



A solution to collaboration and codebase management  
*a presentation by Cole Lawrence*

A black and white puppy is lying on a sandy beach. The puppy has black fur with white markings on its face, chest, and paws. It is looking towards the camera. To the right of the puppy is a colorful, textured ball. The background shows a body of water and some greenery. A semi-transparent red banner is overlaid across the middle of the image, containing the text "Local Repository".

# Local Repository

# Git Repository

The **working copy** of code along with the **complete history** of the code base.

A small, fluffy orange hamster is the central focus of the image. It is sitting upright on a light-colored, textured surface, possibly a carpet. The hamster has large, dark eyes and long, thin whiskers. It is holding a small, dark object in its front paws, which are raised towards its mouth. The background is blurred, showing some indistinct shapes and colors. The overall lighting is soft and warm, giving the image a cozy feel. The word "Commits" is overlaid in white text on the hamster's face.

# Commits

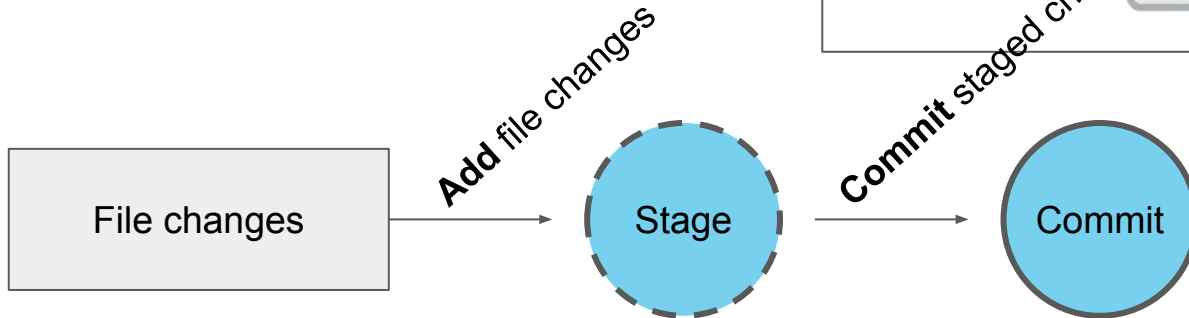
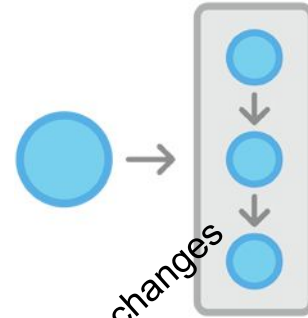
Snapshot of your codebase

