

Version Control with Git

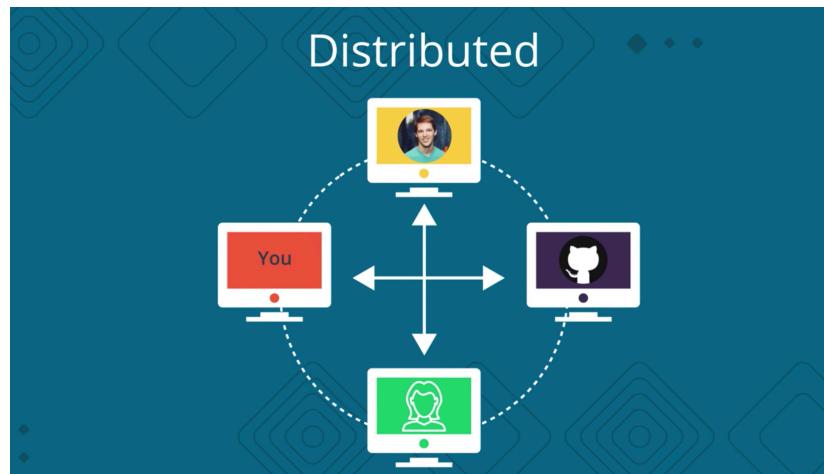
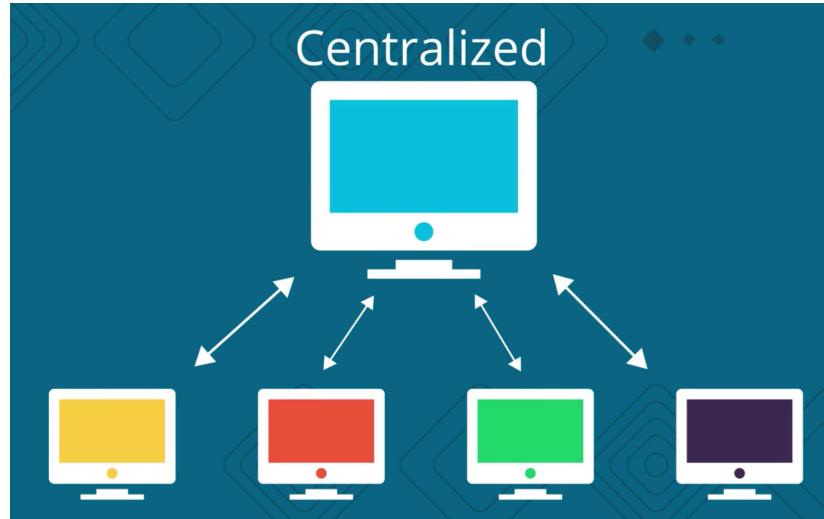
Local Laboratory

What is Version Control

Git

Subversion

Mercurial

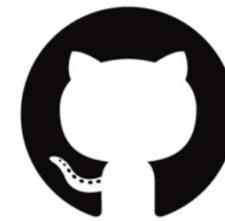


What is Version Control



git

Version control tool



GitHub

Service that hosts
Git projects

Git and Version Control Terminology

Version Control System / Source Code Manager

Commit

Repository / repo

Working Directory

Checkout

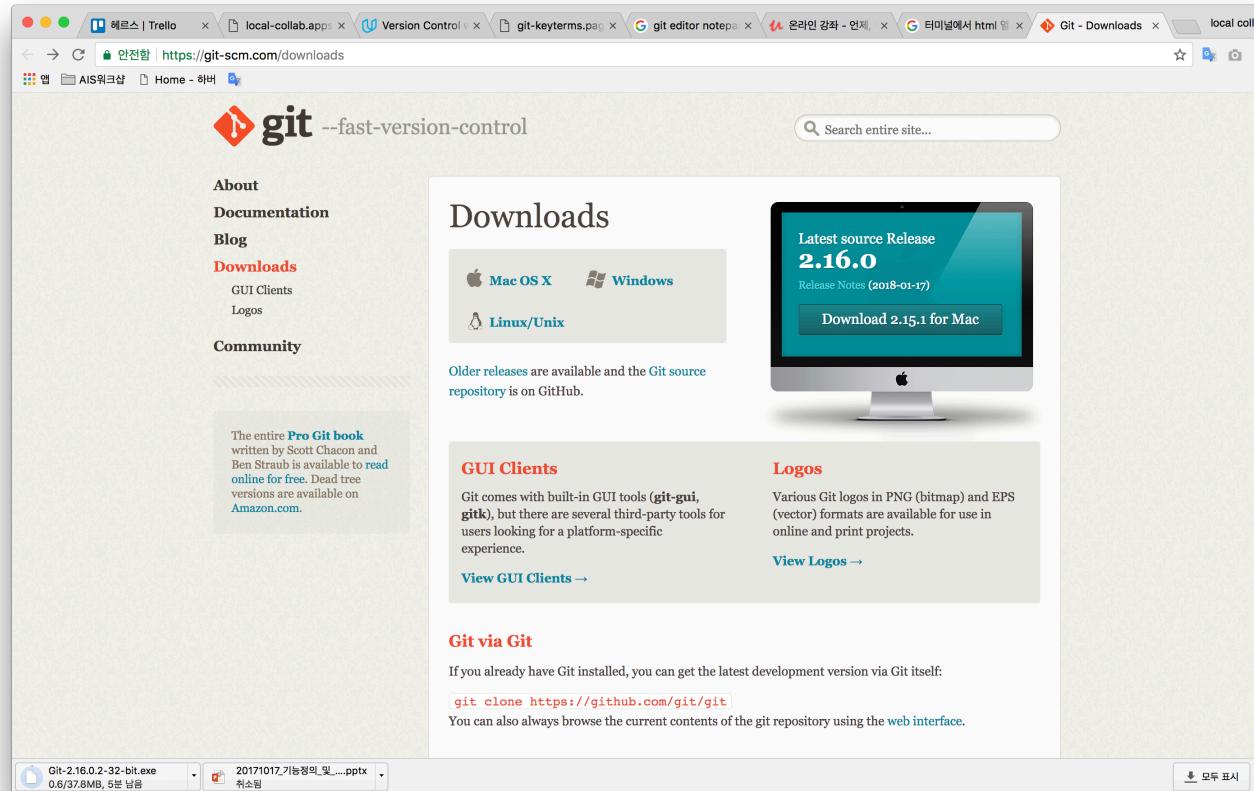
Staging Area

SHA

Branch

Windows Setup

download : <https://git-scm.com/downloads>



First Time Git Configuration

```
# sets up Git with your name
$ git config --global user.name "<Your-Full-Name>

# sets up Git with your email
$ git config --global user.email "<your-email-address>

# makes sure that Git output is colored
$ git config --global color.ui auto

# displays the original state in a conflict
$ git config --global merge.conflictstyle diff3

$ git config --list
```

First Time Git Configuration

```
$ git config --global core.editor "/Applications/Sublime Text 2.app/Contents/SharedSupport/bin/subl" -n -w"  
$ git config --global core.editor " 'C:/Program Files/Sublime Text 3/sublime_text.exe' -n -w"
```

or

```
$ git config --global core.editor "atom --wait"
```

2. Create A Git Repo

git init

Create brand new repositories(repos) on your computer

git clone

Copy existing repos from somewhere else to your local computer

git status

Check the status of a repo

2. Create A Git Repo – Required Commands

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

shell workshop : <https://www.udacity.com/course/shell-workshop--ud206>

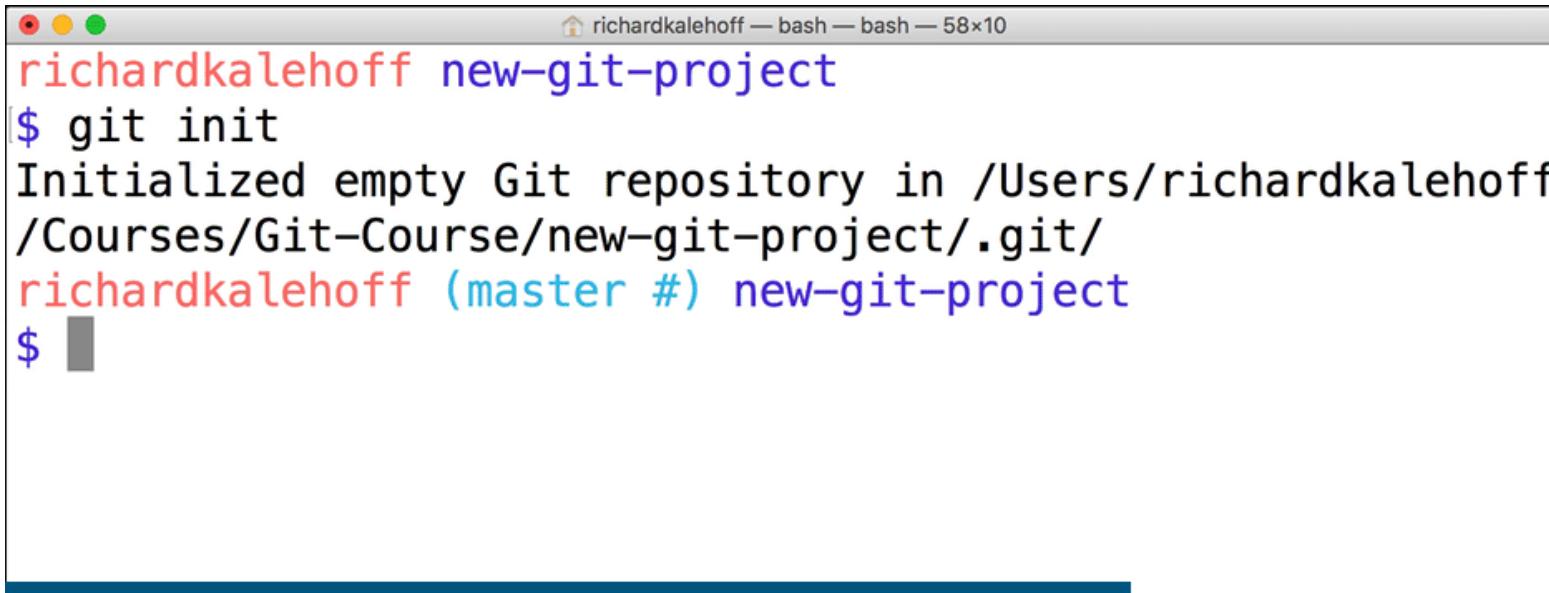
2. Create A Git Repo – Create Course Directory

