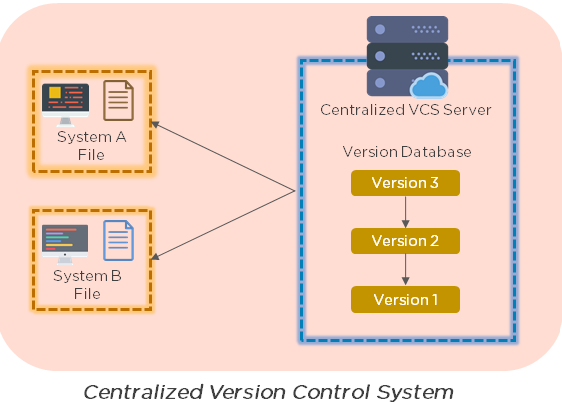
DevOps Interview Questions for Source Code Management — Git

15. Explain the difference between a centralized and distributed version control system (VCS).

Centralized Version Control System

* All file versions are stored on a central server
* No developer has a copy of all files on a local system
* If the central server crashes, all data from the project will be lost



Distributed Control System

- Chaque développeur a une copie de toutes les versions du code sur ses systèmes  
- Permet aux membres de l'équipe de travailler hors ligne et ne dépend pas d'un seul emplacement pour les sauvegardes

- Il n'y a aucune menace, même si le serveur plante

16. What is the git command that downloads any repository from GitHub to your computer?



The git command that downloads any repository from GitHub to your computer is git clone.

17. How do you push a file from your local system to the GitHub repository using Git?

First, connect the local repository to your remote repository:

git remote add origin [copied web address]

// Ex: git remote add origin <https://github.com/Simplilearn-github/test.git>

Second, push your file to the remote repository:

git push origin master

18. How is a bare repository different from the standard way of initializing a Git repository?

Using the standard method:

git init

* You create a working directory with git init
* A .git subfolder is created with all the git-related revision history

Using the bare way

git init --bare

* It does not contain any working or checked out a copy of source files
* Bare repositories store git revision history in the root folder of your repository, instead of the .git subfolder

### 19. Which of the following CLI commands can be used to rename files?

1. git rm
2. git mv
3. git rm -r
4. None of the above

The correct answer is B) git mv

20. Quel est le processus pour annuler un commit qui a déjà été poussé et rendu public?

There are two ways that you can revert a commit:

1. Remove or fix the bad file in a new commit and push it to the remote repository. Then commit it to the remote repository using:  
   git commit –m "commit message"
2. Create a new commit that undoes all the changes that were made in the bad commit. Use the following command:  
     
   git revert <commit id>

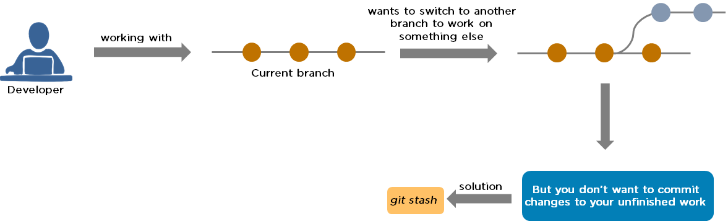
Example: git revert 56de0938f

21. Explain the difference between git fetch and git pull.

|  |  |
| --- | --- |
| Git fetch | Git pull |
| Git fetch télécharge uniquement les nouvelles données à partir d'un référentiel distant | Git Pull met à jour la branche HEAD actuelle avec les dernières modifications du serveur distant |
| N'intègre pas de nouvelles données dans vos fichiers de travail | Télécharge de nouvelles données et l'intègre aux fichiers de travail actuels |
| Les utilisateurs peuvent exécuter une extraction Git à tout moment pour mettre à jour les branches de suivi à distance | Tente de fusionner les modifications à distance avec vos modifications locales |
| Command - git fetch origin  git fetch –-all | Command - git pull origin master |

22. What is Git stash?

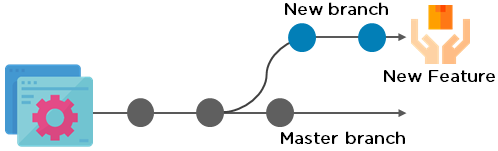
Un développeur travaillant avec une branche actuelle souhaite passer à une autre branche pour travailler sur autre chose, mais le développeur ne souhaite pas valider les modifications apportées à votre travail inachevé. La solution à ce problème est la cachette Git. Git Stash prend vos fichiers suivis modifiés et les enregistre sur une pile de modifications inachevées que vous pouvez réappliquer à tout moment.



23. Explain the concept of branching in Git.

Suppose you are working on an application, and you want to add a new feature to the app. You can create a new branch and build the new feature on that branch.

* By default, you always work on the master branch
* The circles on the branch represent various commits made on the branch
* After you are done with all the changes, you can merge it with the master branch



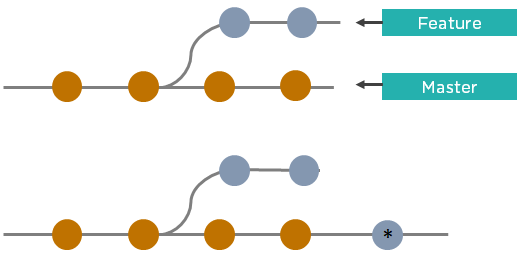
24. What is the difference between Git Merge and Git Rebase?

Suppose you are working on a new feature in a dedicated branch, and another team member updates the master branch with new commits. You can use these two functions:

Git Merge

To incorporate the new commits into your feature branch, use Git merge.

