Essential Git Commands

Contents

[GitHub – Make a repository 2](#_Toc478389954)

[Save a Commit to GitHub 5](#_Toc478389955)

[Making sure you are working on the current commit. 7](#_Toc478389956)

[Branches 8](#_Toc478389957)

[Creating a new Branch 8](#_Toc478389958)

[Creating a commit to the new branch for the first time. 8](#_Toc478389959)

# GitHub – Make a repository

|  |
| --- |
|  |
| Click on the green “New repository” |
|  |
| 1. Enter the Repository name 2. Check the “Initialize this repository with a README” 3. Click the green “Create repository” button |

|  |
| --- |
|  |
| Click the green “Clone or download” Button |

|  |
| --- |
|  |
| Click the copy to clip board button (Red arrow pointing at it in this picture) |

|  |
| --- |
|  |
| Open Git Bash |

|  |
| --- |
|  |
| Change to the directory you want to place the file folder from Git Hub, not “cd” is the command to change directory. |

|  |
| --- |
|  |
| Type “git clone “ (make sure there is a space after clone) than paste in the path you copied from Git Hub and enter. |
|  |
| You will see something like this, and your files are cloned and on your local machine ready to use. |
|  |
|  |

# Save a Commit to GitHub

|  |
| --- |
|  |
| Use the command “cd” to go into the folder of your local repository. |

|  |
| --- |
|  |
| Use the command “git add –A” to stage all of the files that have been changed or added. |

|  |
| --- |
|  |
| Use the command ‘git commit –m “comment on what was done” ‘ to create the commit with the message that will be associated with the commit. |

|  |
| --- |
|  |
| Use the command “git push” to push the changes up to GitHub. |

|  |
| --- |
|  |
| After the push you may have to log into your GitHub account. It may be a pop up window like this, or… |

|  |
| --- |
| the UserName might be in the console with the password also in the console or… |
| The UserName might be in the console and the password in a pop up.  If you have logged into the machine before, you might not need to log in at all, depending on your password settings. |

|  |
| --- |
|  |
| When you have finished logging in you will se something like this, which indicates that the commit has been pushed up to GitHub. |

# Making sure you are working on the current commit.

If you are working on a project on different machines, before you start working you should do a pull to make sure you are working on the latest commit.

|  |
| --- |
|  |
| Run Git Bash and go to the folder using the “cd” that has your repository saved on your local machine. |

|  |
| --- |
|  |
| Use the command “git pull” and the latest version will be merged into your repository.cd |

|  |
| --- |
|  |
| If you do “git pull” and there is nothing new to pull you will get this message. |

# Branches

## Creating a new Branch

|  |
| --- |
|  |
| Use the command ‘git checkout –b “new-branch” to create a new branch and go there. |

## Creating a commit to the new branch for the first time.

|  |
| --- |
|  |
| The first time you do a commit to a new branch this is what you see, if you use the standard commands. To fix the problem type in the push command exactly as git tells you to. |

|  |
| --- |
|  |
| If you remember to on your first commit you can type in the push command as ‘git push –set-upstream origin new-branch’ the first time and you will not get the previous fatal error message. It does not hurt if you forget to do the regular ‘git push’ for git will tell you exactly what you need to do for the first fix. |

|  |
| --- |
|  |
| Use the command ‘git checkout master’ to switch back to a branch that already exists. |

## Merging branches

I would recommend avoiding merging branches till you are comfortable in using git than check out either <https://www.atlassian.com/git/tutorials/what-is-version-control> for more information or if you a Laracasts account <https://laracasts.com/search?q=git>.

# Git commands

These commands were copied from <https://www.siteground.com/tutorials/git/commands.htm>

* git config

Sets configuration values for your user name, email, gpg key, preferred diff algorithm, file formats and more. Example: git config --global user.name "My Name" git config --global user.email "user@domain.com" cat ~/.gitconfig [user] name = My Name email = user@domain.com

* git init

Initializes a git repository – creates the initial ‘.git’ directory in a new or in an existing project. Example: cd /home/user/my\_new\_git\_folder/ git init

* git clone

Makes a Git repository copy from a remote source. Also adds the original location as a remote so you can fetch from it again and push to it if you have permissions. Example: git clone git@github.com:user/test.git

* git add

Adds files changes in your working directory to your index. Example: git add .

* git rm

Removes files from your index and your working directory so they will not be tracked. Example: git rm filename

* git commit

Takes all of the changes written in the index, creates a new commit object pointing to it and sets the branch to point to that new commit. Examples: git commit -m ‘committing added changes’ git commit -a -m ‘committing all changes, equals to git add and git commit’

* git status

Shows you the status of files in the index versus the working directory. It will list out files that are untracked (only in your working directory), modified (tracked but not yet updated in your index), and staged (added to your index and ready for committing). Example: git status # On branch master # # Initial commit # # Untracked files: # (use "git add <file>..." to include in what will be committed) # # README nothing added to commit but untracked files present (use "git add" to track)

* git branch

Lists existing branches, including remote branches if ‘-a’ is provided. Creates a new branch if a branch name is provided. Example: git branch -a \* master remotes/origin/master

* git checkout

Checks out a different branch – switches branches by updating the index, working tree, and HEAD to reflect the chosen branch. Example: git checkout newbranch

* git merge

Merges one or more branches into your current branch and automatically creates a new commit if there are no conflicts. Example: git merge newbranchversion

* git reset

Resets your index and working directory to the state of your last commit. Example: git reset --hard HEAD

* git stash

Temporarily saves changes that you don’t want to commit immediately. You can apply the changes later. Example: git stash Saved working directory and index state "WIP on master: 84f241e first commit" HEAD is now at 84f241e first commit (To restore them type "git stash apply")

* git tag

Tags a specific commit with a simple, human readable handle that never moves. Example: git tag -a v1.0 -m 'this is version 1.0 tag'

* git fetch

Fetches all the objects from the remote repository that are not present in the local one. Example: git fetch origin

* git pull

Fetches the files from the remote repository and merges it with your local one. This command is equal to the git fetch and the git merge sequence. Example: git pull origin

* git push

Pushes all the modified local objects to the remote repository and advances its branches. Example: git push origin master

* git remote

Shows all the remote versions of your repository. Example: git remote origin

* git log

Shows a listing of commits on a branch including the corresponding details. Example: git log commit 84f241e8a0d768fb37ff7ad40e294b61a99a0abe Author: User <user@domain.com> Date: Mon May 3 09:24:05 2010 +0300 first commit

* git show

Shows information about a git object. Example: git show commit 84f241e8a0d768fb37ff7ad40e294b61a99a0abe Author: User <user@domain.com> Date: Mon May 3 09:24:05 2010 +0300 first commit diff --git a/README b/README new file mode 100644 index 0000000..e69de29

* git ls-tree

Shows a tree object, including the mode and the name of each item and the SHA-1 value of the blob or the tree that it points to. Example: git ls-tree master^{tree} 100644 blob e69de29bb2d1d6434b8b29ae775ad8c2e48c5391 README

* git cat-file

Used to view the type of an object through the SHA-1 value. Example: git cat-file -t e69de29bb2d1d6434b8b29ae775ad8c2e48c5391 blob

* git grep

Lets you search through your trees of content for words and phrases. Example: git grep "www.siteground.com" -- \*.php

* git diff

Generates patch files or statistics of differences between paths or files in your git repository, or your index or your working directory. Example: git diff

* **gitk**

Graphical Tcl/Tk based interface to a local Git repository. Example: gitk