Basic Git commands

Git task	Notes	Git commands
Tell Git who you are	Configure the author name and email address to be used with your commits. Note that Git strips some characters (for example trailing periods) from user.name.	git configglobal user.name "Sam Smith" git configglobal user.email sam@example.com
Create a new local repository		git init
Check out a repository	Create a working copy of a local repository: For a remote server, use:	<pre>git clone /path/to/repository git clone username@host:/path/to/repository</pre>
Add files	Add one or more files to staging (index):	git add <filename> git add *</filename>
Commit	Commit changes to head (but not yet to the remote repository):	git commit -m "Commit message"
	Commit any files you've added with git add, and also commit any files you've changed since then:	git commit -a
Push	Send changes to the master branch of your remote repository:	git push origin master
Status	List the files you've changed and those you still need to add or commit:	git status
Connect to a remote repository	If you haven't connected your local repository to a remote server, add the server to be able to push to it:	git remote add origin <server></server>
	List all currently configured remote repositories:	git remote -v

Create a new branch and switch to it: Switch from one branch to another: List all the branches in your repo, and also tell you what branch you're currently in: Delete the feature branch: Push the branch to your remote repository; so others can use it: Push all branches to your remote repository: Delete a branch on your remote repository: Fetch and merge changes on the remote server to your working directory: To merge a different branch into your active branch: View all the merge conflicts: remote repository: Update from the remote server to your working directory: Update from the remote server to your working directory: To merge a different branch into your active branch: View all the merge conflicts: View the conflicts against the base file: Preview changes, before merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant changeset, such as a git checkout prach checkout spranch denkout pranch denkout -d spranch and spranch and spranch and spranch also tell you and spranch and s		S
another: List all the branches in your repo, and also tell you what branch you're currently in: Delete the feature branch: Push the branch to your remote repository, so others can use it: Push all branches to your remote repository: Delete a branch on your remote repository: Delete a branch on your remote repository: Fetch and merge changes on the remote server to your working directory: To merge a different branch into your active branch: To merge a different branch into your active branch: View all the merge conflicts: remote repository View the conflicts against the base file: Preview changes, before merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant Viet tag 1.0.0 committed <a a="" auchhaire<="" href="ma</td><td rowspan=7>Branches</td><td>OTT CHECKOUT -D <pre>pranchname></pre></td></tr><tr><td>your repo, and also tell you what branch you're currently in: Delete the feature branch: Push the branch to your remote repository, so others can use it: Push all branches to your remote repository: Delete a branch on your remote repository: Delete a branch on your remote repository: Delete a branch on your remote repository: Fetch and merge changes on the remote server to your working directory: To merge a different branch into your active branch: Update from the remote repository View all the merge conflicts: Update from the remote repository View the conflicts against the base file: Preview changes, before merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant Quit tag 1.0.0 < commitID></td><td>OTT CHECKOUT SOF ANCHHAME?</td></tr><tr><td>Push the branch to your remote repository, so others can use it: Push all branches to your remote repository: Delete a branch on your remote repository: Delete a branch on your remote repository: Fetch and merge changes on the remote server to your working directory: To merge a different branch into your active branch: View all the merge conflicts: remote repository Update View all the merge conflicts: remote repository View the conflicts against the base file: Preview changes, before merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant Push origin
 git push origin
 print push origin
 git push origin
 print push origin
 print push origin
 push origin
 print push origin
 push origin
 print push origin
 print push origin
 push origin <br/</td><td>your repo, and also tell you git branch what branch you're</td></tr><tr><td>remote repository, so others can use it: Push all branches to your remote repository: Delete a branch on your remote repository: Fetch and merge changes on the remote server to your working directory: To merge a different branch into your active branch: Update from the conflicts: remote repository View all the merge conflicts: remote repository View the conflicts against the base file: Preview changes, before merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant git pushall origin git push origin
git push origin
 git push origin
 git push origin
 spranchname> git push origin
 git push origin
 git push origin
 spranchname> git push origin
 spranchname></td><td>Delete the feature branch: git branch -d <branchname></td></tr><tr><td>remote repository: Delete a branch on your remote repository: Fetch and merge changes on the remote server to your working directory: To merge a different branch into your active branch: Update View all the merge conflicts: remote repository View the conflicts against the base file: Preview changes, before merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant git push origin :
 prull git pull git pull git pull git diff git diff sourcebranch> git diffbase <filename> git diff <sourcebranch> <targetbranch></td><td>remote repository, so git push origin
branchname></td></tr><tr><td>remote repository: Fetch and merge changes on the remote server to your working directory: To merge a different branch into your active branch: Update from the remote repository View all the merge conflicts:</td><td>ull busiiall ullulii</td></tr><tr><td>on the remote server to your working directory: To merge a different branch into your active branch: Update from the remote repository View all the merge conflicts: git diff sourcebranch> <targetbranch> git add <filename> git add <filename></td><td>" nush="" of="" suf="" td="" tuth="" un="">		
branch into your active branch: Update from the remote repository View the conflicts against the base file: Preview changes, before merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant git merge git diff git diffbase <filename> git diffbase <filename> git add <filename> git add <filename></filename></filename></filename></filename>	from the remote	on the remote server to git pull
from the remote repository View the conflicts against the base file: Preview changes, before merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant git diffbase <filename> git diffbase <filename> git diffbase <filename></filename></filename></filename>		branch into your active git merge branchname>
View the conflicts against the base file: Preview changes, before merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant Yiew the conflicts against git diffbase <filename> git diffbase <filename> git diffbase <filename> git diffbase <filename> git diffbase <filename></filename></filename></filename></filename></filename>		
merging: After you have manually resolved any conflicts, you mark the changed file: You can use tagging to mark a significant git tag 1.0.0 < committo>		
resolved any conflicts, you git add <filename> mark the changed file: You can use tagging to mark a significant git tag 1.0.0 <committee< td=""><td>Treview changes, before</td></committee<></filename>		Treview changes, before
mark a significant qit tag 1.0.0 <commit td=""></commit>		resolved any conflicts, you git add <filename></filename>
release:	Tags	mark a significant changeset, such as a git tag 1.0.0 <commitid></commitid>
CommitId is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using:		characters of the changeset ID, up to 10, but must be
Push all tags to remote repository:		utt busiitaus vitutii
Undo local If you mess up, you can git checkout <filename></filename>	Undo local	If you mess up, you can git checkout <filename></filename>

	replace the changes in your working tree with the last content in head:	
changes	Changes already added to the index, as well as new files, will be kept.	
	Instead, to drop all your local changes and commits, fetch the latest	git fetch origin
	history from the server and point your local master branch at it, do this:	git resethard origin/master
Search	Search the working directory for foo():	git grep "foo()"

Ref:- https://confluence.atlassian.com/bitbucketserver/basic-git-commands-776639767.html