$ git branch

$ git logs

GIT CHEAT SHEET

SCALER
Topics

$ git fetch

$ git status

SETUP & INIT
Configuring user information, initializing and cloning repositories.
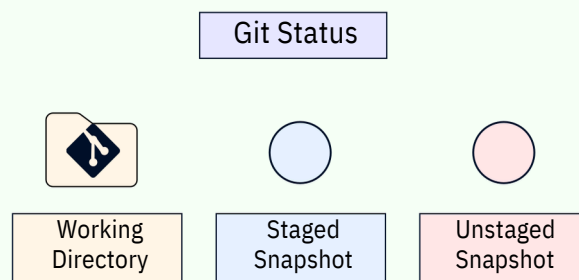
STAGE & SNAPSHOT
Working with snapshots and the Git staging area.

$ git init

Initialize an existing directory as a Git repository.

git init

Normal Directory → Git Repository

$ git status

Show modified files in working directory, staged for your next commit.

Git Status

Working Directory | Staged Snapshot | Unstaged Snapshot

$ git clone [url]

Retrieve an entire repository from a hosted location via URL.

Remote Repository at [URL]

git clone [URL]

Local Copy of the Repository

git clone [URL]

Local Copy of the Repository
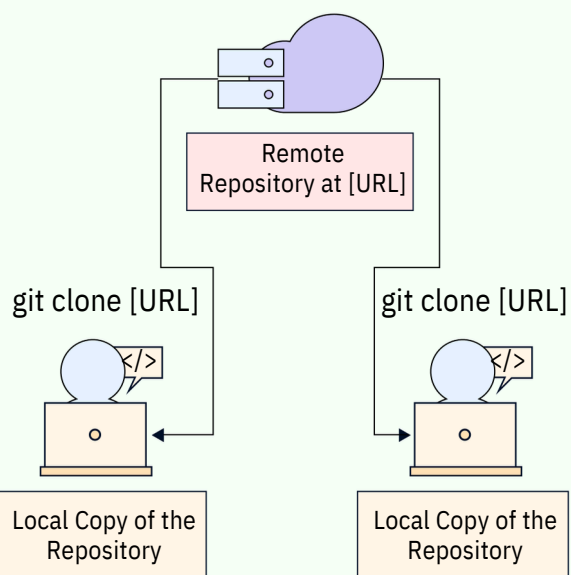
$ git add [file]

Add a file as it looks now to your next commit (stage).

git add [file]
(Stage)

Working Directory → Staging Area

SCALER
Topics

Git Cheatsheet
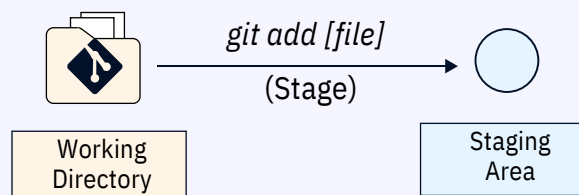
$ git reset [file]

Unstage a file while retaining the changes in working directory.

(Stage)
git add [file]
git reset [file]
(Unstage)

Working Directory

Staging Area

$ git diff

Diff of what is changed but not staged

Working Directory ≠ git diff Staging Area

git diff --staged

Diff of what is staged but not yet commited.

HEAD

Staging Area ≠ git diff --staged Commit History

$ git commit -m "[descriptive message]"

Commit your staged content as a new commit snapshot.

Working Directory

Staging Area

Git Repository (Commit History)

git add

git commit

- -

BRANCH & MERGE
Isolating work in branches, changing context, and integrating changes.
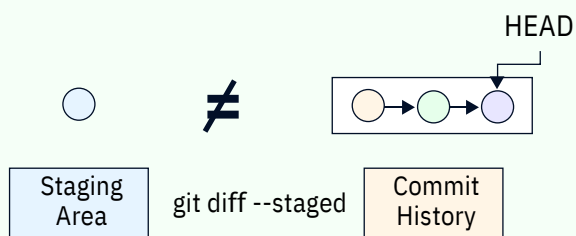
$ git branch

List your branches. A * will appear next to the currently active branch.

