Before you can make commits or do anything else with a git repository, the repository needs to actually exist. To create a new repository with Git, we'll use the git init command.

The init subcommand is short for "initialize", which is helpful because it's the command that will do all of the initial setup of a repository. We'll look at what it does in just a second.

***Required Commands***

*Heads up! We'll be using the following terminal commands in this lesson:*

* *ls - used to list files and directories*
* *mkdir - used to create a new directory*
* *cd - used to change directories*
* *rm - used to remove files and directories*

*If you're not sure how to use them, check out our course*[*Shell Workshop*](https://www.udacity.com/course/shell-workshop--ud206)*!*

*We'll also be using the idea of the current working directory, the directory that your shell is "looking at" right now. Using cd changes your working directory, and using ls (by itself) lists the files in the working directory. If you lose track of what your shell's working directory is, you can print its name with the pwd command (which stands for "print working directory").*

**Create Course Directories**

We're about to create a new project for this course. Since we're all professionals, we want our projects organized. If you already have a location on your computer where you put all your projects, then keep doing what you're doing. I'll be storing all of my work in a directory called udacity-git-course.

If you want to follow along with me:

* create a directory called udacity-git-course
* inside that, create another directory called new-git-project
* use the cd command to move into the new-git-project directory

If you're a copy/paster like me, just run this command on the terminal - mkdir -p udacity-git-course/new-git-project && cd $\_ (Before running this command, make sure you cd to where you want these files stored. For example, if you want the files stored on your Desktop, then make sure you cd to the Desktop before running the command.)

If you're all set, then your terminal should be "inside" the new-git-project directory and look like this:

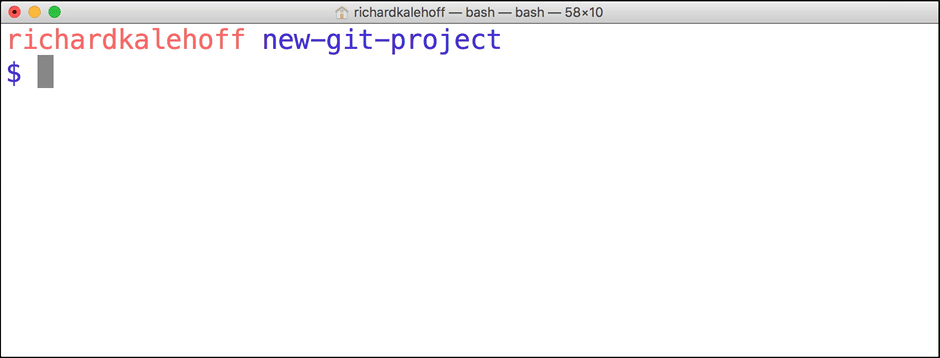
[[](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/fa8f761a-d0a2-4be1-a5b9-60116ea4ecd1)](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/fa8f761a-d0a2-4be1-a5b9-60116ea4ecd1)

*[The Terminal application located in the new-git-project directory.](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/fa8f761a-d0a2-4be1-a5b9-60116ea4ecd1)*

**Git Init**

Fantastic work - we're all set up and ready to start using the git init command!

This is one of the easiest commands to run. All you have to do is run git init on the terminal. That's it! Go ahead, why not give it a try right now!

[[](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/fa8f761a-d0a2-4be1-a5b9-60116ea4ecd1)](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/fa8f761a-d0a2-4be1-a5b9-60116ea4ecd1)

*[The terminal application showing the git init command being run. The command initializes an empty Git repository in the current directory.](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/fa8f761a-d0a2-4be1-a5b9-60116ea4ecd1)*

**QUIZ QUESTION**

Did you run git init yet? If not, do it now because you'll need it to answer this quiz!

After running git init, the text "Initialized empty Git repository in " followed by a path. should have appeared. The question is, has anything changed with your command prompt? If so, what?

* Yes - I now see the word "master".
* No, it looks exactly the same.

