What is GitHub? When was it created? Why? By who? What similar platforms exist? Why would you use such a platform?

Answer:

Git is a command-line tool, but the center around which all things involving Git revolve – effectively, the Hub, is GitHub.com, where developers can store their projects and network with likeminded people.

GitHub was co-founded by Thomas Preston-Werner, Chris Wanstrath and PJ Hyett.

BitBucket.com, GitLab, SVN (Subversion Repository) are similar web-based repository hosting services.

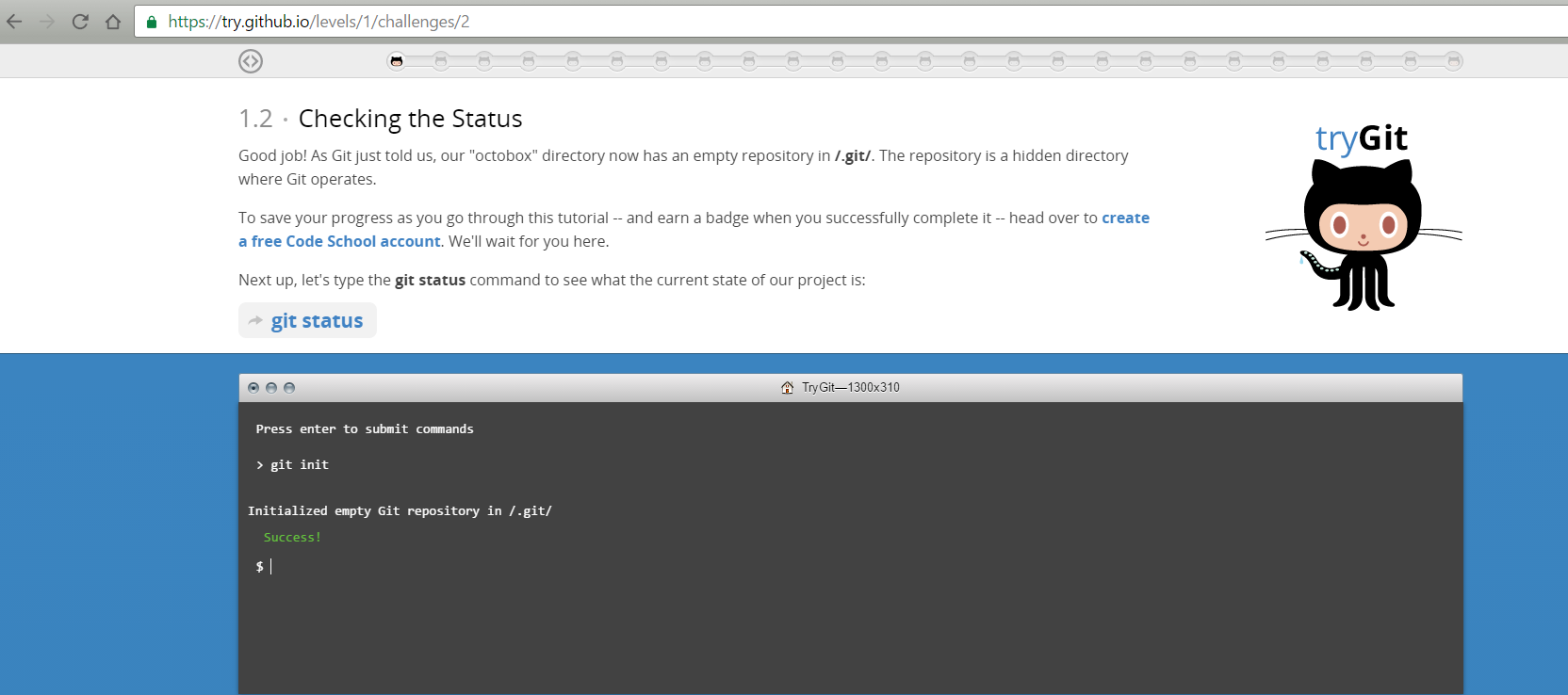
There are few reasons why we should GitHub:

1. The world is moving towards open source.
2. Our team can learn from each other’s work.
3. We can request other developers to contribute their knowledge if needed.

Tutorial steps:

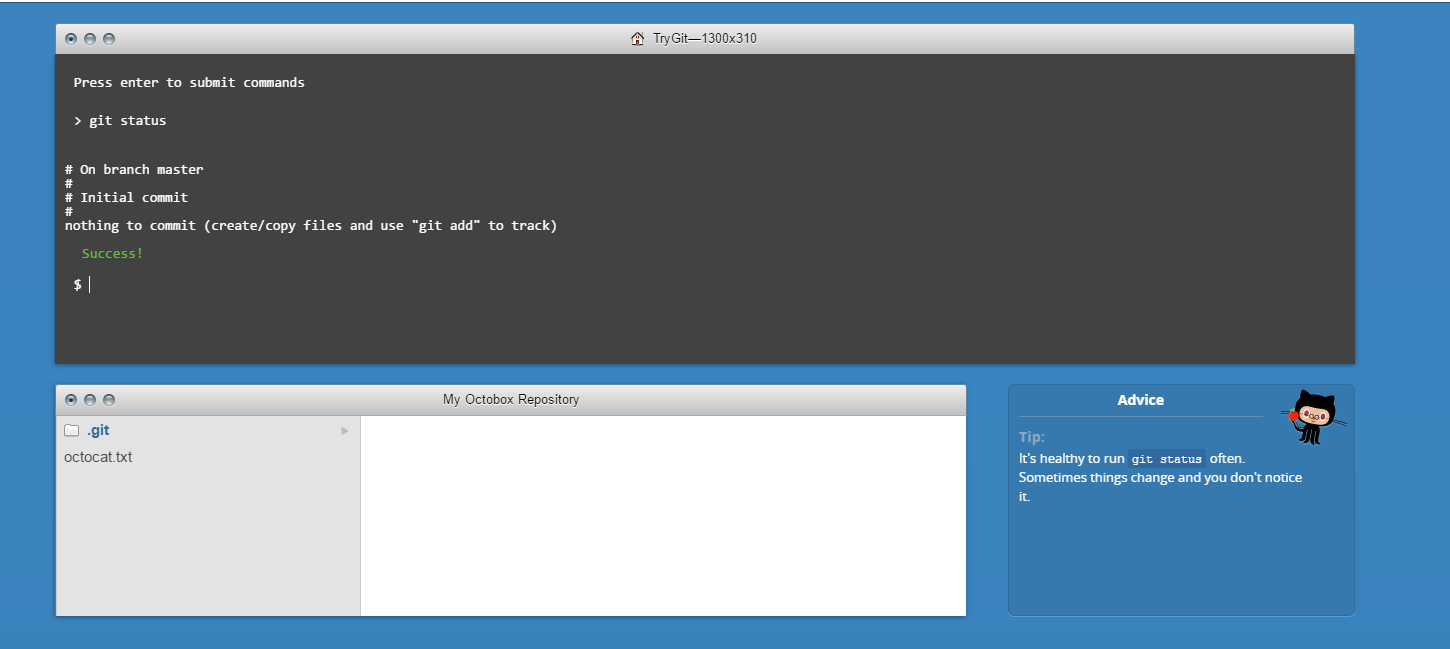
Step 1:

Initializing Git



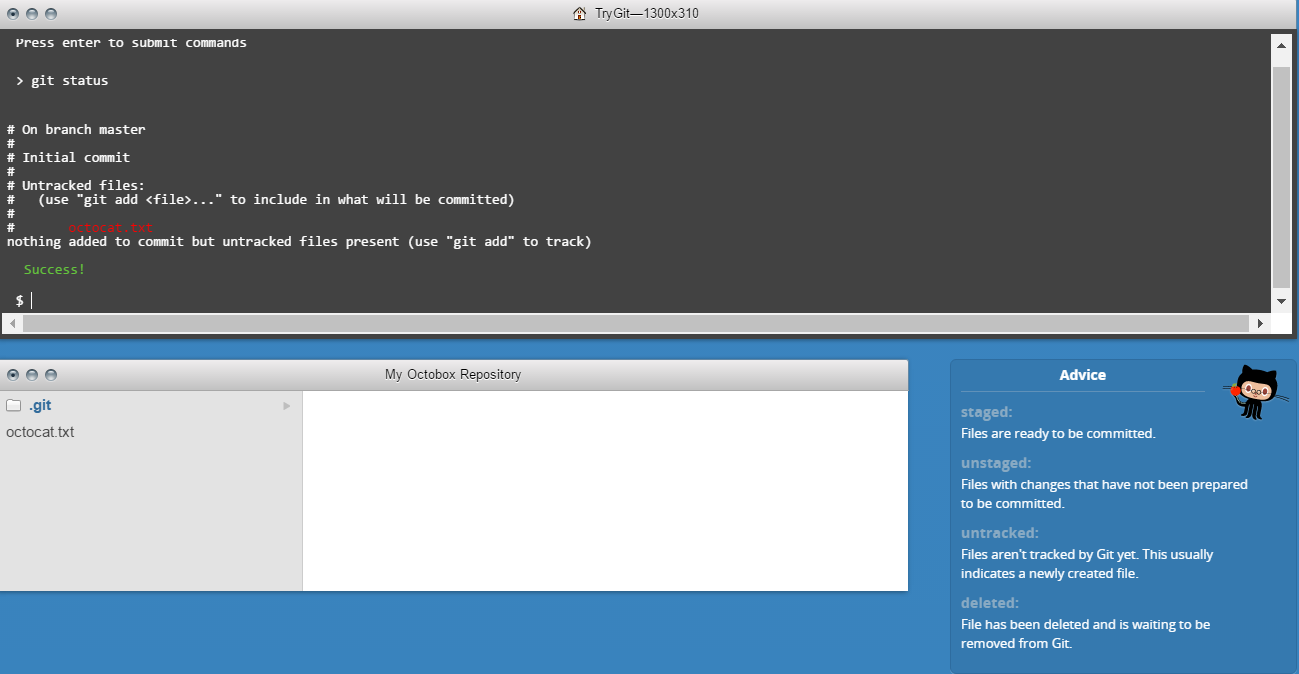
Step 2:

