

Github icon

Overview

0. Install git and create Github account
1. What is version control?
2. What is git?
3. How does git work?
4. What is GitHub?
5. Quick example using git and GitHub
6. Break into teams



You can find the slides at:



0

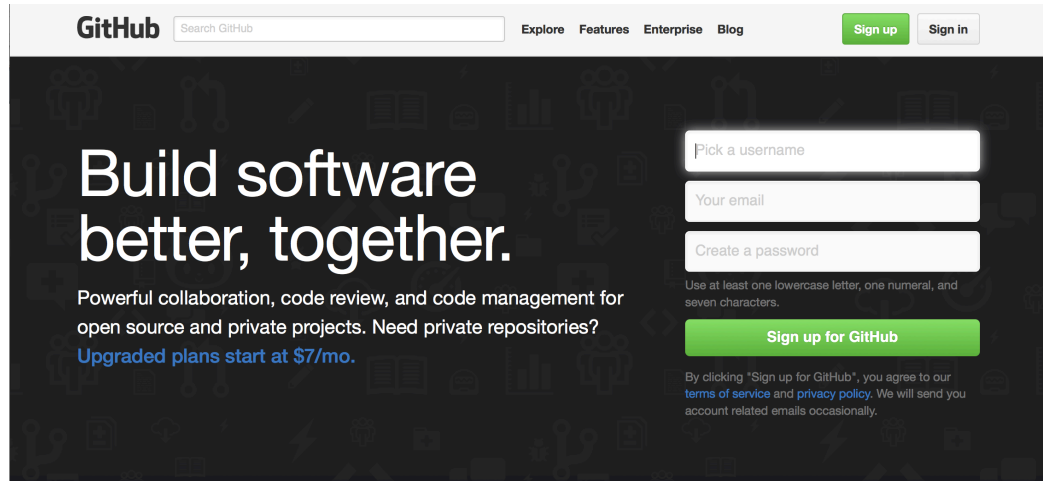
**Install git and create GitHub
account**

Install git

- **Linux (Debian)**
 - Command: `sudo apt-get install git`
- **Linux (Fedora)**
 - Command: `sudo yum install git`
- **Mac**
 - <http://git-scm.com/download/mac>
- **Windows**
 - <http://git-scm.com/download/win>

Create Github account

- **www.github.com**
- **Free for public repositories**



The screenshot shows the GitHub homepage with a focus on the sign-up process. The header includes the GitHub logo, a search bar, and navigation links for Explore, Features, Enterprise, and Blog. On the right, there are 'Sign up' and 'Sign in' buttons. The main content area has a dark background with the text 'Build software better, together.' and a description of GitHub's capabilities. To the right of this text are three input fields for 'Pick a username', 'Your email', and 'Create a password'. Below these fields is a green 'Sign up for GitHub' button. At the bottom, there is a disclaimer about agreeing to terms of service and privacy policy.

GitHub

[Explore](#) [Features](#) [Enterprise](#) [Blog](#) [Sign up](#) [Sign in](#)

Build software better, together.

Powerful collaboration, code review, and code management for open source and private projects. Need private repositories?
[Upgraded plans start at \\$7/mo.](#)

Use at least one lowercase letter, one numeral, and seven characters.

[Sign up for GitHub](#)

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We will send you account related emails occasionally.

Please email your GitHub user names to:

- **git101.hubspot@gmail.com**
- Needed to add everyone as collaborators of the tutorial repository



A quick request...





Ask questions!



