Using a Git Workflow Natalie Belmonte, 28 October 2024

GitHub repositories are a useful tool for program developers that allow them to easily collaborate on a single project. Repositories are very adaptable, and as such can be used in various ways. At the start of a project, a developer team will establish a workflow in order to maintain consistency. When referencing Git repositories, a workflow is a "...recommendation for how to use Git to accomplish work in a consistent and productive manner" ("Git…" 1). Since they depend on team size, project complexity, and applicable release schedules, workflows can be as unique as the projects they are used for. Though developers can freely establish any guidelines needed for a project, there are public Git workflows available for use.

The primary purpose of a workflow is to allow a team of developers to smoothly implement changes to their programming. Workflows should not make mistakes more difficult to undo or be more restrictive than what is needed. One of the most popular Git workflows, which uses a centralized repository, requires each developer to clone their own local version of the project. This allows them to write and test changes individually before pushing them to the central repository.

Before "pushing" any changes, they must first be committed to a local repository. Committing code acts as confirmation that the code segment is completed, without any major errors. Then, the code can be pushed to the central repository, sharing it with all other developers and merging it into the preexisting program. This process allows multiple developers to edit the same sections of code, without worrying about their changes contradicting each other.

If a commit conflicts with what is already written in the central repository, the code will not be pushed. This is known as a merge conflict. Often, this occurs when a developer writes code, then tries to publish it shortly after the central repository is updated. To resolve this conflict, the code would need to be adapted with the updated project in mind.

The prioritization of the central repository is used to keep the history of the program linear, making it easier to identify the origin of any bugs. However, because the number of merge conflicts will increase with the number of developers, the Centralized Workflow is best for relatively small developer teams. Many other Git workflows are modified versions of the Centralized Workflow. Some, such as the Feature Branch Workflow, which divides each feature of a program into different sections, still utilize a central repository.

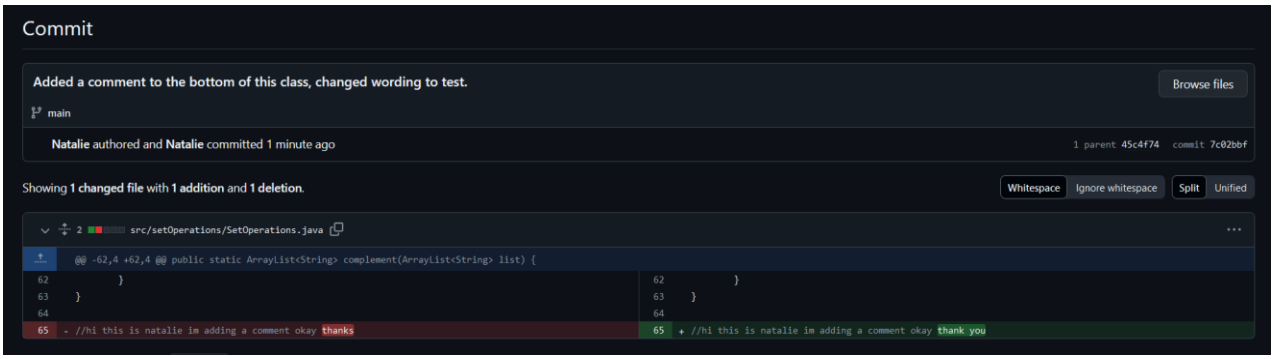
There are plenty of alternative workflows that place less emphasis on a central repository, or do not use them at all. Gitflow, for example, is a feature branch-style workflow that determines a purpose for each branch and how it should interact with any others. Once the project is completed, each branch can then be merged into the central repository. A Forking Workflow, on the other hand, involves each developer using both their local repository and an individual, public repository on the project's server. Branching is most useful for organizing complex, multifaceted programs. Generally, it is beneficial to keep branches separate from the main project for as little time as possible. As programming is carried out on isolated branches, the possibility for merge conflicts increases.

Using temporary branches is important for any Git workflow. Similarly, an effective workflow will minimize the need to revert the program, but still allow for reverts to be executed without much issue. There is no singular perfect Git workflow, as each team and project will have different needs. However, the standardization of the development process that comes from using a workflow makes them essential tools for any large-scale projects.

Works Cited

"Git Workflow: Atlassian Git Tutorial." Atlassian, 2023, www.atlassian.com/git/tutorials/comparing-workflows.

Example of a GitHub Commit Message



The above screenshot shows a GitHub commit to a classmate's repository on September 16, 2024. (The committing comment was edited due to screenshot errors.)