

Cheat Sheets

How Git Work

Git ??????Modified/Untracked/Staged/Committed

1. **Modified/Untracked:** ?????????? `git add`
2. **Staged:** ??? `git add` ????? `git commit`
3. **Committed:** ??? `git commit`

?? 2 ? 3 ?????? Git ????????



HOW GIT WORK

Workspace

.git
src
index.html

Stage



git add

git reset



git commit

git pull

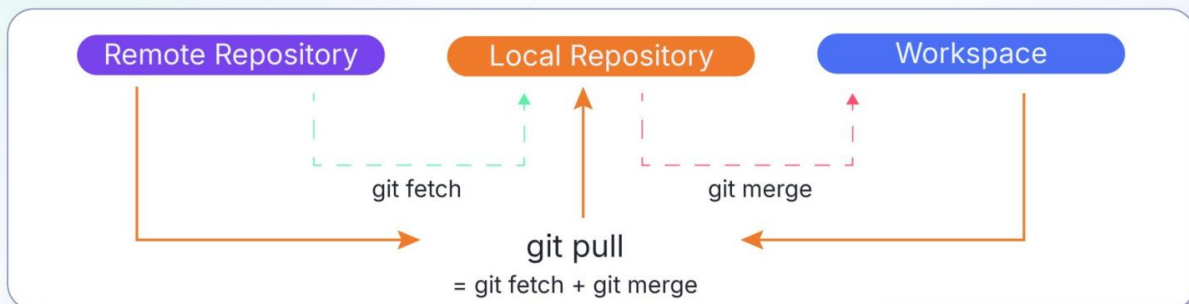
Remote Repository

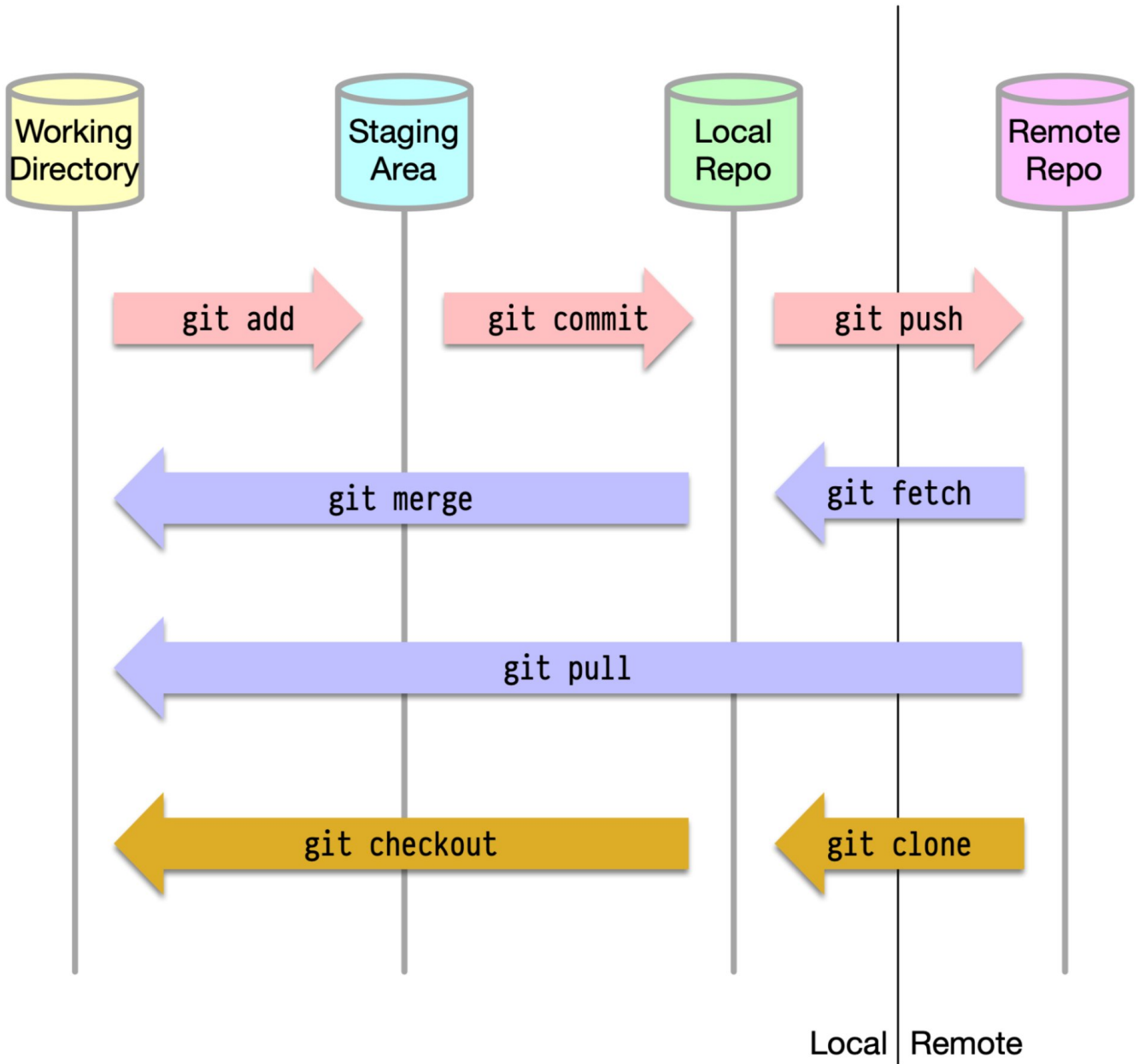
Gitlab
Github
Bitbucket

git push

git fetch

Local Repository





Git Commands

GIT Cheatsheet

Essential Commands for Developers

Repository Setup

```
git init                Initialize repo
git clone <url>         Clone repository
git config --global user.name Set username
                        "Name"
git config --global user.email Set email
                        "email"
```

Basic Commands

```
git status              Check status
git add <file>         Stage file
git add .               Stage all
git commit -m "msg"    Commit changes
git commit -am "msg"  Add & commit
git log                View history
git diff               Show changes
```

Branching

```
