**Getting & Creating Projects**

| **Command** | **Description** |
| --- | --- |
| git config --global user.name "Sam Smith"  git config --global user.email sam@example.com | Configure the author name and email address to be used with your commits. Note that Git [strips some characters](http://stackoverflow.com/questions/26159274/is-it-possible-to-have-a-trailing-period-in-user-name-in-git/26219423#26219423) (for example trailing periods) from user.name.  [**Tell Git who you are**](https://www.atlassian.com/git/tutorials/setting-up-a-repository/git-config) |
| git init | Initialize a local Git repository  [**Create a new local repository**](https://www.atlassian.com/git/tutorials/setting-up-a-repository/git-init) |
| git clone ssh://git@github.com/[username]/[repository-name].git  git clone username@host:/path/to/repository | Create a local copy of a remote repository  [**Check out a repository**](https://www.atlassian.com/git/tutorials/setting-up-a-repository/git-clone) |

**Basic Snapshotting**

| **Command** | **Description** |
| --- | --- |
| git status | Check status  [**Status**](https://www.atlassian.com/git/tutorials/inspecting-a-repository#git-status) |
| git add [file-name.txt] | Add a file to the staging area  [**Add files**](https://www.atlassian.com/git/tutorials/saving-changes#git-add) |
| git add -A | Add all new and changed files to the staging area |
| git commit -m "[commit message]"  git commit -a | Commit changes  [**Commit**](https://www.atlassian.com/git/tutorials/saving-changes#git-commit) |
| git rm -r [file-name.txt] | Remove a file (or folder) |

**Branching & Merging**

| **Command** | **Description** |
| --- | --- |
| git branch | List branches (the asterisk denotes the current branch)  [**Branches**](https://www.atlassian.com/git/tutorials/using-branches) |
| git branch -a | List all branches (local and remote) |
| git branch [branch name] | Create a new branch |
| git branch -d [branch name] | Delete a branch |
| git push origin --delete [branchName] | Delete a remote branch |
| git checkout -b [branch name] | Create a new branch and switch to it |
| git checkout -b [branch name] origin/[branch name] | Clone a remote branch and switch to it |
| git checkout [branch name] | Switch to a branch |
| git checkout - | Switch to the branch last checked out |
| git checkout -- [file-name.txt] | Discard changes to a file |
| git merge [branch name] | Merge a branch into the active branch |
| git merge [source branch] [target branch] | Merge a branch into a target branch |
| git stash | Stash changes in a dirty working directory |
| git stash clear | Remove all stashed entries |

**Sharing & Updating Projects**

| **Command** | **Description** |
| --- | --- |
| git push origin [branch name] | Push a branch to your remote repository  [**Push**](https://www.atlassian.com/git/tutorials/syncing#git-push) |
| git push -u origin [branch name] | Push changes to remote repository (and remember the branch) |
| git push | Push changes to remote repository (remembered branch) |
| git push origin --delete [branch name] | Delete a remote branch |
| git pull | Update local repository to the newest commit |
| git pull origin [branch name] | Pull changes from remote repository |
| git remote add origin ssh://git@github.com/[username]/[repository-name].git | Add a remote repository |
| git remote set-url origin ssh://git@github.com/[username]/[repository-name].git | Set a repository's origin branch to SSH |

**Inspection & Comparison**

| **Command** | **Description** |
| --- | --- |
| git log | View changes |
| git log --summary | View changes (detailed) |
| git diff [source branch] [target branch] | Preview changes before merging |
| git grep "foo()" | Search the working directory for foo(): |

|  |  |  |
| --- | --- | --- |
| Git task | Notes | Git commands |
| [**Connect to a remote repository**](https://www.atlassian.com/git/tutorials/syncing#git-remote) | If you haven't connected your local repository to a remote server, add the server to be able to push to it: | git remote add origin <server> |
| List all currently configured remote repositories: | git remote -v |
| [**Branches**](https://www.atlassian.com/git/tutorials/using-branches) | Create a new branch and switch to it: | git checkout -b <branchname> |
| Switch from one branch to another: | git checkout <branchname> |
| List all the branches in your repo, and also tell you what branch you're currently in: | git branch |
| Delete the feature branch: | git branch -d <branchname> |
| Push the branch to your remote repository, so others can use it: | git push origin <branchname> |
| Push all branches to your remote repository: | git push --all origin |
| Delete a branch on your remote repository: | git push origin :<branchname> |
| [**Update from the remote repository**](https://www.atlassian.com/git/tutorials/syncing) | Fetch and merge changes on the remote server to your working directory: | git pull |
| To merge a different branch into your active branch: | git merge <branchname> |
| View all the merge conflicts:  View the conflicts against the base file:  Preview changes, before merging: | git diff  git diff --base <filename>  git diff <sourcebranch> <targetbranch> |
| After you have manually resolved any conflicts, you mark the changed file: | git add <filename> |
| **Tags** | You can use tagging to mark a significant changeset, such as a release: | git tag 1.0.0 <commitID> |
| CommitId is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using: | git log |
| Push all tags to remote repository: | git push --tags origin |
| [**Undo local changes**](https://www.atlassian.com/git/tutorials/undoing-changes) | If you mess up, you can replace the changes in your working tree with the last content in head:  Changes already added to the index, as well as new files, will be kept. | git checkout -- <filename> |
| Instead, to drop all your local changes and commits, fetch the latest history from the server and point your local master branch at it, do this: | git fetch origin  git reset --hard origin/master |

**.DS\_Store:**

Remove existing files from the repository:

find . -name .DS\_Store -print0 | xargs -0 git rm -f --ignore-unmatch

Add the line

.DS\_Store

to the file .gitignore, which can be found at the top level of your repository (or created if it isn't there already). You can do this easily with this command in the top directory

echo .DS\_Store >> .gitignore

Then

git add .gitignore

git commit -m '.DS\_Store banished!'