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



2. Create A Git Repo – Git Init

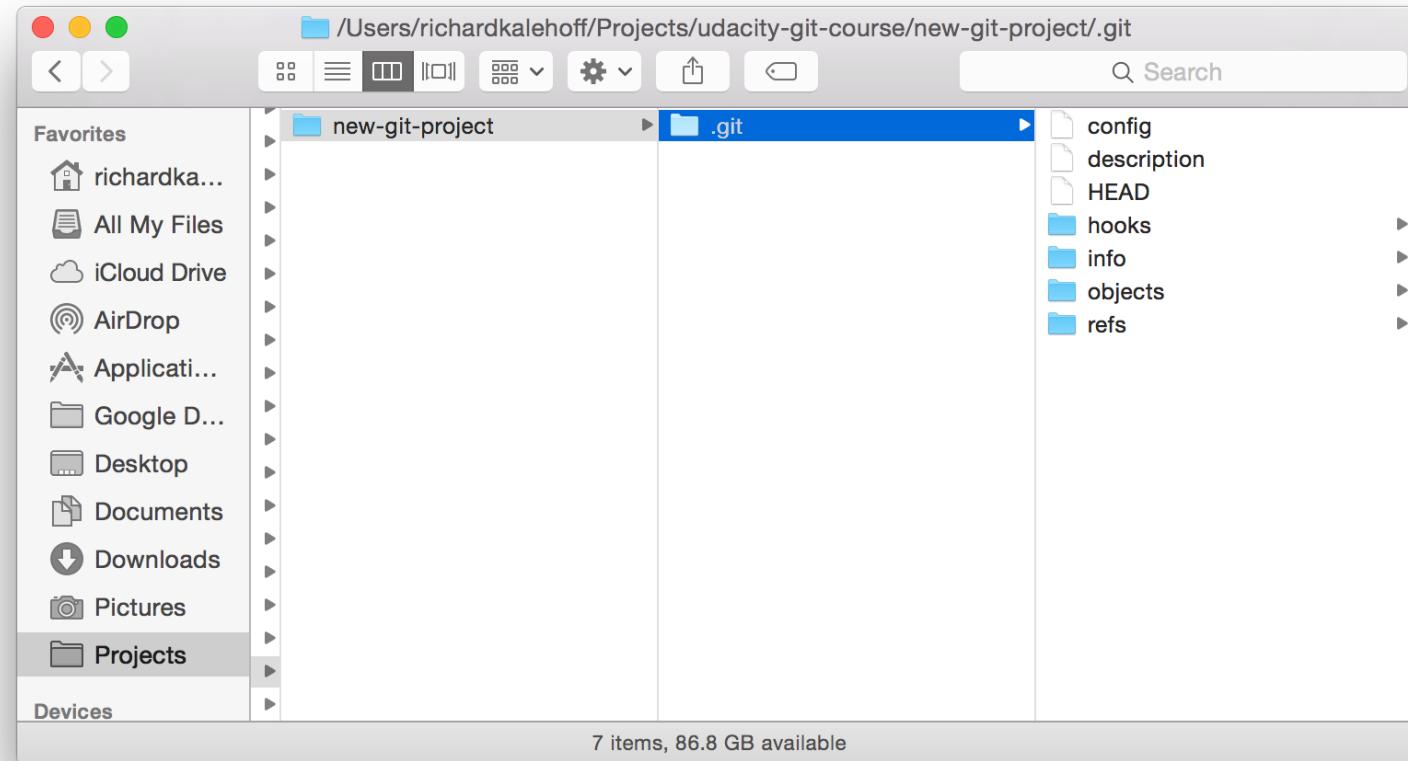
```
$ git init
```



A screenshot of a macOS terminal window titled "richardkalehoff — bash — bash — 58x10". The window shows the following command and its output:

```
richardkalehoff new-git-project
$ git init
Initialized empty Git repository in /Users/richardkalehoff/
/Courses/Git-Course/new-git-project/.git/
richardkalehoff (master #) new-git-project
$
```

2. Create A Git Repo – Git Init's Effect



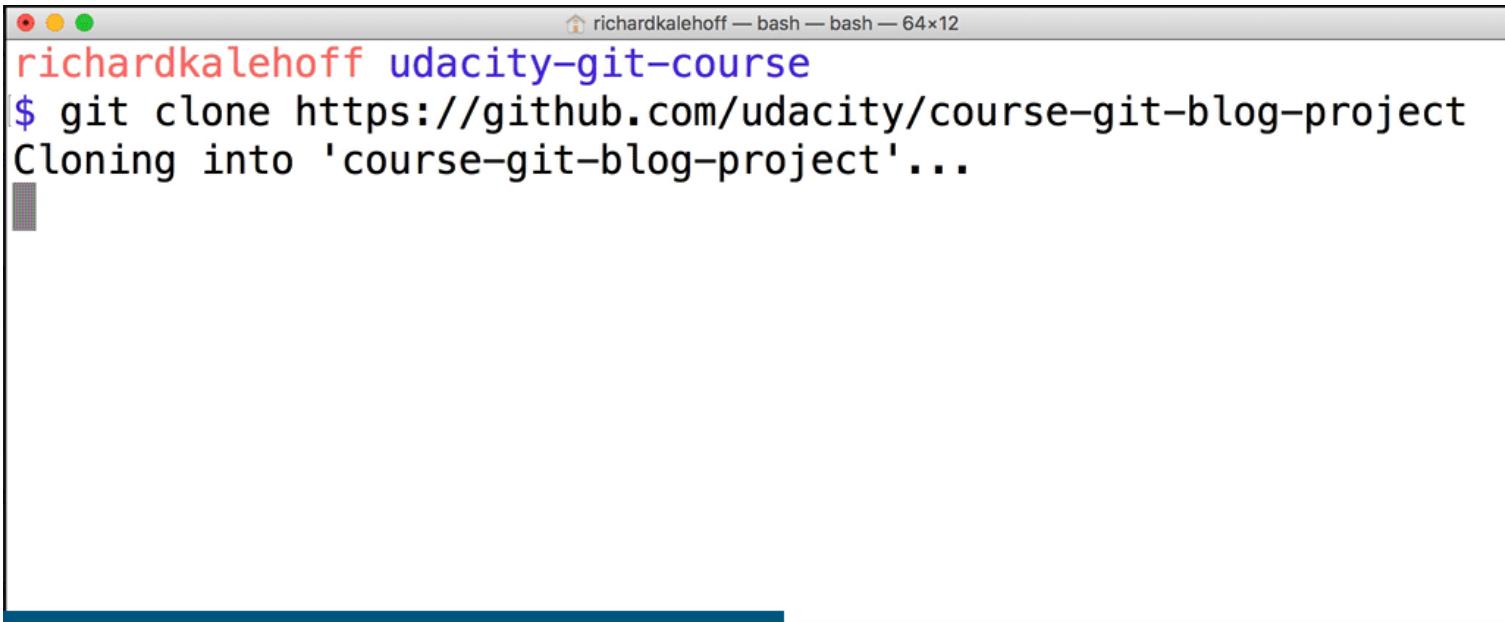
2. Create A Git Repo – Clone An Existing Repo

What is cloning?

to make an identical copy

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

Cloning The Blog Repository



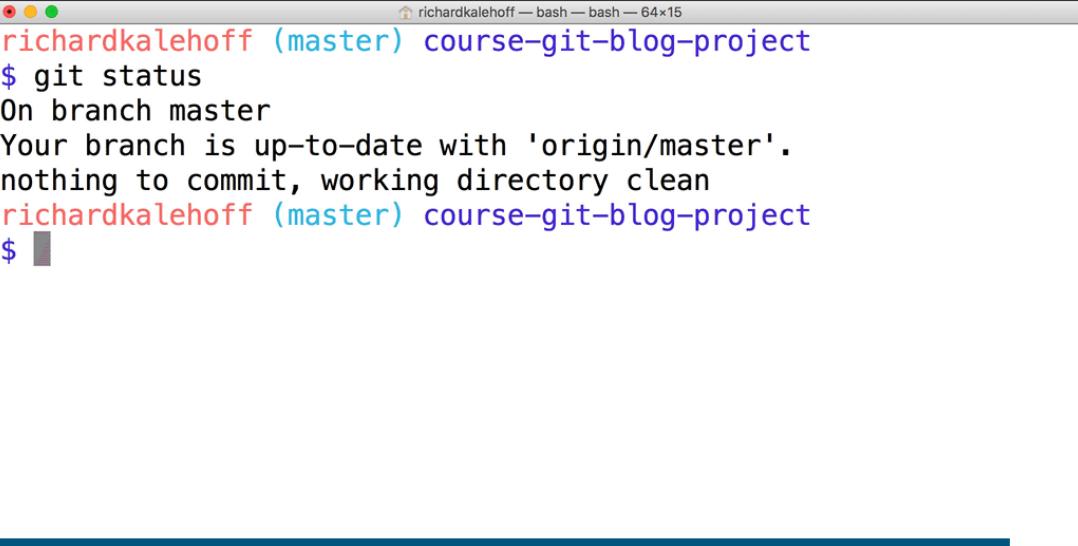
The screenshot shows a terminal window titled "richardkalehoff — bash — bash — 64x12". The window contains the following text:

```
richardkalehoff udacity-git-course
$ git clone https://github.com/udacity/course-git-blog-project
Cloning into 'course-git-blog-project'...
```

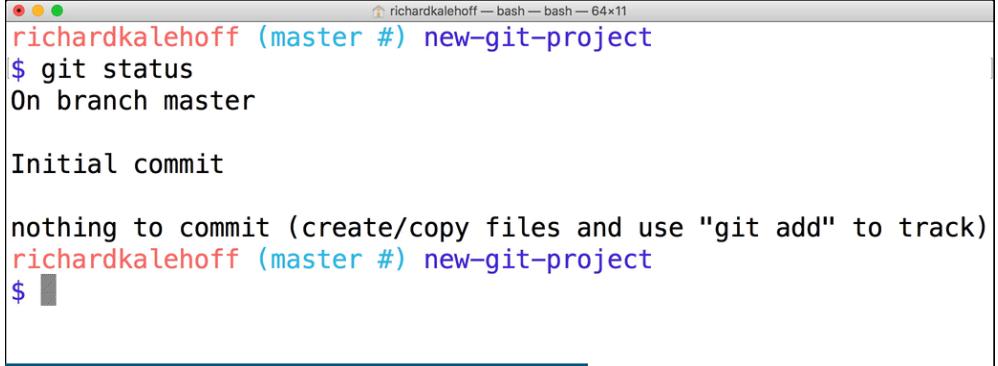
2. Create A Git Repo – Determine A Repo's Status

```
$ git status
```