# How to commit code

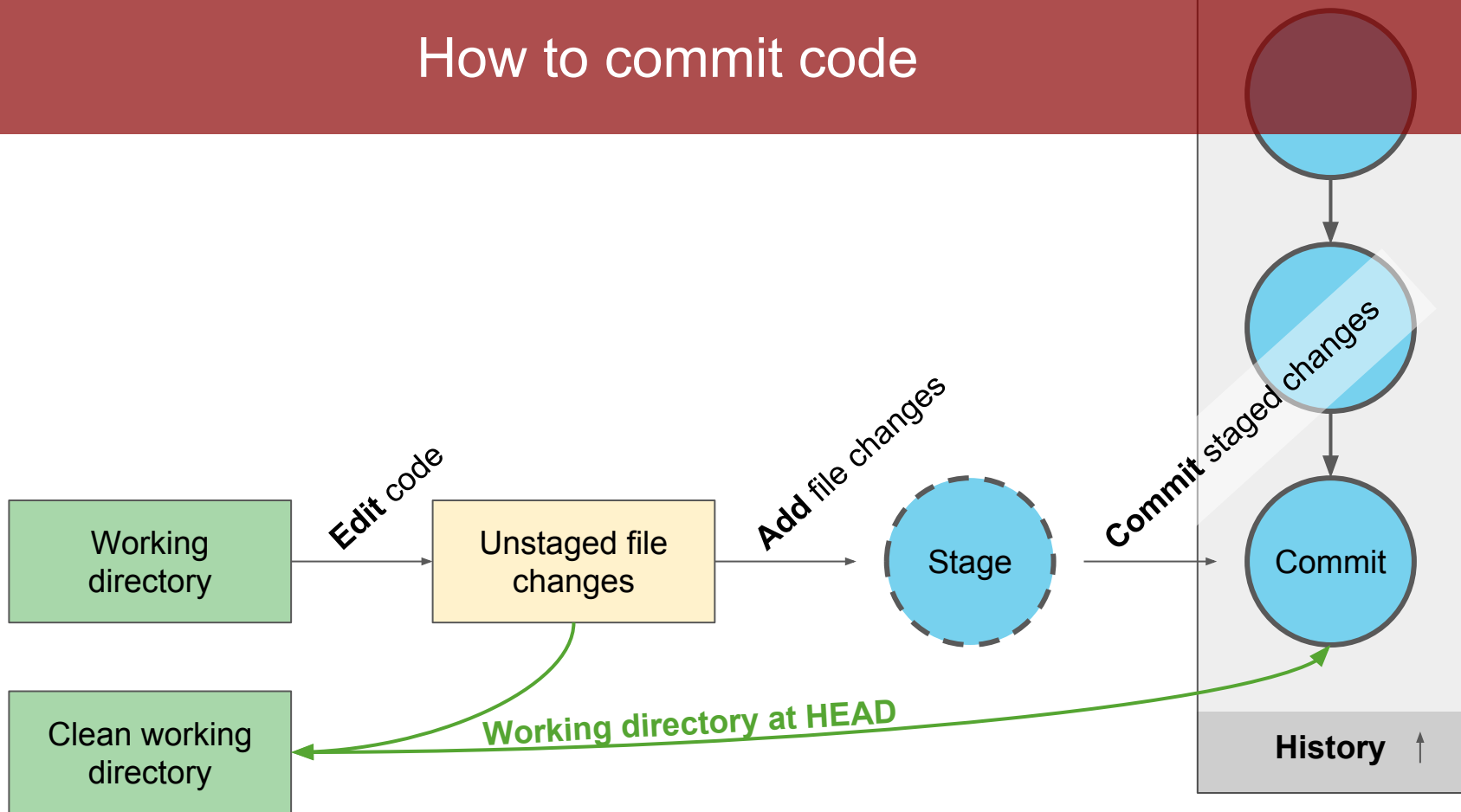
**Add** file changes to stage



**Commit** file changes in the stage to the history



# How to commit code



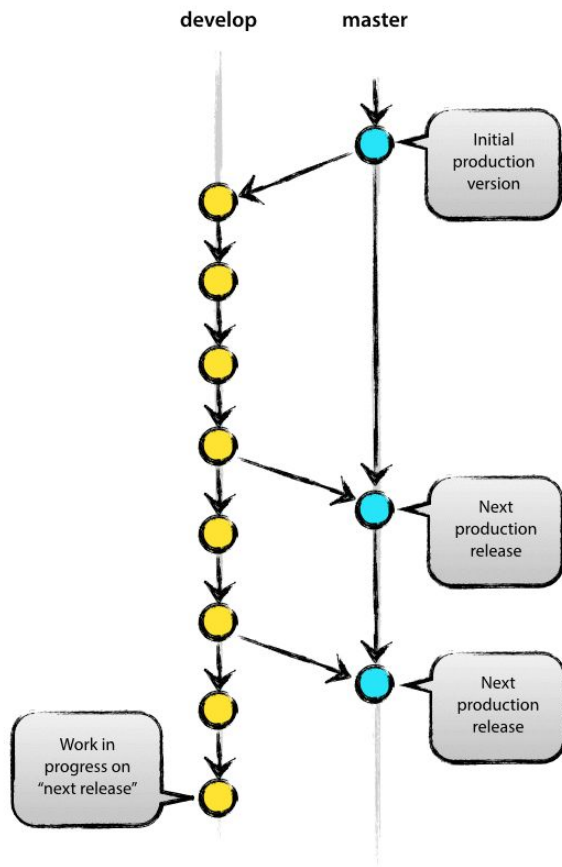


A golden retriever dog is the central focus, looking directly at the camera with a calm expression. Its fur is a rich golden color and appears soft and well-groomed. In the background, an American flag is visible, slightly out of focus, adding a patriotic or traditional feel to the image. The lighting is warm and soft, highlighting the dog's features. The overall composition is centered and balanced.