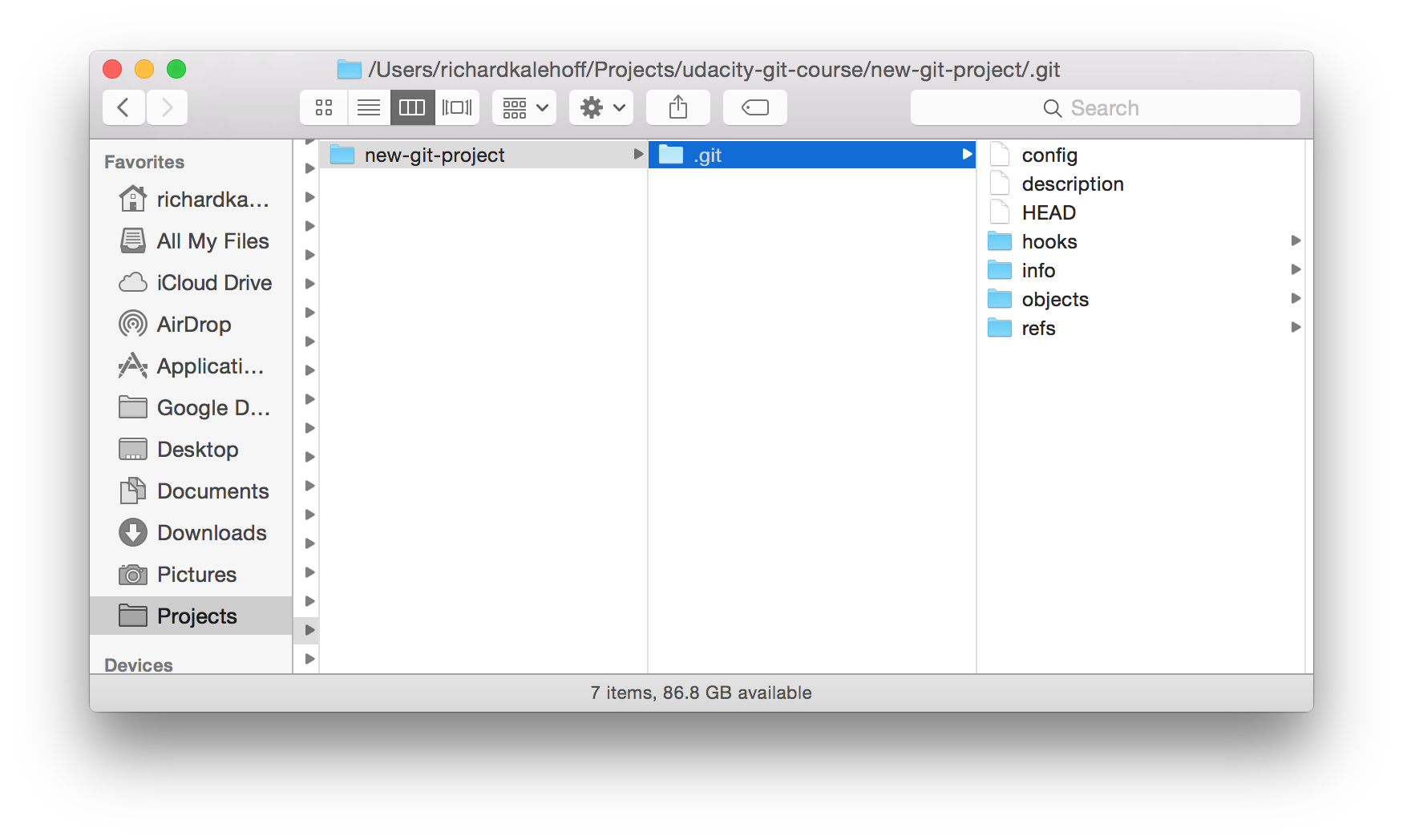
SUBMIT

**Git Init's Effect**

Running the git init command sets up all of the necessary files and directories that Git will use to keep track of everything. All of these files are stored in a directory called .git (notice the . at the beginning - that means it'll be a hidden directory on Mac/Linux). This .git directory *is the "repo"*! This is where git records all of the commits and keeps track of everything!

Let's take a brief look at the contents of the .git directory.

