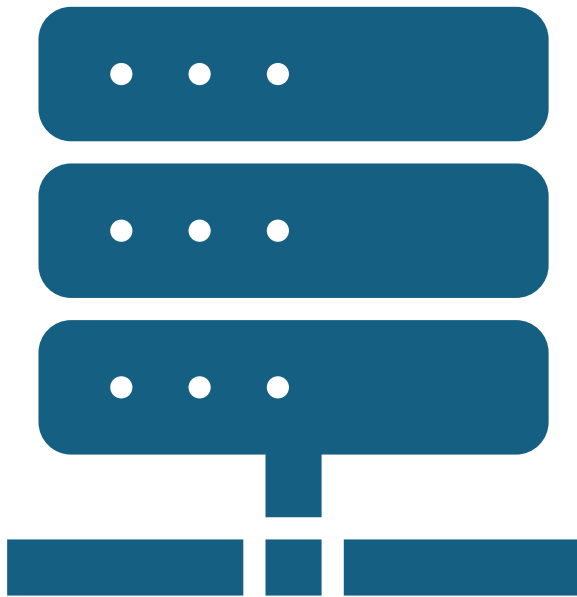




Introduction

- **Git:** A distributed version control system.
- **GitHub:** A platform to host and collaborate on Git repositories.
- 🗝️ *Main goal:* Manage code efficiently and work in teams seamlessly.



Git History

- 2005 – Git is Created by **Linus Torvalds**
- At the time, the **Linux kernel project** was using a proprietary version control system called **BitKeeper**.
- Design goals:
- Distributed
- Fast
- Secure
- Reliable branching and merging

Github History

- 2008 – GitHub Launches
- Founded by **Tom Preston-Werner, Chris Wanstrath, PJ Hyett, and Scott Chacon.**
- Public launch: **April 10, 2008**
- Purpose:
- Make Git easier to use
- Provide hosting for Git repositories
- Enable **collaboration, issue tracking, and code review**



Git != GitHub



GIT

version Control System is a tools that helps to track changes in code

Git is a Version Control System. It is:

popular

free & Open Source

fast and Scalable

Primarily 2 use of Git



1) TRACK THE HISTORY





2) COLLABORATE



GitHub

- Website that allows developers to store and manage their code using git.
- <https://github.com>
- Provides tools for:
 - Code review
 - Issue tracking
 - Pull Requests (PRs)
 - Collaboration
 - Project management

Git vs GitHub

Feature	Git	GitHub
Type	Version Control Tool	Cloud Hosting Platform
Works Locally		 (Needs Internet)
Collaboration	Limited (local only)	Full (PRs, Issues, Comments)
Interface	Command Line	Web-based GUI

GitHub Account



. CREATE A NEW
REPOSITORY






. MAKE OUR FIRST
COMMIT

Create a New Repository

GitHub, Inc. [US] | https://github.com/new

Search or jump to...

Pull requestsIssuesMarketplaceExplore




Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

Repository name

 SiliconOrchid

 /

HelloWorld

Great repository names are short and memorable. Need inspiration? How about [solid-succotash](#).

Description (optional)

My first Git Repository

☐ Public

Anyone can see this repository. You choose who can commit.

☒ Private


You choose who can see and commit to this repository.

☒ Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: Python

Add a license: None



Create repository

Setting up Git



VISUAL STUDIO CODE



GIT --VERSION

Configuring Git

```
git config --global user.name "my name"
```



```
git config --global user.email  
"someone@gmail.com"
```



```
git config --list
```



```
C:\Users\saura>git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.email=sauravacharya236@gmail.com
user.name=SauravAcharya23
core.longpaths=true
```

Clone & Status

Clone - Cloning a repository on our local machine

status - displays the state of code

- `git clone <-some link->`

- `git status`

```
saura@TUFGAMMER MINGW64 ~/OneDrive/Desktop
$ git clone https://github.com/SauravAcharya23/demo.git
Cloning into 'demo'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.

saura@TUFGAMMER MINGW64 ~/OneDrive/Desktop
$
```

```
PS C:\Users\saura\OneDrive\Desktop\demo> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\saura\OneDrive\Desktop\demo>
```

