Les commandes push, pull et revert (1)

Auteur : Romain NOEL	**Date : 02/03/2020**	**Version : 1**
Périmètre de l'opération : Travailler à plusieurs, publier des sauvegardes et gérer les conflits		
Prérequis : Poste de travail configuré		
Outil : Git	**Durée :** 45min	

Opérations (ce qu'il faut faire)	**Points clés** (à quoi il faut faire attention)	**Argumentaire** (pourquoi c'est important)
Partie 1 : Push sans conflits		
1/ Publier vos commits `git push`	Vous copiez vos commits depuis votre dépôt local sur le dépôt distant	

2/ Faire un commit
 	
`touch <votreprenom>`
`git add <votreprenom>`
`git commit -m "Commit de <votreprenom>"`
	Vous créez un commit qui n'existe que sur votre machine locale	

3/ Attendre que votre binôme finisse son action 1.1		➢ Votre binôme ne bénéficiera pas de votre dernier commit puisqu'il n'existe que sur votre machine pour le moment.
4/ Publier son commit `git push`	Vous mettez le dépôt distant à jour avec votre dernier commit.	➢ Votre binôme est en retard d'un commit sur son dépôt par rapport au dépôt distant.
5/ Attendre que votre binôme finisse son action 1.6 (et suivre avec lui)		➢ Votre binôme met son dépôt à jour et pousse sur le repo son propre commit.

6/ Se mettre à jour avec le repo distant
 	
`git pull`
`ls`
`git log`
	Vous avez récupéré la modification de votre binôme.	

Partie 2 : Pull avec conflits		
1/ Créer un fichier « prenoms.txt » et le pousser touch prenoms.txt git add prenoms.txt git commit -m "Ajout d'un fichier prenoms.txt" git push		➢ Création d'un jeu de données qui va devenir conflictuel.
2/ Attendre que votre binôme effectue l'action 2.2		➢ Votre binôme récupère le fichier que vous venez de créer
3/ Ajouter son prénom dans le fichier et faire un commit echo "Jean-Mohammed-Yiao" >> prenoms.txt git add prenoms.txt git commit -m "Ajout de Jean-Mohammed-Yiao"		
4/ Attendre que votre binôme effectue l'action 2.3		➢ Votre binôme écrit également son prénom sur la première ligne de ce fichier et le publie.
5/ Publier son prénom git push	La publication ne passe pas car votre dépôt n'est pas à jour.	
6/ Mettre à jour son dépôt git pull	Git vous indique que vous avez des conflits	
7/ Identifier et résoudre les conflits cat prenoms.txt sed -i "1d;3d;5d" prenoms.txt cat prenoms.txt	Git marque les conflits par les caractères suivants : <<<<<<<<<, ==========, et >>>>>>>>>. La commande sed supprime ces lignes (1, 3 et 5 étant les numéros des lignes qui marquent le conflit).	➢ Ici, nous cherchons à conserver les deux lignes. Il suffit juste de supprimer les marqueurs de conflits. Dans certains cas, il faudra étudier quel bloc conserver, ou encore effectuer les fusions manuellement.
8/ Publier le code fusionné git add prenoms.txt git commit -m "Conflit résolu. Conservation des deux prénoms" git push	Création d'un nouveau commit résultat de la fusion des deux modifications. Et publication de ce dernier.	➢ Le conflit est résolu

Partie 3 : Pull avec conflits		
1/ Attendre que votre binôme effectue l'action 3.1		➢ Votre binôme génère un fichier servira à générer un conflit
2/ Récupérer le jeu de données git pull git log ls	Vous venez de récupérer le jeu de données généré par votre binôme	
3/ Ajouter un animal dans le fichier et faire un commit echo "ornithorynque" >> animaux.txt git add animaux.txt git commit -m "Ajout de ornithorynque" git push		➢ Votre binôme va en faire autant. C'est ce qui va générer un conflit !
4/ Suivre et aider votre binôme pour le reste de l'exercice		
Partie 4 : Revert		
1/ Ajoutez et publiez une ligne dans le fichier animaux echo "chaise" >> animaux.txt git add animaux.txt git commit -m "Ajout de chaise" git push	Vous vous êtes trompés dans votre commit (une chaise n'est pas un animal). Votre commit est déjà publié.	➢ La publication d'un commit est irréversible. Il va falloir réparer votre erreur.
2/ Créer un commit inverse git log copier le # du dernier commit git revert <#commit> echap -> :q -> entrée (sortir de vi) git log	Création d'un commit inverse (suppression de la ligne que vous vener d'ajouter)	➢ Cette commande permet de créer un commit inverse. Il aurait été possible de le l'effectuer à la main mais il vaut mieux faire confiance à git pour gérer des cas plus complexe.
3/ Publier son commit inverse git push	Votre erreur est réparée	➢ Ce mode de réparation d'erreur se voit dans les logs de git.