git branch              List branches
git branch -a           List all branches
git branch <name>      Create branch
git checkout <branch>  Switch branch
git checkout -b <name> Create & switch
git merge <branch>     Merge branch
git branch -d <name>   Delete branch
```

Remote Operations

```
git remote              List remotes
git remote -v           Show remote URLs
git push <remote> <branch> Push to remote
git pull <remote> <branch> Pull from remote
git fetch               Fetch changes
```

Undo Changes

```
git reset <file>       Unstage file
git reset --hard        Reset to HEAD
git checkout <file>     Discard changes
git revert <commit>     Revert commit
```

Advanced

```
git stash               Stash changes
git stash pop           Apply stash
git rebase <branch>    Rebase branch
git tag <name>         Create tag
git log --oneline      Compact log
```

Git Commands Cheat Sheet

Basics

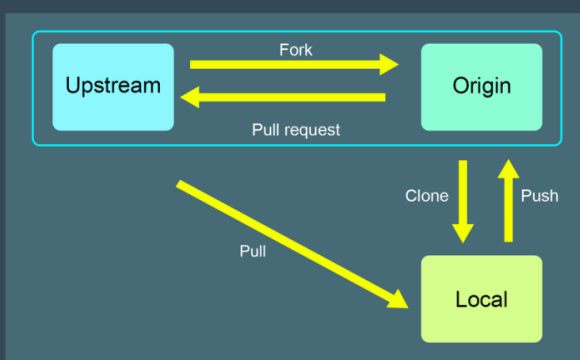
git help [commands]

Use this when you are stuck

Terminology

main	default primary branch
origin	default upstream branch
HEAD	current branch
HEAD^	parent of HEAD
HEAD~3	great grandparent of HEAD