Git Status Output



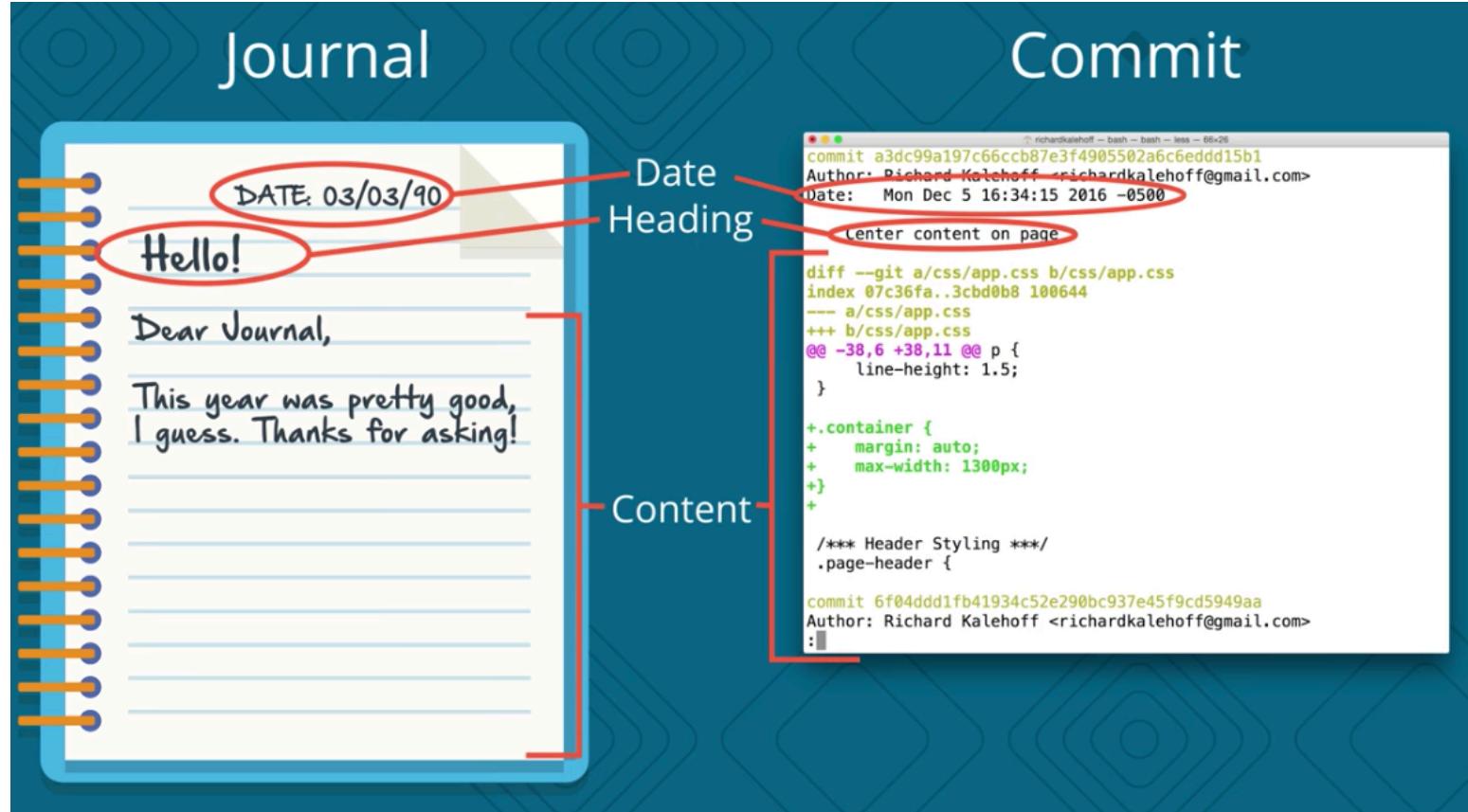
```
richardkalehoff (master) course-git-blog-project
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working directory clean
richardkalehoff (master) course-git-blog-project
$
```



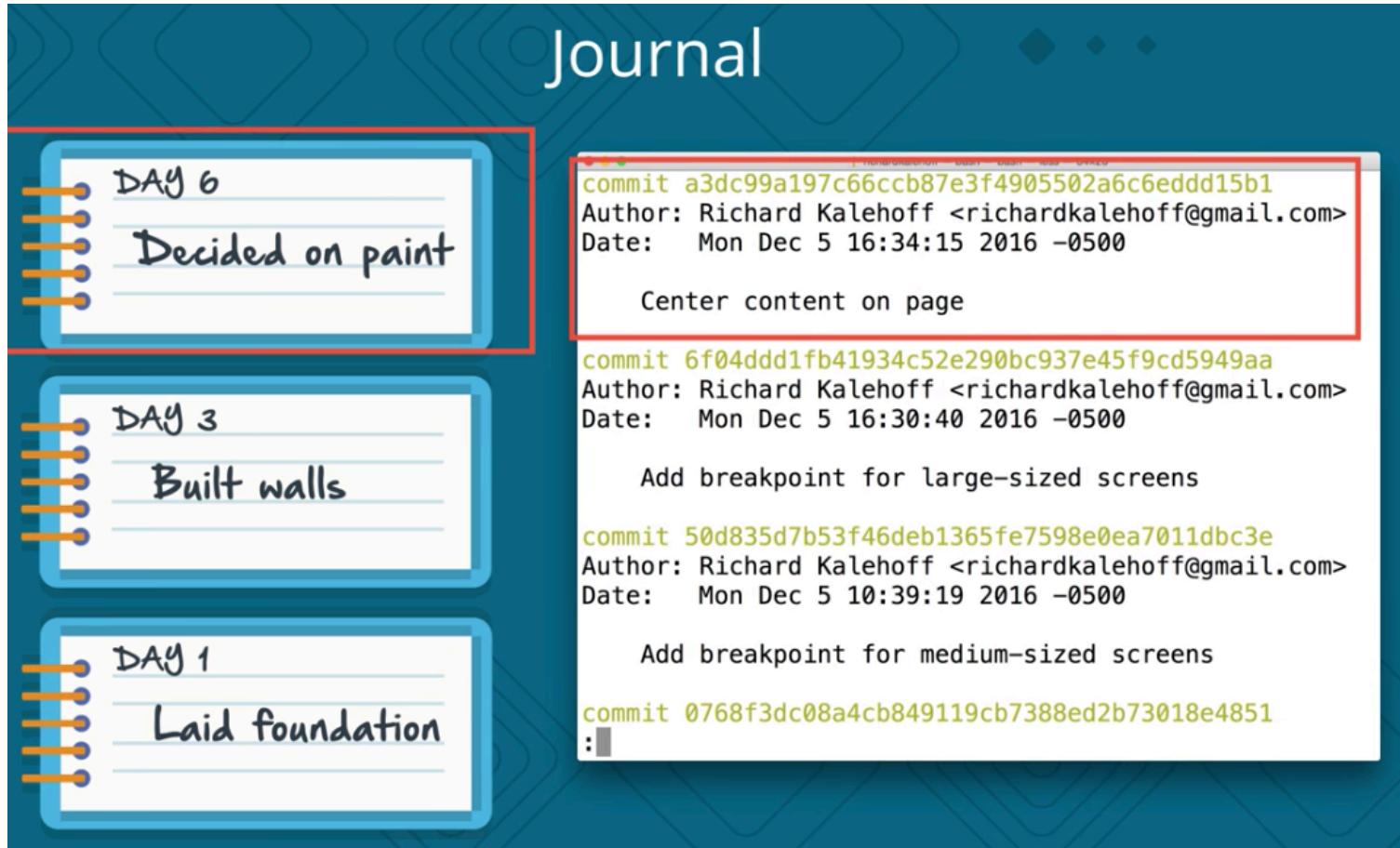
```
richardkalehoff (master #) new-git-project
$ git status
On branch master
Initial commit

nothing to commit (create/copy files and use "git add" to track)
richardkalehoff (master #) new-git-project
$
```

3. Review a Repo's History – Intro



3. Review a Repo's History – Init



3. Review a Repo's History – Init

git log

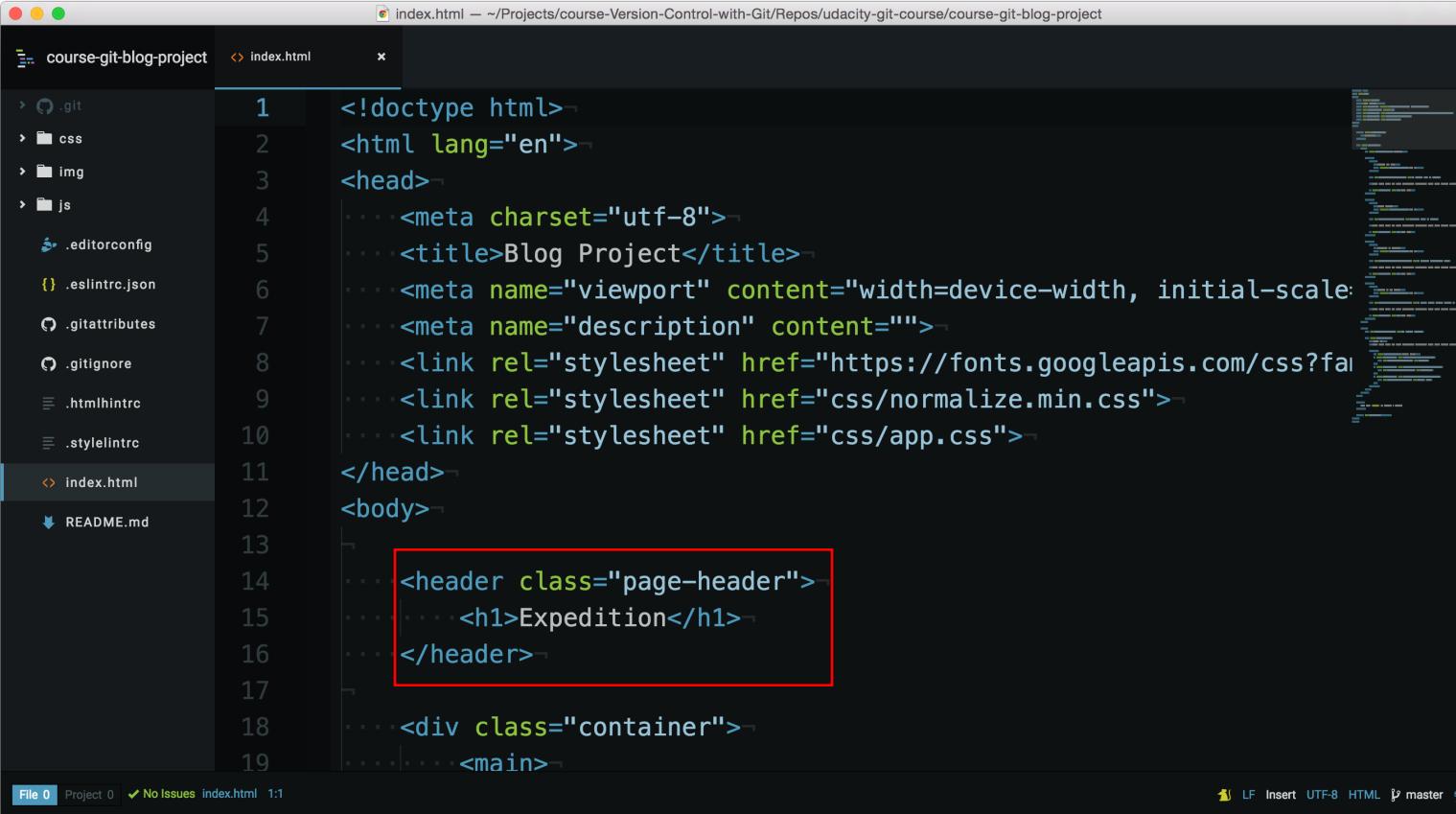
Displays information about the existing commits

git Show

Displays information about the given commit

3. Review a Repo's History – Displaying A Repository's Commits

Opening The Project (course-git-blog-project)