Branch 1

Main Branch

Branch 2

SCALER Topics

Git Cheatsheet
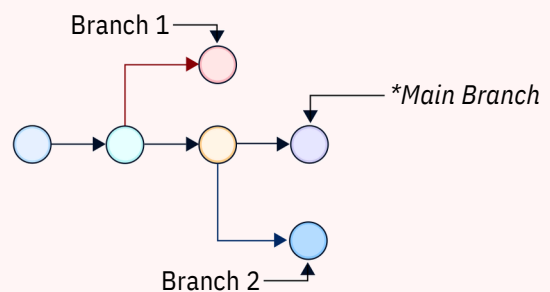
$ git branch [branch-name]

Create a new branch at the current commit.

[branch-name]

git branch [branch-name]

Main Branch
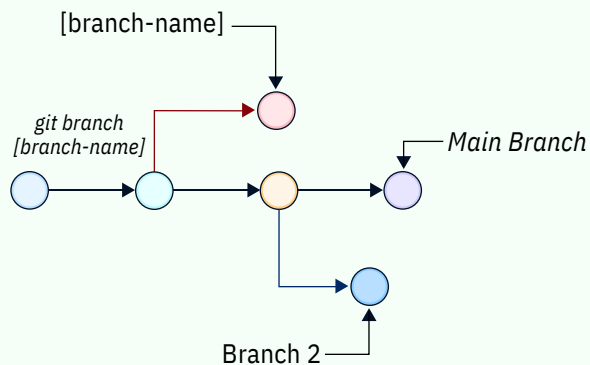
Branch 2

$ git merge [branch]
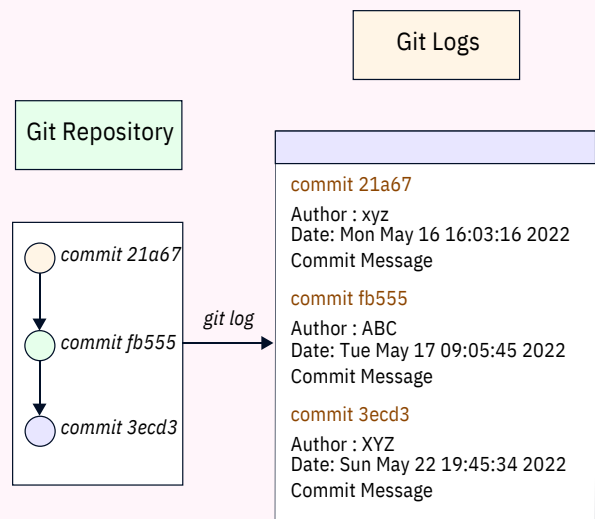
Merge the specified branch's history into the current one.

New Feature

git merge

Main Branch
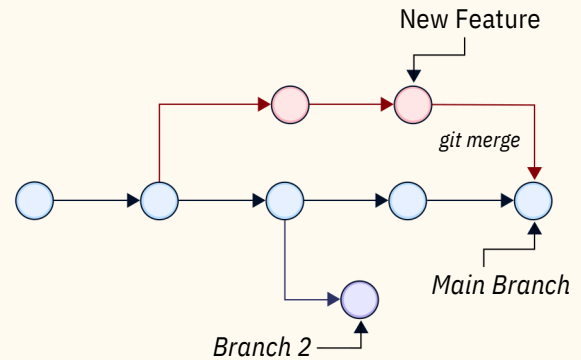
Branch 2

$ git checkout

Switch to another branch and check it out into your working directory.