*WARNING: Don't directly edit any files inside the .git directory. This is the heart of the repository. If you change file names and/or file content, git will probably lose track of the files that you're keeping in the repo, and you could lose a lot of work! It's okay to look at those files though, but don't edit or delete them.*

[[](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/fa8f761a-d0a2-4be1-a5b9-60116ea4ecd1)](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/fa8f761a-d0a2-4be1-a5b9-60116ea4ecd1)

*[Mac's Finder window showing the contents of the hidden ".git" directory.](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/fa8f761a-d0a2-4be1-a5b9-60116ea4ecd1)*

**.Git Directory Contents**

*We're about to take a look at the .git directory...it's not vital for this course, though, so don't worry about memorizing anything, it's here if you want to dig a little deeper into how Git works under the hood.*

Here's a brief synopsis on each of the items in the .git directory:

* **config file** - where all *project specific* configuration settings are stored.  
  From the [Git Book](https://git-scm.com/book/en/v2/Customizing-Git-Git-Configuration):

*Git looks for configuration values in the configuration file in the Git directory (.git/config) of whatever repository you’re currently using. These values are specific to that single repository.*

For example, let's say you set that the global configuration for Git uses your personal email address. If you want your work email to be used for a specific project rather than your personal email, that change would be added to this file.

* **description file** - this file is only used by the GitWeb program, so we can ignore it
* **hooks directory** - this is where we could place client-side or server-side scripts that we can use to hook into Git's different lifecycle events
* **info directory** - contains the global excludes file
* **objects directory** - this directory will store all of the commits we make
* **refs directory** - this directory holds pointers to commits (basically the "branches" and "tags")

Remember, other than the "hooks" directory, you shouldn't mess with pretty much any of the content in here. The "hooks" directory *can* be used to hook into different parts or events of Git's workflow, but that's a more advanced topic that we won't be getting into in this course.

**Further Research**

* [Git Internals - Plumbing and Porcelain](https://git-scm.com/book/en/v2/Git-Internals-Plumbing-and-Porcelain) (advanced - bookmark this and check it out later)
* [Customizing Git - Git Hooks](https://git-scm.com/book/en/v2/Customizing-Git-Git-Hooks)

**Git Init Recap**

Use the git init command to create a new, empty repository in the current directory.

$ git init

Running this command creates a hidden .git directory. This .git directory is the brain/storage center for the repository. It holds all of the configuration files and directories and is where all of the commits are stored.

**Helpful Links**

* [Initializing a Repository in an Existing Directory](https://git-scm.com/book/en/v2/Git-Basics-Getting-a-Git-Repository#Initializing-a-Repository-in-an-Existing-Directory)
* [git init docs](https://git-scm.com/docs/git-init)
* [git init Tutorial](https://www.atlassian.com/git/tutorials/setting-up-a-repository)

## Why Clone?

First, what is cloning?

*to make an identical copy*

What's the value of creating an identical copy of something, and how does this relate to Git and version control?

Why would you want to create an identical copy? Well, when I work on a new web project, I do the same set of steps:

* create an index.html file
* create a js directory
* create a css directory
* create an img directory
* create app.css in the css directory
* create app.js in the js directory
* add starter HTML code in index.html
* add configuration files for linting (validating code syntax)
  + [HTML linting](http://htmlhint.com/)
  + [CSS linting](https://stylelint.io/)
  + [JavaScript linting](http://eslint.org/)
* [configure my code editor](http://editorconfig.org/)

...and I do this every time I create a new project!...which is a lot of effort I'm putting in for each new project. I didn't want to keep doing these same steps over and over, so I did all of the steps listed above one last time and created a starter project for myself. Now when I create a new project, I just make an identical copy of that starter project!

The way that cloning relates to Git is that the command we'll be running on the terminal is git clone. You pass a path (usually a URL) of the Git repository you want to clone to the git clone command.

Wanna try cloning an existing project? Let's see how Git's clone command works!

