**Git Interview Questions with Answer**

**🔰 Beginner Level**

**1. What is Git, and how is it different from other version control systems like SVN?**

**Answer:**  
Git is a **distributed version control system** used for tracking changes in source code during software development. Unlike SVN, which is **centralized**, Git allows each developer to have a full copy of the repository, enabling offline work and faster operations.

**Example:**  
In SVN, if the server goes down, you lose access. In Git, even if the server is down, you can continue working locally and push your changes later.

**2. What is the difference between Git and GitHub?**

**Answer:**

* **Git**: A version control system.
* **GitHub**: A hosting platform for Git repositories that also provides collaboration features.

**Example:**  
You use Git to commit code changes locally, and GitHub to **push** them to a shared repository so your team can collaborate.

**3. What are the basic Git commands you use daily?**

**Answer:**

* git clone <repo\_url> – Clone a repository
* git add . – Stage changes
* git commit -m "message" – Commit changes
* git push – Push to remote
* git pull – Fetch and merge remote changes

**4. What is the purpose of git init?**

**Answer:**  
git init initializes a new Git repository in your local project folder.

**Example:**  
You just created a new project folder my-app. Run git init to start tracking it with Git.

**5. Explain the difference between git add, git commit, and git push.**

**Answer:**

* git add – Moves changes to staging area.
* git commit – Records changes in local repository.
* git push – Uploads commits to the remote repository.

**Example:**  
Modify index.js → git add index.js → git commit -m "Update index" → git push origin main

**6. What does git status show?**

**Answer:**  
It shows the status of files in the working directory – tracked, untracked, modified, staged.

**7. How do you create a new branch in Git?**

**Answer:**  
git branch feature-login – Creates a branch  
git checkout feature-login – Switches to it  
OR  
git checkout -b feature-login – Both in one step

**8. How do you switch between branches?**

**Answer:**  
git checkout branch-name or git switch branch-name (newer syntax)

**9. What is a merge conflict? How do you resolve it?**

**Answer:**  
Occurs when two branches have changes in the same part of the same file.

**Example:**

* feature1 updates line 10 of index.js
* feature2 also updates line 10  
  When merged, Git cannot decide which change to keep. You resolve manually and commit the fix.

**10. Explain git clone vs git pull.**

**Answer:**

* git clone – Copies a remote repo for the first time
* git pull – Fetches and merges changes from remote into your local branch

**🔄 Intermediate Level**

**11. What is the difference between git fetch and git pull?**

**Answer:**

* git fetch – Downloads updates without merging
* git pull – git fetch + git merge  
  Allows review before merging.

**12. How does git rebase differ from git merge?**

**Answer:**

* git merge – Keeps history of both branches.
* git rebase – Rewrites history to make it linear.

**Example:**  
Feature branch appears like it's based on latest main.

**13. What is the use of .gitignore?**

**Answer:**  
Specifies files Git should ignore.

**Example:**  
Ignore node\_modules/, \*.log, or .env in .gitignore.

**14. What is a detached HEAD in Git?**

**Answer:**  
HEAD is not pointing to any branch – you're checking out a specific commit.

**Example:**  
Running git checkout 3a5b8e1 (a commit hash) puts you in detached HEAD state.

**15. How do you revert a commit?**

**Answer:**

* git revert <commit> – Adds a new commit that undoes the changes.

**16. How do you squash commits?**

**Answer:**  
Use interactive rebase:  
git rebase -i HEAD~3  
Then mark old commits as squash or s.

**17. Explain git stash and how to use it.**

**Answer:**  
Temporarily saves uncommitted changes.  
git stash – Stash current changes  
git stash pop – Apply them later

**18. Difference between git reset, git checkout, and git revert**

**Answer:**

* git reset – Moves HEAD and can unstage or delete commits
* git checkout – Switch branches or restore files
* git revert – Reverses changes with a new commit

**19. How can you view commit history in Git?**

**Answer:**  
Use git log  
Or git log --oneline --graph --all for a graphical view.

**20. What is cherry-picking in Git?**

**Answer:**  
Apply a specific commit from one branch to another.  
git cherry-pick <commit-hash>

**🚀 Advanced Level**

**21. What is Git rebase interactive (git rebase -i) and when do you use it?**

**Answer:**  
Used to edit, squash, reorder commits interactively.  
git rebase -i HEAD~n opens a list of last n commits.

**22. How do you resolve conflicts after a rebase?**

**Answer:**  
Git marks conflicts. You manually edit files, then run:

* git add .
* git rebase --continue

**23. How do you handle large binary files in Git?**

**Answer:**  
Use **Git LFS (Large File Storage)**  
git lfs install, git lfs track "\*.zip"

**24. How do you undo a pushed commit?**

**Answer:**  
If not shared:

* git reset --hard HEAD~1  
  If shared:
* git revert <commit>

**25. What are Git hooks?**

**Answer:**  
Scripts that run on Git events (e.g., pre-commit, pre-push).  
Use for auto-formatting or validation.

**26. How do you configure Git in a CI/CD pipeline?**

**Answer:**  
Use .git to fetch source, clone repo, checkout specific branch. Example using GitHub Actions:

yaml

CopyEdit

- uses: actions/checkout@v2

**27. What’s the difference between local, remote, and upstream branches?**

**Answer:**

* **Local**: Exists on your system
* **Remote**: Copy from central repo
* **Upstream**: Tracking branch you pull from/push to

**28. How do you clean up a Git repository?**

**Answer:**

* Delete old branches: git branch -d old-feature
* Remove large files: bfg-repo-cleaner or Git LFS
* Prune: git gc, git prune

**29. Explain the concept of Git reflog.**

**Answer:**  
git reflog tracks every move of HEAD. Useful to recover lost commits.

**Example:**  
You reset hard and lost work? Use git reflog to find and restore.

**30. What is the difference between origin/main and main?**

**Answer:**

* main – Local branch
* origin/main – Remote tracking branch