Git Basic Commands

1. **Git init**

   Suppose developer needs to prepare one new repository in his local environment. For initializing the same they have to use init commands, which helps them for initializing GitHub project repository locally.

2. **Git clone**

   Suppose developer needs to create one specific repository of GITHUB in their local copy from the specific remote location. Then they have to execute clone command for copying the same remote repository in the local environment in specific location.

1. **Git status**

   This GitHub command is mainly used for identifying GIT created status in local repository. It provides proper information between a syncing status of local and GITHUB repository to the end user.

2. **Git add [file_name.doc]:**

   Used for adding one specific file in staging area.

3. **Git add –A**

   Adding all the files whether it is new or modified will be added in the staging area.

4. **Git commits –m [“message for commit”]:**

   Commit all the required changes.

5. **Git rm –r [file_name.doc]:**

   Helping for removing the file or any defined folder.

To remove only from the staging area

```
git rm –cached “ file name”
```

Intermediate Commands

There are several other popular commands which are used by the developer, which are not very basic commands but working with GITHUB more, those commands are very much required to execute. Some of these kinds require intermediate commands which are mentioned below:

1. **Git branch**

   This is very much a useful command for using any kind of GITHUB set up for your projects. It normally provides entire branch information in a list. Asterisk denotation is also given which points the exact current project.

2. **Git branch -a**

   This is helping for listing all the available branches considering all the locations of the branches, it can be local or remote. It is one of the key commands for identifying all the available projects in GITHUB environment.

3. **Git branch [branch_name]**

   Creating a new branch with new name.

4. **Git branch –d [branch_name]**

   Deleting a specific branch.

5. **Git checkout -b [branch_name]**

   Helping to create one specific new branch and switching entire codebase to it.

6. **Git checkout [branch_name]**

   Switching to a defined branch.

7. **Git merge [branch_name]**

Helping for merging one specific branch with one of the other activities available branches.

8. **Git merge [source_branch] [target_branch]**

Helping for merging with specified sourcing branch and specified target branch.

Advanced Commands

Still, some of the critical tasks need to be done by GitHub users frequently. Those tasks also have some advance kind of commands which needs to be executed, and which are mainly used by some of the managerial people for understanding projects status comparing other existing project and also sometimes used for sharing some project knowledge with others. These advanced commands are mentioned below:

1. **Git push origin [branch_name]**

   Helps for pushing one of the created branches in your local environment to a remote directory or repository.

2. **Git push –u origin [branch_name]**

   It does a similar task like above, pushing entire local repository branch to the remote repository, extra utility of using this command is also remembered for this specific branch for future reference.

3. **Git push**

   Again it does the same task, pushing entire local repository data to remote, but do not need to mention the branch name, it will pick up branch name directly from the last remembered branch.

4. **Git pull**

   Helping for updating all the newest commit in the mapping local repository.

5. **Git remote add origin**

   Helping for mapping local repository's origin branch with remote repository through SSH.

Commands for Comparison or specific inspection on GITHUB repository:

1. **Git log**

   Displaying all the log information of one specific project, which can provide every information with committed changes.

2. **Git diff [source_branch] [target_branch]**

   This one normally is used before merging on a specific document. It helps for displaying current changes done on the existing document.