# Branches

Series of commits

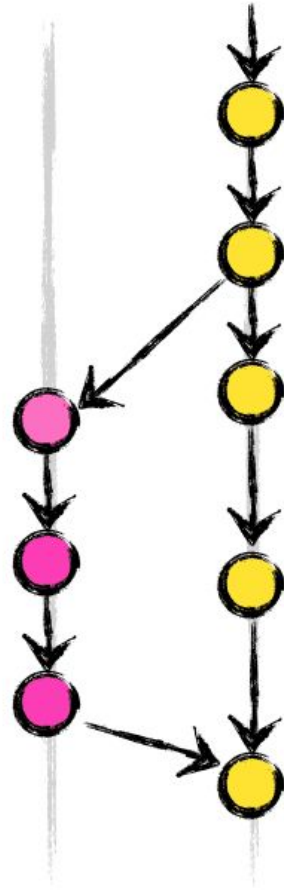
# Branches





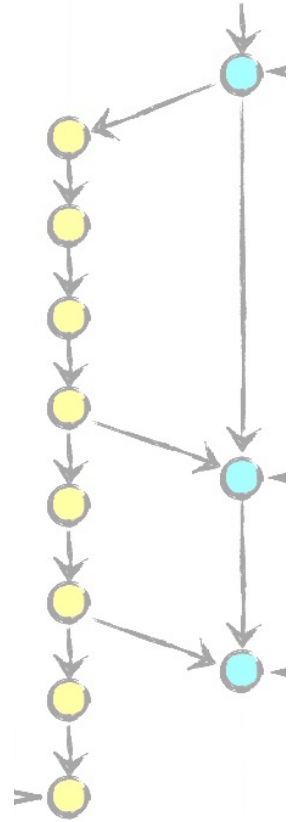
feature  
branches

develop



develop

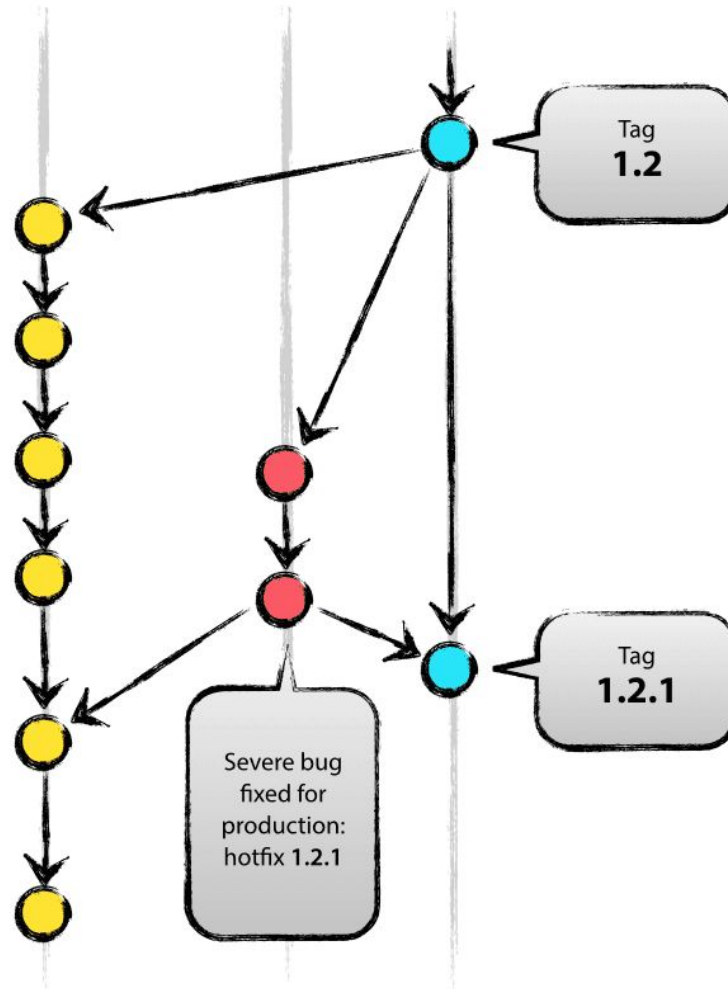
master



develop

hotfixes

master





Doin' Git

# Commits





files/folders

git client

History

Initialize Git in a folder

Create 14 files

Add files

Commit files "Setup website"

New index.html

Edit website to direct people to home page

Add file changes

Commit files "Add home page"

New login.html

Edit website to handle logins

Add file changes

Commit files "Add login page"



A photograph of a flock of sheep in a lush green field. A semi-transparent green horizontal banner is overlaid across the middle of the image, featuring the text "Learn by Doing" in white. The sheep are of various breeds, some with thick wool and some with small horns. The background shows rolling hills and a line of trees under a bright sky.

Learn by Doing

# Atlassian Git Tutorials

<https://www.atlassian.com/pt/git/tutorial/git-basics>

The screenshot shows the 'Git Tutorial: Basics' page on the Atlassian website. The header includes the Atlassian logo and a 'Recommend' button. Below the header, there's a navigation bar with links: Overview, Git Tutorials, Git Workflows, Migration, and Git Resources. A progress bar indicates the current position in the tutorial, with '1. GIT BASICS' highlighted. The main content area is titled '1. Git Basics' and contains an introductory paragraph about the tutorial's purpose. Below the text is a diagram showing a sequence of four circles, with the last one being blue. A sidebar on the left lists the tutorial sections: Overview, git init, git clone, git config, and git add. The main content area also features a section titled 'Setting Up a Git Repository' with the 'git init' command highlighted and a 'Learn more' link.

Git Tutorial: Basics

Atlassian

Recommend

## Git Tutorials

Overview Git Tutorials Git Workflows Migration Git Resources

1. GIT BASICS 2. UNDOING CHANGES 3. GIT BRANCHES 4. REWRITING GIT HISTORY 5. REMOTE REPOSITORIES

### 1. Git Basics

The *Git Basics* tutorial provides a succinct overview of the most important Git commands. First, the *Setting Up a Repository* section explains all of the tools you need to start a new version-controlled project. Then, the remaining sections introduce your everyday Git commands.

By the end of this module, you should be able to create a Git repository, record snapshots of your project for safekeeping, and view your project's history.

