Méthodes de travail Git

Jean-Gabriel Maheux

École d'actuariat
Université Laval, Québec, Canada

Le vendredi 5 juillet 2024

Plan de la présentation

UNIVERSITÉ LAVAL

Introduction

Objectifs de la présentation

- Présenter les caractéristiques de base du logiciel *Git*.
- Présenter une courte procédure d'utilisation du logiciel *Git*.
- Présenter deux méthodes de travail pour utiliser *Git* de manière structurée.

UNIVERSITÉ LAVAL

Git : La Base

UNIVERSITÉ LAVAL

Git, c'est quoi ?

- Gestionnaire de versions bâti en 2005 [Git, 2024b] [Atlassian, 2024a].
- Auteur : Linus Torvalds [Atlassian, 2024a].
- Logiciel libre et gratuit [Git, 2024c].
- Avantage de *Git* : Création de « branches locales indépendantes » [Git, 2024b].

Git, c'est quoi ?

- Gestionnaire de versions bâti en 2005 [Git, 2024b] [Atlassian, 2024a].
- Auteur : Linus Torvalds [Atlassian, 2024a].
- Logiciel libre et gratuit [Git, 2024c].
- Avantage de *Git* : Création de « branches locales indépendantes » [Git, 2024b].

Git, c'est quoi ?

- Gestionnaire de versions bâti en 2005 [Git, 2024b] [Atlassian, 2024a].
- Auteur : Linus Torvalds [Atlassian, 2024a].
- Logiciel libre et gratuit [Git, 2024c].
- Avantage de *Git* : Création de « branches locales indépendantes » [Git, 2024b].

Git, c'est quoi ?

- Gestionnaire de versions bâti en 2005 [Git, 2024b] [Atlassian, 2024a].
- Auteur : Linus Torvalds [Atlassian, 2024a].
- Logiciel libre et gratuit [Git, 2024c].
- Avantage de *Git* : Création de « branches locales indépendantes » [Git, 2024b].

UNIVERSITÉ LAVAL

Comment travailler avec *Git* ?

- Installer le logiciel *Git*.
- Créer un dépôt distant *Git* sur une plateforme comme *GitHub* (travailler uniquement localement est aussi possible.).
- Interagir avec *Git* depuis une ligne de commande (*Invite de commande* sur *Windows* ou *Git Bash* par exemple).

Comment travailler avec *Git* ?

- Installer le logiciel *Git*.
- Créer un dépôt distant *Git* sur une plateforme comme *GitHub* (travailler uniquement localement est aussi possible.).
- Interagir avec *Git* depuis une ligne de commande (*Invite de commande* sur *Windows* ou *Git Bash* par exemple).

Comment travailler avec *Git* ?

- Installer le logiciel *Git*.
- Créer un dépôt distant *Git* sur une plateforme comme *GitHub* (travailler uniquement localement est aussi possible.).
- Interagir avec *Git* depuis une ligne de commande (*Invite de commande* sur *Windows* ou *Git Bash* par exemple).

UNIVERSITÉ LAVAL

Procédure générale d'utilisation de *Git*

- Création d'un dépôt sur un site hôte comme *GitHub*.
- Clonage du dépôt avec la commande `git clone <url du dépôt>`.
- Ajout de fichiers et/ou modification de fichiers.
- Ajout des modifications à l'étape d'« index » avec la commande `git add <chemin d'accès vers le fichier>` [Git, 2024d].
- Préparer les modifications à l'envoi au dépôt central de la présente version du projet avec la commande `git commit -m "Message de version"` [Atlassian, 2024a].
- Envoi de la nouvelle version du projet (avec les modifications) au dépôt central avec la commande `git push`.
- Récupération des modifications par un collaborateur au projet avec la commande `git pull` (facultatif).

Procédure générale d'utilisation de *Git*

- Création d'un dépôt sur un site hôte comme *GitHub*.
- Clonage du dépôt avec la commande `git clone <url du dépôt>`.
- Ajout de fichiers et/ou modification de fichiers.
- Ajout des modifications à l'étape d'« index » avec la commande `git add <chemin d'accès vers le fichier>` [Git, 2024d].
- Préparer les modifications à l'envoi au dépôt central de la présente version du projet avec la commande `git commit -m "Message de version"` [Atlassian, 2024a].
- Envoi de la nouvelle version du projet (avec les modifications) au dépôt central avec la commande `git push`.
- Récupération des modifications par un collaborateur au projet avec la commande `git pull` (facultatif).

