Luis Morales

GitHub Essay

The Centralized workflow "…uses a central repository to serve as the single point-of-entry for all changes to the project." What it means is that it essentially only has one folder called main where are the changes are pushed into. It is similar to SVN, but each person gets their own local copy of main and in their local copy, they can make as many changes as they want and upload their work to the main folder whenever they want. To get started, the host has to create central repository. Next, the developers create a local copy of the project by using the git clone command. After creating the local copy, git adds a shortcut called "origin" which points to the main folder. While in their local copy, the developer can many any changes they want and commit. After the changes have been committed, the changes can be shared with the other developers by using the git push command. If the push command has been previously used and the code conflicts with the new changes, git will output a message indicating there is a problem and will for git pull to be executed first. The developer has to fix the conflicts and push again after the conflicts have been resolved. There are other workflows such as the Feature Branching where all of the development takes place in a specific branch and the main branch contains the final code or code that is not broken. Another type of workflow is called the Gitflow Workflow where it "assigns very specific roles to different branches and defines how and when they should interact." The last one is the Forking Workflow where it assigns every developer a server-side repository. The developers end up having two repositories, one is a private server-side repository and the other is a public-side repository. One important command is the pull command. The pull command downloads the content from a repository and updates the local repository to match it. This command is basically a combination of the fetch and the merge command. After the pull command is called, it executes the fetch command which looks at the HEAD of the local repository. Once the content is downloaded, it calls the merge command which creates a new merge commit and the HEAD points to the at the new commit. Another interesting feature is the pull request however this feature requires the use of Bitbucket. The pull request notifies the other members that a feature has been completed and allows the others to check for any problems and they are able to comment on the problems and push "follow-up commits". GitHub has two commands that are frequently used and those are the add and commit command. Git add saves the changes and similarly, commit "…captures a snapshot of the projects' currently staged changes. Both commands are usually used together to save the changes locally. The merge command is often used to combine two branches. The way it works is that it takes two commit pointers and finds a common base between the two and creates a new merge commit and the new branch merges with the main branch. Usually, the merge command resolves the conflicts when it merges two branches however, sometimes a conflict can happen when two people edit a line or a developer deletes a file while someone is making some changes. There are two types of merge conflicts, one is when Git fails to start the merge and when Git fails during the merge. If Git fails to start the merge, it usually means that there are conflicts with "pending local changes." If Git fails during the merge, it usually means that there is a conflict between the local branch and the other merging branch. Git notifies when a conflict has occurred and to get more information about the conflict, the status command outputs more information. The best way to resolve a

merge conflict is by editing the conflicted file. Some tools that can help fix the conflict are the status command, the log --merge command, diff command, checkout command, reset command, rest --mixed command and the merge abort command.

Works Cited

https://www.atlassian.com/git/tutorials/comparing-workflows