

*TIP: In lesson 2 you used `git clone` to clone the blog project. This is the project we'll be using in this lesson. If you skipped cloning the project in the previous lesson, then run the following command to get the project.*

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

*Don't forget to `cd` into the project after you've cloned it.*

*If you have questions about this, review how to [Clone An Existing Repo](#) or ask in [Knowledge](#).*

#### QUESTION 1 OF 7

After you've cloned the blog project repository, navigate to the project's directory using the command line. Once you're located inside the blog project, what is the very first thing you should do in a Git repository?

- ☒ run the `git status` command
- ☐ open the project in a code editor
- ☐ decide what new feature to work on

SUBMIT

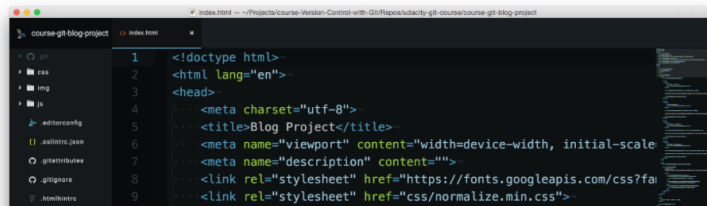
```
richardkalehoff (master) course-git-blog-project
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working directory clean
richardkalehoff (master) course-git-blog-project
$
```

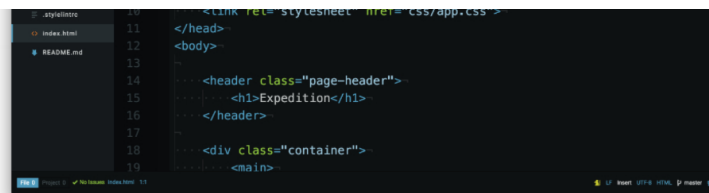
*The Terminal application showing the output of the `git status` command.*

### Git Status & Opening The Project

You can see that `git status` tells us that there's "nothing to commit, working directory clean". That means we're good to go ahead and check out the project!

So open the project in your favorite code editor. If you haven't yet, take a minute or two to look at the project – look over the CSS and the JavaScript files, but look particularly at the HTML file.





```
10 <link rel="stylesheet" href="css/app.css">
11 </head>
12 <body>
13
14   <header class="page-header">
15     <h1>Expedition</h1>
16   </header>
17
18   <div class="container">
19     <main>
```

The course's Blog project open in a code editor. The `index.html` file is being displayed.

#### QUESTION 2 OF 7

In the `index.html` file, take a look at the `<h1>Expedition</h1>` heading around line 15.

Based on what you can see here when was that heading added?

- ☐ It was added on a Tuesday. Yeah, a Tuesday.
- ☐ 3 weeks ago
- ☒ 🙄 I can't tell that by looking at the code.

SUBMIT

#### QUESTION 3 OF 7

Ok, so we're not quite sure *when* the heading was added. How about an easier question - *who* added this heading? Again, what can you tell from just looking at the code?

- ☐ Richard did!
- ☒ No clue

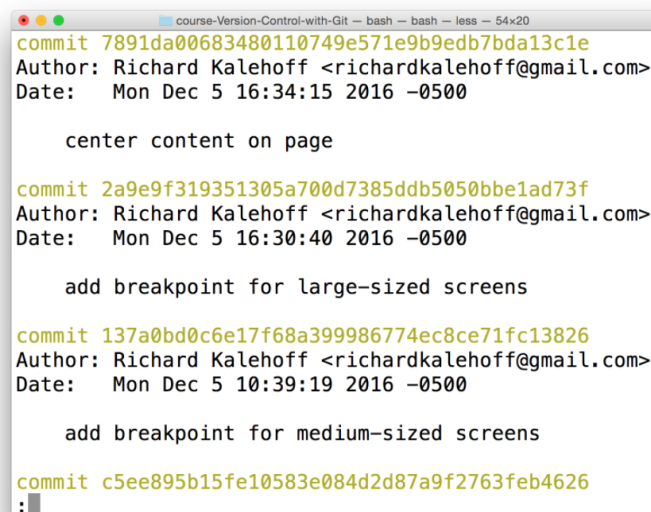
SUBMIT

## The Git Log Command

Finding the answers to these questions is exactly what `git log` can do for us! Instead of explaining everything that it can do for us, let's experience it! Go ahead and run the `git log` command in the terminal:

```
$ git log
```