Google

git tutorial

Web

Videos

Books

News

Images

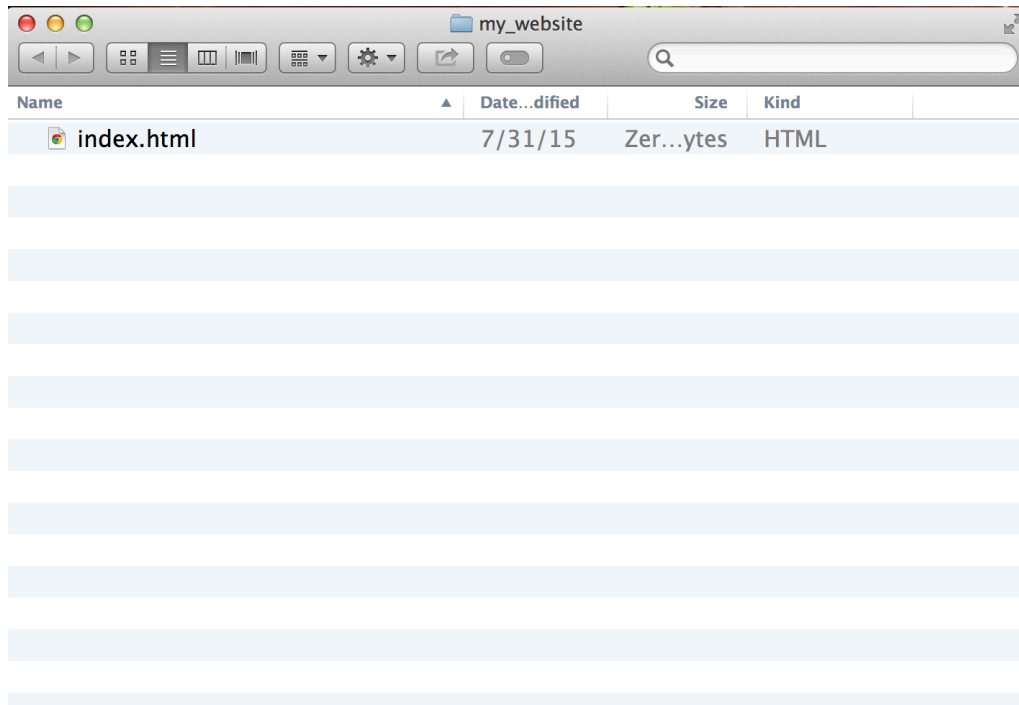
About 28,200,000 results (0.26 seconds)

1

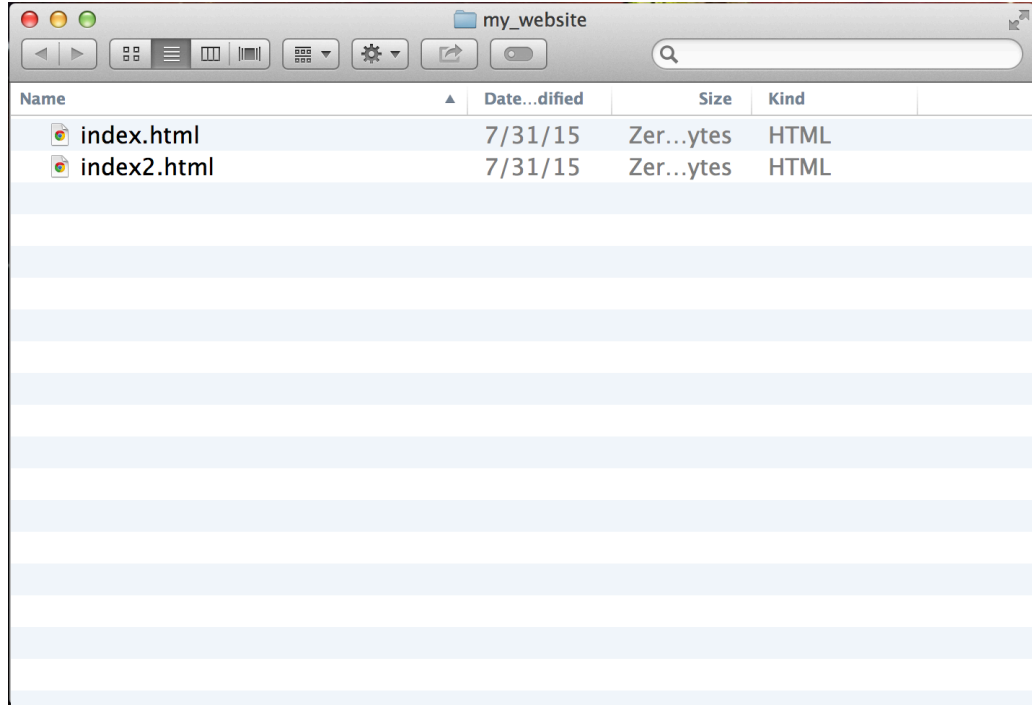
What is version control?