- Untracked
 - new files that git doesn't yet track
- Modified
 - changed
- Staged
 - file is ready to be committed
- Unmodified
 - unchanged

```
Untracked files:
(use "git add <file>..." to include in what will be committed)

about.txt.txt
index.txt.txt
```

```
Changes to be committed:
(use "git rm --cached <file>..." to unstage)

new file:   about.txt.txt
new file:   index.txt.txt
```



Add and Commit

add- adds new or changed files in your working directory to the Git staging area.

git add <-file name->

git add .



commit - It is the record of change

git commit -m "some message"

```
HOME@LAPTOP-MSQ6RBU0 MINGW64 /g/git-tutorial (master)
$ git add index.txt
```

```
HOME@LAPTOP-MSQ6RBU0 MINGW64 /g/git-tutorial (master)
$ git commit -m "first commit"
[master (root-commit) 93b8b09] first commit
2 files changed, 2 insertions(+)
create mode 100644 about.txt.txt
create mode 100644 index.txt.txt
```


Push Command

Push

- upload local repo
content to remote
repo

- git push origin main

```
HOME@LAPTOP-MSQ6RBUO MINGW64 /g/git-tutorial (master)
$ git push origin master
To github.com:thesparkler/git-tutorials.git
 ! [rejected]        master -> master (non-fast-forward)
error: failed to push some refs to 'git@github.com:thesparkler/git-tutorials.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Init Command

- init - used to create a new git repo
- git init
- git remote add origin <-link->
 - This command adds a new remote repository named "origin" to your local Git configuration.
- git remote -v (to verify remote)
 - This command lists all configured remote repositories and their URLs.
 - The -v flag stands for "verbose" and shows both fetch and push URLs.
- git branch (to check branch)
 - This command lists all local branches in your repository.
- git branch -M main (to rename branch)
 - This command renames the current branch to "main".
 - -M is a combination of --move --force, which forces the rename even if "main" already exists.
- git push origin main
 - This command pushes your local "main" branch to the remote repository named "origin".

```
nothing to commit, working tree clean
PS C:\Users\saura\OneDrive\Desktop\demo> get-ChildItem -Hidden

Directory: C:\Users\saura\OneDrive\Desktop\demo

Mode                LastWriteTime         Length Name
----                -
d--h--            6/28/2025   1:11 PM             .git

