**CheatSheet**

**GIT**

**CONFIG:**

**Configuration values for your user name, email, gpg key, preferred diff algorithm, file formats and more:**

#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

**INIT:**

**Creates the initial ‘.git’ directory in a new or in an existing project:**

# cd /home/user/my\_new\_git\_folder/ git init

**PULL:**

**Fetches the files from the remote repository and merges it with your local one:**

# git pull origin master

**PUSH:**

**Pushes all the modified local objects to the remote repository:**

# git push origin master

**CLONE:**

**Makes a Git repository copy from a remote source:**

# git clone git@github.com:user/test.git

**ADD:**

**Adds files changes in your working directory to your index:**

# git add .

**RM:**

**Removes files from your index and you’re working directory so they will not be tracked:**

# git rm filename

**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:**

# git commit -m ‘committing added changes’

**STATUS:**

**Shows you the status of files in the index versus the working directory:**

#git status

**BRANCH:**

**Lists existing branches, including remote branches if ‘-a’ is provided:**

# git branch -a \* master remotes/origin/master

**CHECKOUT:**

**Checks out a different branch – switches branches by updating the index, working tree, and HEAD to reflect the chosen branch:**

# git checkout newbranch

**MERGE:**

**Merges one or more branches into your current branch and automatically creates a new commit if there are no conflicts:**

# git merge newbranchversion

**RESET**

**Resets your index and working directory to the state of your last commit:**

# git reset --hard HEAD

**STASH:**

**Temporarily saves changes that you don’t want to commit immediately:**

#git stash

**TAG:**

**Tags a specific commit with a simple, human readable handle that never moves:**

# git tag -a v1.0 -m 'this is version 1.0 tag'

**FETCH:**

**Fetches all the objects from the remote repository that are not present in the local one:**

# git fetch origin

**REMOTE:**

**Shows all the remote versions of your repository:**

# git remote origin

**LOG:**

**Shows a listing of commits on a branch including the corresponding details:**

# git log commit 84f241e8a0d768fb37ff7ad40e294b61a99a0abe Author: User <user@domain.com> Date: Mon May 3 09:24:05 2010 +0300 first commit

**SHOW:**

**Shows information about a git object:**

# 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

**GREP:**

**Let’s you search through your trees of content for words and phrases:**

# git grep "www.siteground.com" -- \*.php

**DIFF:**

**Generates patch files or statistics of differences between paths or files in your git repository:**

# git diff

**ARCHIVE:**

**Creates a tar or zip file including the contents of a single tree from your repository:**

# git archive --format=zip master^ README >file.zip

**GC:**

**Garbage collector for your repository. Optimizes your repository:**

# git gc

**FSCK:**

**Does an integrity check of the Git file system, identifying corrupted objects:**

# git fsck

**PRUNE:**

**Removes objects that are no longer pointed to by any object in any reachable branch:**

# git prune

|  |
| --- |
| **Resources:**   * http://www.siteground.com/tutorials/git/commands.htm |