Git et Github: Collaborer

1WEBD – Javascript Web Development

Sommaire

1. Collaborer avec GIT
2. Collaborer avec Github

1. Collaborer avec GIT

1. Collaborer avec GIT

Rappel

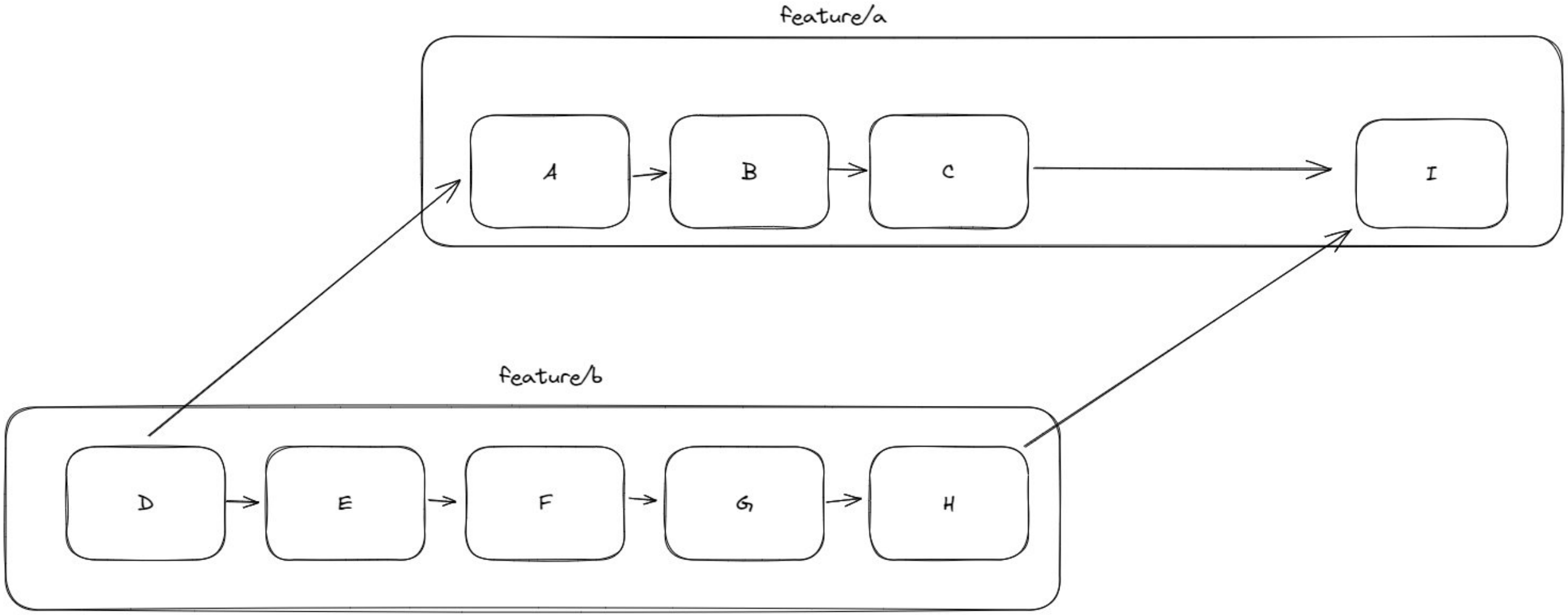
- GIT est un VCS distribué
- Permet à plusieurs personnes de travailler en parallèle (branches)

1. Collaborer avec GIT

Collaborer

- Pour collaborer, on travaille dans des branches différentes
- Ensuite, on rapatrie les commits sur une branche commune

1. Collaborer avec GIT



1. Collaborer avec GIT

Collaborer – Merge Commit

- Quand deux branches divergent, on peut les rapatrier avec un merge
- Git trouve le commit que les branches ont en commun
- Ensuite il applique les changements portés par les commits des branches
- Si jamais il y a un conflit (deux commits modifient le même fichier), git peut demander à l'utilisateur de résoudre ce conflit (choisir quoi garder dans la partie du fichier en question)
- Un nouveau commit est créé: un merge commit

1. Collaborer avec GIT

Collaborer – Bonnes Pratiques

- GIT ne force pas de bonnes pratiques
- Les utilisateurs doivent se mettre d’accord