The screenshot shows a code editor window with the following details:

- Title Bar:** index.html — ~/Projects/course-Version-Control-with-Git/Repos/udacity-git-course/course-git-blog-project
- File Explorer:** course-git-blog-project folder containing .git, css, img, js, .editorconfig, .eslintrc.json, .gitattributes, .gitignore, .htmlintrc, .stylelintrc, index.html (selected), and README.md.
- Code Editor:** Content of index.html:

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>Blog Project</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="">
    <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Open+Sans:400,700&display=block">
    <link rel="stylesheet" href="css/normalize.min.css">
    <link rel="stylesheet" href="css/app.css">
</head>
<body>

    <header class="page-header">
        <h1>Expedition</h1>
    </header>

    <div class="container">
        <main>
```
- Bottom Status Bar:** File 0 Project 0 ✓ No Issues index.html 1:1 LF Insert UTF-8 HTML master

A red box highlights the header section of the HTML code, specifically the `<header>` block.

3. Review a Repo's History – Displaying A Repository's Commits

The Git Log Command (in course-git-blog-project)

```
$ git log
```

```
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
:|
```

Navigating The Log

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)

Less: [https://en.wikipedia.org/wiki/Less_\(Unix\)](https://en.wikipedia.org/wiki/Less_(Unix))

3. Review a Repo's History – Displaying A Repository's Commits

QUESTION

- Use **git log** to find the commit that has a SHA that starts with **f9720a**. Who made the commit?
- Use **git log** to find the commit with the SHA that starts with **8aa6668**. What is the message for that commit?
- Use **git log** to find the commit with the SHA that starts with **f9720a9**. When was that commit made?
- 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.

3. Review a Repo's History – Changing How Git Log Displays Information

git log --oneline

```
$ git log --oneline
```

```
course-Version-Control-with-Git — bash — 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

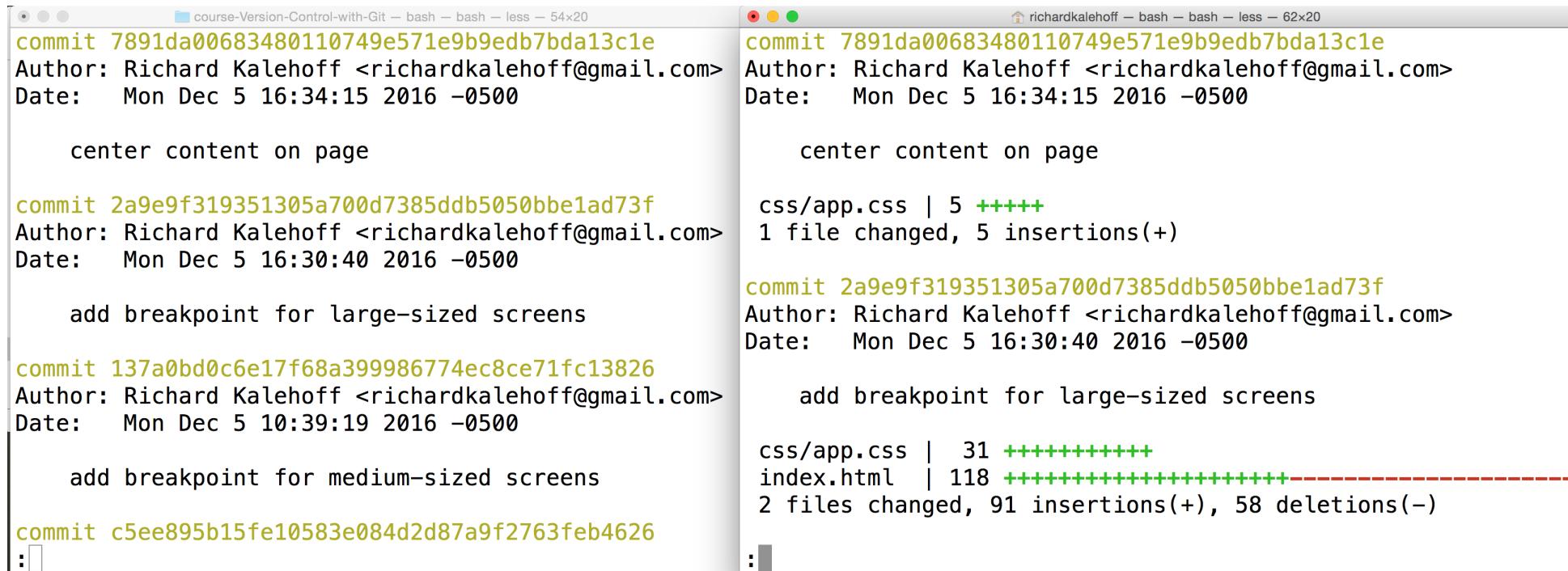
commit c5ee895b15fe10583e084d2d87a9f2763feb4626
:
```

```
richardkalehoff — bash — bash — less — 62x20
7891da0 center content on page
2a9e9f3 add breakpoint for large-sized screens
137a0bd add breakpoint for medium-sized screens
c5ee895 add space around page edge
b552fa5 style page header
f8c87c7 convert social links from text to images
f0521d0 add divider between content/footer
1ef56e2 add divider between main/side content
eb08e03 add missing profile picture
2d19c44 style 'read more' links
cdbaaa8 set paragraph line-height
7ee6d73 set default text color
d8fc39a set article timestamp color
670c0fb align article header content
bf7138e make article images responsive
0efab71 add 'visuallyhidden' helper class
98f6059 add article images
370c23f set default fonts
0e5f182 give body a default color
:
```

3. Review a Repo's History – Viewing Modified Files

git log --stat

```
$ git log --stat
```



The image shows two side-by-side terminal windows. Both windows have a title bar with the text "course-Version-Control-with-Git" and "richardkalehoff" respectively, followed by "bash - bash - less". The left window has a width of 54x20, and the right window has a width of 62x20.

Terminal 1 (Left):

```
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
:
```

Terminal 2 (Right):

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

    center content on page

css/app.css | 5 +++
1 file changed, 5 insertions(+)

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

    add breakpoint for large-sized screens

css/app.css | 31 ++++++
index.html  | 118 ++++++-----+
2 files changed, 91 insertions(+), 58 deletions(-)

:
```

3. Review a Repo's History – Viewing Modified Files

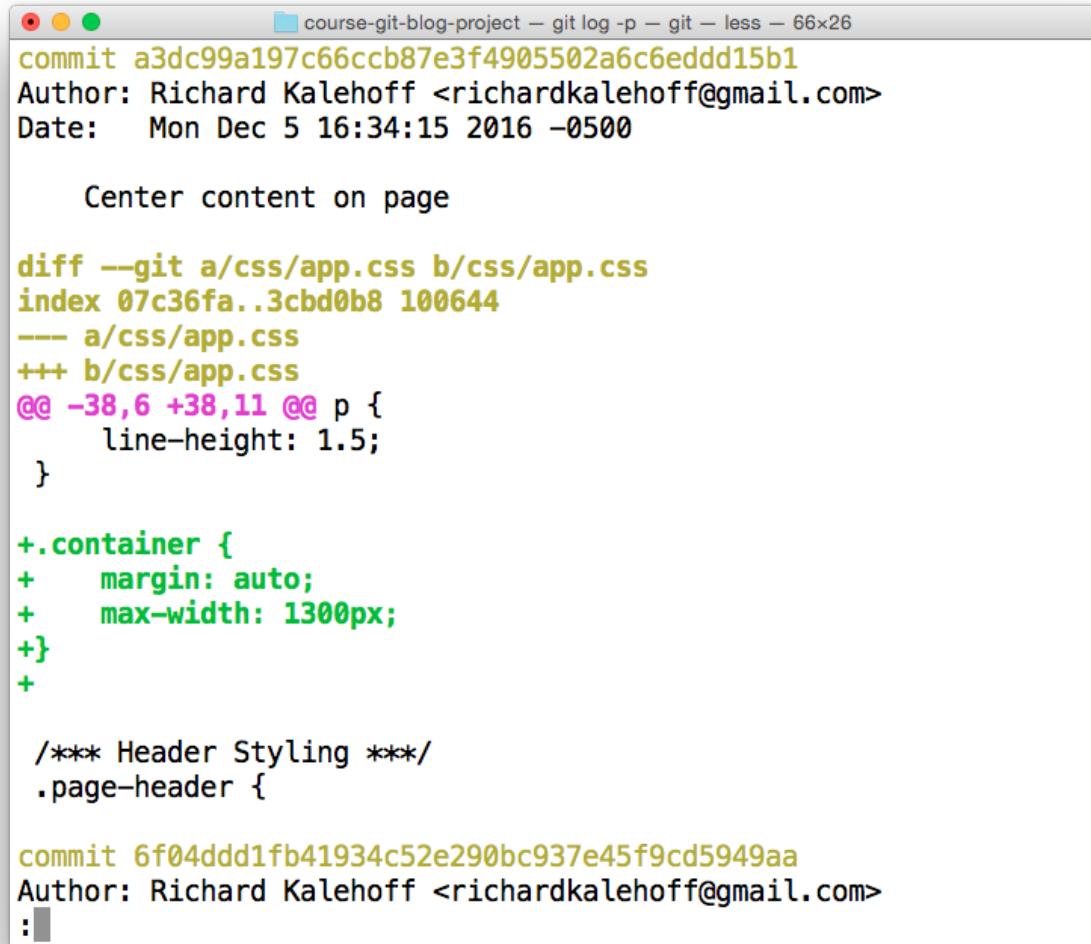
QUESTION

- How many files were modified in the commit with the SHA **6f04ddd**?
- How many files were modified in the commit with the SHA **8d3ea36**?
- How many lines of code were deleted in **index.html** in the commit with the SHA **8d3ea36**?

