Interview Questions

JAVA-112: Session 2 - Git

Answering interview questions is crucial in your journey of applied learning. Review them to ensure your understanding of important topics covered in the previous session and to prepare yourself for upcoming challenges. Remember that it's important to answer these questions on your own before viewing the solution. The solutions are hyperlinked to community posts on our platform.

**Note: The questions below have been sourced from previous interviews**

Questions

1. [What is the difference between Git, GitHub and GitLab?](https://crio.do/learn/crio-community/topic/what-is-the-difference-between-git-github-and-gitlab/169165)
2. [Explain git workflow.](https://crio.do/learn/crio-community/topic/explain-git-workflow/114050)
3. [What is the purpose of the .gitignore file?](https://crio.do/learn/crio-community/topic/what-is-the-purpose-of-the-gitignore-file/209678)
4. [Explain the difference between Git merge and Git rebase.](https://crio.do/learn/crio-community/topic/explain-the-difference-between-git-merge-and-git-rebase/209679)
5. [What is a merge conflict, and how do you resolve it?](https://crio.do/learn/crio-community/topic/what-is-a-merge-conflict-and-how-do-you-resolve-it/209680)
6. [Explain the difference between Git pull and Git fetch.](https://crio.do/learn/crio-community/topic/explain-the-difference-between-git-pull-and-git-fetch/209681)
7. [How do you resolve a pull request conflict?](https://crio.do/learn/crio-community/topic/how-do-you-resolve-a-pull-request-conflict/209682)
8. [What are some best practices for collaborating with Git in a team environment?](https://crio.do/learn/crio-community/topic/what-are-some-best-practices-for-collaborating-with-git-in-a-team-environment/209683)

Here are clear and concise answers to your **Git-related interview questions** from **JAVA-112: Session 2**:

**1. What is the difference between Git, GitHub, and GitLab?**

* **Git**: A **distributed version control system** used to track changes in source code during development.
* **GitHub**: A **cloud-based hosting platform** for Git repositories that adds collaboration features like **pull requests**, **issue tracking**, and **actions** (CI/CD).
* **GitLab**: Similar to GitHub, but also includes **built-in CI/CD**, **DevOps pipelines**, and is often used for **self-hosted** Git repository management.

**2. Explain Git workflow.**

The basic Git workflow includes:

1. **Clone** the repository (git clone).
2. **Create or switch** to a new branch (git checkout -b feature-branch).
3. **Make changes** and **stage** them (git add filename).
4. **Commit** the changes (git commit -m "message").
5. **Pull** the latest changes from remote (git pull).
6. **Push** your changes (git push origin feature-branch).
7. **Create a pull request** (via GitHub/GitLab) for merging into the main branch.

**3. What is the purpose of the .gitignore file?**

The .gitignore file tells Git which files or directories to **ignore and not track**. Common examples:

* Compiled code (\*.class, \*.exe)
* Logs (\*.log)
* OS files (.DS\_Store, Thumbs.db)
* Environment configs (.env)

**4. Explain the difference between Git merge and Git rebase.**

* **Git Merge**: Combines changes from one branch into another by creating a **new merge commit**.
* **Git Rebase**: Re-applies commits from one branch **on top of another**, **rewriting history** to create a linear commit history.

| **Feature** | **Merge** | **Rebase** |
| --- | --- | --- |
| History | Preserves branching history | Creates linear history |
| Use when | Collaboration (safe & traceable) | Clean history before pushing |
| Creates new commit? | Yes | No (rewrites old commits) |

**5. What is a merge conflict, and how do you resolve it?**

A **merge conflict** occurs when Git cannot automatically merge code due to conflicting changes in the **same line or file**.

**To resolve:**

1. Git marks the conflict in the file with <<<<<<<, =======, and >>>>>>>.
2. **Manually edit** the file to fix the conflict.
3. Stage the fixed file (git add filename).
4. Commit the merge (git commit).

**6. Explain the difference between Git pull and Git fetch.**

* **Git Fetch**: Retrieves the latest changes from the remote **without merging** them into your local branch.
* **Git Pull**: Equivalent to git fetch **followed by** git merge, meaning it updates your local branch directly.

**7. How do you resolve a pull request conflict?**

To resolve conflicts in a pull request:

1. Checkout to the branch:  
   git checkout feature-branch
2. Pull latest main branch:  
   git pull origin main
3. Resolve any **merge conflicts manually**.
4. Stage and commit the resolved files.
5. Push the updated branch:  
   git push origin feature-branch

**8. What are some best practices for collaborating with Git in a team environment?**

* Use **feature branches** for each new task or bug.
* Commit often with **meaningful messages**.
* Pull regularly to stay updated.
* Resolve merge conflicts **promptly**.
* Use **.gitignore** to avoid committing unnecessary files.
* Use **pull requests** for code review.
* Avoid force-pushing to shared branches (main, develop).
* Rebase only on **local branches**, never rewrite public history.