Checking the status



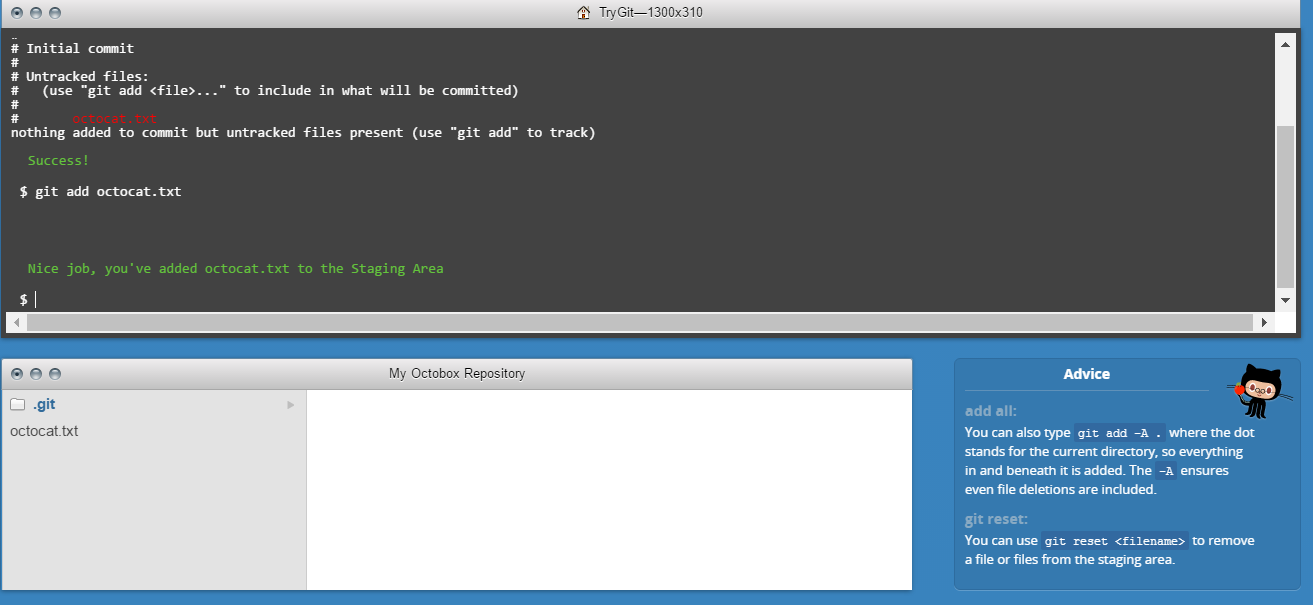
Step 3:

Adding and committing



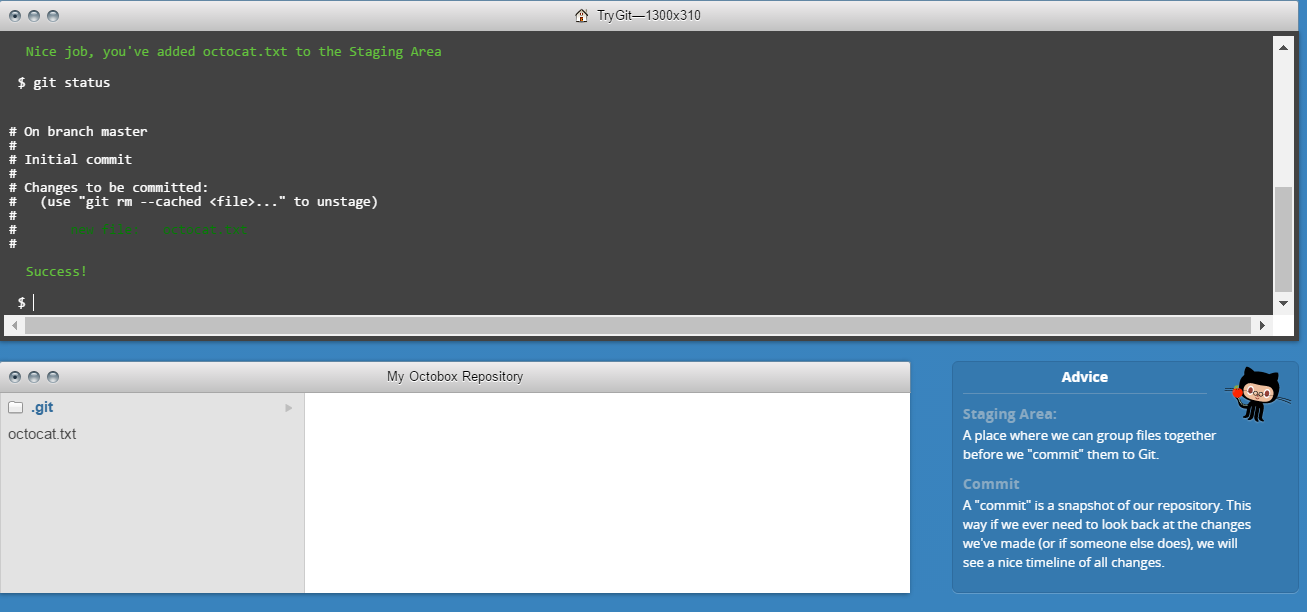
Step 4:

Adding Changes



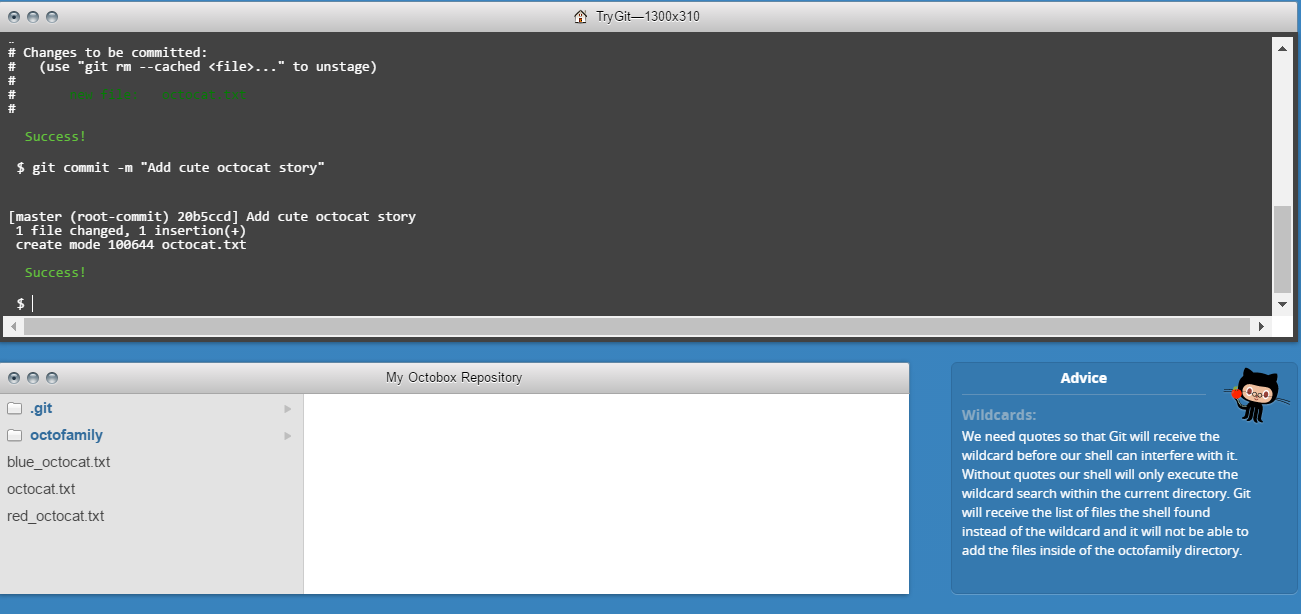
Step 5:

Checking for changes



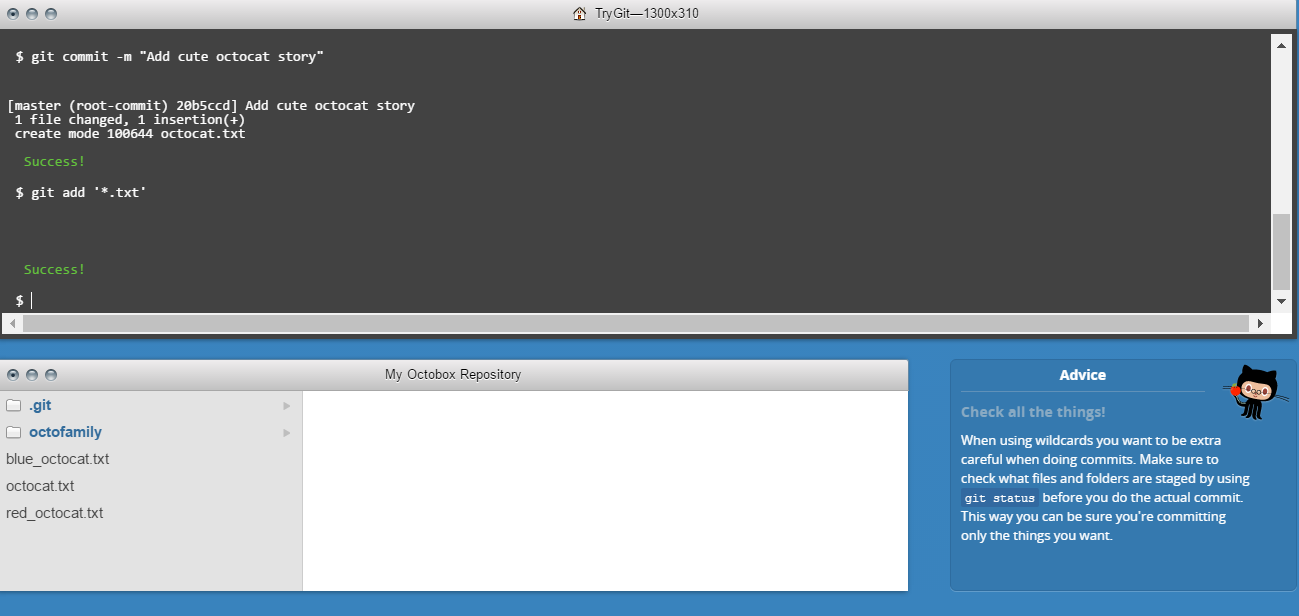
Step 6:

Committing



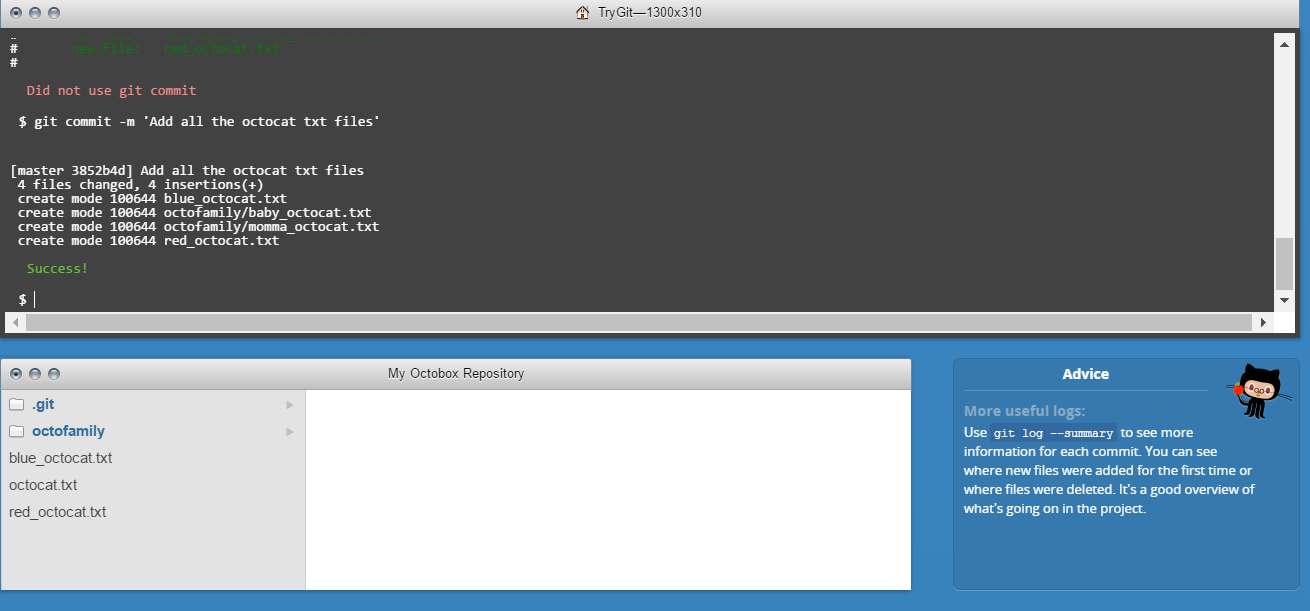
Step 7:

Adding all changes



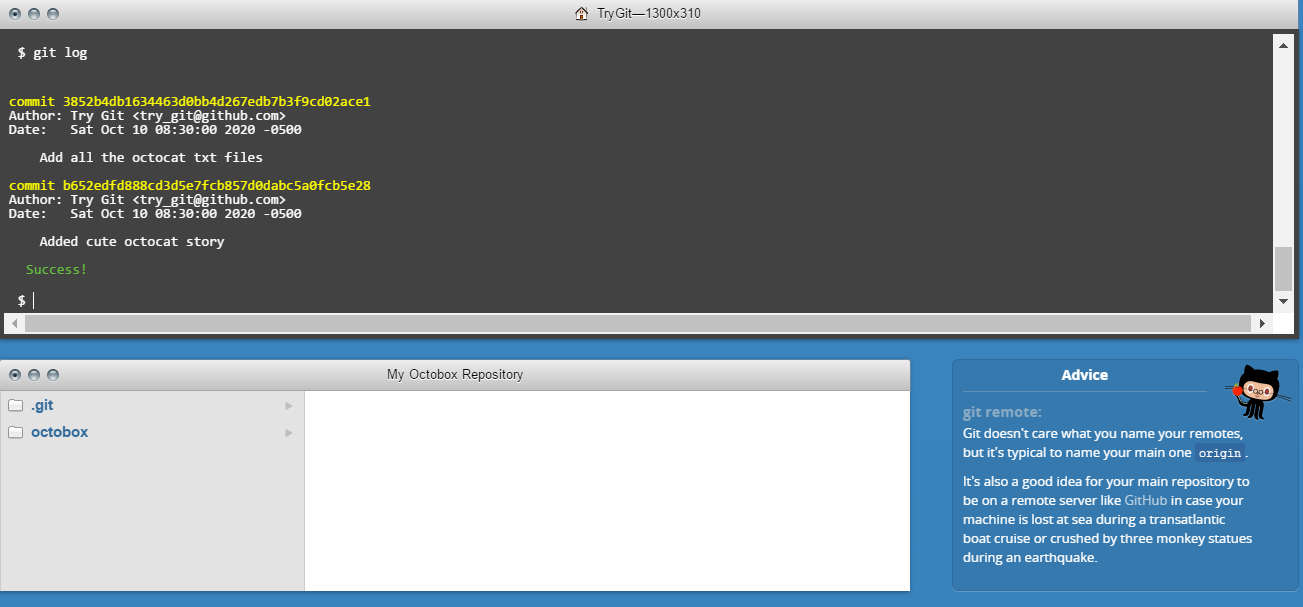
Step 8:

Committing all changes



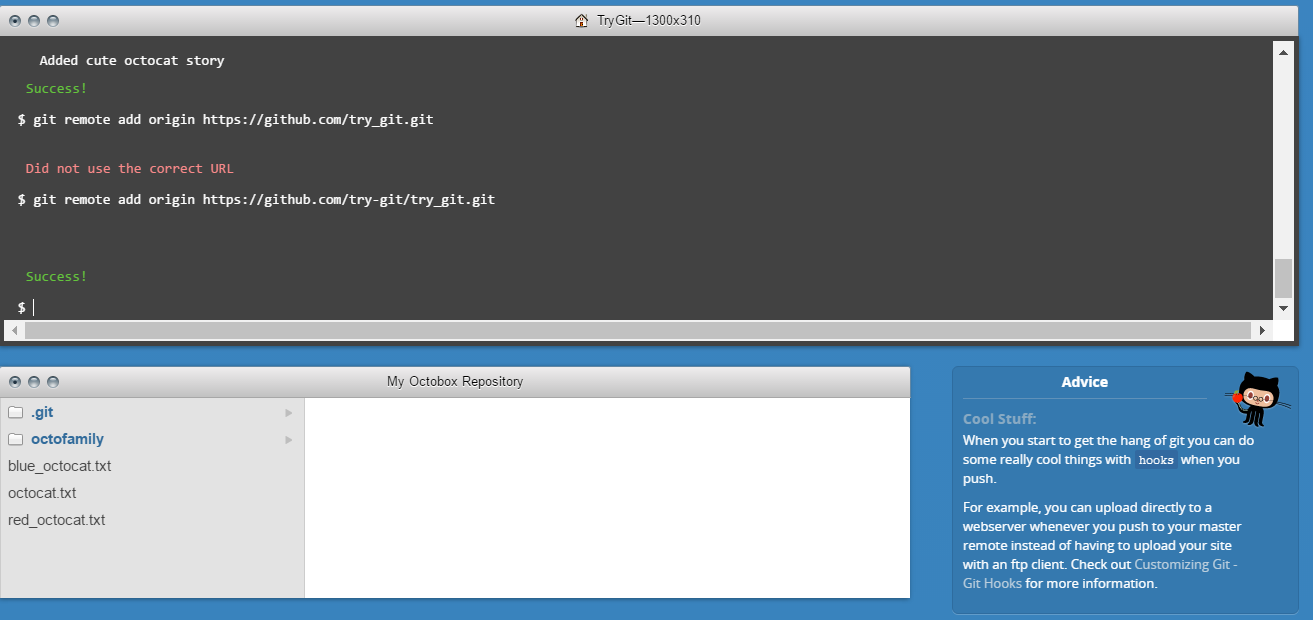
Step 9:

History



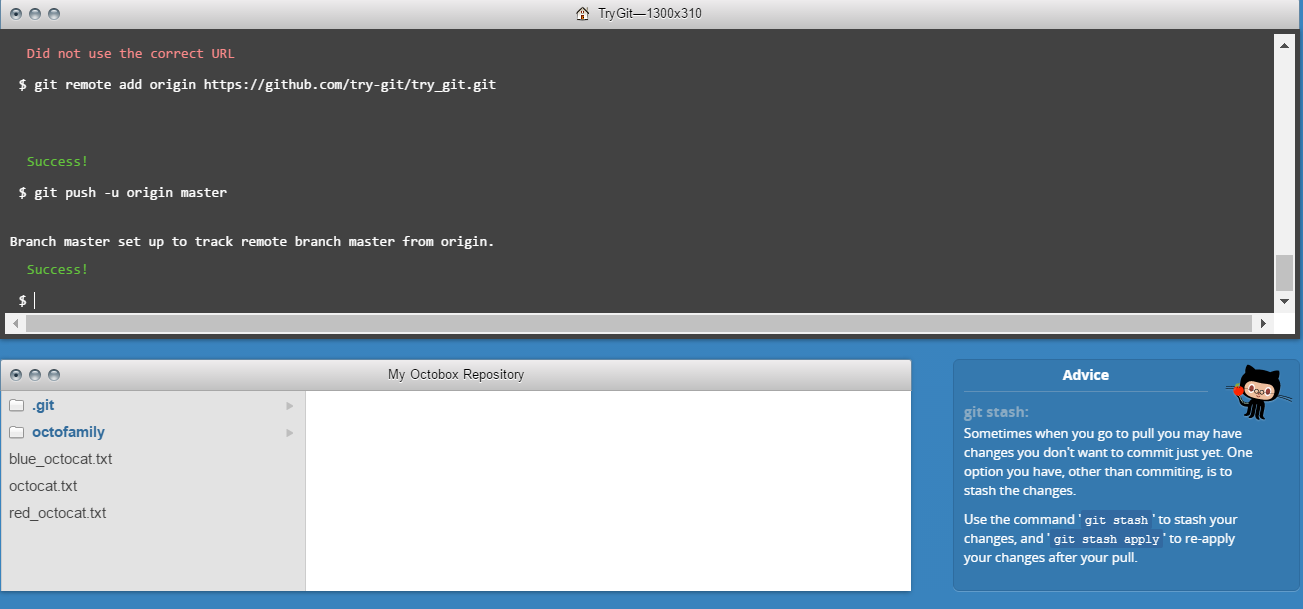
Step 10:

Remote Repositories



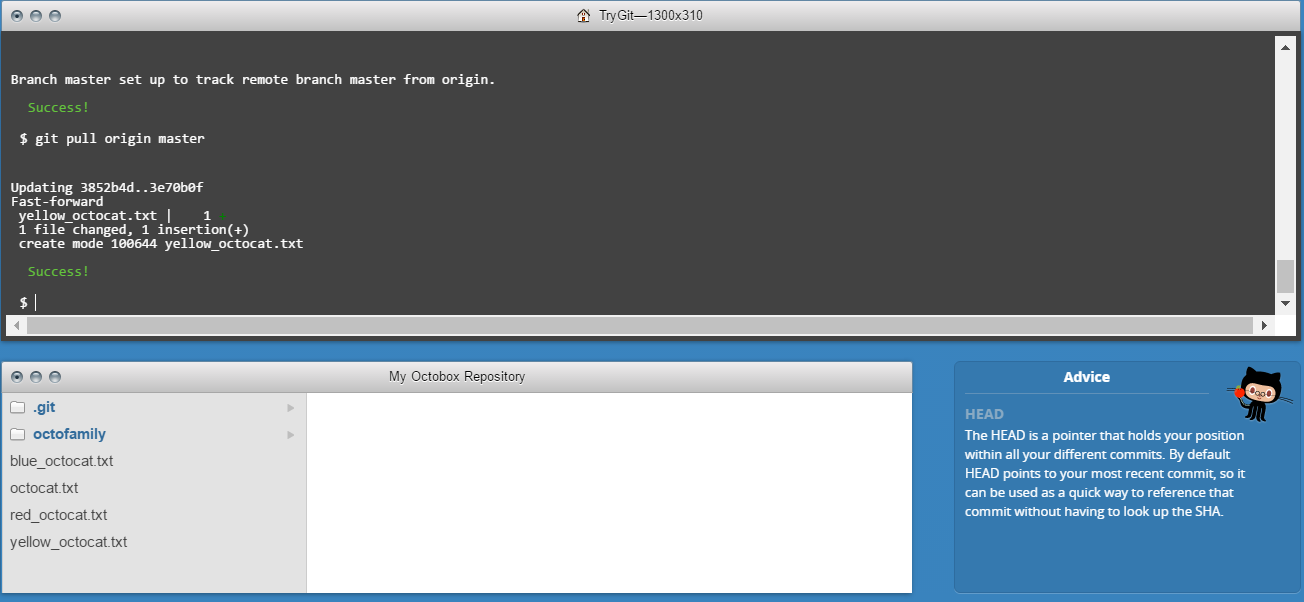
Step 11:

Pushing Remotely



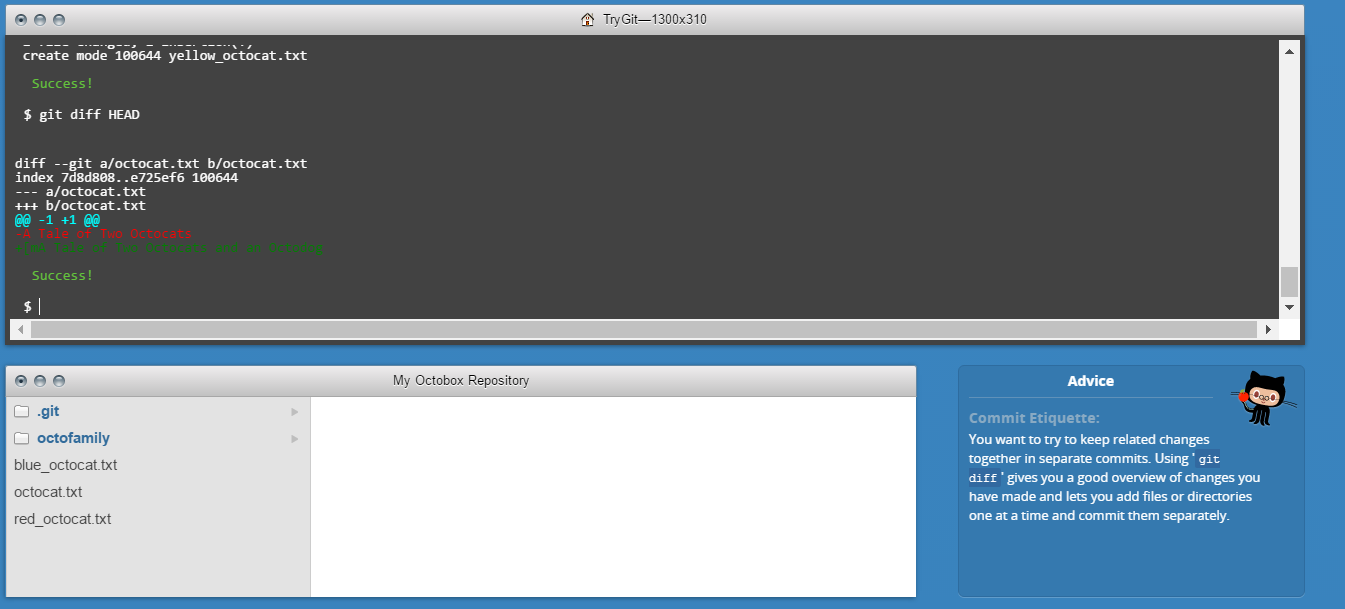
Step 12:

Pulling Remotely



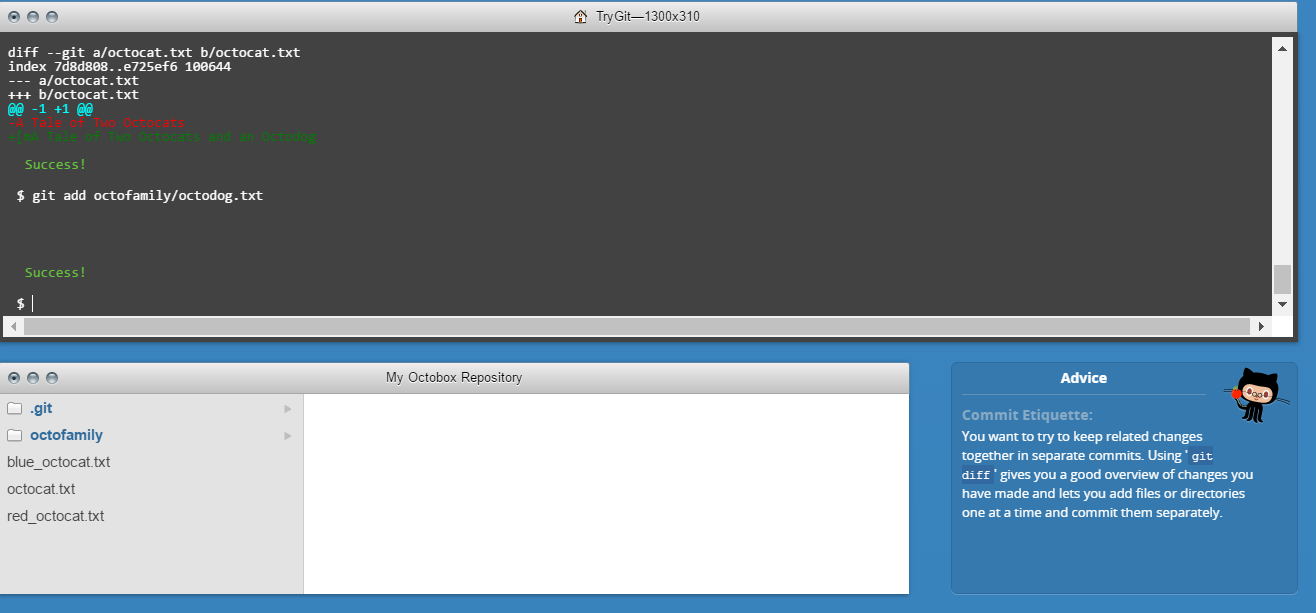
Step 13:

Differences



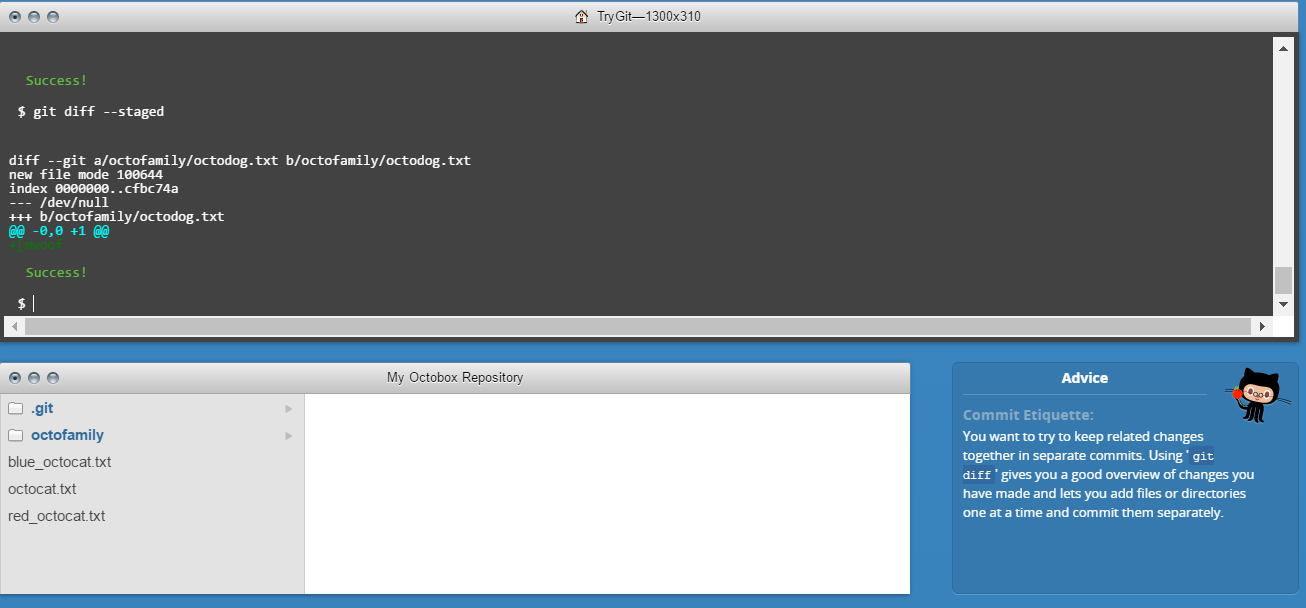
Step 16:

Staged Difference



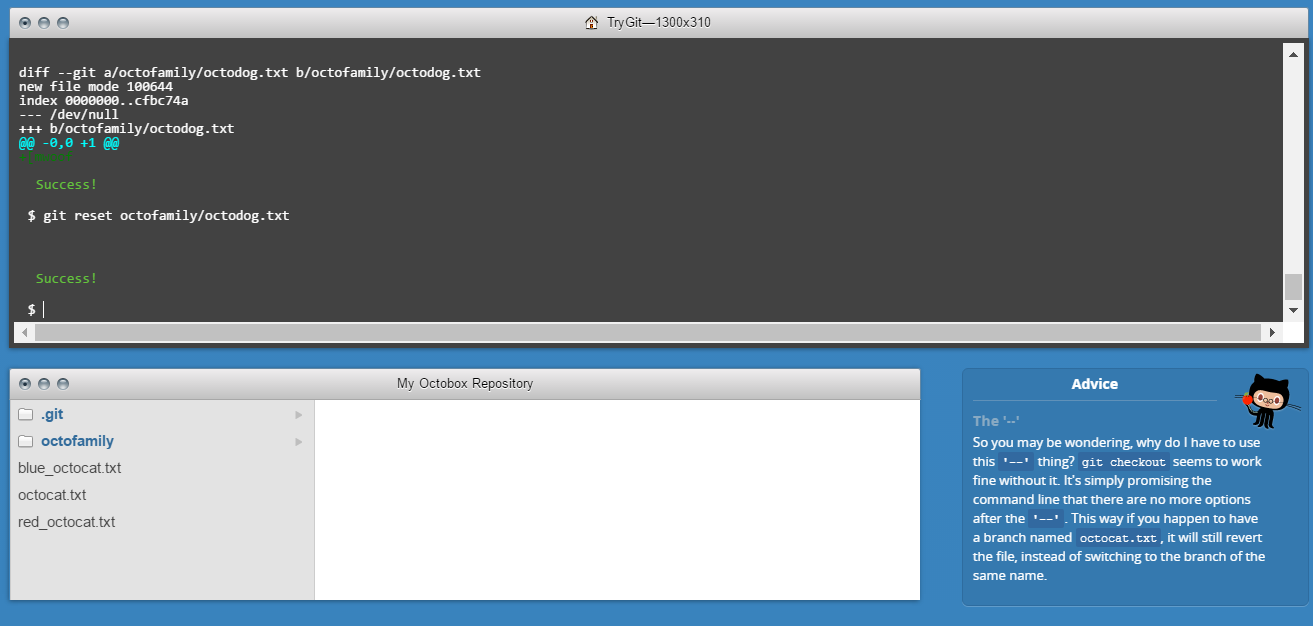
Step 17:

Staged Difference (Cont’d)