PS C:\Users\saura\OneDrive\Desktop\demo> |
```

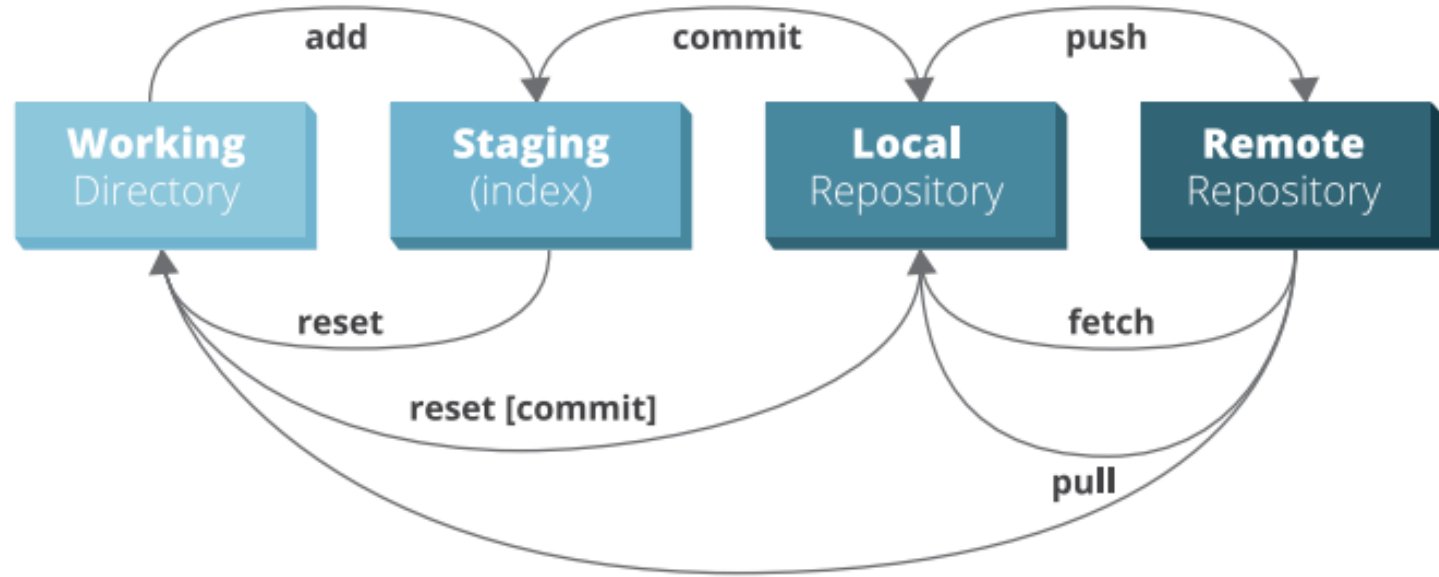
```
PS C:\Users\saura\OneDrive\Desktop\demo> git branch
* main
PS C:\Users\saura\OneDrive\Desktop\demo> |
```

```
HOME@LAPTOP-MSQ6RBU0 MINGW64 /g/git-tutorial
$ git init
Initialized empty Git repository in G:/git-tutorial/.git/
```



WorkFlow

- Clone Repository
- Create Branch
- Make Changes
- Add & Commit
- Push to GitHub
- Create Pull Request
- Review & Merge





Git Branches

Branch Commands

`git branch` (to check branch)

- Lists all local branches in your Git repository.

`git branch -M main` (to rename branch)

- Renames the current branch to main (or another specified name).

`git checkout <-branch name->`
(to navigate)

- Switches to an existing branch.

`git checkout -b <-new branch name->` (to create new branch)

- Creates a new branch and switches to it immediately.
- `-b` flag Create a new branch before checking it out.

`git branch -d <-branch name->`
(to delete branch)

- Deletes a local branch (only if it has been merged).

Alternate Branch Commands

```
git branch <-newbranch  
name->
```

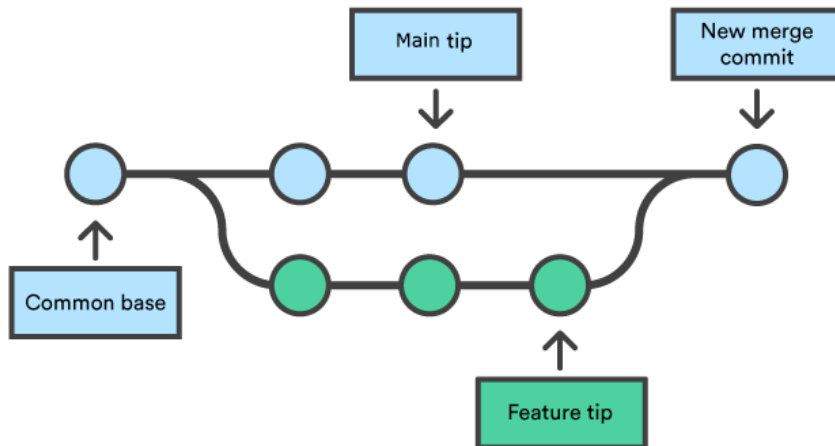
- Create new branch that you have specified

```
git checkout <-  
branchname->
```

- Switches to an existing branch.