## *Verify Terminal Location*

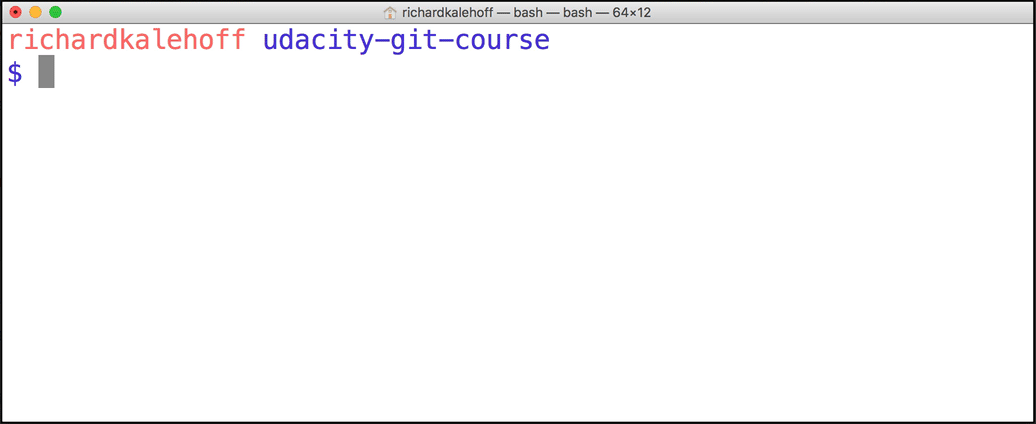
*TIP: Now before you clone anything, make sure you are located in the correct directory on the command line. Cloning a project creates a new directory and places the cloned Git repository in it. The problem is that you can't have nested Git repositories. So make sure the terminal's current working directory isn't located in a Git repository. If your current working directory is not in your shell's prompt, type pwd to print the working directory.*

## Cloning The Blog Repository

Ready? Let's get cloning!

The command is git clone and then you pass the path to the Git repository that you want to clone. The project that we'll be using throughout this course is located at this URL: <https://github.com/udacity/course-git-blog-project> So using this URL, the full command to clone blog project is:

$ git clone https://github.com/udacity/course-git-blog-project

[[](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)

[The](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)*[git clone](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)*[command is used to copy the blog project repository into a](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)*[course-git-blog-project](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)*[folder in the current directory.](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)

## Git Clone Output Explanation

Let's look briefly at the output that git clone displays.

The first line says "Cloning into 'course-git-blog-project'...". Git is creating a directory (with the same name of the project we're cloning) and putting the repository in it...that's pretty cool!

The rest of the output is basically validation - it's counting the remote repository's number of objects, then it compresses and receives them, then it unpacks them.

## Clone Project And Use Different Name

You just cloned the blog project for this course. Awesome job!

The command you ran in the terminal was:

$ git clone https://github.com/udacity/course-git-blog-project

...which created a directory named course-git-blog-project.

What if you want to use a different name instead of the default one? Yes, you could just run the command above and manually rename it in Finder/Windows Explorer or use mv on the terminal. But that's too many steps for us! Instead, we'd rather clone the project and have it use a different name all in one go! But how do we do that?

### QUIZ QUESTION

Why don't you check out [the documentation for git clone](https://git-scm.com/book/en/v2/Git-Basics-Getting-a-Git-Repository#Cloning-an-Existing-Repository) and pick the correct way to do it from the options below. The command should clone the blog project repo and store it in a directory named blog-project.

* git clone-new-dir <https://github.com/udacity/course-git-blog-project> blog-project
* git clone <https://github.com/udacity/course-git-blog-project> --out blog-project
* git clone <https://github.com/udacity/course-git-blog-project> --rename blog-project
* git clone <https://github.com/udacity/course-git-blog-project> blog-project