3. Review a Repo's History – Viewing File Changes

`git log -p`

`$ git log -p(--patch)`



The screenshot shows a terminal window titled "course-git-blog-project — git log -p — git — less — 66x26". It displays a git commit log and a diff patch. The commit details are:

```
commit a3dc99a197c66ccb87e3f4905502a6c6eddd15b1
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date:   Mon Dec 5 16:34:15 2016 -0500
```

The diff patch shows changes to app.css:

```
diff --git a/css/app.css b/css/app.css
index 07c36fa..3cbd0b8 100644
--- a/css/app.css
+++ b/css/app.css
@@ -38,6 +38,11 @@ p {
    line-height: 1.5;
}

+.container {
+  margin: auto;
+  max-width: 1300px;
+}
+

/** Header Styling */
.page-header {
```

At the bottom, another commit is shown:

```
commit 6f04ddd1fb41934c52e290bc937e45f9cd5949aa
Author: Richard Kalehoff <richardkalehoff@gmail.com>
:
```

3. Review a Repo's History – Viewing A Specific Commit

New Command : `git show`

`$ git show`

`$ git show fdf5493`

`$ git log -p fdf5493`

QUESTION

- How many Rulesets are added to the CSS by commit **8d3ea36**?
- There's a commit with the message "**Convert social links from text to images**". How many files were changed by this commit?
- Look at commit **fdf5493**. What's the first HTML heading element that's added by this commit?

Version Control with Git

Local Laboratory

4. Add Commits To A Repo – Intro

git add

Add files from the working directory to the staging index

git commit

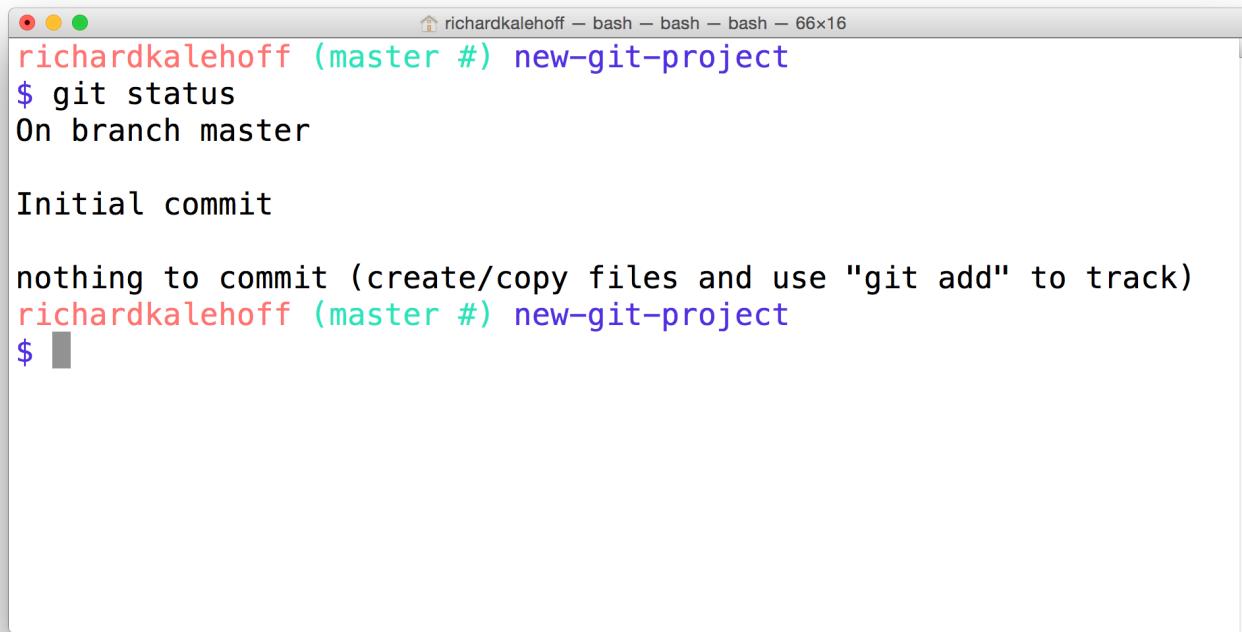
Take files from the staging index and save them in the repository

git diff

Displays the difference between two versions of a file

4. Add Commits To A Repo – Git Add

Status Status Status



```
richardkalehoff (master #) new-git-project
$ git status
On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)
richardkalehoff (master #) new-git-project
$
```

4. Add Commits To A Repo – Git Add

Create An HTML File

index.html

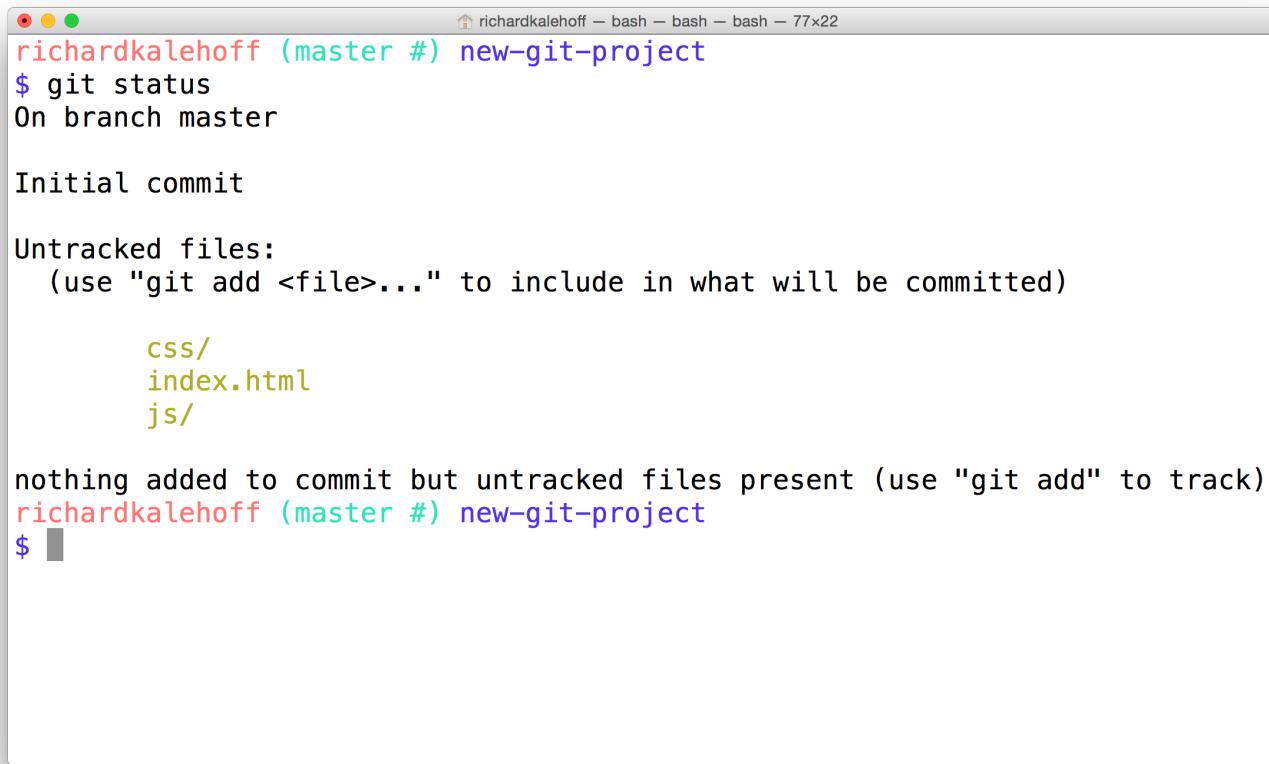
```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Blog Project</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="description" content="">
  <link rel="stylesheet" href="css/app.css">
</head>
<body>
  <script src="js/app.js"></script>
</body>
</html>
```

css/app.css
js/app.js

4. Add Commits To A Repo – Git Add

Quick Git Status Check

\$ git status



```
richardkalehoff (master #) new-git-project
$ git status
On branch master

Initial commit

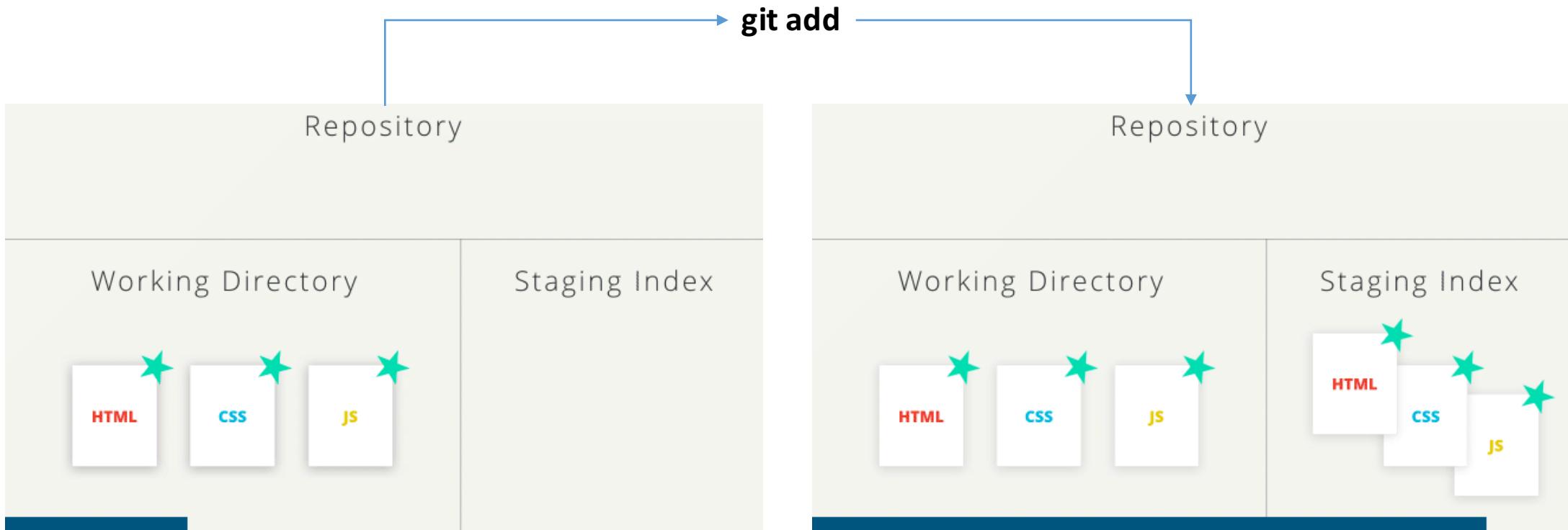
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    css/
    index.html
    js/

nothing added to commit but untracked files present (use "git add" to track)
richardkalehoff (master #) new-git-project
$
```

