# GIT Cheat Sheet

## INSTALLATION

With platform specific installers for Git, GitHub also provides the ease of staying up to date with the latest releases of the command line tool while providing a graphical user interface for day-to-day interaction, review, and repository synchronization.

**GitHub for Windows [**https://windows.github.com**]**

**GitHub for Mac [**https://mac.github.com**]**

For Linux and Solaris platforms, the latest release is available on the official Git web site.

**Git for All Platforms [**http://git-scm.com**]**

## Set-Up

[Configuring user information used across all repositories]

| Command | Description |
|---|---|
| **git config --global user.name "[firstname lastname]"** | set a name that is identifiable for credit when review version history |
| **git config --global user.email "[valid-email]"** | set an email address that will be associated with each history marker |
| **git config --global color.ui auto** | set automatic command line coloring for Git for easy reviewing |

## Getting & Creating Projects

[Configuring user information, initializing and cloning repositories]

| Command | Description |
|---|---|
| **git init** | Initialize an existing directory as a local Git repository |
| **git clone ssh://git@github.com/[username]/[repository-name].git** | Create a local copy of a remote repository |
| **git clone [url]** | retrieve an entire repository from a hosted location via URL |

## Basic Snapshotting

[Working with snapshots and the Git staging area]

| Command | Description |
|---|---|
| **git status** | Check status or show modified files in working directory, staged for your next commit |
| **git add [file-name.txt]** | Add a file to the staging area |
| **git add -A** | Add all new and changed files to the staging area |
| **git commit -m "[commit message]"** | Commit changes |
| **git rm -r [file-name.txt]** | Remove a file (or folder) |
| **git reset [file]** | Un-stage a file while retaining the changes in working directory |
| **git diff** | diff of what is changed but not staged |
| **git diff --staged** | diff of what is staged but not yet committed |
| **git commit -m "[descriptive message]"** | commit your staged content as a new commit snapshot |

**GIT Cheat Sheet**
**Branching & Merging**

[Isolating work in branches, changing context, and integrating changes]

| Command | Description |
|---|---|
| **git branch** | List branches (the asterisk [a*] denotes the current branch) |
| **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 [branch name]** | 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 branch -m [old branch name] [new branch name]** | Rename a local branch |
| **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 |
| **git log** | show all commits in the current branch's history |

**Sharing & Updating Projects**

[Retrieving updates from another repository and updating local repos]

| Command | Description |
|---|---|
| **git push origin [branch name]** | Push a branch to your remote repository |
| **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**

**[**Examining logs, diffs and object information**]**

| Command | Description |
|---|---|
| **git log** | View changes or show the commit history for the currently active branch |
| **git log --summary** | View changes (detailed) |
| **git log --oneline** | View changes (briefly) |
| **git diff [source branch] [target branch]** | Preview changes before merging |
| **git log source_branch..target_branch** | show the commits on target_branch that are not on branch |
| **git log --follow [file]** | show the commits that changed file, even across renames |
| **git show [SHA]** | show any object in Git in human-readable format |

# GIT Cheat Sheet

## TRACKING PATH CHANGES

[Versioning file removes and path changes]

| Command | Description |
|---|---|
| **git rm [file]** | delete the file from project and stage the removal for commit |
| **git mv [existing-path] [new-path]** | change an existing file path and stage the move |
| **git log --stat -M** | show all commit logs with indication of any paths that moved TEMPORARY COMMITS |

## IGNORING PATTERNS

[Preventing unintentional staging or committing of files]

| Command | Description |
|---|---|
| **logs/**<br>***.notes**<br>**pattern*/** | Save a file with desired patterns as .gitignore with either direct string matches or wildcard globs. |
| **git config --global core.excludesfile [file]** | system wide ignore pattern for all local repositories |

## REWRITE HISTORY

[Rewriting branches, updating commits and clearing history]

| Command | Description |
|---|---|
| **git rebase [branch]** | apply any commits of current branch ahead of specified one |
| **git reset --hard [commit]** | clear staging area, rewrite working tree from specified commit |

## TEMPORARY COMMITS

[Temporarily store modified, tracked files in order to change branches]

| Command | Description |
|---|---|
| **git stash** | Save modified and staged changes |
| **git stash list** | list stack-order of stashed file changes |
| **git stash pop** | write working from top of stash stack |
| **git stash drop** | discard the changes from top of stash stack |