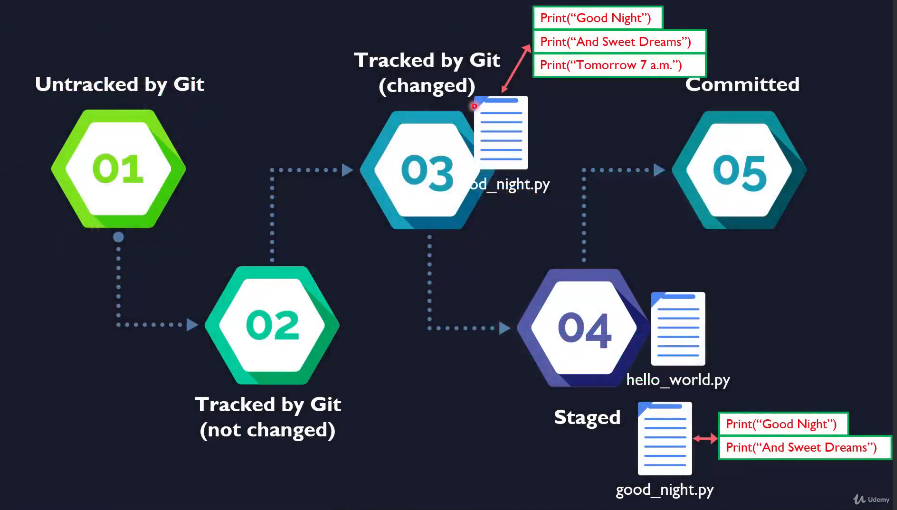
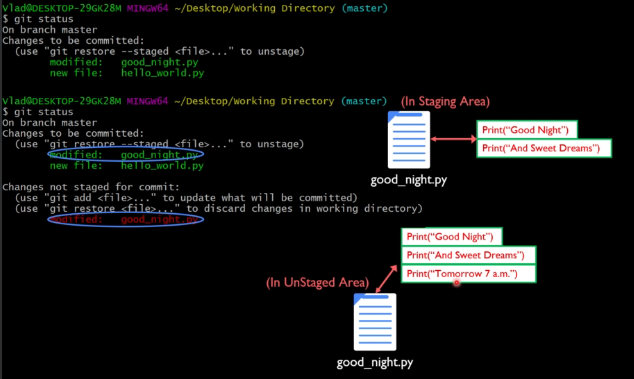


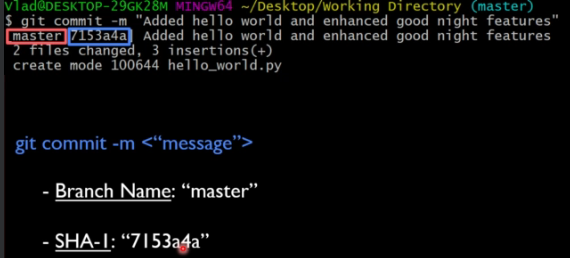
Working directory and /git folder are different things. /git is used for version control and track files in working directory.



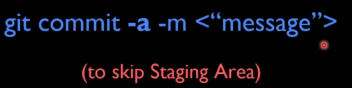


Above, you can first use add and then do commit to have additional change committed.





Below command commits all the modified tracked files.

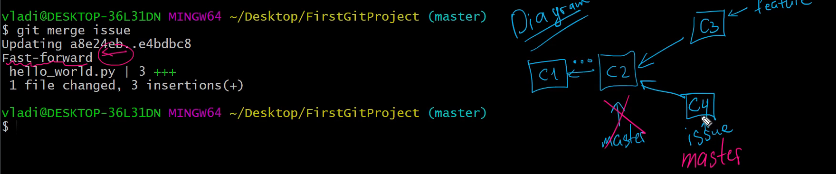


Below is a command to both create a branch and check it out.



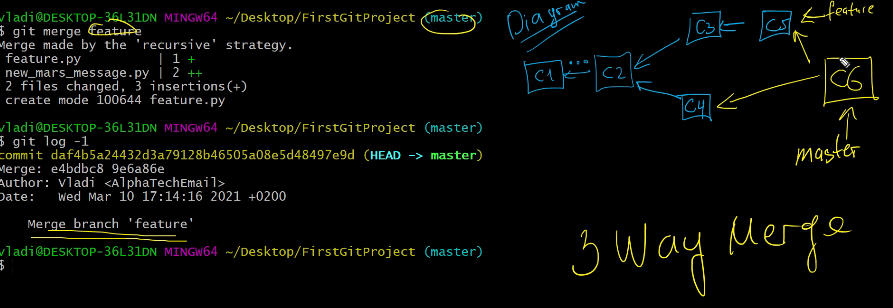
When you change branches using ‘checkout’, it will not allow if there are conflicts for any of the files.

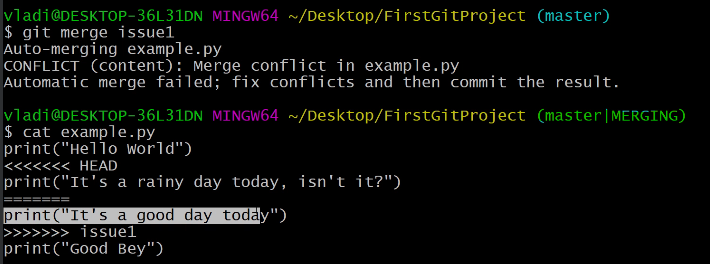
When Git detects that your commit is about to be merged into your project's main branch without the main branch having been modified since your feature branch was first made, it chooses to use a **fast-forward merge** instead of a three-way merge. Fast-forward merges literally move your main branch's tip forward to the [end of your feature branch](https://www.bogotobogo.com/cplusplus/Git/Git_GitHub_Fast-Forward_Merge.php). This keeps all commits created in your feature branch sequential while integrating it neatly back into your main branch.





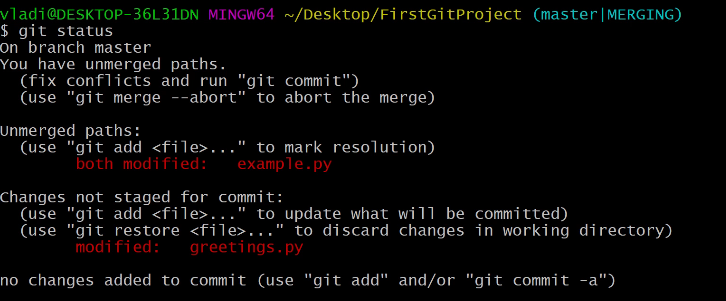
Below is **recursive merging (3-way merge)**. It makes a new commit.



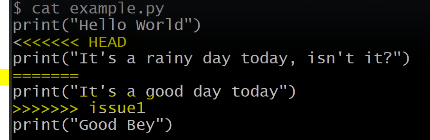


Conflicts can occur at start (which is trivial – happens when working tree is not clear and pending local changes – also called pending local changes conflict) or during merging (example above – also called failure during merge process).

Run below to see which files have not merged.



GIT adds standard conflict markers. Highlighted lines are conflict dividers.



You can now open the file and resolve the conflict. Then add and commit.