A world without version control

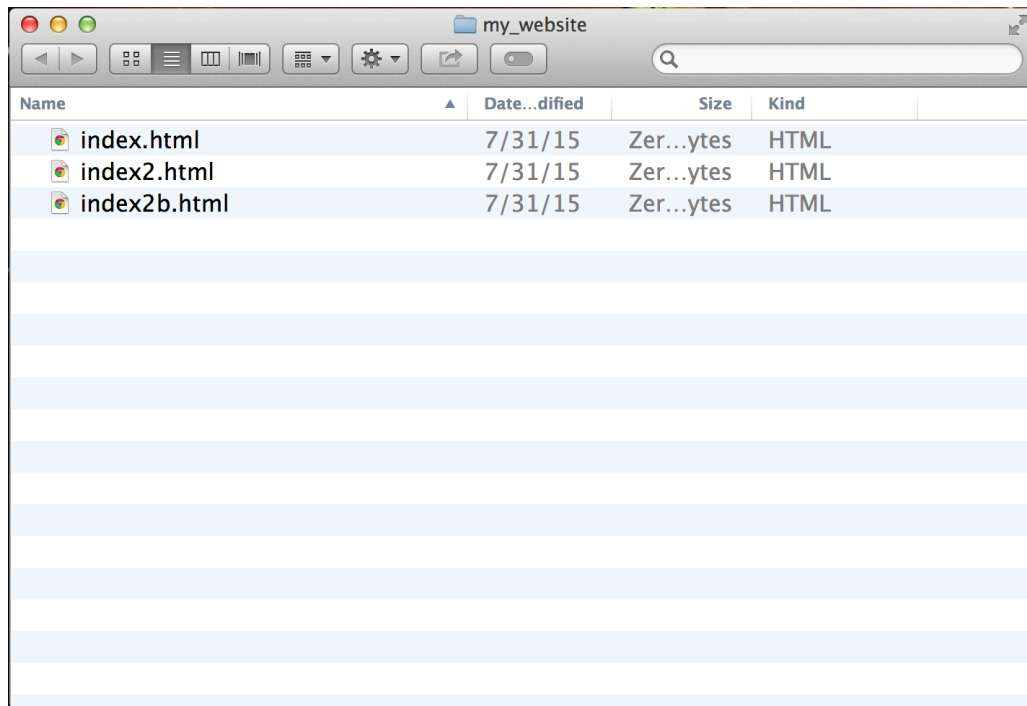
A world without version control



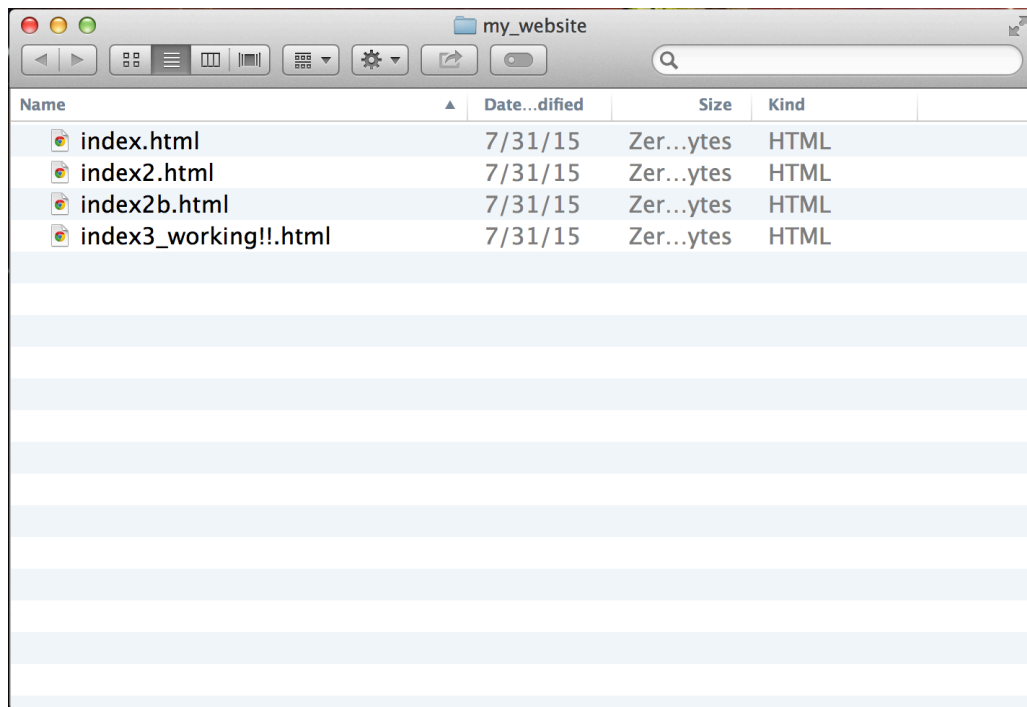
A world without version control



A world without version control



A world without version control



What is version control?

What is version control?

- A system that keeps records of your file changes

What is version control?

- A system that keeps records of your file changes
- Allows for team collaboration

What is version control?

- A system that keeps records of your file changes
- Allows for team collaboration
- Allows you to know who made what change, when

What is version control?

- A system that keeps records of your changes
- Allows for collaborative development
- Allows you to know who made what change, when
- Allows you to revert any changes and go back to a previous state

2

What is git?

What is version control?

- **Distributed version control**
- **Users keep entire code and history on their location machines.**
 - Users can make any changes without internet access
 - (except pushing and pulling changes from a remote server)

What is git?

- **Started in 2005**
- **Created by Linus Torvald to aid in Linux kernel development**



Git icon

What is git?

- **Git isn't the only version control system**

What is git?

- Git isn't the only version control system



What is git?

- Git isn't the only version control system



What is git?

