



# Git

- [Cheat Sheet - GIT](#)

# Cheat Sheet - GIT

Git Cheat Sheet		Rebasing	Review your Repo	Stashing
<h3>Setup</h3> <p>Set the name and email that will be attached to your commits and tags</p> <pre>\$ git config --global user.name "Danny Adams" \$ git config --global user.email "my-email@gmail.com"</pre>	<h3>Branches</h3> <p>List all local branches. Add -r flag to show all remote branches. -a flag for all branches.</p> <pre>\$ git branch</pre> <p>Create a new branch</p> <pre>\$ git branch &lt;new-branch&gt;</pre> <p>Switch to a branch &amp; update the working directory</p> <pre>\$ git checkout &lt;branch&gt;</pre> <p>Create a new branch and switch to it</p> <pre>\$ git checkout -b &lt;new-branch&gt;</pre> <p>Delete a merged branch</p> <pre>\$ git branch -d &lt;branch&gt;</pre> <p>Delete a branch, whether merged or not</p> <pre>\$ git branch -D &lt;branch&gt;</pre> <p>Add a tag to current commit (often used for new version releases)</p> <pre>\$ git tag &lt;tag-name&gt;</pre>	<p>Rebase feature branch onto main (to incorporate new changes made to main). Prevents unnecessary merge commits into feature, keeping history clean</p>  <pre>\$ git checkout feature \$ git rebase main</pre> <p>Interactively clean up a branches commits before rebasing onto main</p> <pre>\$ git rebase -i main</pre> <p>Interactively rebase the last 3 commits on current branch</p> <pre>\$ git rebase -i HEAD~3</pre>	<h3>Review your Repo</h3> <p>List new or modified files not yet committed</p> <pre>\$ git status</pre> <p>List commit history, with respective IDs</p> <pre>\$ git log --oneline</pre> <p>Show changes to unstaged files. For changes to staged files, add --cached option</p> <pre>\$ git diff</pre> <p>Show changes between two commits</p> <pre>\$ git diff commit1_ID commit2_ID</pre>	<p>Store modified &amp; staged changes. To include untracked files, add -u flag. For untracked &amp; ignored files, add -a flag.</p> <pre>\$ git stash</pre> <p>As above, but add a comment.</p> <pre>\$ git stash save "comment"</pre> <p>Partial stash. Stash just a single file, a collection of files, or individual changes from within files</p> <pre>\$ git stash -p</pre> <p>List all stashes</p> <pre>\$ git stash list</pre> <p>Re-apply the stash without deleting it</p> <pre>\$ git stash apply</pre> <p>Re-apply the stash at index 2, then delete it from the stash list. Omit stash@{n} to pop the most recent stash.</p> <pre>\$ git stash pop stash@{2}</pre> <p>Show the diff summary of stash 1. Pass the -p flag to see the full diff.</p> <pre>\$ git stash show stash@{1}</pre>
<h3>Start a Project</h3> <p>Create a local repo (omit &lt;directory&gt; to initialise the current directory as a git repo)</p> <pre>\$ git init &lt;directory&gt;</pre> <p>Download a remote repo</p> <pre>\$ git clone &lt;url&gt;</pre>	<h3>Merging</h3> <p>Merge branch a into branch b. Add --no-ff option for no-fast-forward merge</p>  <p>New Merge Commit (no-ff)</p> <pre>\$ git checkout b \$ git merge a</pre> <p>Merge &amp; squash all commits into one new commit</p> <pre>\$ git merge --squash a</pre>	<h3>Undoing Things</h3> <p>Move (&amp;/or rename) a file &amp; stage move</p> <pre>\$ git mv &lt;existing_path&gt; &lt;new_path&gt;</pre> <p>Remove a file from working directory &amp; staging area, then stage the removal</p> <pre>\$ git rm &lt;file&gt;</pre> <p>Remove from staging area only</p> <pre>\$ git rm --cached &lt;file&gt;</pre> <p>View a previous commit (READ only)</p> <pre>\$ git checkout &lt;commit_ID&gt;</pre> <p>Create a new commit, reverting the changes from a specified commit</p> <pre>\$ git revert &lt;commit_ID&gt;</pre> <p>Go back to a previous commit &amp; delete all commits ahead of it (revert is safer). Add --hard flag to also delete workspace changes (BE VERY CAREFUL)</p> <pre>\$ git reset &lt;commit_ID&gt;</pre>	<h3>Synchronizing</h3> <p>Add a remote repo</p> <pre>\$ git remote add &lt;alias&gt; &lt;url&gt;</pre> <p>View all remote connections. Add -v flag to view urls.</p> <pre>\$ git remote</pre> <p>Remove a connection</p> <pre>\$ git remote remove &lt;alias&gt;</pre> <p>Rename a connection</p> <pre>\$ git remote rename &lt;old&gt; &lt;new&gt;</pre> <p>Fetch all branches from remote repo (no merge)</p> <pre>\$ git fetch &lt;alias&gt;</pre> <p>Fetch a specific branch</p> <pre>\$ git fetch &lt;alias&gt; &lt;branch&gt;</pre> <p>Fetch the remote repo's copy of the current branch, then merge</p> <pre>\$ git pull</pre> <p>Move (rebase) your local changes onto the top of new changes made to the remote repo (for clean, linear history)</p> <pre>\$ git pull --rebase &lt;alias&gt;</pre> <p>Upload local content to remote repo</p> <pre>\$ git push &lt;alias&gt;</pre> <p>Upload to a branch (can then pull request)</p> <pre>\$ git push &lt;alias&gt; &lt;branch&gt;</pre>	
<h3>Make a Change</h3> <p>Add a file to staging</p> <pre>\$ git add &lt;file&gt;</pre> <p>Stage all files</p> <pre>\$ git add .</pre> <p>Commit all staged files to git</p> <pre>\$ git commit -m "commit message"</pre> <p>Add all changes made to tracked files &amp; commit</p> <pre>\$ git commit -am "commit message"</pre>	<h3>Basic Concepts</h3> <p>main: default development branch origin: default upstream repo HEAD: current branch HEAD^: parent of HEAD HEAD~4: great-great grandparent of HEAD</p> <p>By @DoableDanny</p>			