Git Basics Cheat Sheet

3 Objects

The git data model

Commit

Contains the tree object sha, parent commit hash, author, committer, date and message.

Tree

Contains the directory listing for the commit

Blob (binary large object)

Contains the content of each file in a repository

3 Trees

The git state model

HEAD

Points to the currently checked out branch, which points to the last commit from that branch. Stores the commit history.

Index

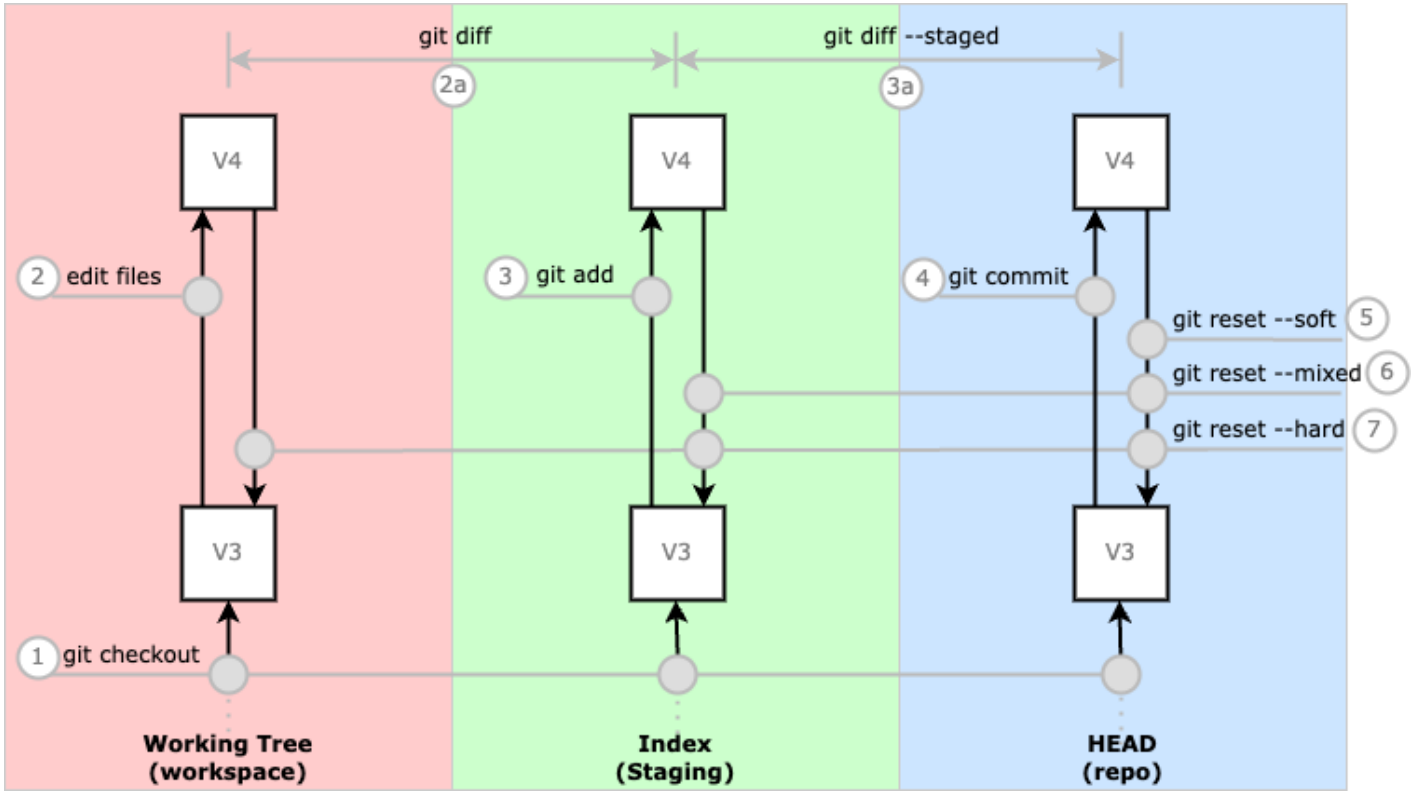
Points to the staging area. Where you add files prior to commit.

Working Tree

Stores files whose contents can be changed (files checked out on disk). Also called the working directory.

Managing Local Changes

A round-trip diagram demonstrating changing files from version 3 to version 4, and back. (The numbers trace a typical path for this round trip).