SUBMIT

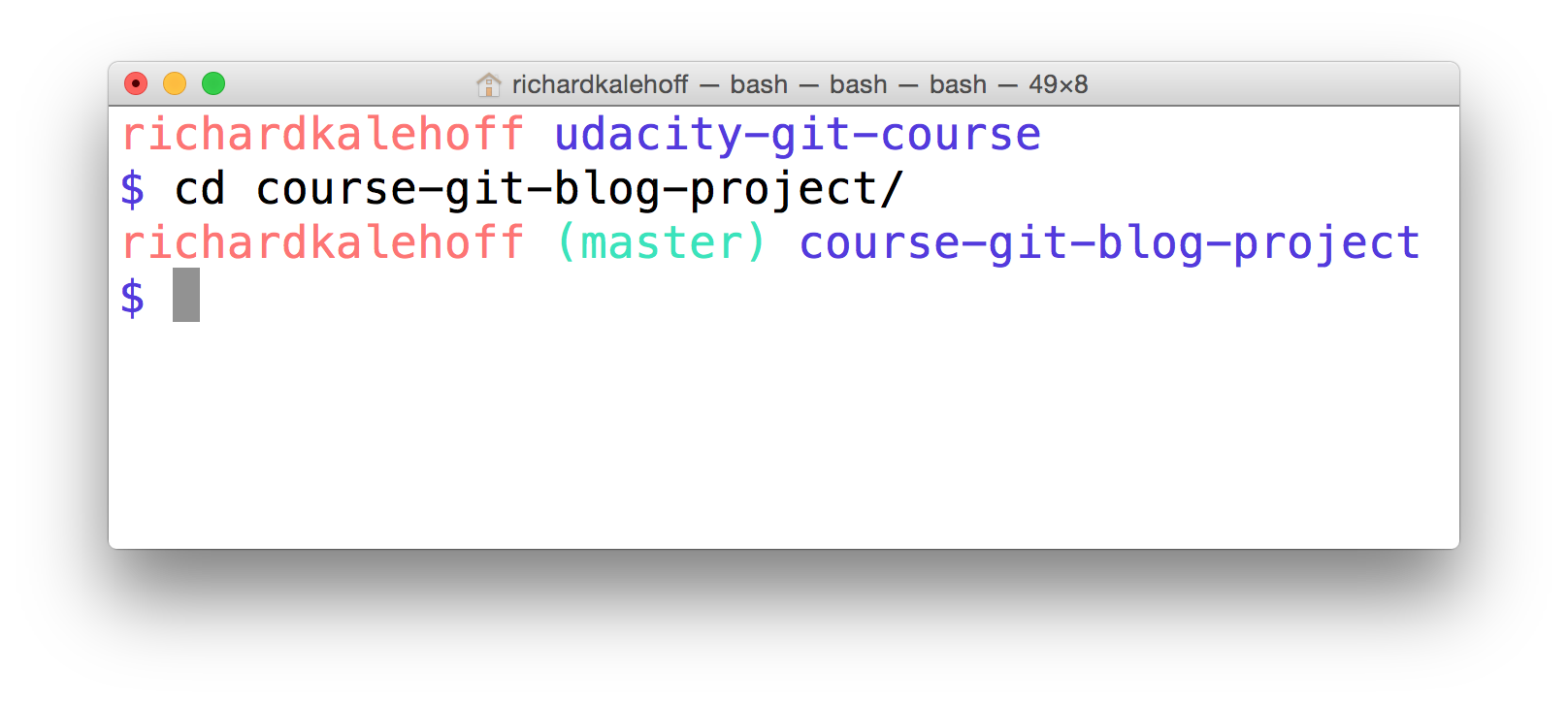
## *Not In A Git Repository?*

*WARNING: Here's a*very important*step that often gets missed when you first start working with Git. When the git clone command is used to clone a repository, it creates a new directory for the repository...you already know this.*But, it doesn't change your shell's working directory.*It created the new repo inside the current working directory, which means that the current working directory is still*outside*of this new Git repo! Make sure you cd into the new repository.*

*Remember to use the Terminal's command prompt as an aid - if you're in a directory that is a Git repository, the command prompt will include a name in parentheses.*

[[](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)

[Mac's Terminal application. The terminal shows the starting directory.](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)