- Git isn't the only version control system



What is git?

- Git isn't the only version control system



- But it's the best [citation needed]



3

How does git work?

How does git work?

- Can be complicated at first, but there are a few key concepts
- Important git terminology in following slides are **blue**

Key Concepts: **Snapshots**

- The way git keeps track of your code history
- Essentially records what all your files look like at a given point in time
- You decide when to take a snapshot, and of what files
- Have the ability to go back to visit any snapshot
 - Your snapshots from later on will stay around, too

Key Concepts: **Commit**

- The act of creating a snapshot
- Can be a noun or verb
 - “I committed code”
 - “I just made a new commit”
- Essentially, a project is made up of a bunch of commits

Key Concepts: **Commit**

- Commits contain three pieces of information:
 1. Information about how the files changed from previously
 2. A reference to the commit that came before it
 - Called the “parent commit”
 3. A **hash code** name
 - Will look something like:
fb2d2ec5069fc6776c80b3ad6b7cbde3cade4e

Key Concepts: **Repositories**

- Often shortened to **'repo'**
- **A collection of all the files and the history of those files**
 - Consists of all your commits
 - Place where all your hard work is stored

Key Concepts: **Repositories**

- Can live on a local machine or on a remote server (GitHub!)
- The act of copying a repository from a remote server is called **cloning**
- Cloning from a remote server allows teams to work together

Key Concepts: **Branches**

- All commits in git live on some branch
- But there can be many, many branches

Key Concepts: **Branches**

- All commits in git live on some branch
- But there can be many, many branches
- The main branch in a project is called the **master** branch

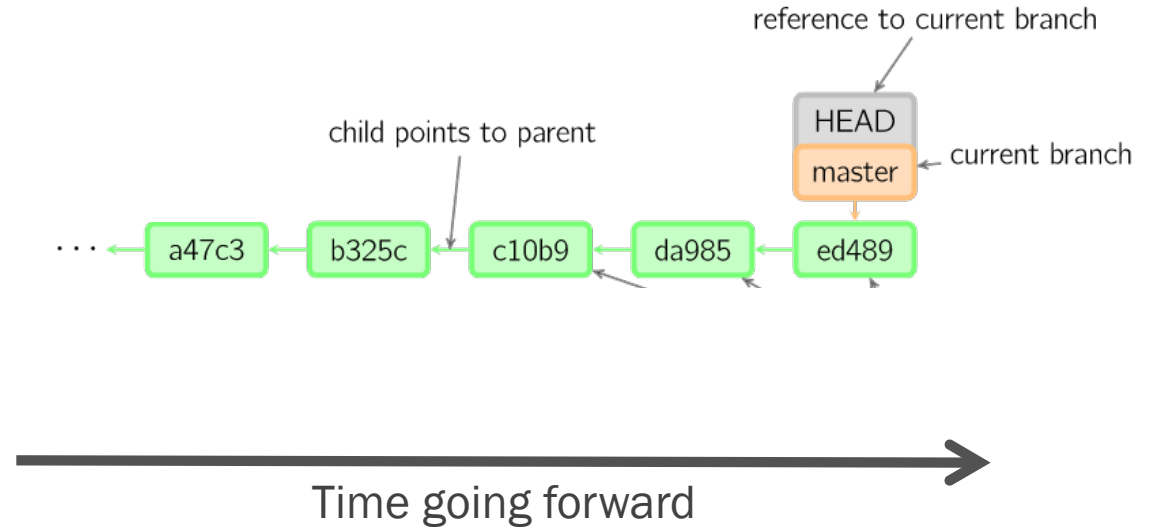
Key Concepts: **Branches**

- All commits in git live on some branch
- But there can be many, many branches
- The main branch in a project is called the **master** branch

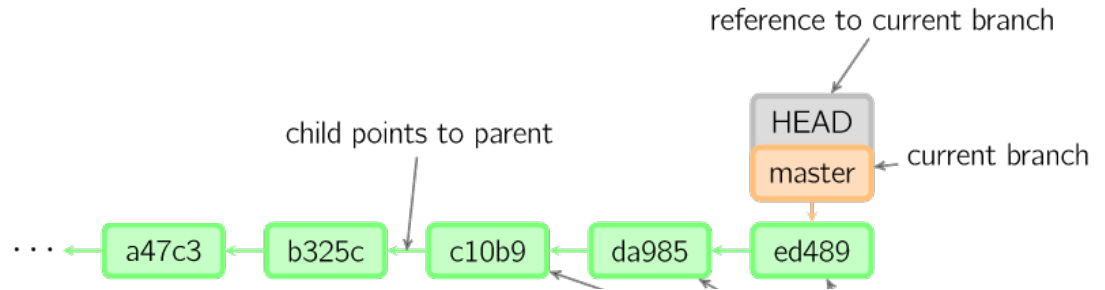
So, what does a typical project look like?