4. Add Commits To A Repo – Git Add

Big Picture Review

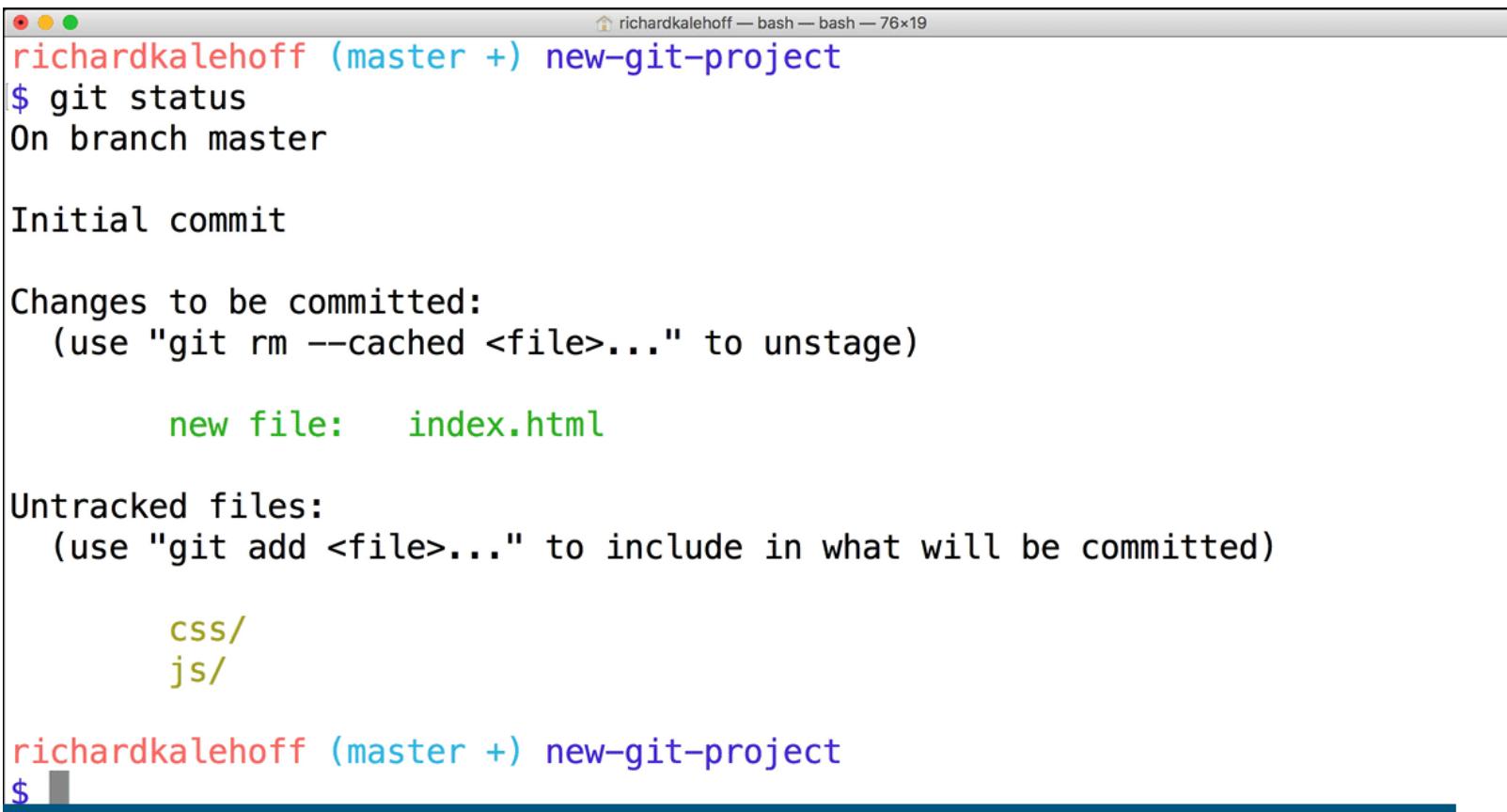


The untracked HTML, CSS, and JavaScript files add from the Working Directory to the Staging Index

4. Add Commits To A Repo – Git Add

Staging Files

```
$ git add index.html
```



A screenshot of a terminal window titled "richardkalehoff — bash — bash — 76x19". The window displays the following text:

```
richardkalehoff (master +) new-git-project
$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   index.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    css/
    js/

richardkalehoff (master +) new-git-project
$
```

4. Add Commits To A Repo – Git Add

Staging Remaining Files

```
$ git add css/app.css js/app.js
```

The Prid .

```
$ git add .
```



A screenshot of a terminal window titled "richardkalehoff — bash — bash — bash — 56x17". The window shows the following text:

```
richardkalehoff (master +) new-git-project
$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

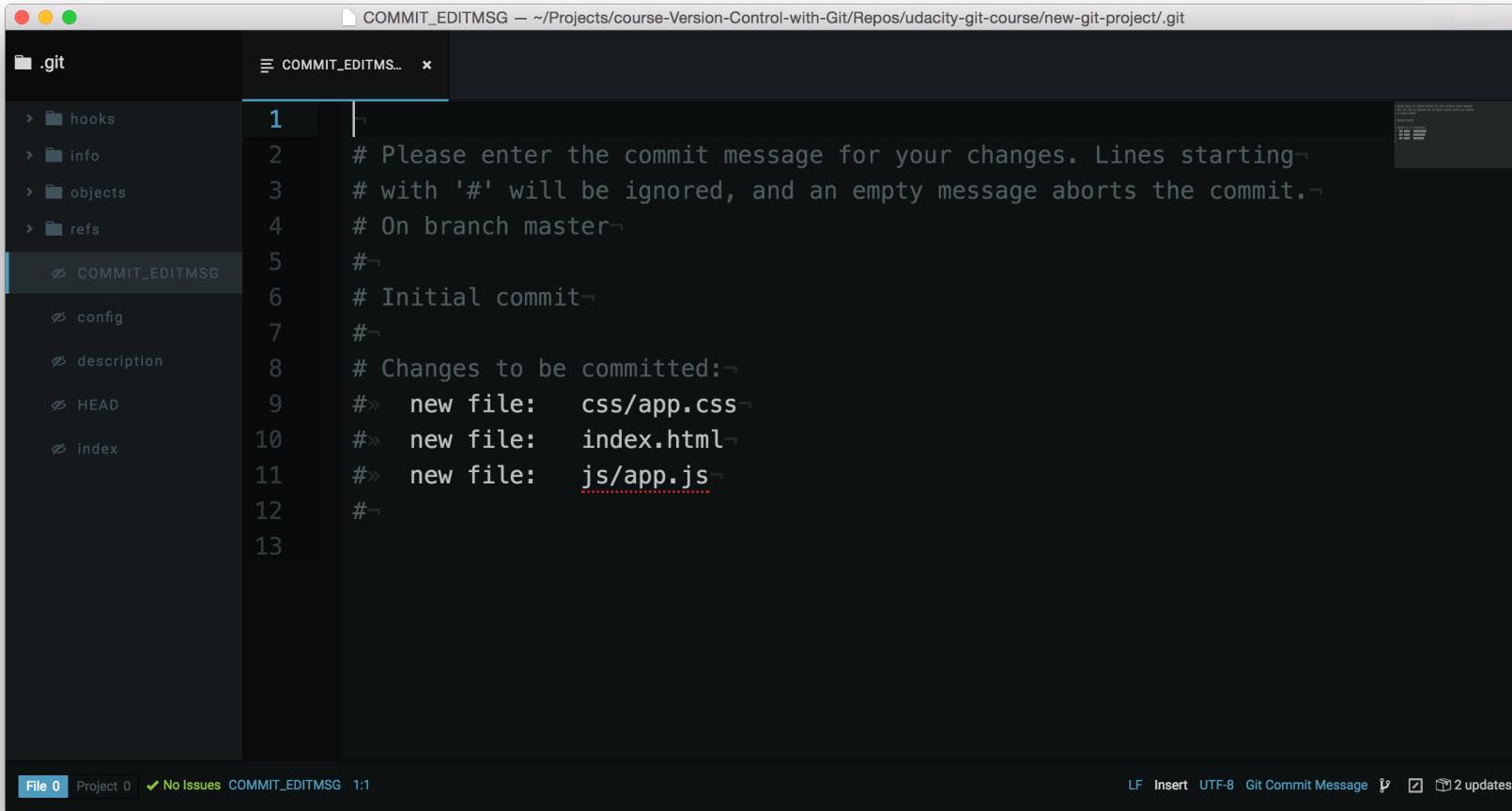
    new file:  css/app.css
    new file:  index.html
    new file:  js/app.js

richardkalehoff (master +) new-git-project
$
```