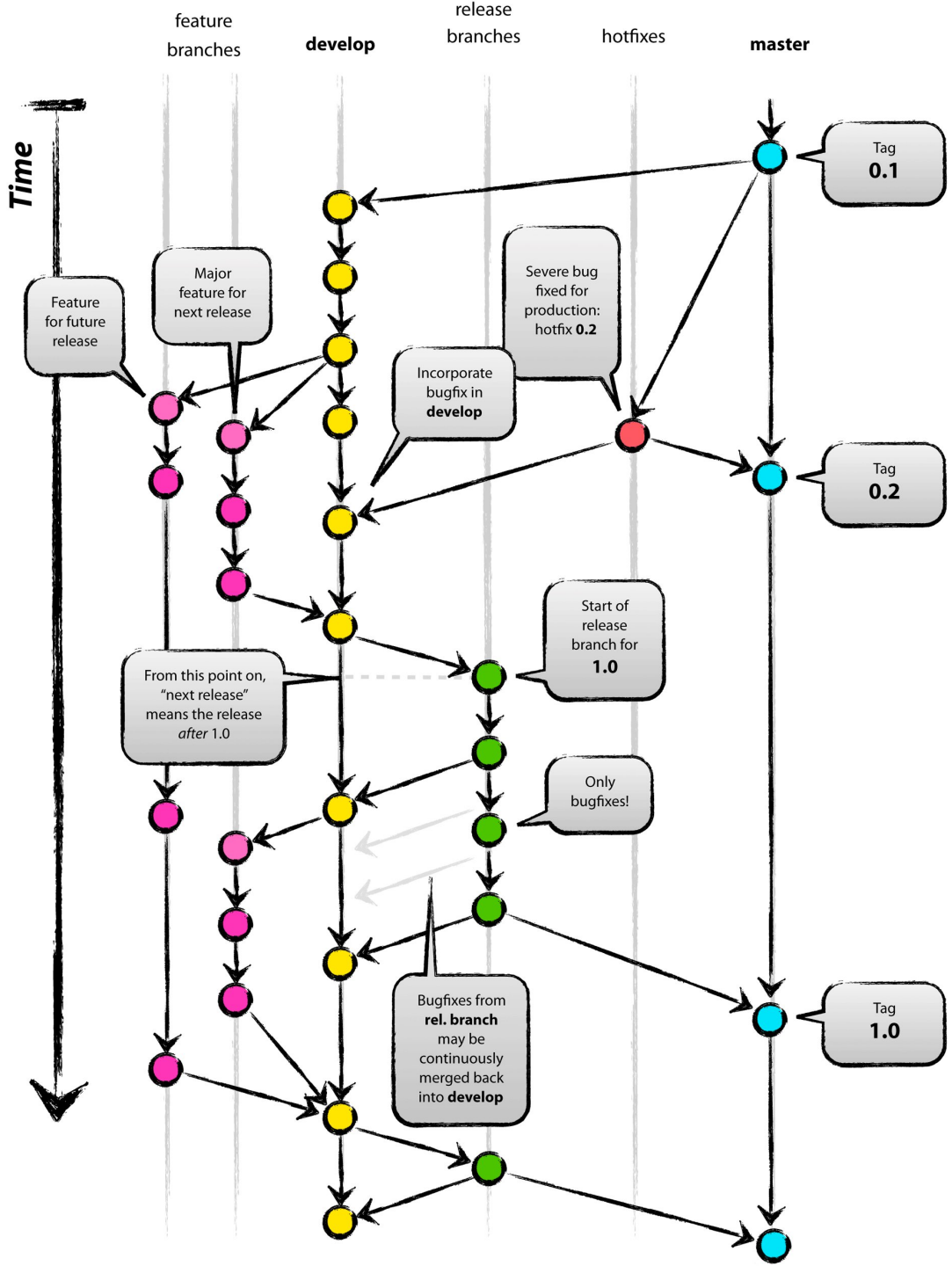
1. Collaborer avec GIT

Collaborer – Bonnes Pratiques – Conventional Commits

- Norme de commits
- <type>[scope]: <descriptions>
- Par exemple: feat(api): send email
- Plus d’infos: conventioncommits.org

1. Collaborer avec GIT

Collaborer – Bonnes Pratiques – Git Flow



1. Collaborer avec GIT

Collaborer – Bonnes Pratiques – Git Flow

- Branche main|master = branche de production (dernier commit = version en prod)
- Branche develop = branche d'intégration (ou sont merge les autres branches)
- Une branche « feature » part de develop et sera merge dans develop
- Une branche « hotfix » sert pour corriger un bug important et peut être merge dans la branche de production
- Pour publier une nouvelle version, une branche « release » peut être crée depuis develop et merge dans la branche principale
- Plus d'infos: https://www.atlassian.com