- **A bunch of commits linked together that live on some branch, contained in a repository**
- **Following images taken and modified from:**
 - <http://marklodato.github.io/visual-git-guide/index-en.html>
 - Also a good tutorial!

So, what does a typical project look like?



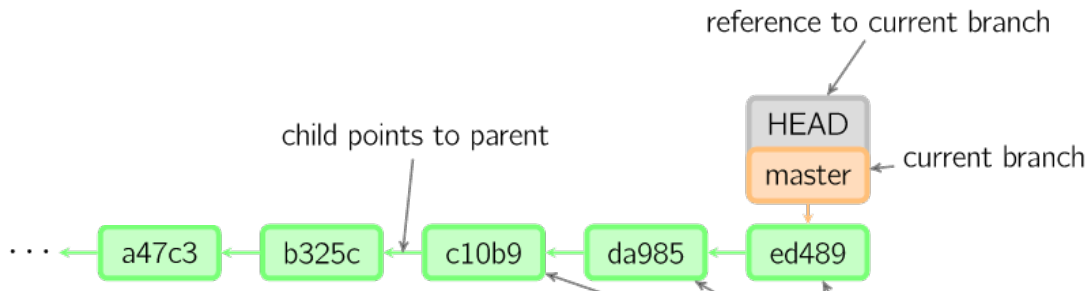
So, what is **HEAD**?



Time going forward →

So, what is **HEAD**?

- A reference to the most recent commit

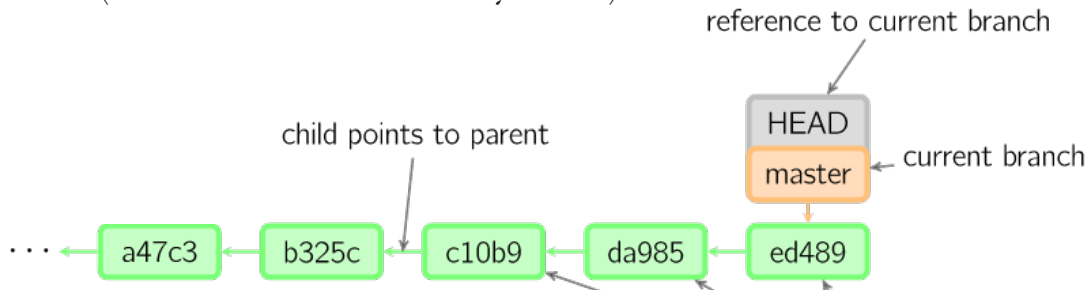


Time going forward →

So, what is **HEAD**?

- A reference to the most recent commit

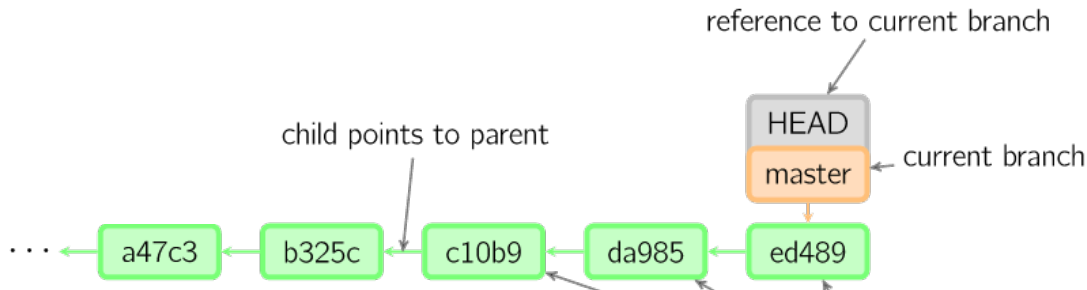
- (in most cases – not always true!)



Time going forward →

So, what is **MASTER**?

- The main branch in your project
- Doesn't *have* to be called master, but almost always is!

