{saperlipopette}, un paquet R pour progresser en Git en toute sérénité

Maëlle Salmon*

Résumé (max 300 mots)

Une pratique de Git confiante peut changer la vie de tout développeur·se de paquets R : un historique utile, une capacité à travailler en parallèle sur différents aspects dans différentes branches, etc. Cependant, il faut en arriver là à la sueur de son front, car Git, ce n'est pas simple ! Notamment, un obstacle peut être de ne pas savoir où et comment s'entraîner à utiliser les commandes moins basiques telles que `git commit --amend` pour changer son dernier commit, `git rebase -i` pour retravailler l'historique d'une branche, `git bisect` pour trouver quel commit a introduit ce fichu bug... Le paquet R saperlipopette est là pour vous aider ! Il offre pour le moment 12 exercices, inspirés du célèbre site "Dangit, Git!?!", ou de la pratique Git de l'autrice. Chaque exercice est commencé en appelant une fonction telle que `saperlipopette::exo_committed_to_wrong()` qui concerne le scénario "zut, j'ai fait mon commit sur la mauvaise branche". La fonction crée l'exercice dans un dossier qu'on lui indique, temporaire par exemple. L'utilisateur·rice ouvre R dans ce dossier, et lit les instructions qui apparaissent. Si cela ne suffit pas, il·elle peut appeler la fonction `tip()` qui lui donne des indices en plus. Ainsi, on s'entraîne à Git sur un scénario utile et réaliste, avec ses outils habituels, depuis notre chère console R, dans un dossier où il n'y a rien à casser. Dans cette présentation, je vous expliquerai pourquoi j'ai créé ce paquet, comment il s'utilise, et pourquoi les fichiers `.Rprofile` sont les rois de son implémentation.
Mots-clefs (3 à 5) : Git - Package - Enseignement - Bonnes pratiques - Développement

Git, un outil crucial mais difficile à prendre en main

L'utilisation compétente d'un logiciel de contrôle de version améliore la productivité du développement logiciel des individus et des équipes en leur permettant de travailler simultanément sur différentes fonctionnalités, tout en enregistrant l'historique du projet. Améliorer ses compétences en contrôle de version au-delà des basiques est un investissement utile, qui se traduit rapidement par des gains de temps : par exemple on peut utiliser des commandes spécifiques pour annuler les modifications au lieu d'effacer et de copier-coller manuellement. Cela permet également de réduire les risques de perte de travail, car on sait comment éviter les erreurs et comment les réparer. Parmi les outils de contrôle de version, Git est le plus populaire ; en outre, une grande partie du développement de logiciels libres dans le monde se fait sur GitHub et, dans une moindre mesure, sur GitLab, qui utilisent tous deux Git. Cependant, Git est notoirement déroutant et difficile à apprendre, ce qui rend le perfectionnement potentiellement décourageant.

saperlipopette, un astucieux paquet R pour s'entraîner à Git en confiance

Pour tenter de résoudre ce problème, nous avons développé un paquet R, saperlipopette https://maelle.github.io/saperlipopette/. Le paquet saperlipopette permet de pratiquer localement les commandes Git avancées, qui ne sont généralement pas enseignées dans les ateliers pour débutant·e·s. Le public cible de saperlipopette est constitué de professionnel·le·s

- qui développent des scripts ou paquets R,
- qui travaillent habituellement avec R et connaissent les commandes Git de base telles que add, commit, push, et pull, et qui comprennent un peu les branches,

*rOpenSci, msmaellesalmon@gmail.com

- mais ne se sentent pas encore à l'aise avec Git, et donc sous-utilisent la palette de fonctionnalités de Git.

Par le paquet saperlipopette, nous fournissons aux apprenant·e·s 12 exercices (12 fonctions), inspirés du célèbre site "Dangit, Git!?!", ou de la pratique Git de l'autrice. Les exercices sont centrés sur un scénario réaliste et utile, comme l'annulation d'un changement ou l'utilisation de la machine à remonter le temps de Git pour réparer un projet, et qui sont tous contenus dans un petit répertoire temporaire donc jetable : le code de saperlipopette crée le dossier, recrée l'historique et la situation Git nécessaires à l'exercice, stocke les instructions dans le `.Rprofile` du dossier. Par conséquent, les utilisateur·rice·s de saperlipopette peuvent s'exercer de manière réaliste à l'utilisation de nouvelles commandes Git plus avancées sans mettre en danger leur travail ni avoir à réfléchir à la manière de créer artificiellement un scénario d'exercice pour eux-mêmes.

Un outil d'apprentissage actif

Le principe de fonctionnement de saperlipopette est basé sur des stratégies pédagogiques d'apprentissage actif qui consistent à travailler sur des tâches authentiques, avec une pratique guidée. De plus, les exercices sont locaux, ce qui signifie que les utilisateurs peuvent s'exercer sur leur machine, avec les outils qu'ils connaissent déjà ou avec lesquels ils essaient de se familiariser : Git en ligne de commande, Git avec une interface telle que RStudio IDE, ou d'autres clients Git à usage général. La recréation d'un exercice se fait simplement par un appel de fonction R, de sorte que les utilisateurs peuvent s'exercer à tout moment, et même se rafraîchir la mémoire sur une commande avant de l'appliquer à leur projet de travail réel. Cette approche permet de réduire la charge cognitive des apprenant·e·s. Il est important de noter que notre logiciel fournit deux niveaux de guidage pour chaque exercice :

- une description générale du problème à résoudre à l'ouverture de la session R,
- la possibilité de demander plus d'indications si nécessaire en appelant la fonction `tip()`.

Enfin, du point de vue du formateur ou de la formatrice, les exercices prêts à l'emploi de saperlipopette permettent de gagner du temps dans la démonstration des flux de travail Git.

Présentation de saperlipopette

Dans cette présentation, nous couvrirons

1. L'utilité des commandes Git au-delà des bases ;
2. Le fonctionnement de saperlipopette pour un exercice simple ;
3. La liste d'exercices et les raisons de leurs choix ;
4. Le fonctionnement de saperlipopette pour un exercice plus compliqué ;
5. L'infrastructure interne de saperlipopette comme son utilisation de fichiers `.Rprofile`;
6. Des pistes d'amélioration du paquet.

L'objectif sera de rendre l'audience curieuse de saperlipopette et Git, et aussi de récolter des retours pour améliorer le paquet.

Références

- saperlipopette https://maelle.github.io/saperlipopette/
- Dangit, Git!?! https://dangitgit.com/fr