

**Git command cheats**

**Install**

$ apt-get install git

**Removal**

$ rm -r .git

**Configuration**

$ git config --global user.name your\_username

$ git config --global user.email your\_email address

$ git config --global credential.helper "cache --timeout=3600"

$ git config core.fileMode false # git will not track chmod

**View username and email**

$ git config user.name

$ git config user.email

**HTTPS Remote Origin**

$ git remote add origin https://github.com/user/repo.git # add remote origin

$ git remote rm origin # remove remote origin

$ git remote set-url origin https://github.com/user/repo.git # reset remote origin

$ git remote -v # view remote origins

**SSH Remote Origin**

$ git remote add production ssh://[USER]@[HOST]/PATH/TO/TARGET/DIRECTORY/

or (if using non standard SSH port):

$ git remote add production ssh://[USER]@[HOST]:[SSH\_PORT]/PATH/TO/TARGET/DIRECTORY/

**Cloning repositories (HTTPS)**

$ git clone https://github.com/username/repo.git

**Cloning repositories (SSH)**

$ git clone git@github.com:username/repo.git

**Clone raw GitHub content**

$ curl -o https://raw.githubusercontent.com/username/path\_to\_file /output\_path

**Push**

Push to existing repository

$ git add --all # adds everything in current path

$ git add . # adds everything in current path

$ git add [file] # adds only the specific file

$ git commit -m "first"

$ git push

Push to new empty repository

$ git init

$ git remote add origin https://github.com/[USERNAME]/[REPO].git

$ git add .

$ git commit "say something"

$ git push --set-upstream origin master

**Pull changes**

$ git pull origin master

**Git Ignore**

Global

$ git config --global core.excludesfile ~/.gitignore\_global

$ nano ~/.gitignore\_global

Resync .gitignore

$ git rm -r --cached

$ git add --all # or git add .

$ git commit -m "resync"

$ git push

Untrack files already commited

$ echo [FILE] >> ~/.gitignore

$ git rm -r --cached [FILE]

$ git add -u

$ git commit -m "remove cached git files"

$ git pull

$ git push

# Create a new branch with git and manage branches

Fabian St. George edited this page on 26 Jan 2015 · [6 revisions](https://github.com/Kunena/Kunena-Forum/wiki/Create-a-new-branch-with-git-and-manage-branches/_history)

### Pages 11

* [**Home**](https://github.com/Kunena/Kunena-Forum/wiki)
* [**Create a new branch with git and manage branches**](https://github.com/Kunena/Kunena-Forum/wiki/Create-a-new-branch-with-git-and-manage-branches)
* [**Developing in PhpStorm with Git**](https://github.com/Kunena/Kunena-Forum/wiki/Developing-in-PhpStorm-with-Git)
* [**Documentation**](https://github.com/Kunena/Kunena-Forum/wiki/Documentation)
* [**Git Issues Guidelines**](https://github.com/Kunena/Kunena-Forum/wiki/Git-Issues-Guidelines)
* [**github**](https://github.com/Kunena/Kunena-Forum/wiki/github)
* [**How do i contribute?**](https://github.com/Kunena/Kunena-Forum/wiki/How-do-i-contribute%3F)
* [**How to handle conflicts with git**](https://github.com/Kunena/Kunena-Forum/wiki/How-to-handle-conflicts-with-git)
* [**Overview of branches**](https://github.com/Kunena/Kunena-Forum/wiki/Overview-of-branches)
* [**Tips with git bash**](https://github.com/Kunena/Kunena-Forum/wiki/Tips-with-git-bash)
* [**Useful git commands**](https://github.com/Kunena/Kunena-Forum/wiki/Useful-git-commands)

##### Clone this wiki locally



[**Clone in Desktop**](github-windows://openRepo/https:/github.com/Kunena/Kunena-Forum.wiki)

In your github fork, you need to keep your master branch clean, by clean I mean without any changes, like that you can create at any time a branch from your master. Each time, that you want to commit a bug or a feature, you need to create a branch for it, which will be a copy of your master branch.

When you do a pull request on a branch, you can continue to work on another branch and make another pull request on this other branch.

Before creating a new branch pull the changes from upstream, your master needs to be up to date.

Create the branch on your local machine and switch in this branch :

$ git checkout -b [name\_of\_your\_new\_branch]

Push the branch on github :

$ git push origin [name\_of\_your\_new\_branch]

When you want to commit something in your branch, be sure to be in your branch.

You can see all branches created by using :

$ git branch

Which will show :

\* approval\_messages

master

master\_clean

Add a new remote for your branch :

$ git remote add [name\_of\_your\_remote]

Push changes from your commit into your branch :

$ git push origin [name\_of\_your\_remote]

Update your branch when the original branch from official repository has been updated :

$ git fetch [name\_of\_your\_remote]

Then you need to apply to merge changes, if your branch is derivated from develop you need to do :

$ git merge [name\_of\_your\_remote]/develop

Delete a branch on your local filesystem :

$ git branch -d [name\_of\_your\_new\_branch]

To force the deletion of local branch on your filesystem :

$ git branch -D [name\_of\_your\_new\_branch]

Delete the branch on github :

$ git push origin :[name\_of\_your\_new\_branch]

The only difference is the : to say delete, you can do it too by using github interface to remove branch : <https://help.github.com/articles/deleting-unused-branches>.

If you want to change default branch, it's so easy with github, in your fork go into Admin and in the drop-down list default branch choose what you want.