Procédure générale d'utilisation de *Git*

- Création d'un dépôt sur un site hôte comme *GitHub*.
- Clonage du dépôt avec la commande `git clone <url du dépôt>`.
- Ajout de fichiers et/ou modification de fichiers.
- Ajout des modifications à l'étape d'« index » avec la commande `git add <chemin d'accès vers le fichier>` [Git, 2024d].
- Préparer les modifications à l'envoi au dépôt central de la présente version du projet avec la commande `git commit -m "Message de version"` [Atlassian, 2024a].
- Envoi de la nouvelle version du projet (avec les modifications) au dépôt central avec la commande `git push`.
- Récupération des modifications par un collaborateur au projet avec la commande `git pull` (facultatif).

Procédure générale d'utilisation de *Git*

- Création d'un dépôt sur un site hôte comme *GitHub*.
- Clonage du dépôt avec la commande `git clone <url du dépôt>`.
- Ajout de fichiers et/ou modification de fichiers.
- Ajout des modifications à l'étape d'« index » avec la commande `git add <chemin d'accès vers le fichier>` [Git, 2024d].
- Préparer les modifications à l'envoi au dépôt central de la présente version du projet avec la commande `git commit -m "Message de version"` [Atlassian, 2024a].
- Envoi de la nouvelle version du projet (avec les modifications) au dépôt central avec la commande `git push`.
- Récupération des modifications par un collaborateur au projet avec la commande `git pull` (facultatif).

Procédure générale d'utilisation de *Git*

- Création d'un dépôt sur un site hôte comme *GitHub*.
- Clonage du dépôt avec la commande `git clone <url du dépôt>`.
- Ajout de fichiers et/ou modification de fichiers.
- Ajout des modifications à l'étape d'« index » avec la commande `git add <chemin d'accès vers le fichier>` [Git, 2024d].
- Préparer les modifications à l'envoi au dépôt central de la présente version du projet avec la commande `git commit -m "Message de version"` [Atlassian, 2024a].
- Envoi de la nouvelle version du projet (avec les modifications) au dépôt central avec la commande `git push`.
- Récupération des modifications par un collaborateur au projet avec la commande `git pull` (facultatif).

Procédure générale d'utilisation de *Git*

- Création d'un dépôt sur un site hôte comme *GitHub*.
- Clonage du dépôt avec la commande `git clone <url du dépôt>`.
- Ajout de fichiers et/ou modification de fichiers.
- Ajout des modifications à l'étape d'« index » avec la commande `git add <chemin d'accès vers le fichier>` [Git, 2024d].
- Préparer les modifications à l'envoi au dépôt central de la présente version du projet avec la commande `git commit -m "Message de version"` [Atlassian, 2024a].
- Envoi de la nouvelle version du projet (avec les modifications) au dépôt central avec la commande `git push`.
- Récupération des modifications par un collaborateur au projet avec la commande `git pull` (facultatif).

Procédure générale d'utilisation de *Git*

- Création d'un dépôt sur un site hôte comme *GitHub*.
- Clonage du dépôt avec la commande `git clone <url du dépôt>`.
- Ajout de fichiers et/ou modification de fichiers.
- Ajout des modifications à l'étape d'« index » avec la commande `git add <chemin d'accès vers le fichier>` [Git, 2024d].
- Préparer les modifications à l'envoi au dépôt central de la présente version du projet avec la commande `git commit -m "Message de version"` [Atlassian, 2024a].
- Envoi de la nouvelle version du projet (avec les modifications) au dépôt central avec la commande `git push`.
- Récupération des modifications par un collaborateur au projet avec la commande `git pull` (facultatif).