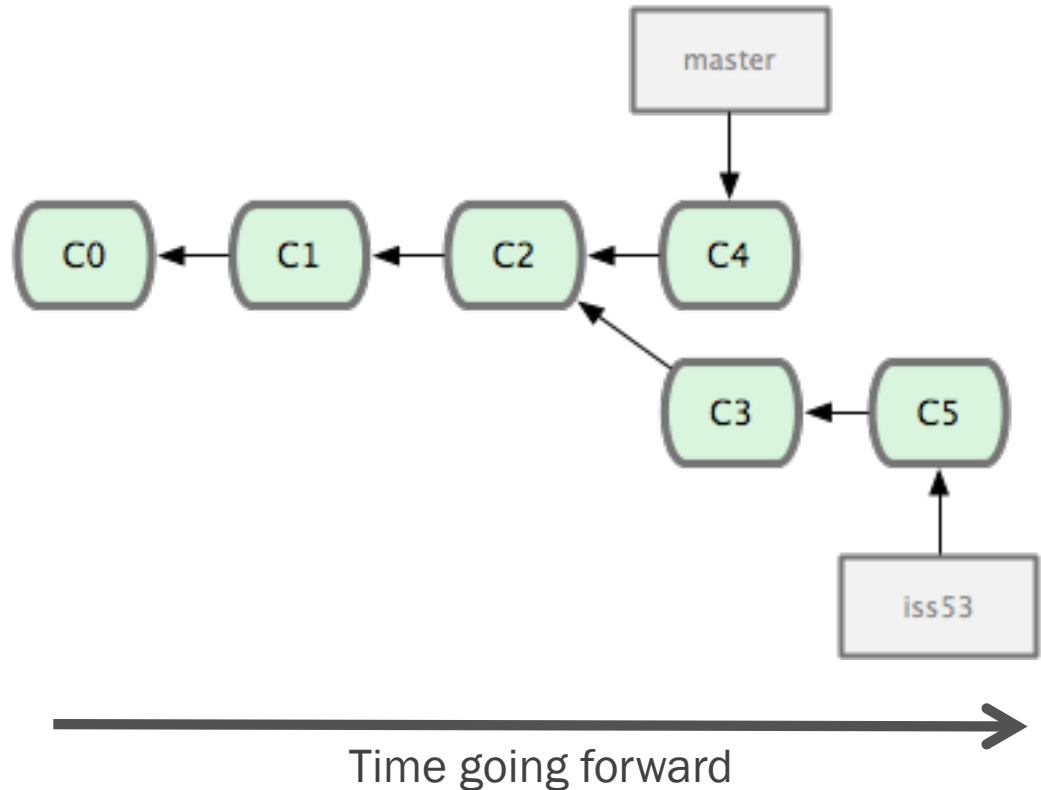


Time going forward →

Key Concepts: Branching off of the **master** branch

- The start of a branch points to a specific commit
- When you want to make any changes to your project you make a new branch based on a commit
- Okay, you don't *strictly* have to, but it's good practice.

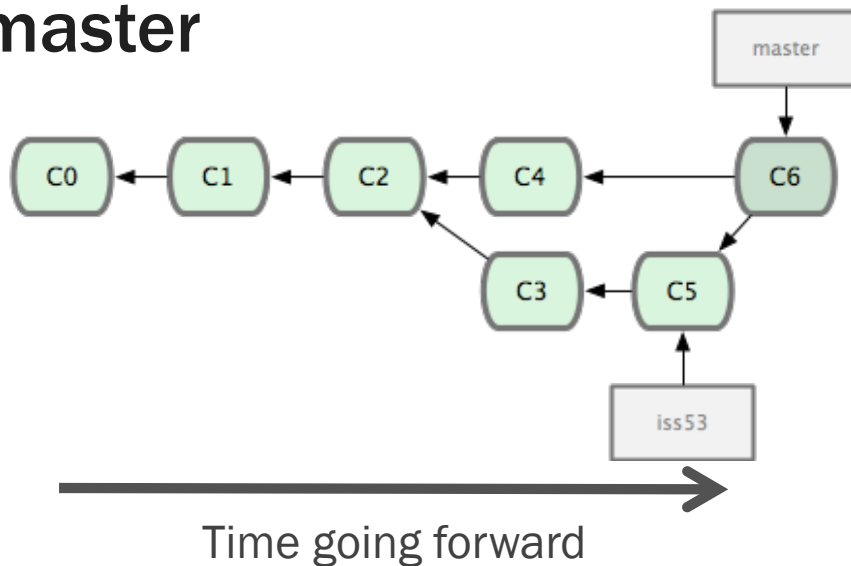
Key Concepts: Branching off of the **master** branch