Overview

- git init
- git clone
- git config
- git add

### Setting Up a Git Repository

**git init**

The `git init` command initializes a new Git repository. If you want to place a project under revision control, this is the first command you need to learn.

[Learn more »](#)

<https://www.atlassian.com/git/tutorials/setting-up-a-repository>

The screenshot shows the 'Setting up a repository' page in the Atlassian Git Tutorials. The header includes the 'Tutorials' title and a search icon. A sidebar on the left lists the tutorial sections: Getting Started, Setting up a repository, Saving changes, Inspecting a repository, Viewing old commits, Undoing Changes, Rewriting history, Collaborating, Migrating to Git, and Advanced Tips. The main content area is titled 'Setting up a repository' and features a large icon of a database cylinder with a plus sign. Below the title is a paragraph introducing the tutorial's purpose. At the bottom, there's a paragraph about the expected learning outcomes.

Setting up a repository | Atlassian Git Tutorial

## Tutorials

Getting Started

Setting up a repository

- git init
- git clone
- git config

Saving changes

Inspecting a repository

Viewing old commits

Undoing Changes

Rewriting history

Collaborating

Migrating to Git

Advanced Tips

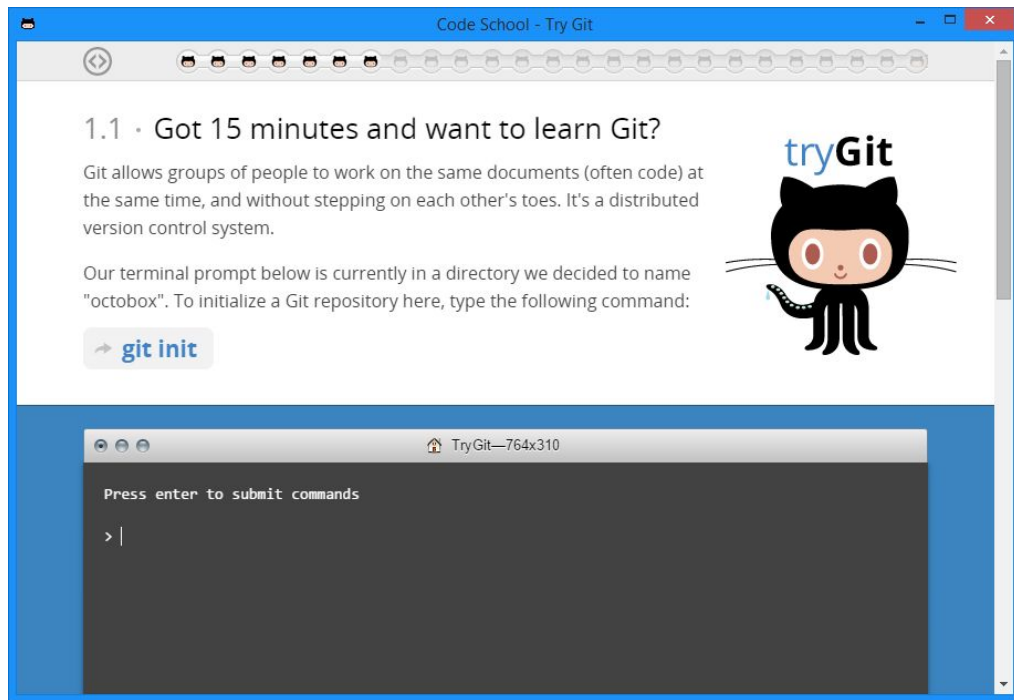
# Setting up a repository

This tutorial provides a succinct overview of the most important Git commands. First, the *Setting Up a Repository* section explains all of the tools you need to start a new version-controlled project. Then, the remaining sections introduce your everyday Git commands.

By the end of this module, you should be able to create a Git repository, record snapshots of your project for safekeeping, and view your project's history.

# GitHub Git Tutorials

Guided git commits in the browser - <https://try.github.io/levels/1/challenges/1>





# Git Illustrated

Less information than the git-help pages, and illustrated - <https://illustrated-git.readthedocs.org/en/latest/>

The screenshot shows the 'Git Illustrated Cheatsheet' web page. The sidebar on the left contains a 'Project Versions' section with 'latest' selected, an 'RTD Search' box, and a 'Table Of Contents' with links to 'GIT Model', 'Object & SHA1', 'Blob', 'Tree', 'Commit', 'Tags', 'Branch', 'Basic operations', 'Configuration', 'Staging', 'Commit', 'See your repository', 'Stash', 'Undoing', 'Branching and merging', 'Create a branch', 'Branch diverging', 'Merge', 'Rebase', and 'Cherry Pick'.