```
HOME@LAPTOP-MSQ6RBUO MINGW64 /g/git-tutorial (master)  
$ git checkout newsBranch  
error: pathspec 'newsBranch' did not match any file(s) known to git.
```

Merging Code



```
minikube-demo git:(main) x git diff README.md
diff --git a/README.md b/README.md
index fb65379..6fc463a 100644
--- a/README.md
+++ b/README.md
@@ -1,3 +1,8 @@
+# Changelog
+
+3.21.24: Updated golang image
+
+# Getting started with minikube
```

Way 1

- `git diff <-branch name->` (to compare commits, branches, files & more)
- `git merge <-branch name->` (to merge 2 branches)

Way 2

- Create a PR (Pull Request)

Pull Request

- It lets you tell others about changes you've pushed to a branch in a repository on GitHub.

The screenshot shows the GitHub interface for the repository `axolo-co / api.axolo.co`. A yellow notification bar at the top states: `newbranch` had recent pushes less than a minute ago. A green button labeled `Compare & pull request` is next to it. Below the notification, the repository's branch status is shown: `newbranch` (selected), 18 branches, and 0 tags. A green box highlights the `newbranch` dropdown and the `Compare & pull request` button. The main content area shows the commit history for the `newbranch` branch, which is 1 commit ahead of `main`. The commit list includes:

Commit	Author	Message	Time
<code>4faf810</code>	cosydney	Update README.md	1 minute ago
<code>1,758</code>		commits	

The commit list also includes a table of files and their commit history:

File	Commit	Time
<code>.circleci</code>	committed circleci	3 months ago
<code>.github/workflows</code>	Update main.yml	5 months ago
<code>.husky</code>	test husky	6 months ago
<code>api</code>	PROD-39 Airtable DB for org and engineers (#773)	3 days ago
<code>config</code>	sentryconfig	2 months ago
<code>extensions</code>	sendChaosSlack avec notifications sans s (#739)	18 days ago
<code>json-data</code>	everything seems good !	6 months ago
<code>public</code>	first commit	8 months ago

On the right side, the `Contributors` section lists three contributors: `cosydney` (Sydney Cohen), `adrienpsl` (adrien), and `acoudouy`. The `Environments` section shows 21 environments, with `emojiback` being `Active` and `axolo-pipeli-feature-de-hjym6f` and `axolo-pipeli-handling-g-okdxg6` being `Inactive`. A link to `+ 18 environments` is also present.

Pull Command



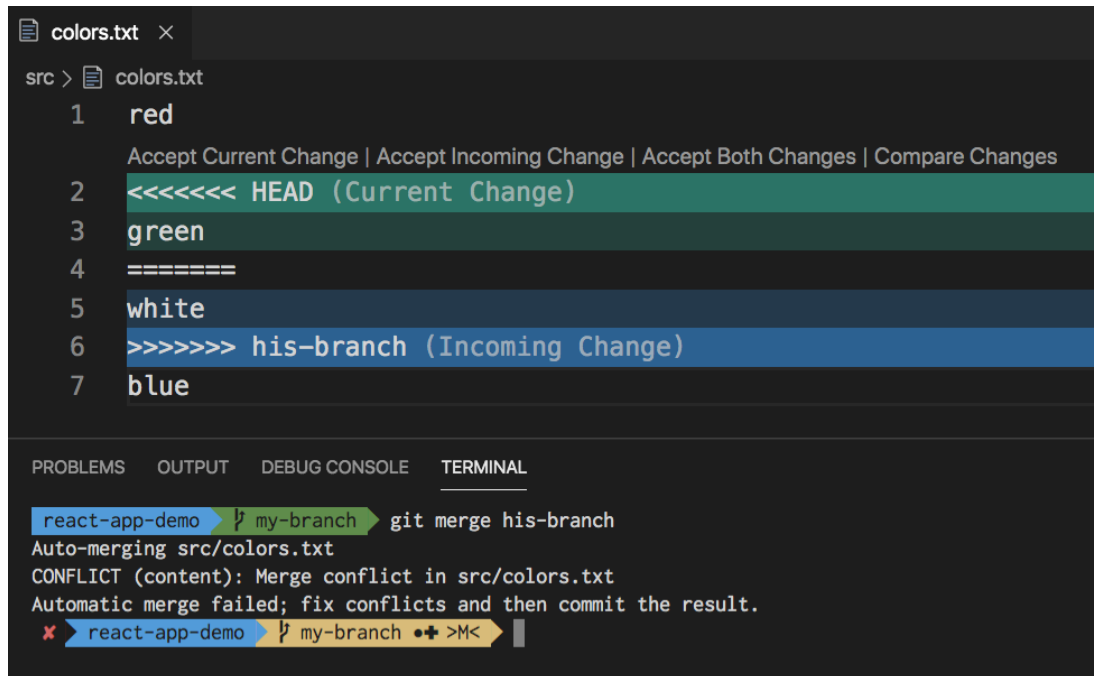
git pull origin main



- used to fetch and download content from a remote repo and immediately update the local repo to match that content.

