GIT CHEAT SHEET

TARA-TECHNOLOGIES

Git is the free and open source distributed version control system that's responsible for everything GitHub related that happens locally on your computer. This cheat sheet features the most important and commonl used Git commands for easy reference.

INSTALLATION & GUIS

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 offi cial Git web site.

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

SETUP

Configuring user information used across all local repositories

!"#$017*"!$//!+18&+$'%,.97&2,$3)*".%#7&2,$+&%#7&2,-6
set a name that is identifiable for credit when review version history
!"#$017*"!$//!+18&+$'%,.9,2&"+$3)5&+"(/,2&"+-6
set an email address that will be associated with each history marker
!"#$017*"!$//!+18&+$01+1.9'"$&'#1
set automatic command line coloring for Git for easy reviewing

SETUP & INIT

Configuring user information, initializing and cloning repositories

!"#$"7"#
initialize an existing directory as a Git repository
!"#$0+17,$)'.+-
retrieve an entire repository from a hosted location via URL

STAGE & SNAPSHOT

Working with snapshots and the Git staging area

!"#$%#&#'%
show modified files in working directory, staged for your next commit
!"#$&(($)*"+,-
add a file as it looks now to your next commit (stage)
!"#$.,%,#$)*"+,-
unstage a file while retaining the changes in working directory
!"#$("**
diff of what is changed but not staged
!"#$("**$//%#&!,(
diff of what is staged but not yet commtted
!"#$0122"#$/2$3)(,%0."4#"5,$2,%%&!,-6
commit your staged content as a new commit snapshot

BRANCH & MERGE

Isolating work in branches, changing context, and integrating changes

!"#$8.&70:
list your branches. a * will appear next to the currently active branch
!"#$8.&70:$)8.&70:/7&2,-
create a new branch at the current commit
!"#$0:,0;1'#
switch to another branch and check it out into your working directory
!"#$2,.!,$)8.&70:-
merge the specified branch's history into the current one
!"#$+1!
show all commits in the current branch's history

INSPECT & COMPARE

Examining logs, diffs and object information

!"#$+1!
show the commit history for the currently active branch
!"#$+1!$8.&70:@998.&70:A
show the commits on branchA that are not on branchB
!"#$+1!$//*1++1>$)*"+,-
show the commits that changed file, even across renames
!"#$("**$8.&70:@9998.&70:A
show the diff of what is in branchA that is not in branchB
!"#$%:1>$)BCA-
show any object in Git in human-readable format

SHARE & UPDATE

Retrieving updates from another repository and updating local repos

!"#$.,21#,$&(($)&+"&%-$)'.+-
add a git URL as an alias
!"#$*,#0:$)&+"&%-
fetch down all the branches from that Git remote
!"#$2,.!,$)&+"&%-<)8.&70:-
merge a remote branch into your current branch to bring it up to date
!"#$4'%:$)&+"&%-$)8.&70:-
Transmit local branch commits to the remote repository branch
!"#$4'++
fetch and merge any commits from the tracking remote branch

TRACKING PATH CHANGES

Versioning file removes and path changes

!"#$.2$)*"+,-
delete the file from project and stage the removal for commit
!"#25),="%#"7!/4&#:-$)7,>/4&#:-
change an existing file path and stage the move
!"#$+1!$//%#&#$/?
show all commit logs with indication of any paths that moved

REWRITE HISTORY

Rewriting branches, updating commits and clearing history

!"#$.,8&%,$)8.&70:-
apply any commits of current branch ahead of specified one
!"#$.,%,#$//:&.($)0122"#-
clear staging area, rewrite working tree from specified commit

IGNORING PATTERNS

Preventing unintentional staging or commiting of files

+1!%< D971#,% 4&##,.7D<
Save a file with desired patterns as .gitignore with either direct string matches or wildcard globs.
!"#$017*"!$//!+18&+$01.,9,=0+'(,%*"+,$)*"+,-
system wide ignore pattern for all local repositories

TEMPORARY COMMITS

Temporarily store modified, tracked files in order to change branches

!"#$%#&%:
Save modified and staged changes
!"#$%#&%:$+"%#
list stack-order of stashed file changes
!"#$%#&%:$414
write working from top of stash stack
!"#$%#&%:$(.14
discard the changes from top of stash stack

TARA TECHNOLOGIES

Teach and learn better, together. GitHub is free for students and teachers. Discounts available for other educational uses.

info@hmftj.com
https://hmftj.com