The main content area is titled 'GIT Illustrated Cheatsheet' and features a diagram of the 'GIT Model'. The diagram shows a 'Commit' (green box) with attributes: `+tree = aabbcc1`, `+author = Axel H.`, `+committer = Axel H.`, `+parent = aabbbs`, and `+comment = A single commit`. A 'Tag' (pink box) with attributes: `+object = abc123f`, `+type = commit`, `+tagger = Axel H.`, and `+comment = My tag comment` points to the commit. The commit points to a 'Tree' (blue box) with attributes: `+aabbccf: blob = file.txt`, `+ddeeff1: blob = other.txt`, and `+aabbcc2: tree = folder`. This tree points to another 'Tree' (blue box) with attributes: `+fffeeel: blob = nested.txt` and `+ddeee22: blob = other.txt`. Both trees point to 'Blob' (yellow boxes). The first tree points to 'Blob aabbccf' (content: file.txt content) and 'Blob ddeeff1' (content: other.txt content). The second tree points to 'Blob ffeeel' (content: folder/nested.txt content) and 'Blob ddeee22' (content: folder/other.txt content).

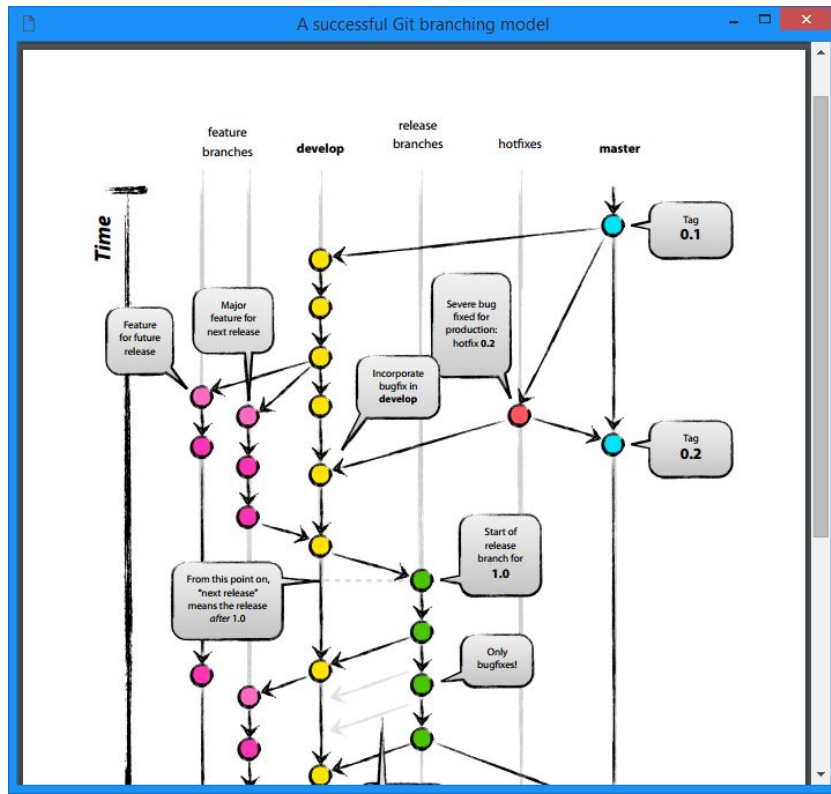
Below the diagram, the 'Object & SHA1' section lists the following points:

- Each object has a **type**, a **size** and a **content**
- Each object is identified by a 40-digit SHA1 hash of attributes
  - 6ff87c4664981e4397625791c8ea3bbb5f2279a3
- Each SHA1 can be shortened to the first digits
  - 6ff87c4664981e4397625
  - 6ff87c4

A 'Read the Docs' logo is visible in the bottom right corner.

# Git Branching Model

Git branching best-practices - <http://nvie.com/files/Git-branching-model.pdf>



# Sources

## Background images

- “Dog with ball” Andrew Pons, February 23, 2015, via *unsplash.com*
- “Hamster” Ricky Kharawala, February 27, 2015, via *unsplash.com*
- “Dog” Caleb Fisher, January 29, 2015, via *unsplash.com*
- “Bear” Thomas Lefebvre, August 18, 2014, via *unsplash.com*
- “Sheep” Ariana Prestes, January 27, 2015, via *unsplash.com*

## Branches images

- “A successful Git branching model” <http://nvie.com/posts/a-successful-git-branching-model/>

## How to commit code miscellaneous images

- “Git Basics - Atlassian” <https://www.atlassian.com/pt/git/tutorial/git-basics>

A photograph of a flock of sheep in a lush green field. The sheep are of various breeds, some with thick wool and some with more sparse fur. They are standing in a line, facing the camera. The background shows rolling hills and a line of trees under a clear sky. The entire image has a green tint.

# Cloning a Repository

Clone a git repository locally.



Pulling updates from Origin



# Why we are using **git**.

**Merge** file changes.

Team of developers may work on the same codebase, then merge their changes to the same files.