4. Add Commits To A Repo – Git Commit

Make A Commit

\$ git commit



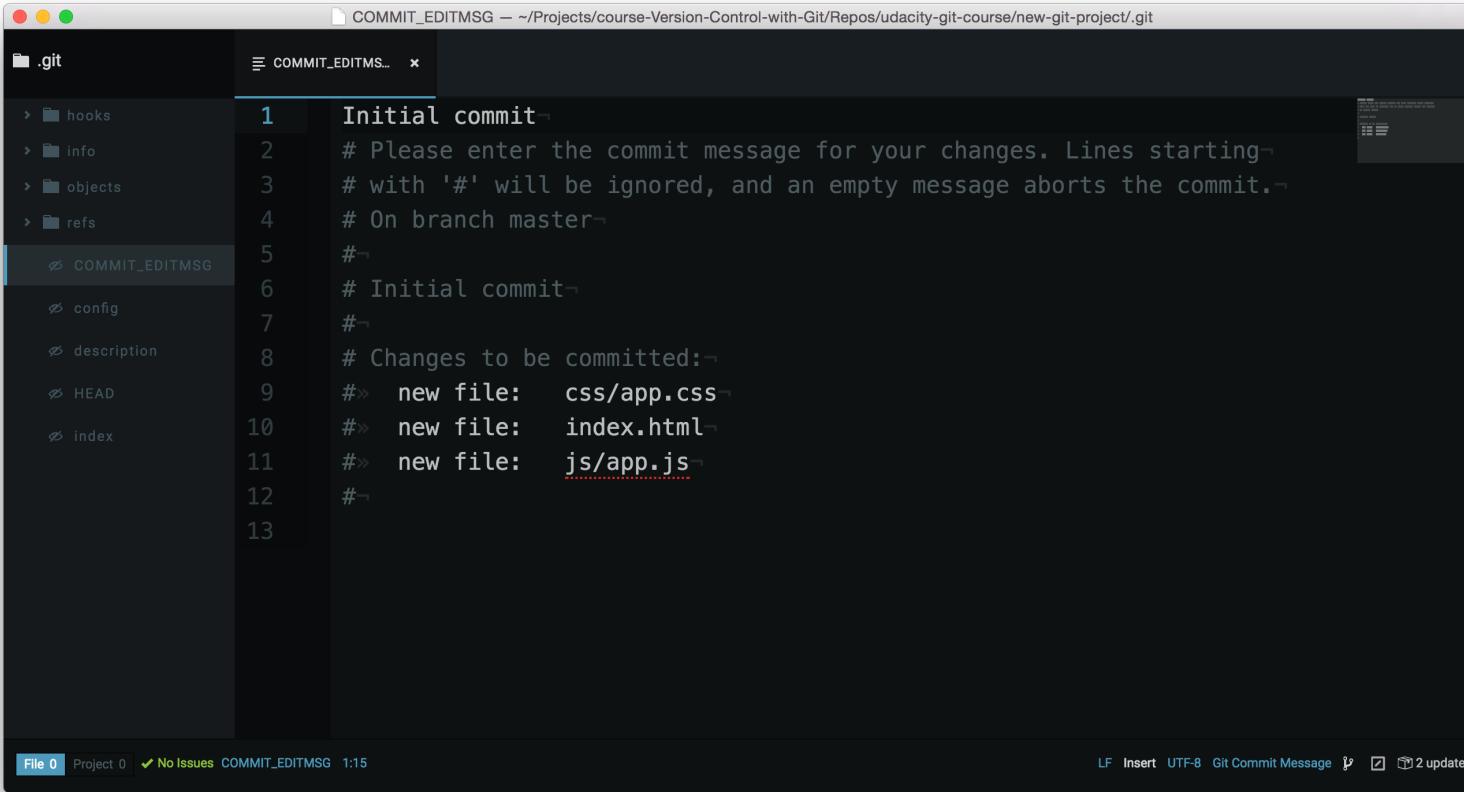
The screenshot shows a terminal window titled "COMMIT_EDITMSG" with the path "/Projects/course-Version-Control-with-Git/Repos/udacity-git-course/new-git-project/.git". The window displays a text editor with the following commit message:

```
1
2 # Please enter the commit message for your changes. Lines starting
3 # with '#' will be ignored, and an empty message aborts the commit.
4 # On branch master
5 #
6 # Initial commit
7 #
8 # Changes to be committed:
9 #> new file: css/app.css
10 #> new file: index.html
11 #> new file: js/app.js
12 #
13
```

The status bar at the bottom of the terminal window indicates "File 0 Project 0 ✓ No Issues COMMIT_EDITMSG 1:1" and "LF Insert UTF-8 Git Commit Message ⌘ ⌘ 2 updates".

4. Add Commits To A Repo – Git Commit

Code Editor Commit Message Explanation



The screenshot shows a code editor window titled "COMMIT_EDITMSG" with the file path "/Projects/course-Version-Control-with-Git/Repos/udacity-git-course/new-git-project/.git". The left sidebar shows a tree view of the repository structure, including ".git", "hooks", "info", "objects", "refs", "COMMIT_EDITMSG" (which is selected), "config", "description", "HEAD", and "index". The main editor area contains the following text:

```
1 Initial commit
2 # Please enter the commit message for your changes. Lines starting
3 # with '#' will be ignored, and an empty message aborts the commit.
4 # On branch master
5 #
6 # Initial commit
7 #
8 # Changes to be committed:
9 #> new file: css/app.css
10 #> new file: index.html
11 #> new file: js/app.js
12 #
13
```

At the bottom of the editor, there is a status bar with the following information: "File 0 Project 0 ✓ No Issues COMMIT_EDITMSG 1:15". To the right of the status bar are several icons: LF, Insert, UTF-8, Git Commit Message, a diff icon, and a 2 updates icon.

Bypass The Editor With The -m Flag

```
$ git commit -m "Initial commit"
```

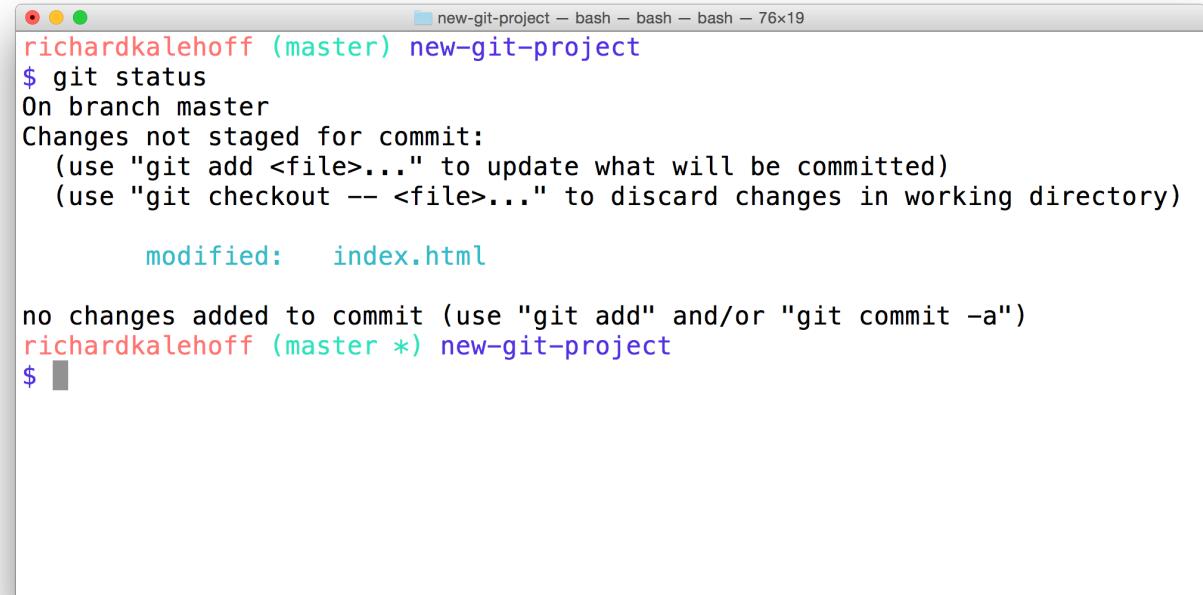
4. Add Commits To A Repo – Git Commit

2nd Commit – Add Changes

index.html의 body tag내에 아래 코드 추가

```
<header>
  <h1>Expedition</h1>
</header>
```

\$ git status



```
richardkalehoff (master) new-git-project
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
richardkalehoff (master *) new-git-project
$
```

4. Add Commits To A Repo – Git Commit

Second Commit

```
$ git commit -m "Add header to blog"
```

Multipurpose Git Add

Even though we used git add to add newly created files to the Staging Index, **we use the same command to move modified files to the Staging Index.**

4. Add Commits To A Repo – Git Commit

What To Include In A Commit

- The goal is that each commit has a **single focus**.
- Each commit should record a **single-unit change**.
- Now this can be a bit subjective (which is totally fine),
but each commit should make a **change to just one aspect of the project**.

4. Add Commits To A Repo – Commit Messages

Good Commit Messages

Do

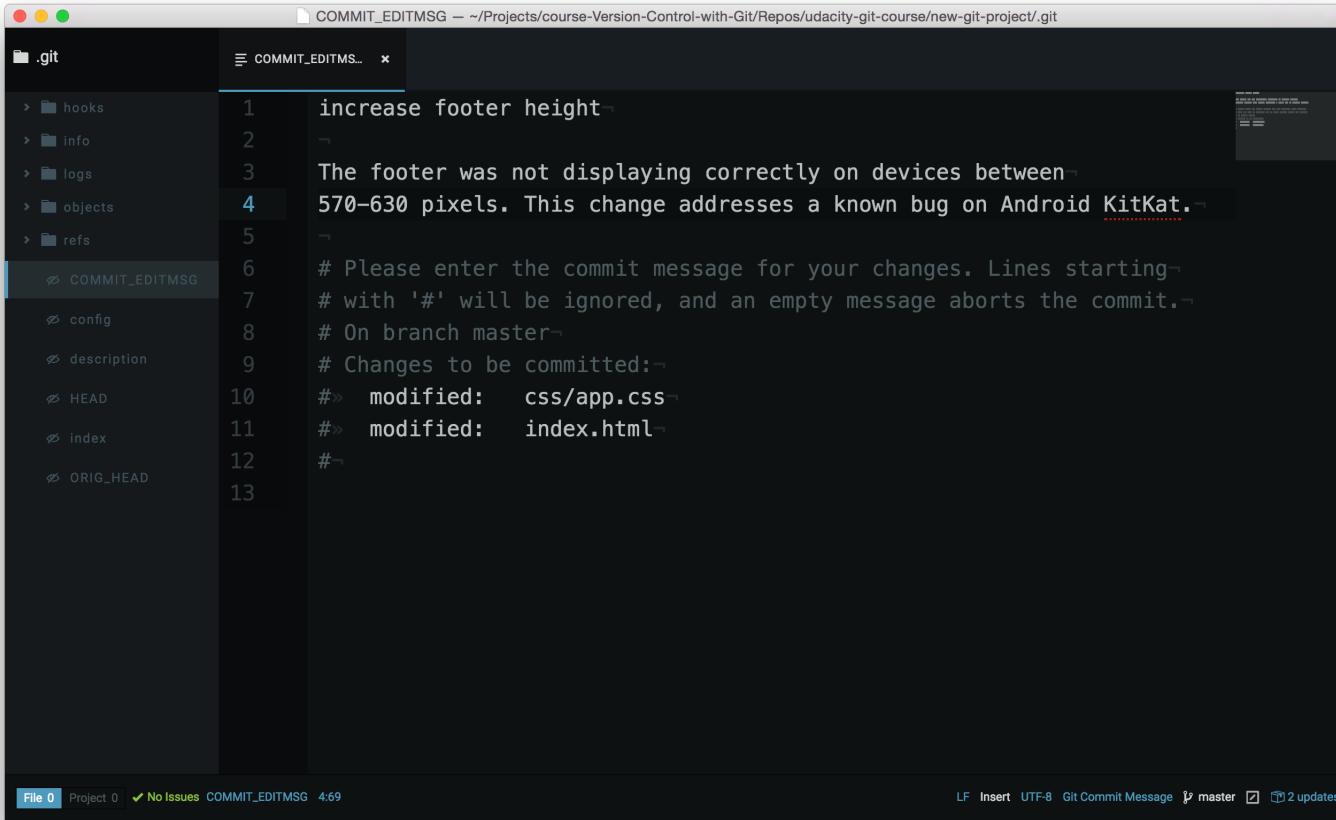
- do keep the **message short** (less than 60-ish characters)
- do **explain what the commit does** (not how or why!)