Branch_1

Main Branch

git checkout Branch_1

*Branch_1

Main Branch
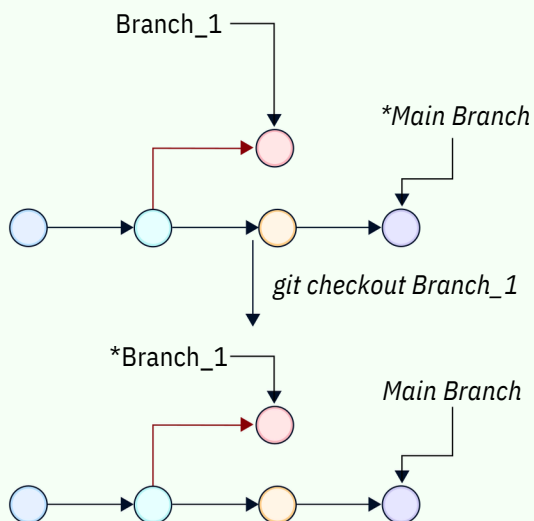
$ git log

Add a file as it looks now to your next commit (stage).

Git Logs

Git Repository

commit 21a67

commit fb555

commit 3ecd3

git log

commit 21a67
Author : xyz
Date: Mon May 16 16:03:16 2022
Commit Message

commit fb555
Author : ABC
Date: Tue May 17 09:05:45 2022
Commit Message

commit 3ecd3
Author : XYZ
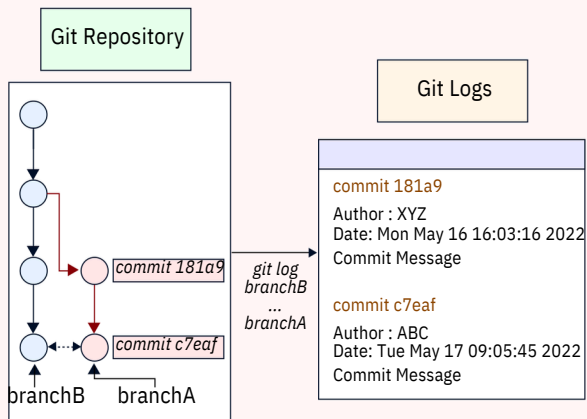Date: Sun May 22 19:45:34 2022
Commit Message

SCALER Topics

INSPECT & COMPARE
Configuring user information,
initializing and cloning repositories.

$ git log branchB..branchA

Show the commits on branchA that
are not on branchB.

Git Repository

Git Logs

commit 181a9

commit 181a9
Author : XYZ
Date: Mon May 16 16:03:16 2022
Commit Message

git log
branchB
...
branchA

commit c7eaf
Author : ABC
Date: Tue May 17 09:05:45 2022
Commit Message

commit c7eaf

branchB   branchA

$ git diff branchB...branchA

Show the diff of what is in branchA
that is not in branchB.
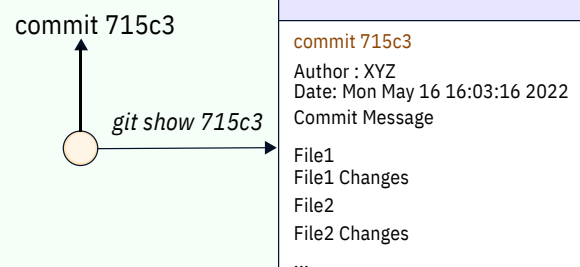
branchA

branchB

git diff branchB...branchA
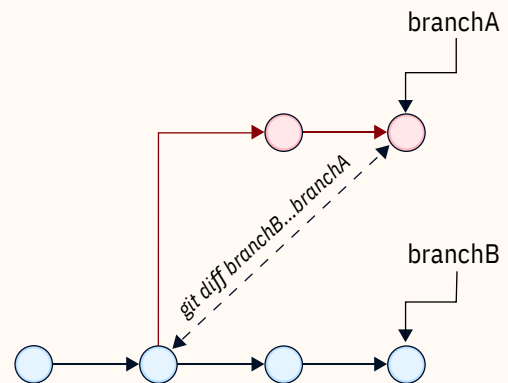
$ git log --follow [file]

Show the commits that changed file,
even across renames.

$ git show [SHA]

Show any object in Git in
human-readable format.

commit 715c3

commit 715c3
Author : XYZ
Date: Mon May 16 16:03:16 2022
Commit Message

git show 715c3

File1
File1 Changes
File2
File2 Changes
...

TRACKING PATH CHANGES
Versioning file removes and
path changes.

