Study guide: Conflict resolution

In Git, merge conflicts, or conflicts that occur when merged branches have competing commits, are not uncommon when working with a team of developers or when working with open-source software. This study guide provides you with tips for resolving merge conflicts.

Tips for resolving merge conflicts

Here are some tips to keep in mind when you're resolving merge conflicts:

- After running Git merge, a message will appear informing that a conflict occurred on the file.
- Read the error messages that imply you cannot push your local changes to GitHub, especially the remote changes with Git pull.
- Use the command line or GitHub Desktop to push the change to your branch on GitHub after you make a local clone of the repository for all other types of merge conflicts.
- Before merging any commits to the master branch, push it into a remote repository so that collaborators can view the code, test it, and inform you that it's ready for merging.
- Use the Git `rebase` command to replay the new commits on top of the new base and then merge the feature branch back into the master.

Key takeaways

It is important to effectively resolve merge conflicts because local changes cannot be made to Git until the merge conflicts have been locally resolved. Once all conflicts have been resolved, changes can be pushed to Git and merged in a pull request.

Mark as completed

Like **Dislike** **Report an issue**