1. Première partie

2. Collaborer avec Github

2. Collaborer avec Github

Rappel

- Serveur GIT
- Fournit beaucoup d’outils supplémentaires pour travailler sur des projets
- Utilisé par beaucoup de projets open source (comme le kernel Linux)

2. Collaborer avec Github

Alternatives

- Gitlab
- Gitea
- GIT (oui, on peut simplement l’installer sur un serveur)

2. Collaborer avec Github

Github Student Pack

- Education.github.com
- Donne accès aux fonctionnalités premium de Github + plateformes partenaires
- Il faut avoir son mail @supinfo.com comme adresse principale sur Github

2. Collaborer avec Github

Github Student Pack - Avantages

- Crédits Azure, Digital Ocean
- Nom de domaine gratuit
- Emails gratuits
- Compte datadog
- Etc etc

2. Collaborer avec Github

Collaborer – Markdown

- Le plus souvent, les documents sur Github sont écrits en Markdown (.md)
- Le texte qui apparait en naviguant dans un repo est contenu dans un README.md
- Plus rapide (pour certains) que d'ouvrir un Word, Pages etc…
- Peut facilement être convertie en d'autres formats (HTML notamment)

2. Collaborer avec Github

Collaborer – Markdown

```
# ceci est un titre

## un titre plus petit

- une liste
- encore un élément de la liste

[un lien](https://example.com)
un texte en *italique* ou en **gras**
```

```
<h1>ceci est un titre</h1>

<h2>un titre plus petit</h2>

<ul>
  <li>une liste</li>
  <li>encore un élément de la liste</li>
</ul>

<a href="https://example.com">un lien</a>
un texte en <i>italique</i> ou en <b>gras</b>
```

2. Collaborer avec Github

Collaborer – Permissions

- Les collaborateurs peuvent directement éditer le code du repo
- Si on veut contribuer mais qu’on n'est pas collaborateur, on peut « fork » le repo

2. Collaborer avec Github

Collaborer – Forks

- Un fork consiste à dupliquer un repo
- Possibilité de créer des pull requests vers le repo d’origine
- Utile pour collaborer sans faire partie de l’équipe interne d’un projet
- Très utilisé pour des projets FOSS (Free Open Source Software)

2. Collaborer avec Github

Collaborer – Pull Requests

- Equivalent d’un merge commit
- Permet d’inspecter les changements des fichiers modifiés par cette branche
- Permet une discussion + revue de code
- Peut être bloqué tant que certaines conditions ne sont pas remplies
  - Plusieurs revues effectuées
  - Validation par des tests automatisés

2. Collaborer avec Github

Collaborer – Pull Requests

- Utile pour trier les « features » (fonctionnalitées) et les bugs
- Peuvent être liée à une PR (pull request)
- Peut être liée à une tâche

2. Collaborer avec Github

Collaborer – Suivi de Projet

- Équivalent Trello, Jirah etc
- Permet de faire de la gestion de t^aches
- Permet d’assigner des t^aches à des membres
- Les t^aches peuvent être liées à des issues

2. Collaborer avec Github

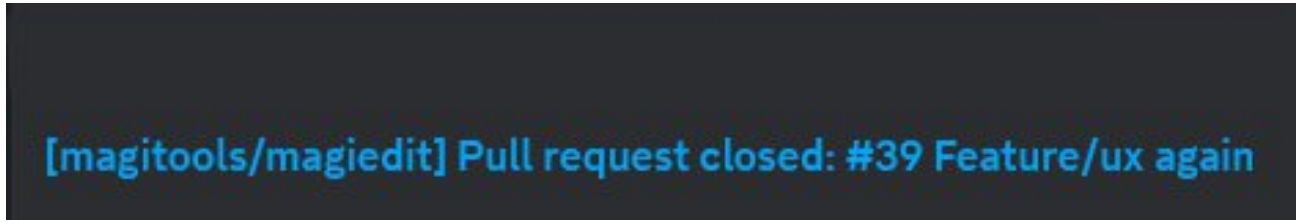
Collaborer – CI (Continuous Integration)

- Divisé en pipelines
  - Chaque pipeline est une suite d'instructions
- Lance des scripts selon certaines conditions (un push sur une branche, une PR…)
- Peut s'exécuter sous Windows, Linux et/ou MacOS
- Peut-être une façon de bloquer l'approbation d'une PR en cas d'échec

2. Collaborer avec Github

Collaborer – Webhooks

- Github peut envoyer une requête vers une URL quand une action se produit (commit, pr etc…)
- Utile pour des automatisations (IFTTT, Zapier etc…)



Example de webhook envoyé dans un canal Discord lors de la fermeture d’une PR

2. Deuxième partie