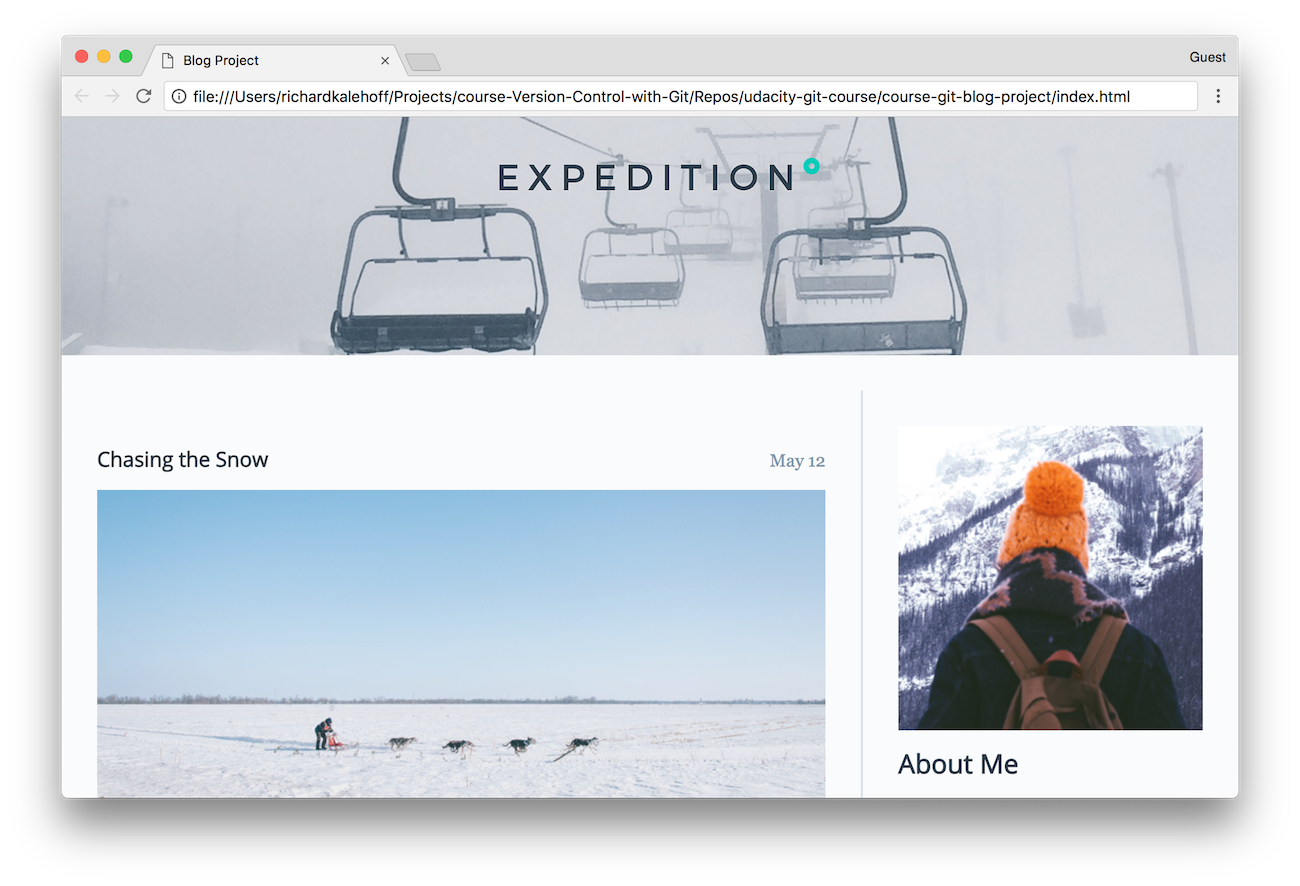
[[](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)

[Mac's Terminal application. The terminal uses the](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)*[cd](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)*[command to move from the base directory into the course-git-blog-project which is a Git repository.](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)

## Look At The Project

So you've cloned the project to your computer, and you've cded into it. Don't you think it's time you checked it out in a browser to see what it looks like?

Open up the index.html file in your favorite browser.

[[](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)

[The blog project loaded in Chrome.](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/59082be0-c1af-4839-8ae5-0a182dc5bfe8)

## Git Clone Recap

The git clone command is used to create an identical copy of an existing repository.

$ git clone <path-to-repository-to-clone>

This command:

* takes the path to an existing repository
* by default will create a directory with the same name as the repository that's being cloned
* can be given a second argument that will be used as the name of the directory
* will create the new repository inside of the current working directory

## Helpful Links

* [Cloning an Existing Repository](https://git-scm.com/book/en/v2/Git-Basics-Getting-a-Git-Repository#Cloning-an-Existing-Repository)
* [git clone docs](https://git-scm.com/docs/git-clone)
* [git clone Tutorial](https://www.atlassian.com/git/tutorials/setting-up-a-repository)

## Status Update

At this point, we have two Git repositories:

* the empty one that we created with the git init command
* the one we cloned with the git clone command

How can we find any information about these repositories? Git's controlling them, but how can we find out what Git knows about our repos? To figure out what's going on with a repository, we use the git status command. Knowing the status of a Git repository is extremely important, so head straight on over to the next concept: Determine A Repo's Status.

#### **Supporting Materials**

[**Git Repository .zip file**](https://video.udacity-data.com/topher/2020/March/5e714611_course-git-blog-project-master/course-git-blog-project-master.zip)

Working with Git on the command line can be a little bit challenging because it's a little bit like a [black box](https://en.wikipedia.org/wiki/Black_box). I mean, how do you know when you should or shouldn't run certain Git commands? Is Git ready for me to run a command yet? What if I run a command but I think it didn't work...how can I find that out? The answer to all of these questions is the git status command!