Procédure générale d'utilisation de *Git*

- Création d'un dépôt sur un site hôte comme *GitHub*.
- Clonage du dépôt avec la commande `git clone <url du dépôt>`.
- Ajout de fichiers et/ou modification de fichiers.
- Ajout des modifications à l'étape d'« index » avec la commande `git add <chemin d'accès vers le fichier>` [Git, 2024d].
- Préparer les modifications à l'envoi au dépôt central de la présente version du projet avec la commande `git commit -m "Message de version"` [Atlassian, 2024a].
- Envoi de la nouvelle version du projet (avec les modifications) au dépôt central avec la commande `git push`.
- Récupération des modifications par un collaborateur au projet avec la commande `git pull` (facultatif).

UNIVERSITÉ LAVAL

Qu'est-ce qu'une branche *Git* ?

- Définition 1 : Version du projet contenant un historique de modifications.
- Définition 2 de [Git, 2024a] : « pointeur » vers une version (« commit ») pouvant être déplacé.
- Création d'une branche : commande `git branch <nom de la branche>`.
- Changement de branche : commande `git checkout <nom de la branche>`.

Qu'est-ce qu'une branche *Git* ?

- Définition 1 : Version du projet contenant un historique de modifications.
- Définition 2 de [Git, 2024a] : « pointeur » vers une version (« commit ») pouvant être déplacé.
- Création d'une branche : commande `git branch <nom de la branche>`.
- Changement de branche : commande `git checkout <nom de la branche>`.

Qu'est-ce qu'une branche *Git* ?

- Définition 1 : Version du projet contenant un historique de modifications.
- Définition 2 de [Git, 2024a] : « pointeur » vers une version (« commit ») pouvant être déplacé.
- Création d'une branche : commande `git branch <nom de la branche>`.
- Changement de branche : commande `git checkout <nom de la branche>`.

Qu'est-ce qu'une branche *Git* ?

- Définition 1 : Version du projet contenant un historique de modifications.
- Définition 2 de [Git, 2024a] : « pointeur » vers une version (« commit ») pouvant être déplacé.
- Création d'une branche : commande `git branch <nom de la branche>`.
- Changement de branche : commande `git checkout <nom de la branche>`.

Méthode par fonctionnalité

Idée générale

- Création d'une branche (ex. : *fonctionnaliteA*), différente de la branche principale nommée *main* ou *master*, pour chaque ajout de fonctionnalité majeure [Atlassian, 2024b].
- Complétion d'une *pull request* (pour une revue par les pairs) ou utilisation de la commande `git merge <nom de la branche pour la fusion>` depuis la branche *main* lorsque le développement de la fonctionnalité est terminé.

Exemple 1

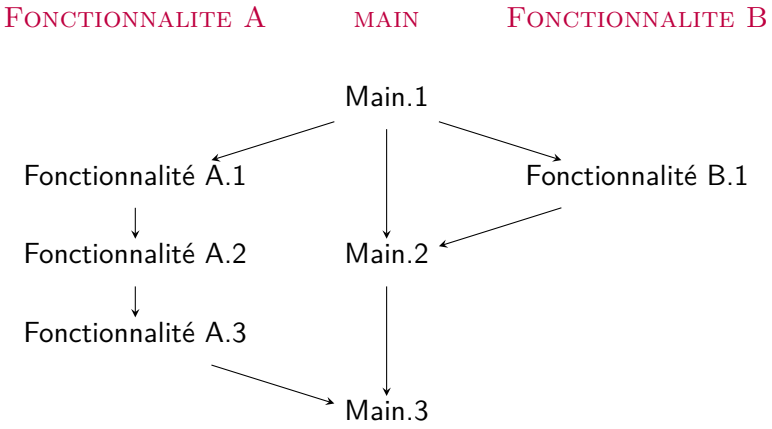

Figure – Schéma conceptuel de la méthode de travail *Git* par fonctionnalité, inspiré de schémas trouvés sur la page web [Atlassian, 2024b].

Méthode *GitFlow* de M. Vincent Driessen

Idée générale

- Cinq branches ou types de branches : *main*, *develop*, *release*, *feature* et *hotfix* [Atlassian, 2024c].
- Développements effectués dans des branches de fonctionnalités (*features*) découlant de la branche *develop* [Atlassian, 2024c].
- Fusion des développements à la branche *main* par l'intermédiaire d'une branche *release* [Atlassian, 2024c].
- Correction de bogues dans la branche *main* avec des branches *hotfix*.