```
HOME@LAPTOP-MSQ6RBU0 MINGW64 /g/git-tutorial (master)
$ git pull origin master
From github.com:thesparkler/git-tutorials
* branch      master      -> FETCH_HEAD
Merge made by the 'recursive' strategy.
 portfolio.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 portfolio.txt
```

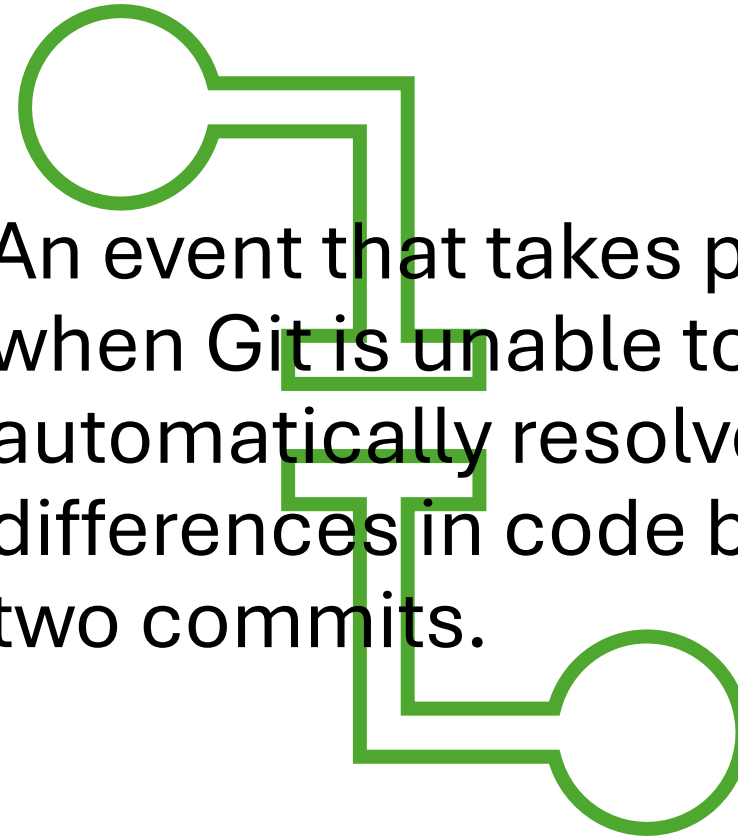
Resolving Merge conflicts



```
src > colors.txt
1  red
   Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
2  <<<<<< HEAD (Current Change)
3  green
4  =====
5  white
6  >>>>>> his-branch (Incoming Change)
7  blue

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
react-app-demo my-branch git merge his-branch
Auto-merging src/colors.txt
CONFLICT (content): Merge conflict in src/colors.txt
Automatic merge failed; fix conflicts and then commit the result.
x react-app-demo my-branch
```

- An event that takes place when Git is unable to automatically resolve differences in code between two commits.



Fork

A fork is a repository that shares code and visibility settings with the original "upstream" repository.

Fork is a rough copy.

