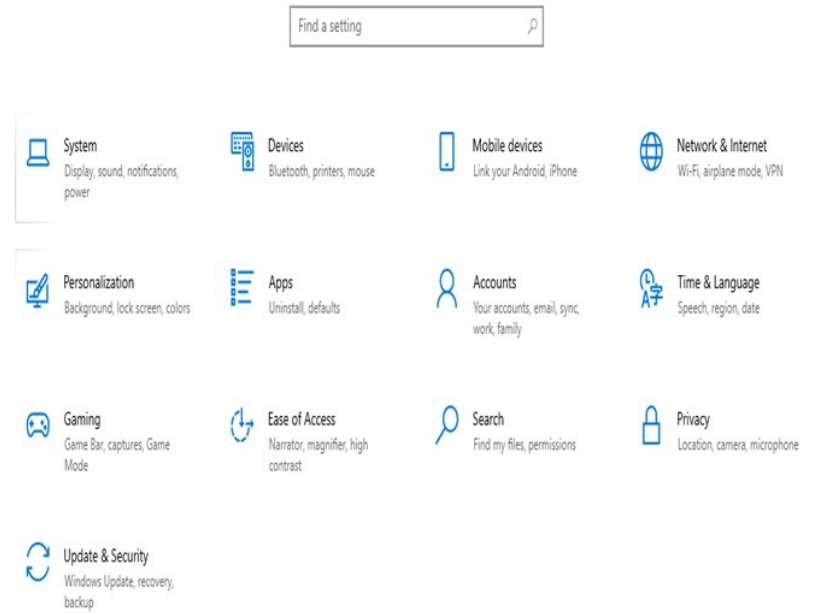
Code along

Introduction aux dotfiles



Avec Linux, tout est fichier

- Les réglages sont souvent des fichiers texte placé à des endroits spécifiques
- Tu peux tout configurer, pas besoin d'interface graphique
- Cela inclut le shell, les éditeurs de texte, les outils système...



```
#This file shows how to configure the Kube Agent Management Plugin
kube:
podTemplatesConfiguration:
  templates:
    - containers:
      - args: "cat"
        command: "/bin/sh -c"
        image: "nginx"
        livenessProbe:
          failureThreshold: 0
          initialDelaySeconds: 0
          periodSeconds: 0
          successThreshold: 0
          timeoutSeconds: 0
        name: "container0"
        ttyEnabled: true
        workingDir: "/home/jenkins/agent"
      hostNetwork: false
      name: "template0"
      yamlMergeStrategy: "override"
    - hostNetwork: false
      name: "template1"
      yamlMergeStrategy: "override"
```

.zshrc

- Définit :
 - les alias
 - les scripts
 - les plugins (oh-my-zsh)
- Se trouve dans ton /home/user path

EN BREF

1) configurer une seule fois dans le bon fichier ex: zshrc

2) mettre ces configurations en contrôle de version avec git.

Avantages:

- Réutilisation dans d'autre système
- Formule only do it once



Code along