**Pull** remote repository



**Push** local repository

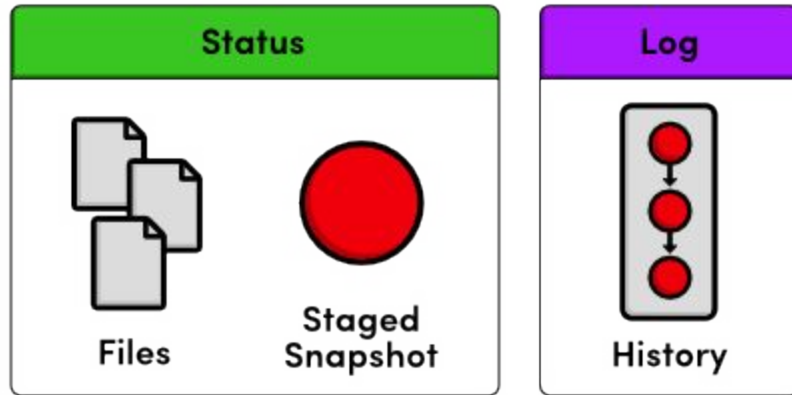


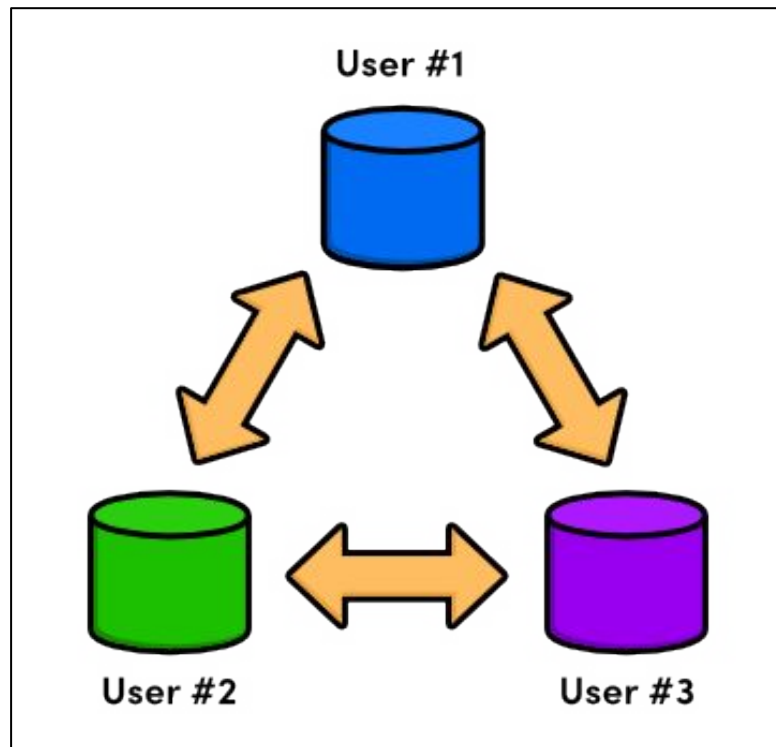
# Resources

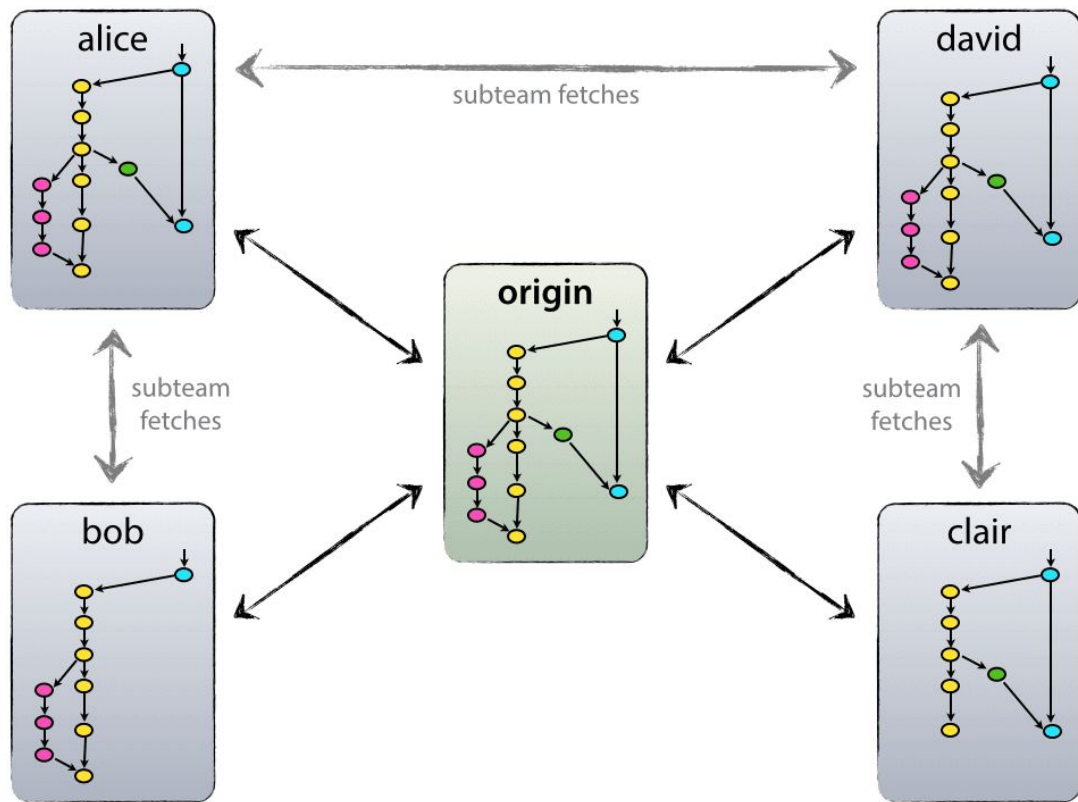
- Learn
  - [Documentation and getting started resources](#)
  - [Try Git in browser](#)
- Cool repositories
  - [google/deepdream](#) IPython Notebook combining images by movement, a study of neural networks, [blog post](#)
  - [golang/go](#) A new compiled language developed for productivity and asynchronous programming
  - [codemirror/CodeMirror](#) An in-browser code editor
  - [/explore](#) Explore up and coming developing software trending on Github