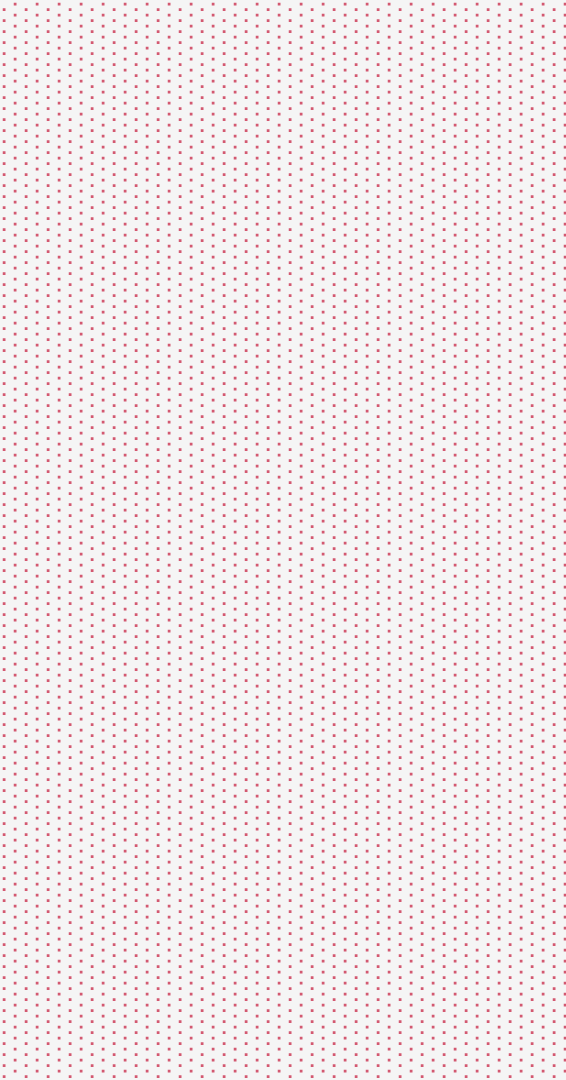


Images from:
[http://codingdomain.com/
git/merging/](http://codingdomain.com/git/merging/)

Key Concepts: Merging

- Once you're done with your feature, you **merge** it back into master



A vertical decorative bar on the left side of the slide, filled with a dense pattern of small red dots.

Key Concepts: How do you make a commit anyway?

Key Concepts: How do you make a commit anyway?

- **There are a lot of ‘states’ and ‘places’ a file can be**

Key Concepts: How do you make a commit anyway?

- **There are a lot of ‘states’ and ‘places’ a file can be**
- **Local on your computer: the ‘working directory’**

Key Concepts: How do you make a commit anyway?

- There are a lot of ‘states’ and ‘places’ a file can be
- Local on your computer: the ‘**working directory**’
- When a file is ready to be put in a commit you add it onto the ‘**index**’ or ‘**staging**’

Key Concepts: How do you make a commit anyway?

- There are a lot of ‘states’ and ‘places’ a file can be
- Local on your computer: the ‘**working directory**’
- When a file is ready to be put in a commit you add it onto the ‘**index**’ or ‘**staging**’
 - Staging is the new preferred term – but you can see both ‘index’ and ‘staging’ being used.

Key Concepts: How do you make a commit anyway?

- **All git commands start with 'git'**

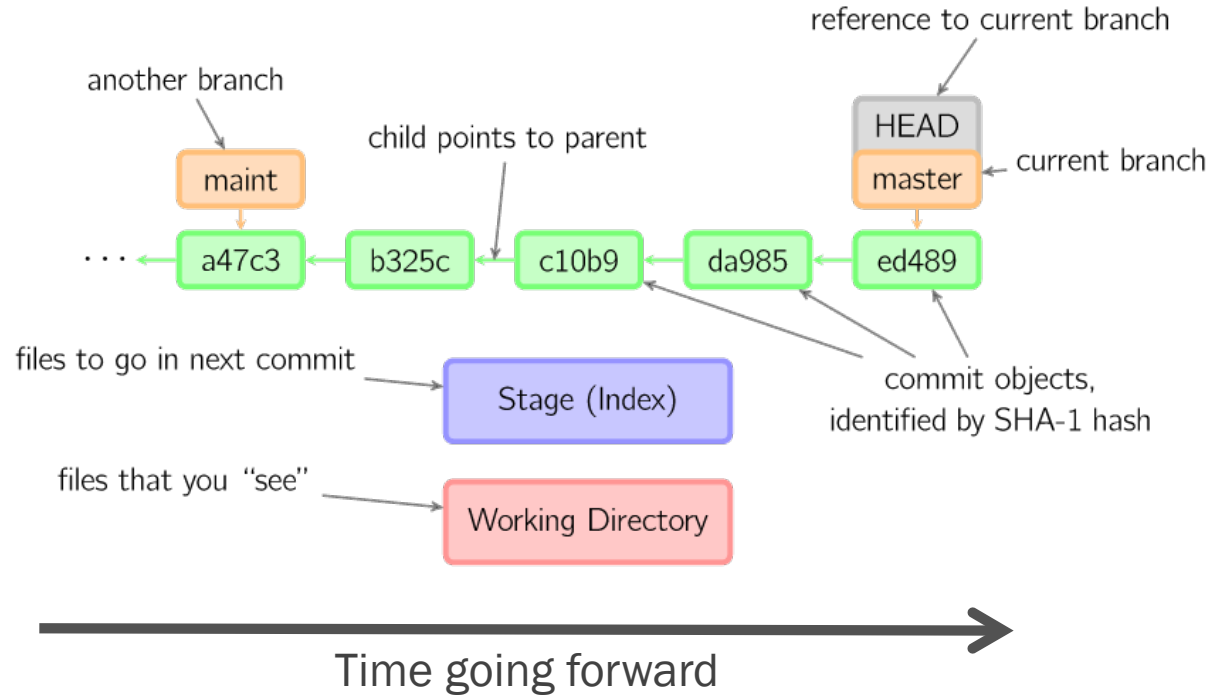
Key Concepts: How do you make a commit anyway?

