Git II : corriger des erreurs

Ces exercices vont vous guider pour faire une gestion très basique des erreurs qu'on peut voir dans git. Ils servent également à montrer l'intérêt du *workflow* de git pour (r)attraper des erreurs. Les exercices sont scénarisés pour créer des erreurs et les résoudre par la suite.

Il est demandé de ne pousser (*push*) que lorsque c'est indiqué dans la feuille, pour le bon fonctionnement des exercices.

Exercice 1 Avant de commencer

Cette feuille d'exercices est prévue pour être faite après les exercices sur les pipelines. À ce moment, vous devriez avoir un tag '**fin-ex-pipelines**' qui marquera le début du travail sur cette séance.

Exercice 2 Défaire un commit erroné après l'avoir poussé

L'idée de cet exercice est de montrer une possibilité qu'offre git pour corriger des erreurs après les avoir poussées sur le dépôt distant.

Exercice 2.a Créez un commit erroné au milieu d'autres

Ajoutez après à la fin du journal de bord une section pour la séance d'aujourd'hui. Vous pouvez utiliser une section type "# Séance 3" et/ou une sous-section spécifique git avec "## exercices git". Committez avec le message "Ajout section git pour la séance 3".

Ajoutez une à quelques lignes bidons [1] au début de votre journal de bord et committez avec le message "I AM ERROR".

Ajoutez une nouvelle ligne où vous indiquerez un message comme "Cette ligne doit rester après correction". Committez avec le message de votre choix.

Poussez vos changements.

Exercice 2.b Défaire le commit

Récupérez l'identifiant SHA (longue chaîne de lettres et chiffres) du commit au message "I AM ERROR" dans votre terminal. Quelle commande devez-vous utiliser pour cela ?

À l'aide de cet identifiant, défaites le commit. Quelle commande devez-vous utiliser ? Regardez l'historique de vos commits, que pouvez-vous dire par rapport au commit que vous avez défait et ce que cela peut impliquer ?

Exercice 2.c Créez un nouveau tag

Créez un tag "git-seance3-defaire" avec le message "Git : apprendre à défaire un commit, séance 3".

Exercice 3 Défaire des changements avant de pousser

Comme suggère le titre de l'exercice, ne faites pas faire de push hâtif sous peine de casser le scénario de l'exercice.

L'idée de cet exercice est de voir comment il est possible de corriger des erreurs avant qu'elle ne soient poussées vers le dépôt distant.

1. Si vous manquez d'inspiration, vous pouvez mettre un petit *lorem ipsum*.

Nous allons effectuer une série de commits dans un mauvais fichier que vous appellerez "oups.md". Vous pouvez utiliser "oups.md" à la place de votre journal de bord pour le reste de la séance. Vous pouvez uniquement faire des `add` et des `commit`, mais aucun `push`.

Exercice 3.a Retourner à la bonne version

Lorsque vous aurez fini votre travail (poussé votre dernier commit), vous devrez retourner vers la version du tag "git-seance3-defaire". La commande doit satisfaire les contraintes suivantes :

— vous ne devez pas perdre les changements
— les changements ne doivent pas être *staged* après le retour au tag

Exercice 3.b Basculer les changements vers le bon fichier

Récupérez les changements effectués dans "oups.md" et copiez-les dans le journal de bord.

Supprimez le fichier "oups.md".

Une fois que vous aurez tout récupéré dans le bon journal de bord, committez et poussez vos changements.

Quelle(s) différence(s) voyez-vous entre les deux méthodes ?

Exercice 4 Garder de côté des changements

L'idée de cet exercice est de créer un conflit entre votre version locale et la version distante qui vous empêche de mettre à jour votre dépôt et d'ajouter vos changements au dépôt en ligne. Nous verrons ensuite une approche possible pour résoudre ce problème.

Exercice 4.a Créer un conflit

Pour créer un conflit, nous allons d'abord modifier le même fichier à deux endroits : sa version en ligne et sa version locale. Sur votre dépôt en ligne, modifiez dans votre journal la ligne "Exercices git" pour ajouter à la fin de la ligne "correction d'erreurs". Modifiez ensuite votre journal en ajoutant une ligne à la fin indiquant que vous allez mettre cette ligne de côté.

Exercice 4.b Anticiper et régler un conflit potentiel

On souhaite à présent voir sur la version locale si nous sommes à jour par rapport à la version en ligne. Quelle(s) commande(s) pouvez-vous utiliser ?

Normalement, vous verrez que vous avez 1 commit de retard par rapport au dépôt en ligne. Tentez de récupérer les changements, que se passe-t-il ?

On souhaite garder dans un coin les changements effectués en local pour les appliquer après avoir récupéré les modification de la version en ligne. On souhaite indiquer le message "Conservation des changements séance 3". On ne souhaite pas encore appliquer ces changements. Quelle(s) commande(s) utiliser ?

Supposons que nous avons laissé quelques jours notre dépôts et que notre mémoire nous fait défaut. Quelle(s) commande(s) utiliser pour voir la liste des changements conservés et ce qu'ils contiennent ?

En théorie, vous n'avez qu'un seul tas de changements conservés. Une fois appliqués, on ne souhaite plus voir les changements en utilisant les commandes du paragraphe précédent. Quelle(s) commande(s) utiliser ?

Une fois les changements appliqués, effectuez les opérations nécessaires pour les pousser sur le dépôt distant.

Exercice 5 Créez un tag

Créez un tag "git-seance3-fin" avec le message "séance 3, version finale des exercices de git" et poussez-le.