Basic Commands

Command	Description
`git add`	Copy changes from the working directory to the index (stage).
`git bisect`	Use a binary search to find a bad commit.
`git branch`	See branches.
`git checkout -b <branch name>`	Create <branch name> and check it out.
`git cherry-pick`	Copy a commit to the current branch.
`git commit –m "description"`	Create a new commit containing staged changes with a message and your description.
`git diff`	See differences between the working tree and index.
`git diff --staged`	See differences between the index and HEAD.
`git merge`	Combine changes from 1 or more branches into the current branch.
`git rebase <branch>`	Rebase current branch to start from <branch>.
`git rebase -i`	Use interactive mode to launch the configured editor and accomplish any of these things (and more): • Squash several commits into one or a few • Remove commits • Reorder commits • Reword comments
`git reflog`	See where HEAD has been (only applies to activity in your local repo).
`git reset --soft`	Point current branch to a commit. Leave index and working tree alone.
`git reset --mixed`	Point current branch and index to a commit. Leave working tree alone.
`git reset --hard`	Point current branch, index, and working tree to a commit.
`git remote -v`	Show remotes configured for this repo – names and URLs.
`git show`	Show the diff between HEAD and previous commit (except for merge commits; shows sha's of its parents);
`git stash`	Save changes pending in working tree and index to a "stash stack", and come back to them later (with git stash pop).
`git status`	See status of your trees - untracked, modified, and staged files.

Collaborative Development Models

Fork and Pull Model

1. Fork an existing repository
2. Push changes to your personal fork
3. Open pull requests so the owner can merge your changes from your fork into the source repository

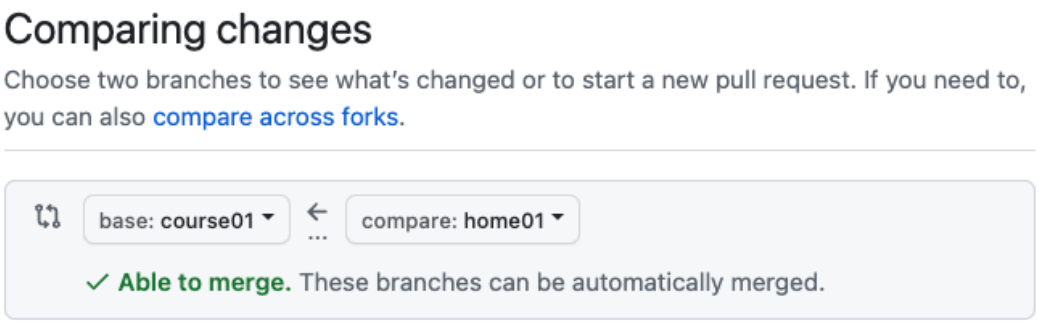
Shared Repository Model

1. Owner grants collaborators push access to single shared repository
2. Create a branch, make changes, and push to the shared repository
3. On GitHub, open a pull request to initiate a code review for approval and merge

Compare Changes

On GitHub, type “/compare” in the address bar to access the "Compare changes" page. For example: https://github.com/walquis/git-basics-sample-project-repo/compare

Use the drop-down controls to compare changes across branches, commits, tags, or forks.



Git Graph

To see the graph: `git log --all --decorate --oneline --graph`

```
*   e96947f (HEAD -> main, origin/main, origin/HEAD) Merge branch 'main' into flex
|\
| *   d0095aa Merge branch 'padding-border-margin' into main
| |\
| | * 4e46c64 Add the class refs to the Sol image.  Tweak the CSS a bit.
| | * eeafe14 Added one-column style for Sol image width
| * | 9c6c222 defined the css class to add 'Caesar Dressing' font to h1 HTML tags
| * | 97b4162 Update the head section in the layouts with stylesheet ref to Caesar
| |/
* | 1b960d8 Apply the display-flex CSS to the planets
* | f3c9add Make the CSS classes ffor putting planets in flex display.
|/
*   434e70c Merge branch 'more-about' into main
```

Graph Components

Lines: The lines at left trace version histories based on each commit's parent(s).

Sha: Uniquely identifies commits; abbreviated to 7 digits, for example, e96947f.

Asterisk: Shows the path on which each commit falls.

HEAD: Points to the checked-out branch. HEAD, main, origin/main, and origin/HEAD all correspond to commit e96947f.

main: The checked out branch (it's pointed to by HEAD).

origin/**main**: The main branch on the remote repository called "origin".

Merges: A merge commit is signaled by multiple parent commits (2 or more lines meeting).

Vocabulary

Term	Definition
sha (sha-1)	A hash function that uniquely identifies a git object by its content.
content-addressable filesystem	Instead of referring to files by location, git refers to content by hash value. "The content determines the address."
fetch versus pull	Fetch gets all changes from a remote but does not merge. Pull performs a fetch, followed by a merge.
fast-forward merge	Applicable when no divergence between current branch and branch being merged; moves the current branch to "catch" up with the branch being merged. (This is the ONLY kind of merge a 'git push' will do).
'detached HEAD' state	Instead of pointing to a branch, HEAD points to a commit.
git symbolic ref	A reference to a name (such as a branch name) that resolves to a commit. "HEAD -> main" is an example.