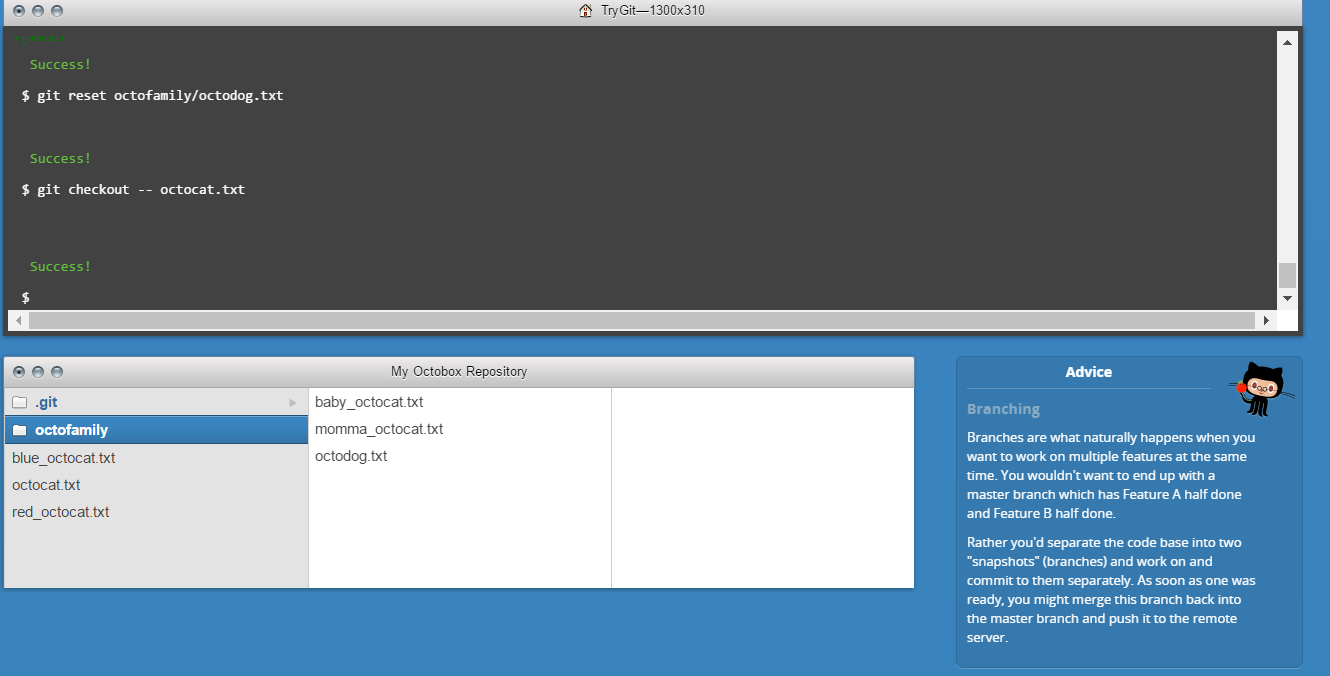
Step 18:

Resetting the stage



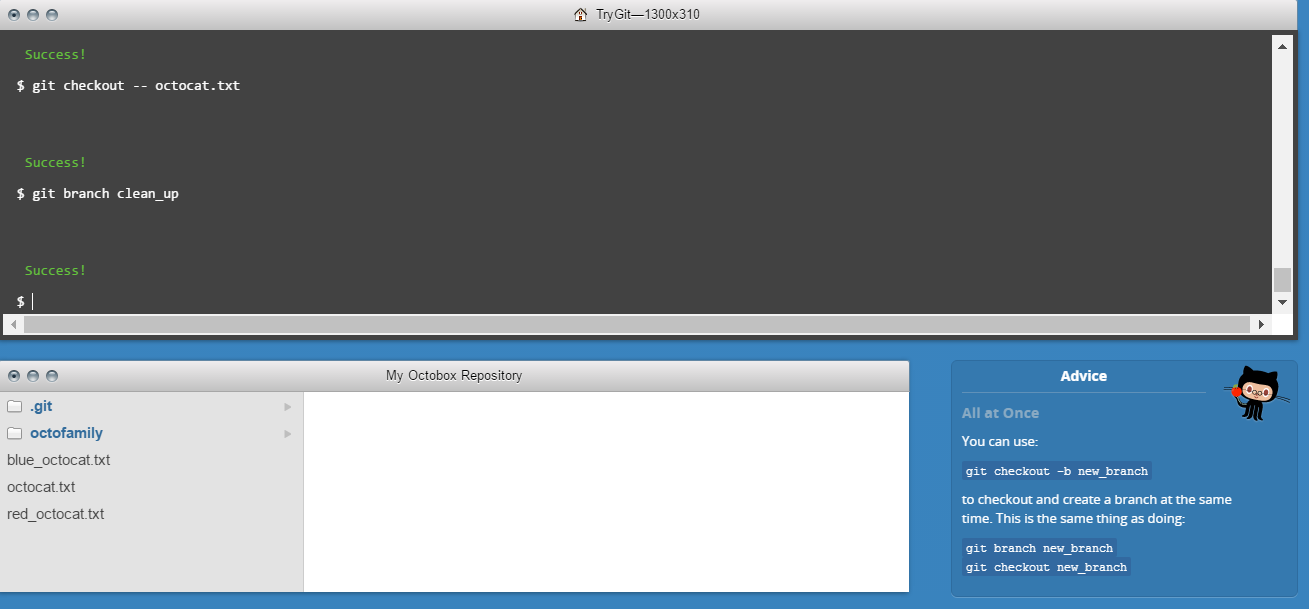
Step 19:

Undo



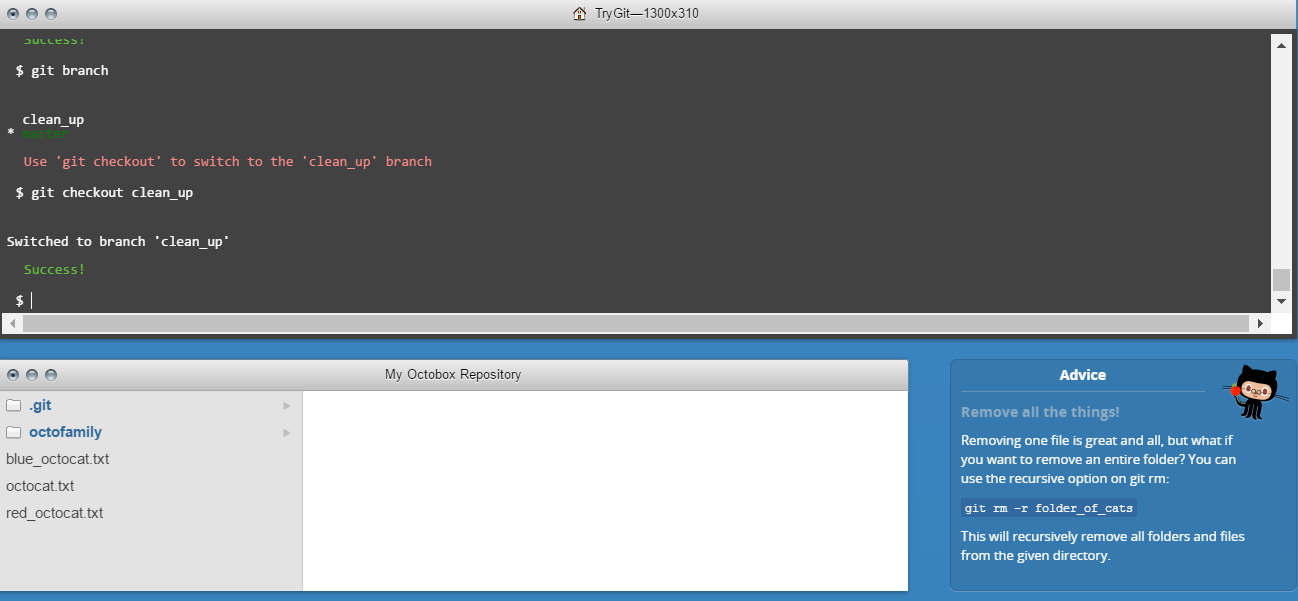
Step 20:

Branching out



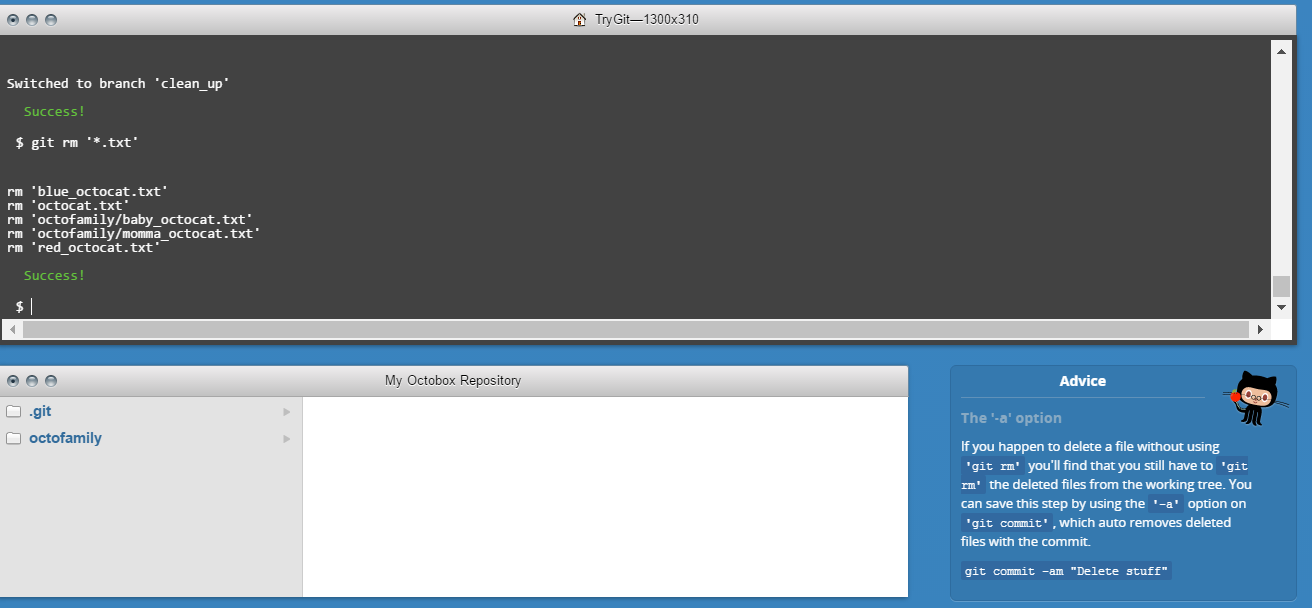
Step 21:

Switching Branches



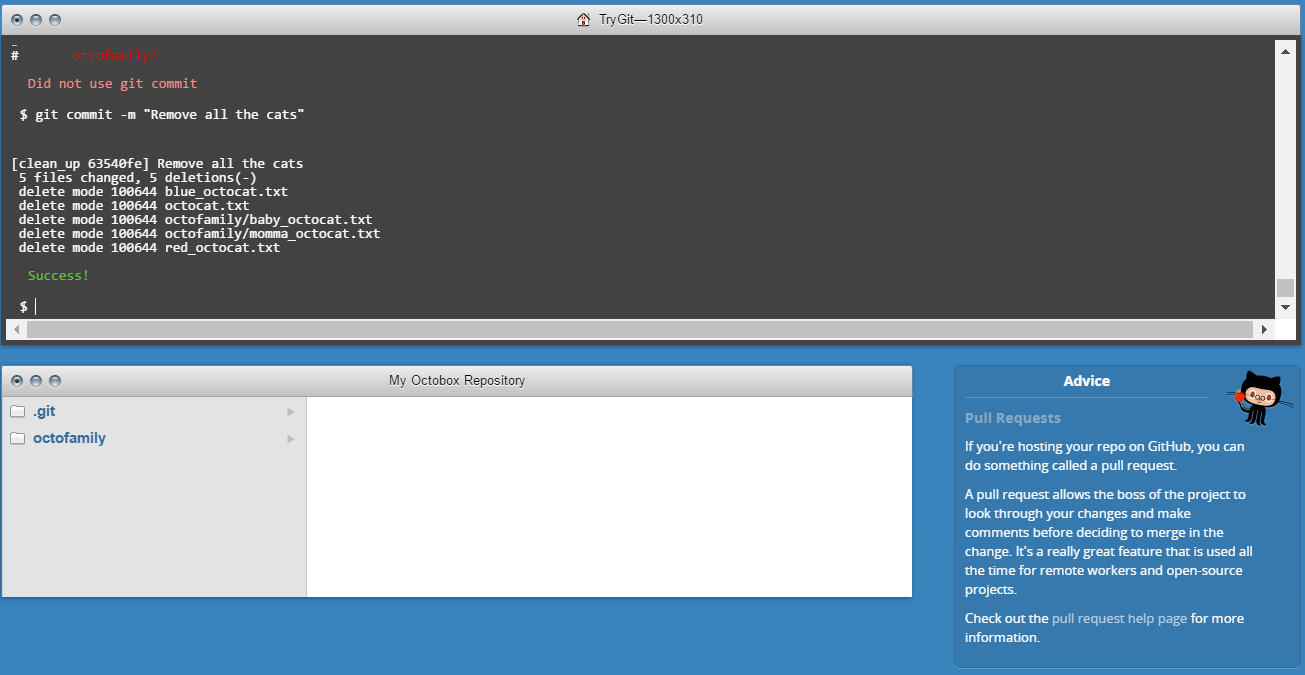
Step 22:

Removing all things



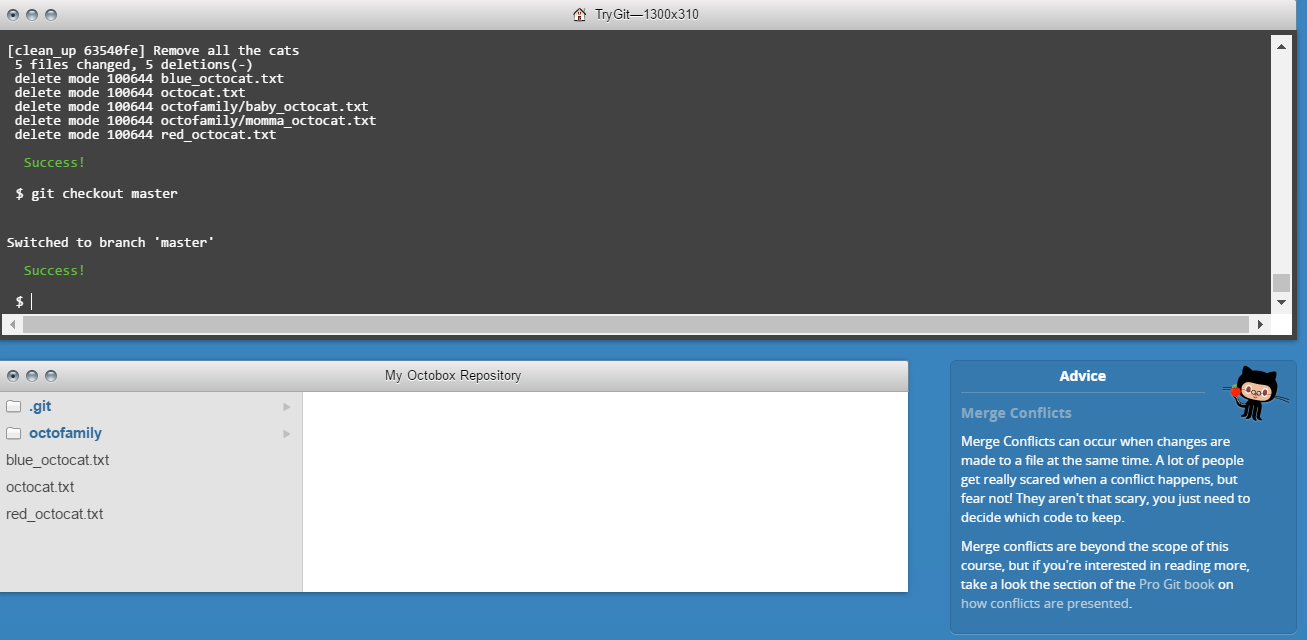
Step 23:

Committing Branch Changes



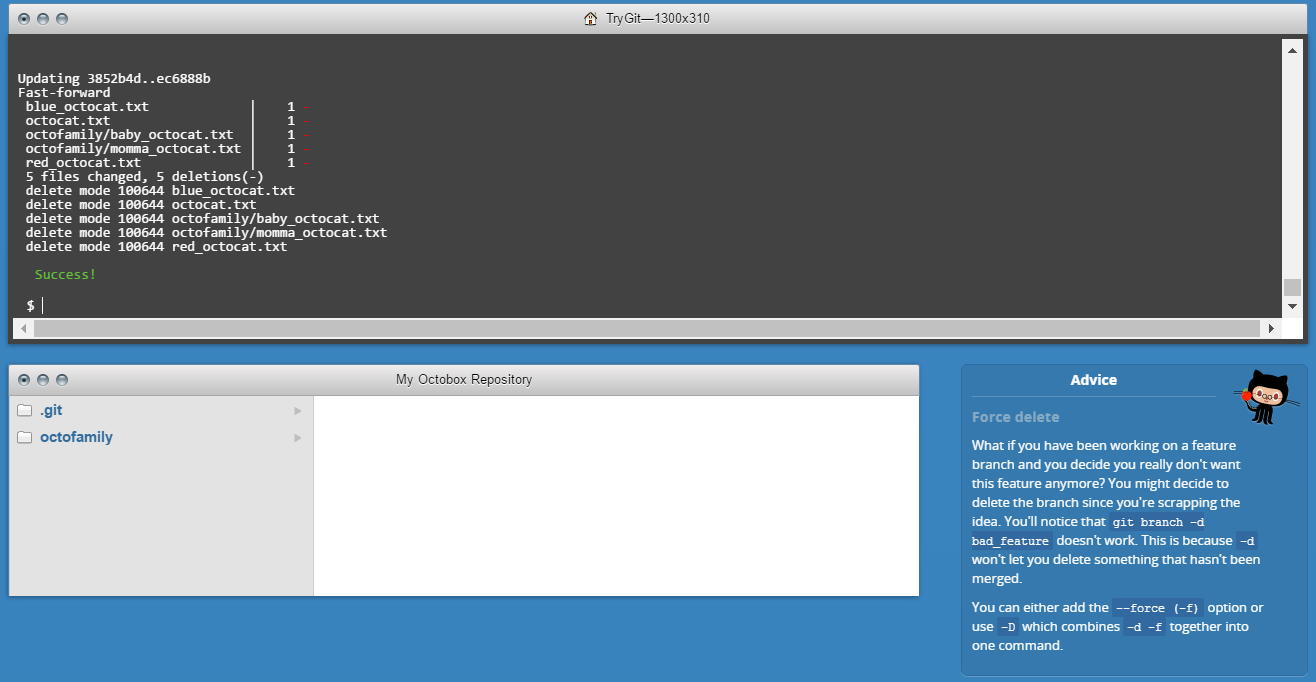
Step 24:

Switching back to master



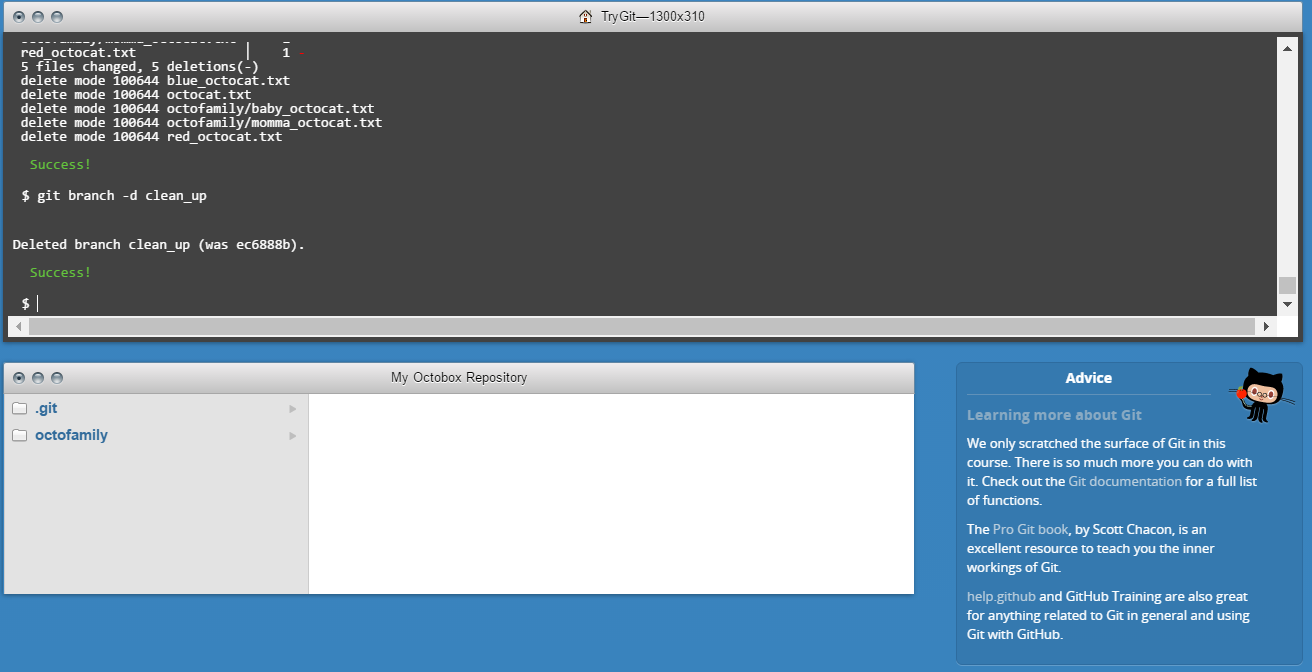
Step 25:

Preparing to merge



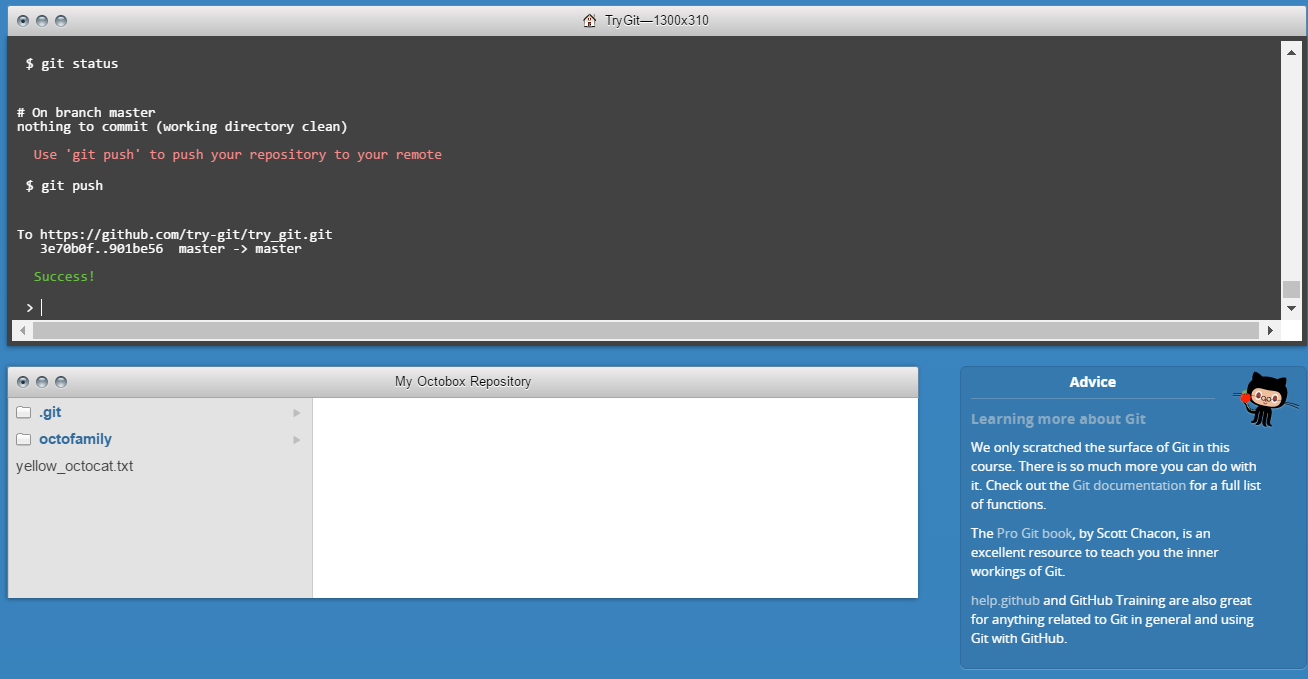
Step 26:

Keeping things clean



Step 27:

The final push



Define the following terms in the context of Git (2 lines maximum):

Answer:

1. Repository: A repository is a central place in which information is aggregated and is maintained in a organized way.
2. Commit: Commits the tracked changes and prepares the data to be pushed to a remote repository. Makes sure the changes made to the repo are saved.
3. Push: Pushes the changes into your local repositories up to the remote repository you specified as the origin.
4. Branch: It is a lightweight movable pointer to all the commits performed while developing a project. The main branch is called as the Master Branch and a clone can also be created where all the bug fixes and features can be tested.
5. Fork: It is a copy of repository. If you want to make changes in repositories without effecting the master repo, you can create a copy using fork to do the same.
6. Merge: If you think that the bug fixes or features applied on a branch are up and running without any warnings or errors, you can merge it with the master. i.e. Applying changes to master branch.
7. Clone: Cloning a repo can be called as, “Creating a copy of a repo provided by developer”. By doing this, anyone can make changes on the clone and check for desired output.
8. Pull: Incorporates changes from a remote repo into the current branch. In its default mode, git pull is short hand for *git fetch*.
9. Pull request: Pull request let others know about the changes you made to a GitHub repo. A revision history can be checked via a Pull request.

**Part 7:**

1. I created a **fork** of the file in **mypseudo/courses** to make certain changes in the file without modifying the master file.
2. I changed the file in my fork to update changes.
3. Later, I **pulled** **request** to update it in the master file.
4. Changes can be only seen when the owner of the master file **merges** it.
5. When the pull request was merged, the changes are visible in the master file.