$ git rm [file]

Delete the file from the project and
stage the removal for commit.

git rm [file]

+ ◯

(Stage the removal)

Git Repository
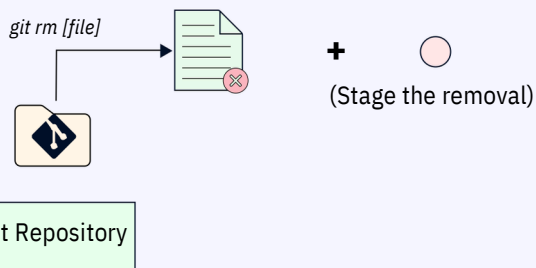
$ git mv [existing-path] [new-path]

Change an existing file path and
stage the move.

git mv [file]

+ ◯

(Stage the removal)

Git Repository
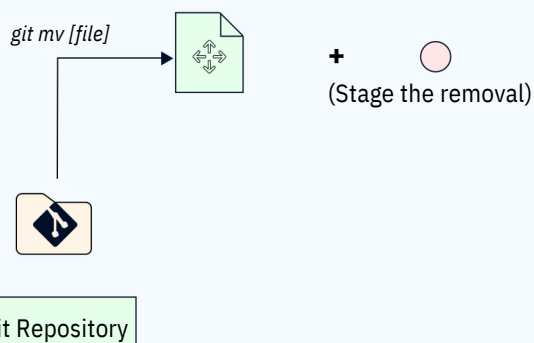
$ git log --stat -M

Show all commit logs with indication
of any paths that moved.

- -

IGNORING PATTERNS
Preventing unintentional staging
or commiting of files.

```
logs/
*.notes
pattern*/
```

Save a file with desired patterns as
.gitignore with either direct string matches
or wildcard globs.

.gitignore

```
logs/
*.notes
pattern*/
```
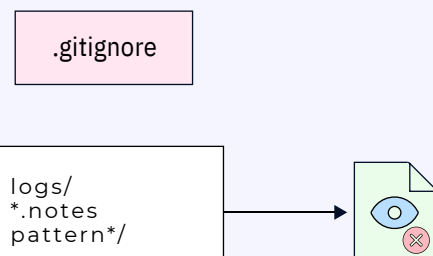
$ git config --global core.excludesfile [file]

System wide ignore pattern for all
local repositories

SCALER
Topics

Git Cheatsheet

SHARE & UPDATE
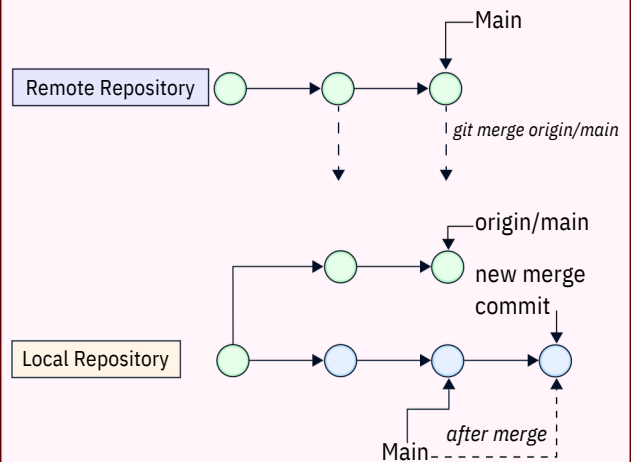Retrieving updates from another repository and updating local repos.