- **The process:**
 - Make some changes to a file
 - Use the ‘**git add**’ command to put the file onto the **staging environment**

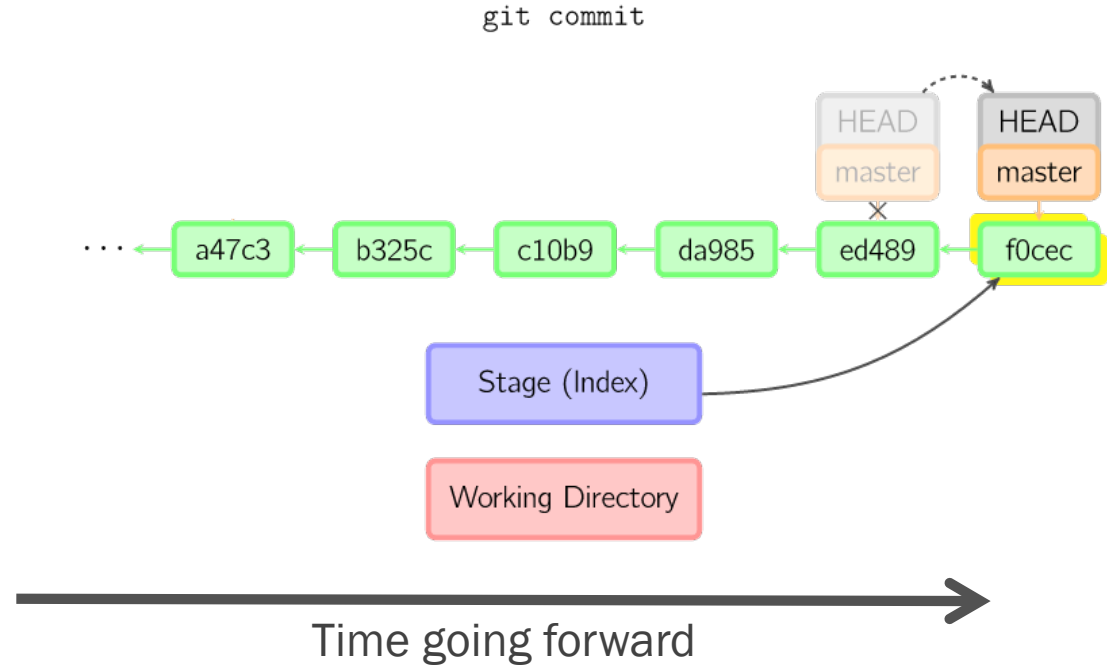
Key Concepts: How do you make a commit anyway?

- **The process:**
 - Make some changes to a file
 - Use the ‘`git add`’ command to put the file onto the `staging environment`
 - Use the ‘`git commit`’ command to create a new commit’.

Key Concepts: How do you make a commit anyway?



Key Concepts: How do you make a commit anyway?



4

What is GitHub?

What is GitHub?

- www.github.com
- Largest web-based git repository hosting service
 - Aka, hosts 'remote repositories'
- Allows for code collaboration with anyone online
- Adds extra functionality ontop of git
 - UI, documentation, bug tracking, feature requests, pull requests, *and more!*



Octocat!

What is GitHub?

- Founded in 2008
- Also has an Enterprise edition for businesses





Any questions?

5

Quick example using git and GitHub



Time to break into teams!





<http://bit.ly/1Elnh0Z>

Git up, git up, git around!



Helpful Commands

- `git init`
 - Creates a new local repository
- `git status`
 - Lists all new and modified files that can be put in a commit
- `git add [file]`
 - Stages one or more files so it can be put in a commit
 - Can be performed multiple times
- `git commit -am [message]`
 - Commits the staged snapshots



Additional Resources

Additional Resources

- **Official git site and tutorial:**
<https://git-scm.com/>
- **GitHub guides:**
<https://guides.github.com/>
- **Command cheatsheet:**
<https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf>
- **Interactive git tutorial:**
<https://try.github.io/levels/1/challenges/1>
- **Visual/interactive cheatsheet:**
<http://ndpsoftware.com/git-cheatsheet.html>