Start to Work



git init

create a new local Git repo

git clone

copy a repo

git pull

fetch remote version and update local branch

git add [file]

stage changes to tracked and untracked files

git commit

commit previously staged changes

git push origin HEAD

push local changes to the origin

Branches

Git Branch Workflow



git branch --all

list all the local and remote branches

git checkout hotfix

change to an existing branch called hotfix

git checkout main

git merge hotfix

merge branch changes from hotfix to main

git log --graph --oneline

show a pretty branch history

Conflicts

git diff

see specific local changes

git diff --ours

compare the working tree with our branch

git diff --theirs

compare the working tree with their branch

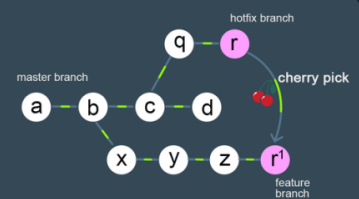
Useful Tools

git archive

create release tarball

git cherry-pick [commit-id]

enable arbitrary Git commits to be picked by reference





12 GIT COMMANDS

git init



Initializes a new Git repository in the current directory.

```
$ mkdir railsapp
$ cd railsapp
$ git init
```

git add



Stages changes in the current directory and sub directories.

```
$ echo "# RAILS APP" > README.md
$ echo "Demo" >> README.md
$ git add README.md
```

git commit



Commits staged changes with a commit message.

```
$ git add app.css
$ git commit -m "Initial commit"
```

git push



Pushes local changes to a remote repository.

```
$ git add .
$ git commit -m "Update files"
$ git push origin main
```

git pull



Pulls changes from a remote repository and merges into the local repository.

```
$ git status
$ git pull origin main
$ git log --oneline
```

git remote



Add a remote repository, view it, and rename it.

```
$ git remote add origin <url>
$ git remote -v
$ git remote rename origin upstr
```

git branch



List all branches, create a new branch with a specified name, and confirm it's created.

```
$ git branch
$ git branch feature-login
$ git branch
```

git fetch



Retrieves the latest data from a remote repository - but it doesn't integrate any of this new data into your working files.

```
$ git fetch origin
```

git checkout



Switches to the specified branch.

```
$ git branch
$ git checkout feature-login
$ git branch
```

git merge



Merges the specified branch into the current branch (in this case main).

```
$ git checkout main
$ git merge feature-login
$ git log --oneline
```

git status



Displays the status of the repository, including any uncommitted changes

```
$ git status
$ echo "Data Added" >> hello.txt
$ git status
```

git reset



Resets the current branch to the specified commit

```
$ git log --oneline
$ git reset --hard <commit-hash>
$ git status
```

GIT FUNDAMENTALS

Git Fundamentals

Version Control System

Track changes in code with history, enabling collaboration and reverting to previous states

Distributed Architecture

Works offline with full local repository; no constant network connection needed

Powerful Branching

Create isolated environments for feature development, fixes, and experiments

Lightweight

Small footprint with high performance even for large projects

Open Source

Free and widely supported across platforms

Multiple Remote Support

Can connect to multiple remote repositories simultaneously

Basic Operations

```
git status # Check status of working directory
git add <file> # Add file to staging area
git add . # Add all changes to staging area
git commit -m "Commit message" # Commit staged changes
git log # View commit history
git diff # Show changes between working directory and staging
git diff --staged # Show changes between staging and last commit
```

Core Git Concepts

Repository States

Working Directory:

Where you edit files

Staging Area:

Prepared changes for commit

Local Repository:

Committed history on your machine

Remote Repository:

Shared server repository (GitHub, GitLab, BitBucket, etc.)



Branching & Merging

```
git branch # List branches
git branch <branch-name> # Create new branch
git checkout <branch-name> # Switch to branch
git checkout -b <branch-name> # Create and switch to new branch
git merge <branch-name> # Merge branch into current branch
git branch -d <branch-name> # Delete branch locally
```

Basic Workflow



Git Remote Repositories



Essential Git Commands

```
git config --global user.name "Your Name"
git config --global user.email "your.email@example.com"
git init #Initialize new repository
git clone <repository-url> #Clone existing repository
```

Remote Operations

```
git remote -v # List remote repositories
git remote add <name> <url> # Add remote repository
git fetch <remote> # Download objects from remote
git pull <remote> <branch> # Fetch and merge changes
git push <remote> <branch> # Push commits to remote
git push -u <remote> <branch> # Push and set upstream
```

sysxplore.com

Revision #11

Created 2025-04-01 20:15:49 CST by A-Lang (Admin)

Updated 2025-09-15 09:45:43 CST by A-Lang (Admin)