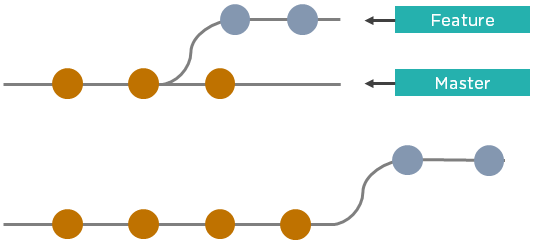
* Creates an extra merge commit every time you need to incorporate changes
* But, it pollutes your feature branch history



Git Rebase

As an alternative to merging, you can rebase the feature branch on to master.

* Incorporates all the new commits in the master branch
* It creates new commits for every commit in the original branch and rewrites project history



25. How do you find a list of files that have been changed in a particular commit?

The command to get a list of files that have been changed in a particular commit is:

git diff-tree –r {commit hash}

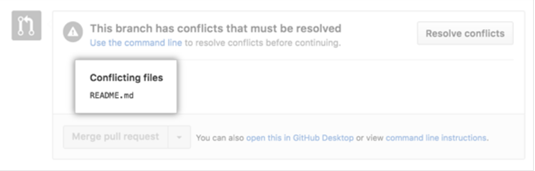
Example: git diff-tree –r 87e673f21b

* -r flag instructs the command to list individual files
* commit hash will list all the files that were changed or added in that commit

26. What is a merge conflict in Git, and how can it be resolved?

A Git merge conflict

se produit lorsque vous avez merge branches avec des validations de validations, et Git a besoin de votre aide pour décider des modifications à intégrer dans final merge.

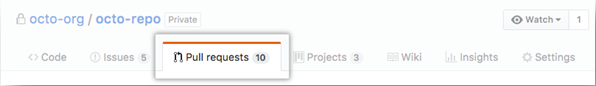


Modifiez manuallement le fichier en conflit pour sélectionner les modifications que vous souhaitez conserver lors de la fusion finale.

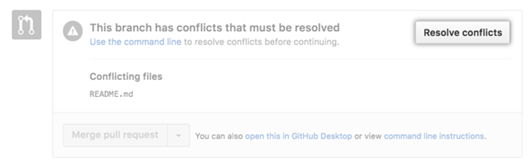
Resolve using GitHub conflict editor

This is done when a merge conflict is caused after competing for line changes. For example, this may occur when people make different changes to the same line of the same file on different branches in your Git repository.

* Resolving a merge conflict using conflict editor:
* Under your repository name, click "Pull requests."



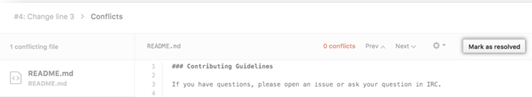
* In the "Pull requests" drop-down, click the pull request with a merge conflict that you'd like to resolve
* Near the bottom of your pull request, click "Resolve conflicts."



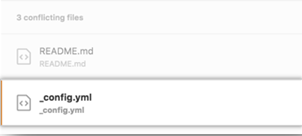
* Decide if you only want to keep your branch's changes, the other branch's changes, or make a brand new change, which may incorporate changes from both branches.
* Delete the conflict markers <<<<<<<, =======, >>>>>>> and make changes you want in the final merge.



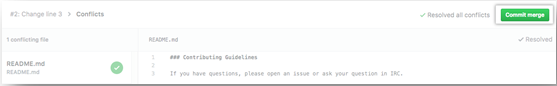
* If you have more than one merge conflict in your file, scroll down to the next set of conflict markers and repeat steps four and five to resolve your merge conflict.
* Once you have resolved all the conflicts in the file, click Mark as resolved.



* If you have more than one file with a conflict, select the next file you want to edit on the left side of the page under "conflicting files" and repeat steps four to seven until you've resolved all of your pull request's merge conflicts.



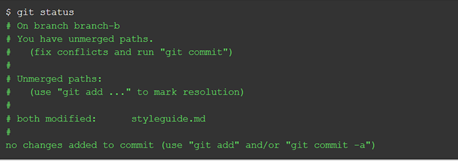
* Once you've resolved your merge conflicts, click Commit merge. This merges the entire base branch into your head branch.



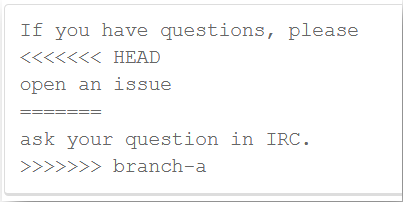
* To merge your pull request, click Merge pull request.
* A merge conflict is resolved using the command line.
* Open Git Bash.
* Navigate into the local Git repository that contains the merge conflict.



* Generate a list of the files that the merge conflict affects. In this example, the file styleguide.md has a merge conflict.



* Open any text editor, such as Sublime Text or Atom, and navigate to the file that has merge conflicts
* To see the beginning of the merge conflict in your file, search the file for the conflict marker "<<<<<<<. " Open it, and you'll see the changes from the base branch after the line "<<<<<<< HEAD."
* Next, you'll see "=======", which divides your changes from the changes in the other branch, followed by ">>>>>>> BRANCH-NAME".



* Decide if you only want to keep your branch's changes, the other branch's changes, or make a brand new change, which may incorporate changes from both branches.
* Delete the conflict markers "<<<<<<<", "=======", ">>>>>>>" and make the changes you want in the final merge.

In this example, both the changes are incorporated into the final merge:



* Add or stage your changes.



* Commit your changes with a comment.



Now you can merge the branches on the command line, or push your changes to your remote repository on GitHub and merge your changes in a pull request.