$ git remote add [alias] [url]

Add a git URL as an alias.
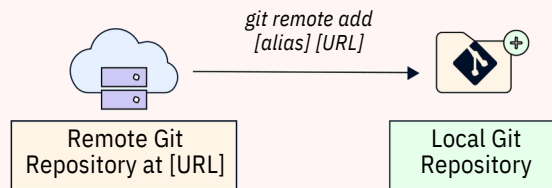
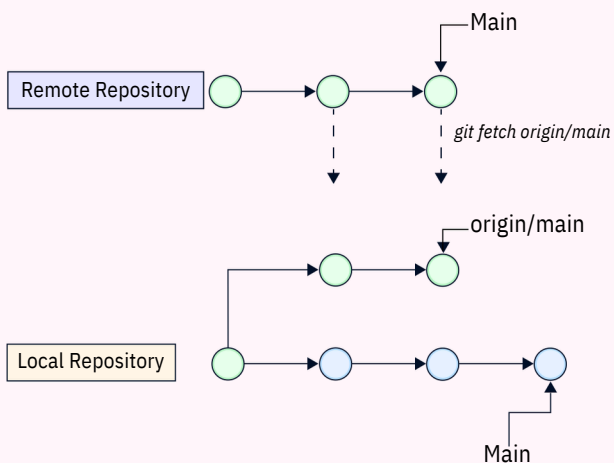## $ git merge [alias]/[branch]

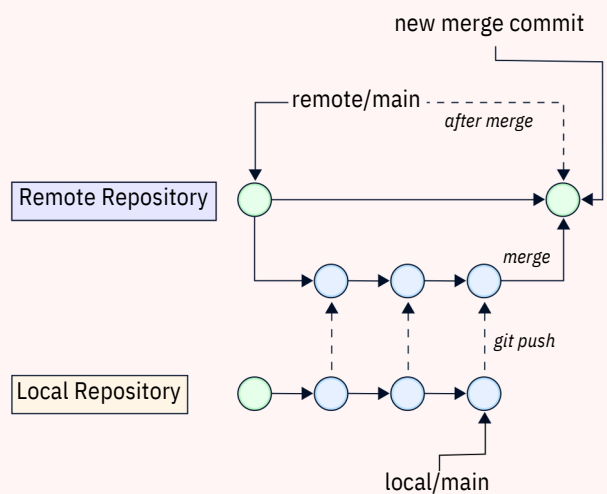Merge a remote branch into your current branch to bring it up to date.

$ git fetch [alias]

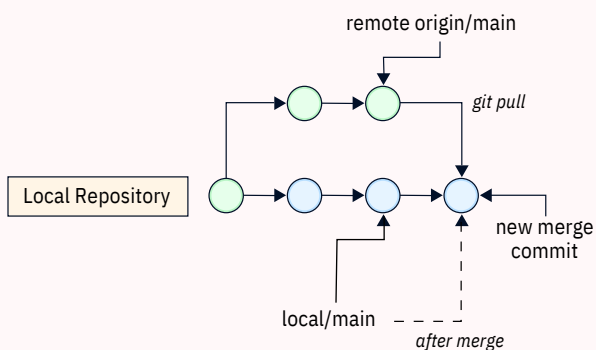Fetch down all the branches from that Git remote.

$ git push [alias] [branch]

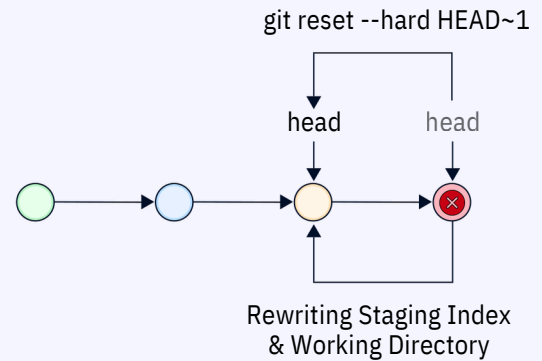Transmit local branch commits to the remote repository branch.

$ git pull

Fetch and merge any commits from the tracking remote branch.

$ git reset --hard [commit]

Clear staging area, rewrite working tree from specified commit.

git reset --hard HEAD~1

Rewriting Staging Index & Working Directory
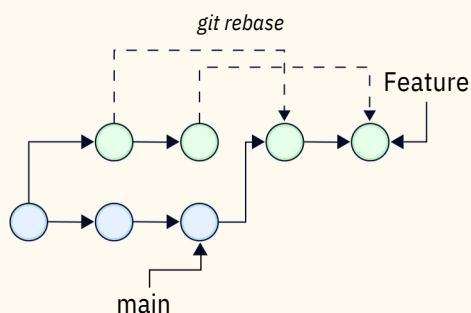

REWRITE HISTORY
Rewriting branches, updating commits and clearing history.