$ git status

The git status is our key to the mind of Git. It will tell us what Git is thinking and the state of our repository as Git sees it. When you're first starting out, you should be using the git status command all of the time! Seriously. You should get into the habit of running it after any other command. This will help you learn how Git works and it'll help you from making (possibly) incorrect assumptions about the state of your files/repository.

### QUESTION 1 OF 2

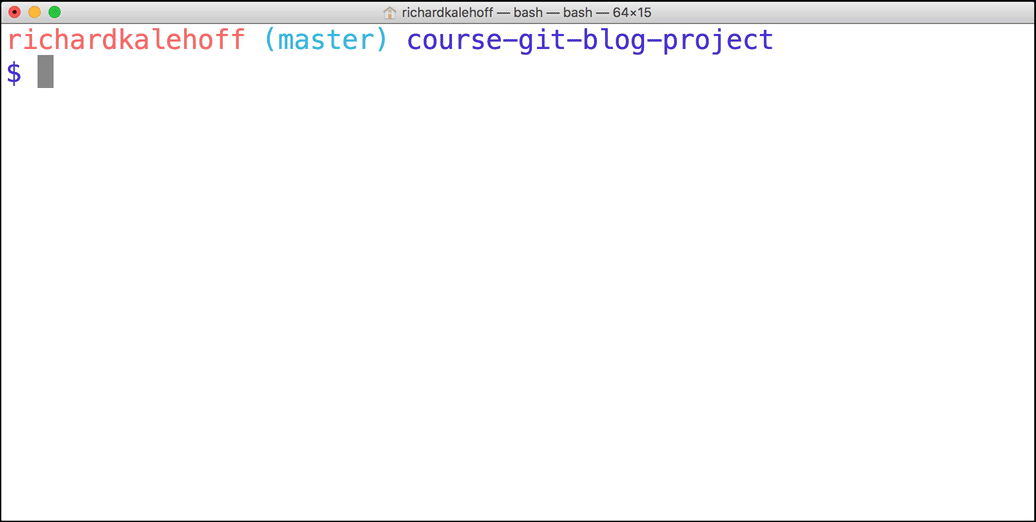
To answer this quiz, make sure you've cded into the course-git-blog-project project. If you've been following along in this lesson and haven't added any files to this project, then what does running git status display?

* Status: good
* On branch master  
  Your branch is up-to-date with 'origin/master'.  
  Initial commit  
  nothing to commit (create/copy files and use "git add" to track)
* master branch  
  Please commit some files
* On branch master  
  Your branch is up-to-date with 'origin/master'.  
  nothing to commit, working directory clean

SUBMIT

## Git Status Output

The git status command will display a lot of information depending on the state of your files, the working directory, and the repository. You don't need to worry too much about these, though...just run git status and it will display the information you need to know.

[[](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)

[An animated gif of the Terminal application. The](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)*[git status](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)*[command is run in the](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)*[course-git-blog-project](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)*[project.](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)

## Git Status Explanation

As you can see in the GIF above, running git status in the course-git-blog-project project produces the following output:

On branch master

Your branch is up-to-date with 'origin/master'.

nothing to commit, working directory clean

The output tells us two things:

1. On branch master – this tells us that Git is on the master branch. You've got a description of a branch on your terms sheet so this is the "master" branch (which is the default branch). We'll be looking more at branches in lesson 5
2. Your branch is up-to-date with 'origin/master'. – Because git clone was used to copy this repository from another computer, this is telling us if our project is in sync with the one we copied from. We won't be dealing with the project on the other computer, so this line can be ignored.
3. nothing to commit, working directory clean – this is saying that there are no pending changes.

Think of this output as the "resting state" (that's not an official description - it's how I like to describe it!). This is the resting state because there are no new files, no changes have been made in files, nothing is in the staging area about to be committed...no change or action is pending, so that's why I like to call it the resting state.

So this is what it looks like when running git status in a repository that already has commits. Let's switch to the new-git-project project to see what the git status output will produce.