# Status output vs. Log output



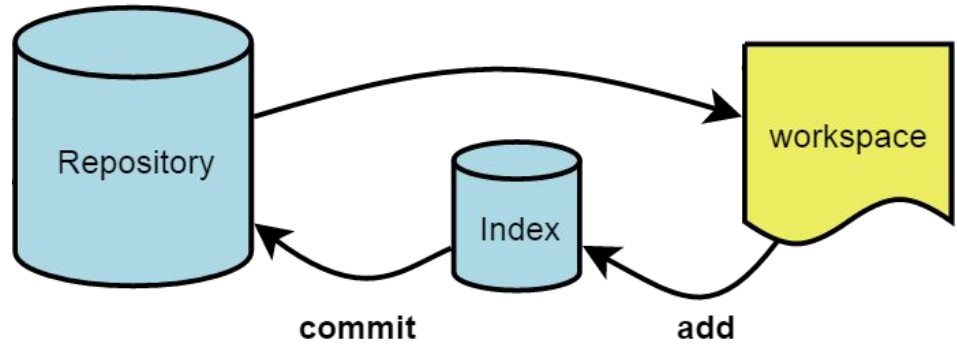
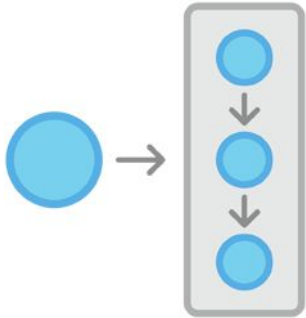




