Git Commands

Created	@November 2, 2022 2:10 PM
Last Edited Time	@November 3, 2022 11:30 AM
Type	
Status	
Created By	
Last Edited By	
Stakeholders	

Fundamentals

`<files>` is a stand-in for any file glob pattern, ex: `filename.md` or `*.md` or `lib/*`

`<name>` is a stand-in for any label/value you want to use.

`<commit id>` is a stand-in for a commit identifier.

`<ref>` is a stand-in for anything that references a commit

Stage & Commit

- `git add <files>`
  - Stage changes
- `git commit`
  - Lock in changes as a new commit
- `git commit -m "Some message"`
  - Commit with a message
- `git show <ref>`
  - Inspect a commit

Undoing / Cleanup

- `git restore --staged <files>`
  - Clear staging / Un-stage
- `git restore <files>`
  - Clear working directory
- `git clean -i` or `-f`
  - Throw out untracked files (interactive / forced)
- `git reset HEAD^`

- `git log --oneline`
  - View the log

  - Undo the last commit on this branch
- `git reset HEAD~`
  - Undo the last commit on this branch
- `git commit --amend`
  - Redo the last commit on this branch

Branching

- `git branch`
  - List branches
- `git branch -a`
  - List branches + remote refs
- `git branch <name>`
  - Create a branch
- `git branch -m <new name>`
  - Rename a branch
- `git branch -D <name>`
  - Delete a branch
- `git branch <name> <commit id>`
  - Create a branch from a specific commit

Merging

- `git merge <ref>`
  - Merge into current branch
  - `--no-ff`
    - No fast-forward merges
  - `--ff-only`
    - Only fast-forward
  - `-X ours` or `-X theirs`
    - Conflict resolution option
- `git merge --abort`
  - Cancel the merge in progress
- `git merge --continue`
  - Finalize the merge in progress
- `git reset ORIG_HEAD --hard`
  - Undo the merge I just did

Remotes

- `git remote add <remote name>`
  - Add a remote bookmark
- `git push origin <ref>`
  - Push a ref to the remote
  - Merges local branch into the remote branch
- `git pull origin <ref>`
  - Merge down updates from a ref on the remote
- `git fetch origin`
  - Download the latest refs and objects from the remote

Log & Compare

- `git log`
  - View the log
- `git log -S <search string>`
  - Search diffs for a string
- `git log -G <search regex>`
  - Regex search through diffs
- `git log -- <files>`
  - Limit log by file pattern
- `git log --grep <search string>`
  - Search commit messages
- `git diff <A>..<B>`
  - Changes to apply to make A look like B
- `git diff <A>...<B>`
  - Compare merge base with B
  - Changes to apply to make the merge base between A and B, look like B
- `git log <A>..<B>`
  - History of B, excluding history of A
- `git log <A>...<B>`
  - History of A and B, excluding history of their merge base

Cherry Pick & Revert

- `git cherry-pick <ref>`
  - Copy changes from a commit into a new commit
  - `--no-commit`
    - Don’t auto-commit
- `git cherry-pick <ref>..<ref>`
- `git revert <ref>`
  - Undo changes from a commit, using a new commit
  - `--no-commit`
- `git revert <ref>..<ref>`

Bisect

- `git bisect start`
  - Start a bisect
- `git bisect good <commit id>`
  - Indicate a commit is good
- `git bisect bad <commit id>`
  - Indicate a commit is bad
- `git bisect reset`
  - Stop the bisect

Rebase & Squash

- `git rebase <base>`
  - Re-set the merge base commit
- `git rebase --onto <base> <exclude>`
  - Rebase and exclude a portion of history
- `git rebase --keep-base <base>`
  - Rebase without bringing in updates
- `git rebase -i <base>`
  - Interactive rebase

Reset & Reflog

- `git reset <commit id>`
  - `--soft`
    - Only affect the branch
  - `--mixed`
    - Affect the branch
    - Clear staging
  - `--hard`
    - Affects the branch
    - Clear staging

        - Clear working directory
- `git reflog`
    - History of your actions
- `git reflog <branch name>`
- `git reflog HEAD`
- `git fsck --lost-found`
    - Dig through the trash for lost objects