### *💡 Changes in Git v2.14 💡*

*In Git version 2.14, running the git status command in an empty directory changed the wording of "Inital commit" to the much clearer "No commits yet". So the output would be:*

*On branch master*

*No commits yet*

*nothing to commit (create/copy files and use "git add" to track)*

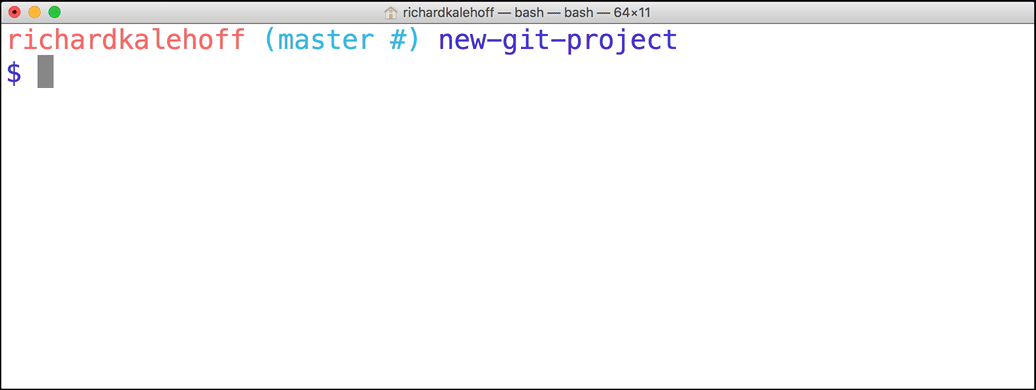
### QUESTION 2 OF 2

To answer this quiz, make sure you've cded into the new-git-project project.

If you've been following along in this lesson and haven't added any files to this project, then what does running git status display?

* Status: good
* On branch master  
  Initial commit  
  nothing to commit (create/copy files and use "git add" to track)
* master branch  
  Please commit some files
* On branch master  
  nothing to commit, working directory clean

SUBMIT

[[](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)

[An animated GIF of the Terminal application. The](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)*[git status](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)*[command is run in the](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)*[new-git-project](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)*[project.](https://classroom.udacity.com/nanodegrees/nd104/parts/912d84b6-bf81-4311-8e00-b3b2f63dc965/modules/90163ae6-479b-42ef-a088-e5c5f4c10101/lessons/ffd30796-2065-420f-90be-97ff6d02ba24/concepts/ce648229-7d6c-4ad3-805e-af6a77f38fd0)

## Explanation Of Git Status In A New Repo

This is the output of running git status in the new-git-project project:

$ git status

On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)

To be completely clear, I haven't made any commits in my project yet. If you have made a commit, then your output should look exactly like that of the course-git-blog-project project.

If you compare this to the git status output from the course-git-blog-project project, then you'll see that they're pretty similar. The thing to note that's different is that this output includes the line Initial commit. This is the tiniest bit confusing because there actually aren't any commits in this repository yet! We haven't discussed making a commit yet, but when we do, we will be able to make an initial commit.

Wanna have a sneak peak of the next lesson and at the same time prove that there aren't any commits in this repo yet? Great, I knew you did! Try running the command git log and check out its response:

$ git log

fatal: your current branch 'master' does not have any commits yet

Well, that's kind of scary looking. "Fatal"? Fortunately, it turns out that just means that the Git program is exiting because it can't find any work to do. Git tells us this as if it were an error, but it's really not a problem. We know we haven't put any commits into this repo yet.

It's pretty clear from the response that there aren't any commits!

We've just taken a very brief look at the git status command. Remember that the output of git status will change depending on if files have been added/deleted/modified, what's on the staging index, and the state of the repository. We'll be using the git status command throughout this entire course, so get comfortable running it!

## Git Status Recap

The git status command will display the current status of the repository.

$ git status

I can't stress enough how important it is to use this command all the time as you're first learning Git. This command will:

* tell us about new files that have been created in the Working Directory that Git hasn't started tracking, yet
* files that Git is tracking that have been modified
* a whole bunch of other things that we'll be learning about throughout the rest of the course ;-)

## Helpful Links

* [Checking the Status of Your Files](https://git-scm.com/book/en/v2/Git-Basics-Recording-Changes-to-the-Repository#Checking-the-Status-of-Your-Files)
* [git status docs](https://git-scm.com/docs/git-status)
* [git status Tutorial](https://www.atlassian.com/git/tutorials/inspecting-a-repository/git-status)