Arek Ohanian

The article first talks about how much Git is used within the industry as it is the standard version control system and is used in most professional settings. Then it talks about what a git workflow is which is a recipe or recommendation on how to use git to accomplish work in a consistent and productive way. You can use it to manage your work in many different ways. You should make sure with your team on what workflow your project will use so that everyone is on the same page when working on it. Workflows are guidelines and not strict rules so they are modifiable depending on the work being done, so you can pick and choose from multiple different workflows if the project calls for it. What makes a good workflow is if it is flexible so you can undo any mistakes, if it is scalable to different sizes of teams and project size, and if it is easy to understand as to not make the work more complicated this is to help not hurt. The first workflow mentioned is the centralized workflow. Which is where there is a central repository of the main repository as the single point of entry for everyone. Then all work is done to the main repository or a clone of it on their respective computers. Then it is merged with the main repository. This also allows us to go back to a previous version in case a commit or push breaks something. Then everyone makes a clone of that repository and starts to work on the project. The non-cloned repository or the origin as it is called is not affected by any work done to the clones. At least until someone pushes their local clone to the origin. When you try to push to the origin initially you can commit everything or just specific parts of code that are not the same as the origin, but they are not pushed yet to the main code. From the commit you can push to the origin updating any changes you committed. Though if someone else has also committed something before you need to pull from the origin taking the updated code first that someone else pushed so that you can push your code. This is to make sure that any code is not lost or deleted in the push. When you pull there may be conflicts in the code. These are lines of code that you pull that do not match up with your local code. You need to get rid of any merge conflicts before you can push to the main branch. With that git also has version control so say for example I get a merge conflict and instead of handling it normally I just delete everything that was conflicting with my code and commit it. The other project members can revert that push to the one before it so that their code isn’t deleted just from my single change. This is all for the centralized workflow which is the basis for all other git workflows. For example, there is the Feature Branching workflow which is where all features are done withing their own dedicated branch which all act as mini origins for their respective features then at the end they are all combined and merged into the final project at the end. But this allows for the groups to be separated instead of all pulling for things that might not matter to what they are working on. There is GitFlow workflow which is where it assigns specific roles to each of the branches and defines how and when they should interact with each other. There is also the Forking Workflow which adds forking a repository which is where you create a new origin for your fork of the project creating a local private repository and a public server one to work with. Ultimately there is no one size fits all for git workflows, so try to mix and match from each of these or others that you may know to help with creating a workflow to make sure your project is organized and efficient. Though some guidelines in the article are to make sure that branches aren’t too long as the longer they are worked on the higher chance of big merge conflicts happening which can be a huge hassle. Minimize the amount of reverts you have to do because otherwise you will be going back and forth on your project slowing down the speed of completion. Finally, to match a release schedule to make sure everything is done one time.