day-1

Git & GitHub Fundamentals

> **Note**
>
> You should have installed git yesterday or already have it on your system

Today we're diving into Git and GitHub - the backbone of modern development.

Version Control with Git

Think of Git as your code's time machine. Every developer has been there - you make changes, break something, and wish you could go back. That's exactly what Git lets you do.

Git tracks every change in your code as a series of snapshots. Each snapshot (or commit) represents a specific version of your project. The beauty is you can jump between these versions anytime.

The place in which git tracks your files is called a repository (or repo).

- Git in 100 seconds

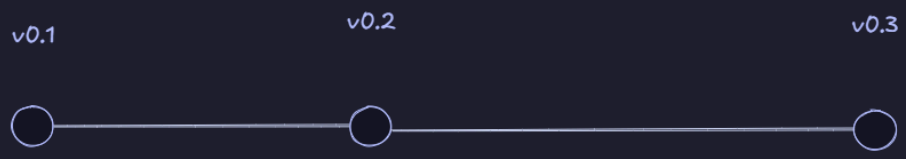
v0.1

```c
#include <stdio.h>


int main(){
  printf("All things working!\n");
  return 0;
}
```

v0.2

```c
#include <stdio.h>


int main(){
  printf("All things working!\n");

  prrrrint("Errors Now!\n");
  return 0;
}
```

... So What if those errors were in 50 files?



At each version you take a Snapshot of the source code (imagine a screenshot of the edited files)

---

v0.1

```c
#include <stdio.h>


int main(){
  printf("All things working!\n");
  return 0;
}
```

initial version

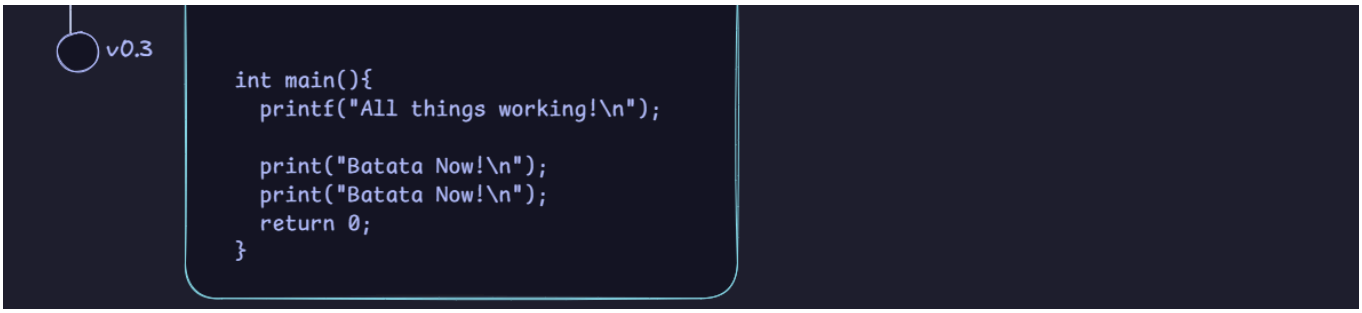
v0.2

```c
#include <stdio.h>


int main(){
  printf("All things working!\n");

  print("Batata Now!\n");
  return 0;
}
```

modified version

```c
#include <stdio.h>
```

modified version



Setting up Git

See this on how to git-started

Essential Git Commands

Let's speed-run through the commands you'll use 99% of the time:

```
# Initialize a new Git repository
git init

# Check the status of your files
git status

# Stage changes for commit
git add <filename>     # Stage specific file
git add .             # Stage all changes

# Create a commit (snapshot)
git commit -m "your message here"

# Send changes to remote repository
# NOTE that you might have the origin branch as master instead of main
git push origin main #origin = remote repository


# Get latest changes from remote
git pull origin main
```

> Tip

> Your commit messages matter! Write them as if you're explaining to your future self what changed and why, example: `git commit -m 'feat: login feature'`.

GitHub - Your Code's Social Network

GitHub is where Git gets superpowers. It's like Instagram for code - but instead of sharing photos, you're sharing and collaborating on code projects.