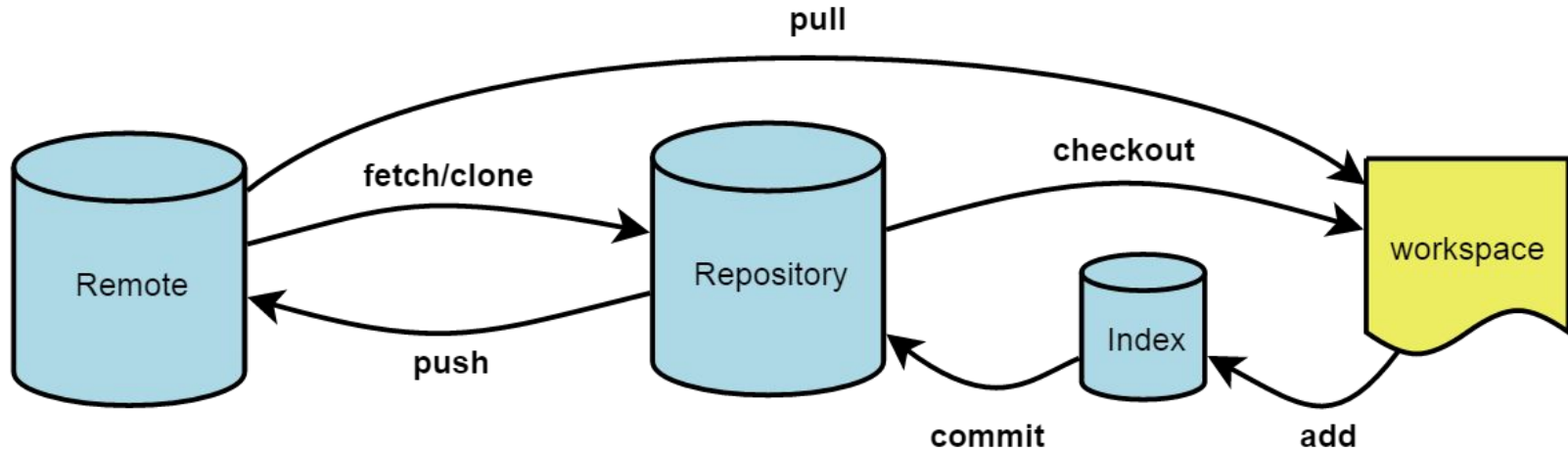
## Add file changes to stage



## Commit file changes



# Syncing your database





# Commands

```
git init
```

**Create** a Git repository in the current folder

```
git status
```

View the **status of each file** in a repository

```
git add <file>
```

**Stage** a file for the **next commit**

```
git commit
```

**Commit the staged files** with a descriptive message

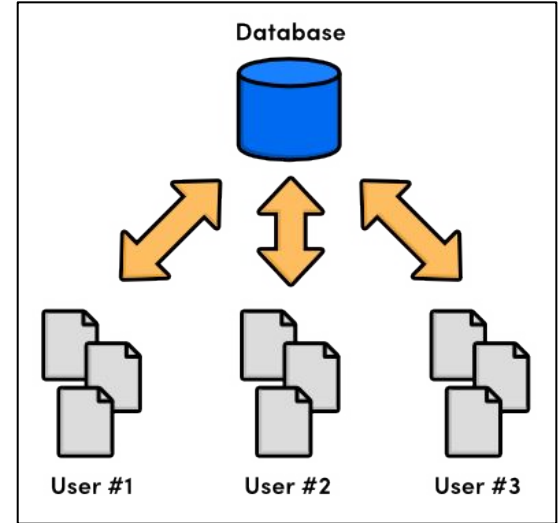
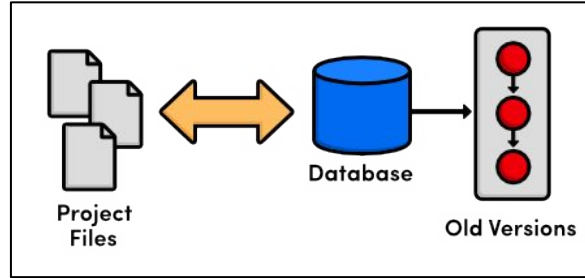
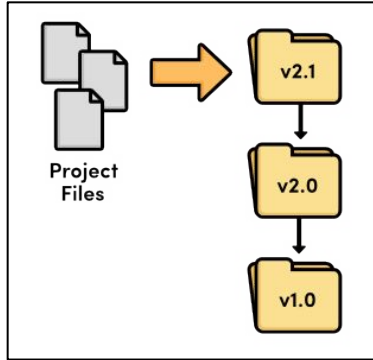
```
git log
```

View a repository's **commit history**.

# Version Control System

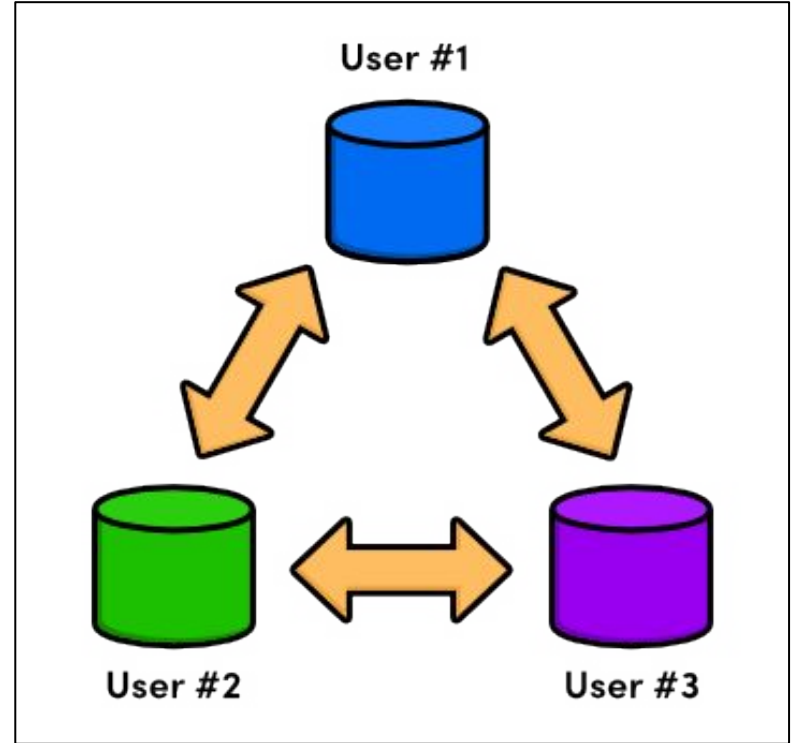
VCS

# How have we solved version control?



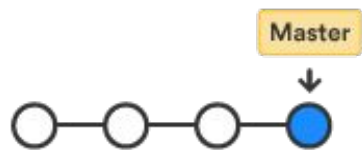
# Today

- **Local database of history**
- Everyone can **sync DBs easily**
- **Easy codebase management**



# Branching

## History



## Branches

