**GIT**

1. **Tell Git who you are**

* git config --global user.name "Sam Smith"
* git config --global user.email sam@example.com #*Configure the author*

*name and email address to be used with your commits.*

**Note:** Git strips some characters (for example trailing periods) from user.name

**2)Create a new local repository**

* git init

**3)Check out a repository**

* git clone /path/to/repository #*Create a working copy of a local repository*
* git clone username@host:/path/to/repository #*For a remote server, use this*

**4)Add files**

* git add <filename> # *to add one file to index*
* git add -A # *to add one or more files*

**5)Commit**

* git commit -m "Commit message" # *(for single file commit) Commit changes to head (but not yet to the remote repository)*
* git commit -a -m # *to commit all files*
* git commit -a # *Commit any files you've added with git add, and also commit any files you've changed since then*

**6)Push**

* git push origin master # *Send changes to the master branch of your remote repository*

**7)Status**

* git status #*List the files you've changed and those you still need to add or commit*

**8)Connect to a remote repository**

* git remote add origin <server link address> #*If you haven't connected your local repository to a remote server, add the server to be able to push to it*
* git remote -v #*List all currently configured remote repositories*

**9)Branches**

* git checkout -b <branchname> #*Create new branch and switch to it*
* git checkout <branchname> #*Switch from one branch to another*
* git branch #*List all the branches in your repo, and also tell you what branch you're currently in*
* git branch -d <branchname> #*Delete the feature branch*
* git push origin <branchname> #*Push the branch to your remote repository, so others can use it*
* git push --all origin #*Push all branches to your remote repository*
* git push origin :<branchname> #*Delete a branch on your remote repository*

**10)Update from the remote repository**

* git pull origin master #*Fetch and merge changes on the remote server to your working directory*
* git merge <branchname> #*To merge a different branch into your active branch*
* git diff
* git diff --base <filename>
* git diff <sourcebranch> <targetbranch> #*To View all the merge conflicts*

View the conflicts against the base file:

Preview changes, before merging

* git add <filename> #*After you have manually resolved any conflicts, you mark the changed file*

**11)INSPECT & COMPARE**  #*Examining logs, diffs and object information*

* git log #*Commit-Id is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using.show all commits in the current branch’s history*
* git log branchB..branchA #*show the commits on branchA that are not on branchB*
* git log --follow [file] #*show the commits that changed file, even across renames*
* git diff branchB...branchA #*show the diff of what is in branchA that is not in branchB*
* git show [SHA] #*show any object in Git in human-readable format. simply list the files that changed as part of the commit*

**12)REWRITE HISTORY** #*Rewriting branches, updating commits and clearing history*

* 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*

**13)TEMPORARY COMMITS** #*Temporarily store modified, tracked files in order to change branches*

* 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*
* git stash apply #*back to index*