Key GitHub features:

- Remote repository hosting
- Pull requests for code review
- Issue tracking
- Project management tools
- Documentation hosting

{Insert diagram showing GitHub's role in collaboration, including concepts like pull requests and issues}

Creating Your First GitHub Repository

To put your code online! Here's the important steps:

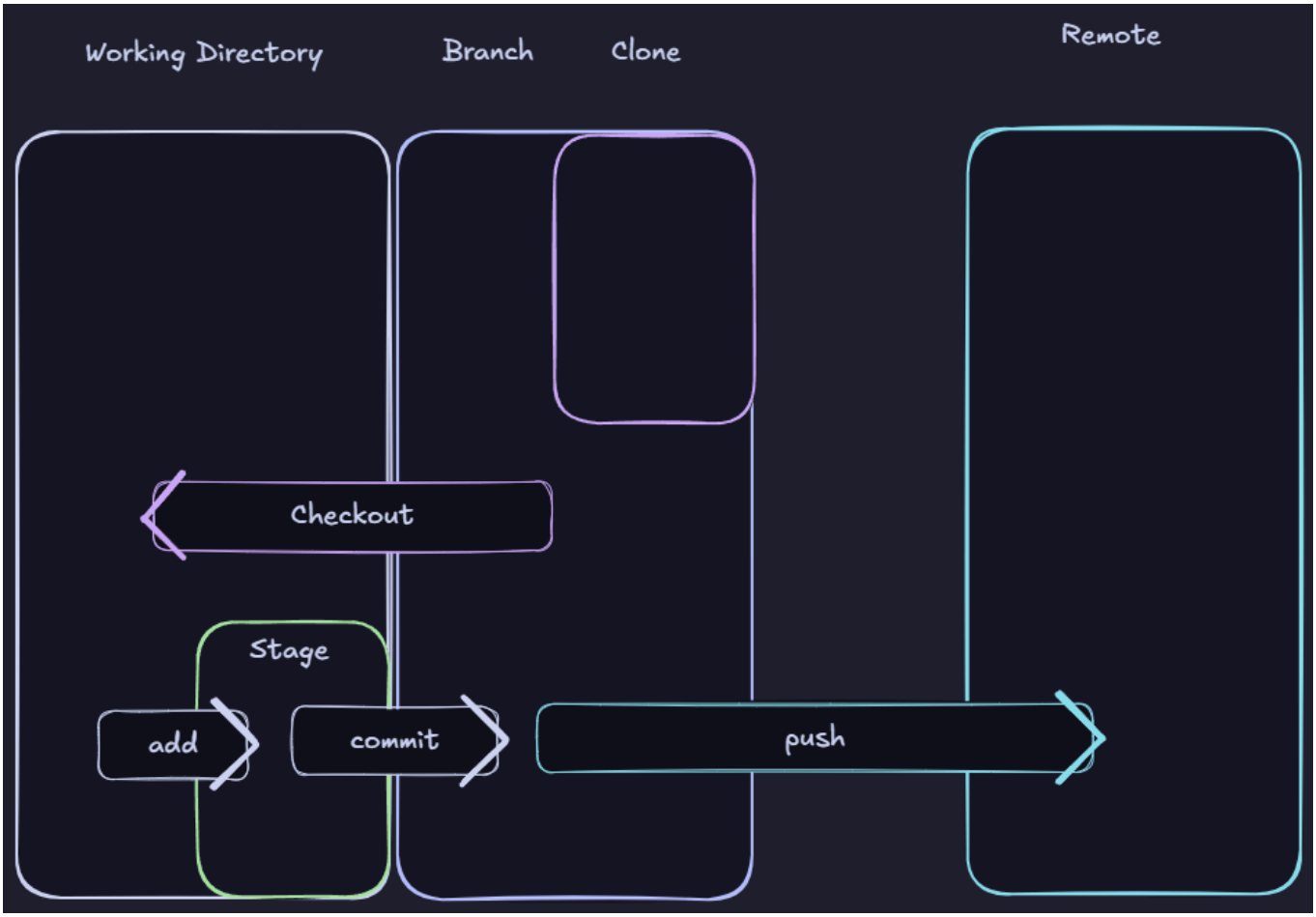
1. Head to github.com and log in
2. Click the "+" in the top right
3. Choose "New repository"
4. Fill in:
    - Repository name
    - Description (optional)
    - Public/Private setting
    - Initialize with README option (choose without for now)

After creating, GitHub gives you commands to connect your local repo:

```
# If starting fresh
git remote add origin https://github.com/username/repo.git
git branch -M main
git push -u origin main

# If you have existing code
```

```
git remote add origin https://github.com/username/repo.git
git push -u origin main
```

Visual Cheat sheet



Useful First Repositories to Create

- A learning notes repository
- A personal portfolio site
- Practice projects to learn Git workflows

Learn and Practice

Resources to level up:

- Git by Example Learn Git with examples

- Learn Git Branching Learn Git as a Game
- Awesome Git: A list of many Git Resources

> **Note**
>
> 💡 Remember: Git is a skill. The more you use it, the more comfortable you'll become. Don't be afraid to make mistakes - that's how you learn!

Next time we'll dive into Markdown. Until then 🚀

Questions? Drop them in our Discord Server!