The terminal should display the following screen.



```
course-Version-Control-with-Git — bash — less — 54x20
commit 7891da00683480110749e571e9b9edb7bda13c1e
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date: Mon Dec 5 16:34:15 2016 -0500

    center content on page

commit 2a9e9f319351305a700d7385ddb5050bbe1ad73f
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date: Mon Dec 5 16:30:40 2016 -0500

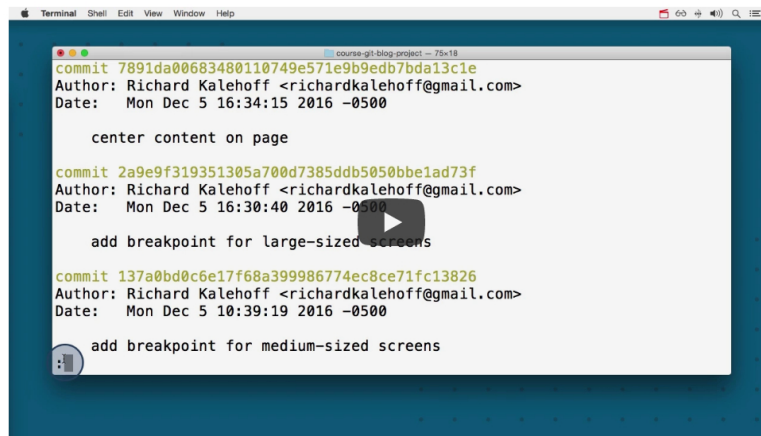
    add breakpoint for large-sized screens

commit 137a0bd0c6e17f68a399986774ec8ce71fc13826
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date: Mon Dec 5 10:39:19 2016 -0500

    add breakpoint for medium-sized screens

commit c5ee895b15fe10583e084d2d87a9f2763feb4626
:
```

The Terminal application showing the output of the `git log` command.



```
commit 7891da00683480110749e571e9b9edb7bda13c1e
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date: Mon Dec 5 16:34:15 2016 -0500

    center content on page

commit 2a9e9f319351305a700d7385ddb5050bbe1ad73f
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date: Mon Dec 5 16:30:40 2016 -0500

    add breakpoint for large-sized screens

commit 137a0bd0c6e17f68a399986774ec8ce71fc13826
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date: Mon Dec 5 10:39:19 2016 -0500

    add breakpoint for medium-sized screens
```

## Navigating The Log

If you're not used to a pager on the command line, navigating in `Less` can be a bit odd. Here are some helpful keys:

- to scroll **down**, press
  - `j` or `↓` to move *down* one line at a time
  - `d` to move by half the page screen
  - `f` to move by a whole page screen
- to scroll **up**, press
  - `k` or `↑` to move *up* one line at a time
  - `u` to move by half the page screen
  - `b` to move by a whole page screen
- press `q` to **quit** out of the log (returns to the regular command prompt)

Git records *a ton* of information when a commit is made. See if you can use `git log` to answer the following questions!

### QUESTION 4 OF 7

Use `git log` to find the commit that has a SHA that starts with `f9720a`. Who made the commit?

- ☐ James Parkes
- ☒ Richard Kalehoff
- ☐ Colt Steele
- ☐ Julia Van Cleve
- ☐ Godzilla McFiddlebrunches

SUBMIT

### What Is The Message?

Use `git log` to find the commit with the SHA that starts with `8aa6668`. What is the message for that commit?

Convert social links from text to images

### QUESTION 6 OF 7

Use `git log` to find the commit with the SHA that starts with `f9720a`. When was that commit made?

☐ Mon Dec 5 10:25:22 2016

☒ Mon Dec 5 10:11:51 2016

☐ Sat Dec 3 16:09:00 2016

☐ Fri Dec 2 16:58:27 2016

SUBMIT

What Is The SHA?

Use `git log` to find the commit that has the message `Set article timestamp color`. Which commit belongs to that SHA? Provide the first 7 characters of the SHA.

5de135a

## Git Log Recap

Fantastic job! Do you feel your Git-power growing?

Let's do a quick recap of the `git log` command. The `git log` command is used to display all of the commits of a repository.

```
$ git log
```

By *default*, this command displays:

- the SHA
- the author
- the date
- and the message

...of every commit in the repository. I stress the "By default" part of what Git displays because the `git log` command can display a lot more information than just this.

Git uses the command line pager, Less, to page through all of the information. The important keys for Less are:

- to scroll down by a line, use `j` or `↓`
- to scroll up by a line, use `k` or `↑`
- to scroll down by a page, use the spacebar or the Page Down button
- to scroll up by a page, use `b` or the Page Up button
- to quit, use `q`

We'll increase our `git log`-wielding abilities in the next lesson when we look at displaying more info.

Why wait?!? Click the link to move to the next lesson!

NEXT