Idée générale

- Cinq branches ou types de branches : *main*, *develop*, *release*, *feature* et *hotfix* [Atlassian, 2024c].
- Développements effectués dans des branches de fonctionnalités (*features*) découlant de la branche *develop* [Atlassian, 2024c].
- Fusion des développements à la branche *main* par l'intermédiaire d'une branche *release* [Atlassian, 2024c].
- Correction de bogues dans la branche *main* avec des branches *hotfix*.

Idée générale

- Cinq branches ou types de branches : *main*, *develop*, *release*, *feature* et *hotfix* [Atlassian, 2024c].
- Développements effectués dans des branches de fonctionnalités (*features*) découlant de la branche *develop* [Atlassian, 2024c].
- Fusion des développements à la branche *main* par l'intermédiaire d'une branche *release* [Atlassian, 2024c].
- Correction de bogues dans la branche *main* avec des branches *hotfix*.

Idée générale

- Cinq branches ou types de branches : *main*, *develop*, *release*, *feature* et *hotfix* [Atlassian, 2024c].
- Développements effectués dans des branches de fonctionnalités (*features*) découlant de la branche *develop* [Atlassian, 2024c].
- Fusion des développements à la branche *main* par l'intermédiaire d'une branche *release* [Atlassian, 2024c].
- Correction de bogues dans la branche *main* avec des branches *hotfix*.

Exemple 2

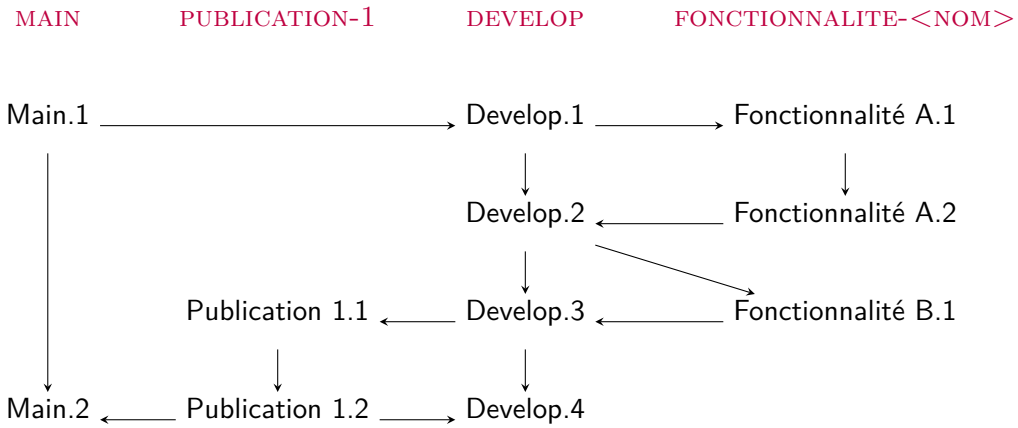

Figure – Schéma conceptuel de la méthode de travail *Git Gitflow*, inspiré de schémas de [Atlassian, 2024c].

Conclusion

Conclusion

Résumé :

- Procédure d'utilisation du logiciel *Git*.
- Méthodes de travail *Git*.

Ressources additionnelles :

- Site web officiel de *Git* : https://git-scm.com/
- Tutoriels de la compagnie Atlassian : https://www.atlassian.com/git

Merci !

Bibliographie I


Atlassian (2024a).
Getting git right.
https://www.atlassian.com/git.
Consulté le jeudi 27 juin 2024.


Atlassian (2024b).
Git feature branch workflow.
https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow.
Consulté le jeudi 27 juin 2024.

Bibliographie II


Atlassian (2024c).
Gitflow workflow.
https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow.
Consulté le jeudi 27 juin 2024.


Git (2024a).
3.1 git branching - branches in a nutshell.
https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell.

Consulté le jeudi 4 juillet 2024.

Bibliographie III


Git (2024b).
Branching and merging.
https://git-scm.com/about/branching-and-merging.
Consulté le jeudi 27 juin 2024.


Git (2024c).
Free and open source.
https://git-scm.com/about/free-and-open-source.
Consulté le jeudi 27 juin 2024.


Git (2024d).
Staging area.
https://git-scm.com/about/staging-area.
Consulté le jeudi 27 juin 2024.