TEMPORARY COMMITS
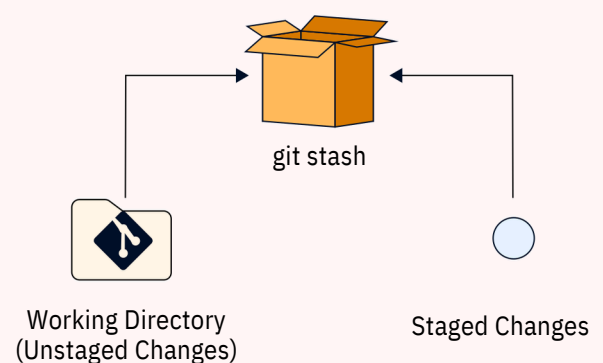Temporarily store modified, tracked files in order to change branches.

$ git rebase [branch]

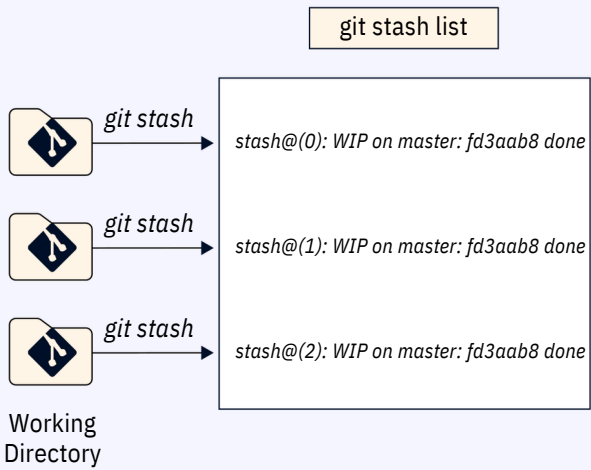Apply any commits of the current branch ahead of specified one.
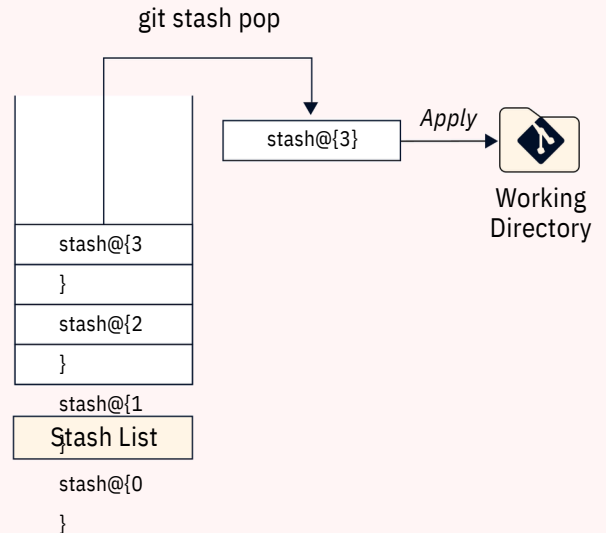

git rebase

Feature

main

$ git stash

Save modified and staged changes.

git stash

Working Directory
(Unstaged Changes)

Staged Changes

$ git stash list

List stack-order of stashed file changes.

git stash list

git stash → stash@(0): WIP on master: fd3aab8 done

git stash → stash@(1): WIP on master: fd3aab8 done

git stash → stash@(2): WIP on master: fd3aab8 done

Working
Directory

$ git stash pop

Write working from the top of
the stash stack.

git stash pop

stash@{3} —*Apply*→ Working
Directory

stash@{3
}
stash@{2
}
stash@{1
}
Stash List
stash@{0
}

$ git stash drop

Discard the changes from the top
of the stash stack.

git stash drop

stash@{3} ⊗
(Discard)

stash@{3
}
stash@{2
}
stash@{1
}
Stash List
stash@{0
}

Git Cheatsheet