Do not

- do not explain **why the changes are made** (more on this below)
- do not explain **how the changes are made** (that's what git log -p is for!)
- do not use the word "**and**"
 - if you have to use "and", your commit message is probably **doing too many changes** - break the **changes into separate commits**
 - e.g. "make the background color pink and increase the size of the sidebar"

4. Add Commits To A Repo – Commit Messages

Explain the *Why*



A screenshot of a terminal window titled "COMMIT_EDITMSG" showing a commit message for a file named ".git". The message is:

```
1 increase footer height
2
3 The footer was not displaying correctly on devices between
4 570–630 pixels. This change addresses a known bug on Android KitKat.
5
6 # Please enter the commit message for your changes. Lines starting
7 # with '#' will be ignored, and an empty message aborts the commit.
8 # On branch master
9 # Changes to be committed:
10 #> modified: css/app.css
11 #> modified: index.html
12 #
13
```

The terminal window also shows a sidebar with various git files like .git, hooks, info, logs, objects, refs, COMMIT_EDITMSG, config, description, HEAD, index, and ORIG_HEAD.

\$ git log
\$ git log --oneline

8a11b3f

4. Add Commits To A Repo – Git Diff

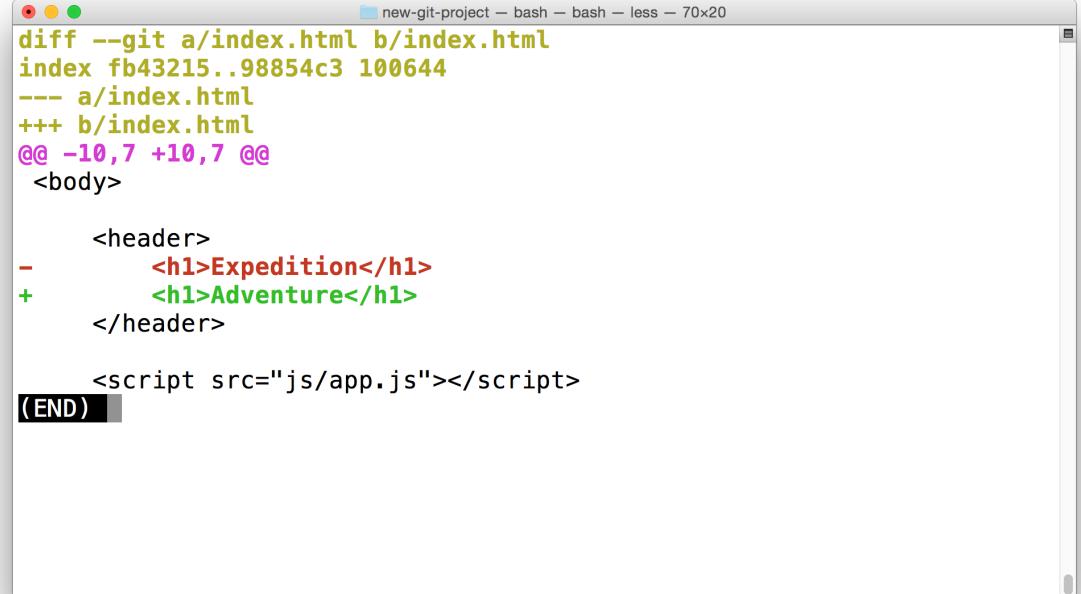
Why Do We Need This

git diff

index.html의 h1태그 내용 변경

```
<header>
  <h1>Adventure</h1>
</header>
```

\$ git diff



```
diff --git a/index.html b/index.html
index fb43215..98854c3 100644
--- a/index.html
+++ b/index.html
@@ -10,7 +10,7 @@
<body>

  <header>
-    <h1>Expedition</h1>
+    <h1>Adventure</h1>
  </header>

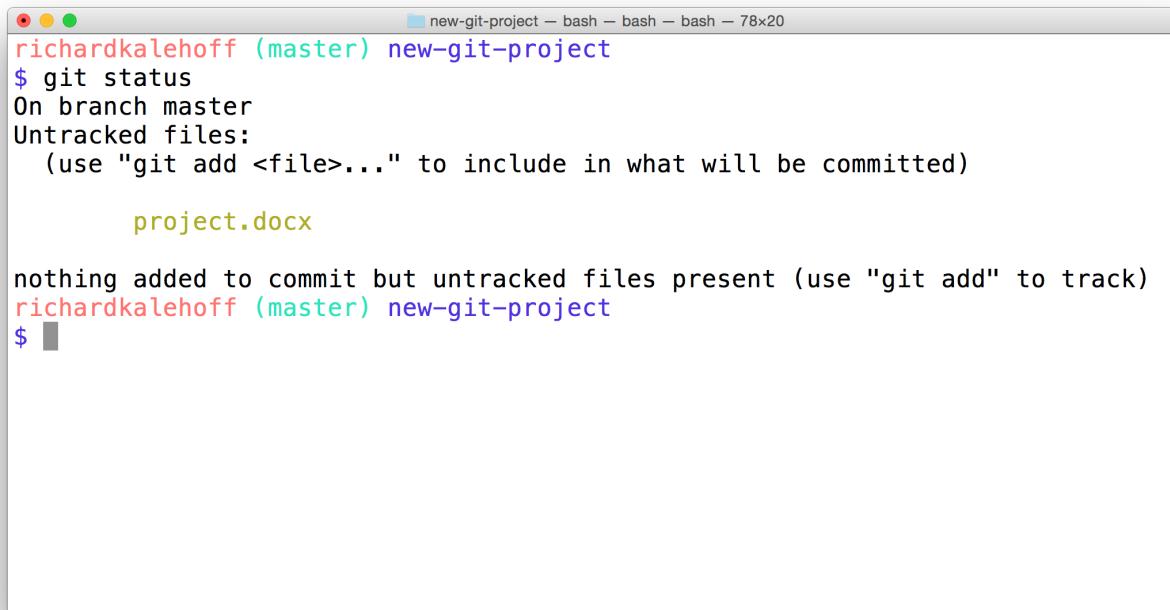
  <script src="js/app.js"></script>
(END)
```

4. Add Commits To A Repo – Having Git Ignore Files

Why Should Files Be Ignored

The Problem

```
$ touch project.docx  
$ git status
```



A screenshot of a terminal window titled "new-git-project — bash — bash — bash — 78x20". The window shows the following command and its output:

```
richardkalehoff (master) new-git-project  
$ git status  
On branch master  
Untracked files:  
  (use "git add <file>..." to include in what will be committed)  
  
    project.docx  
  
nothing added to commit but untracked files present (use "git add" to track)  
richardkalehoff (master) new-git-project  
$
```

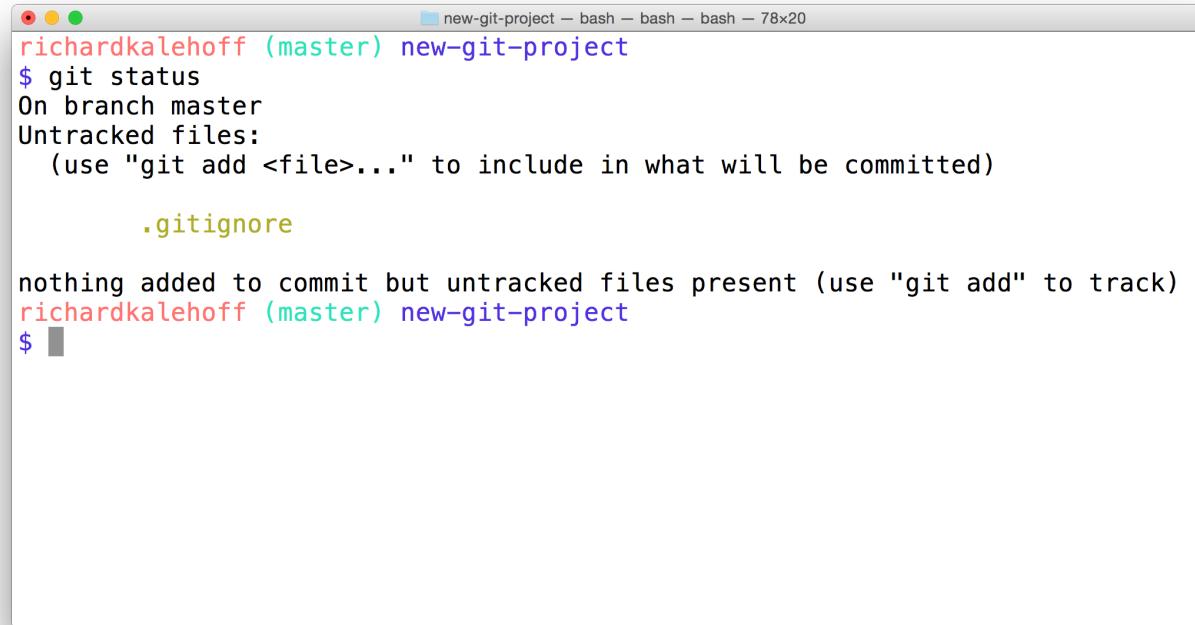
git add . 실행시 문제 발생

4. Add Commits To A Repo – Having Git Ignore Files

Git Ignore

- git project 내에 .gitignore 파일 생성
- .gitignore 파일에 제외할 파일명(project.docx) 추가

\$ git status



```
richardkalehoff (master) new-git-project
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    .gitignore

nothing added to commit but untracked files present (use "git add" to track)
richardkalehoff (master) new-git-project
$
```

\$ git add .

\$ git commit -m "Add .gitignore file"

4. Add Commits To A Repo – Having Git Ignore Files

Grobbig Crash Course

blank lines can be used for spacing

- # - marks line as a comment
- * - matches 0 or more characters
- ? - matches 1 character
- [abc] - matches a, b, or c
- ** - matches nested directories - a/**/z matches
 - a/z
 - a/b/z
 - a/b/c/z

ex) samples/*.jpg

Version Control with Git

Local Laboratory

5. Tagging, Branching, and Merging – Intro

git tag

Add tags to specific commits

git branch

Allows multiple lines of development

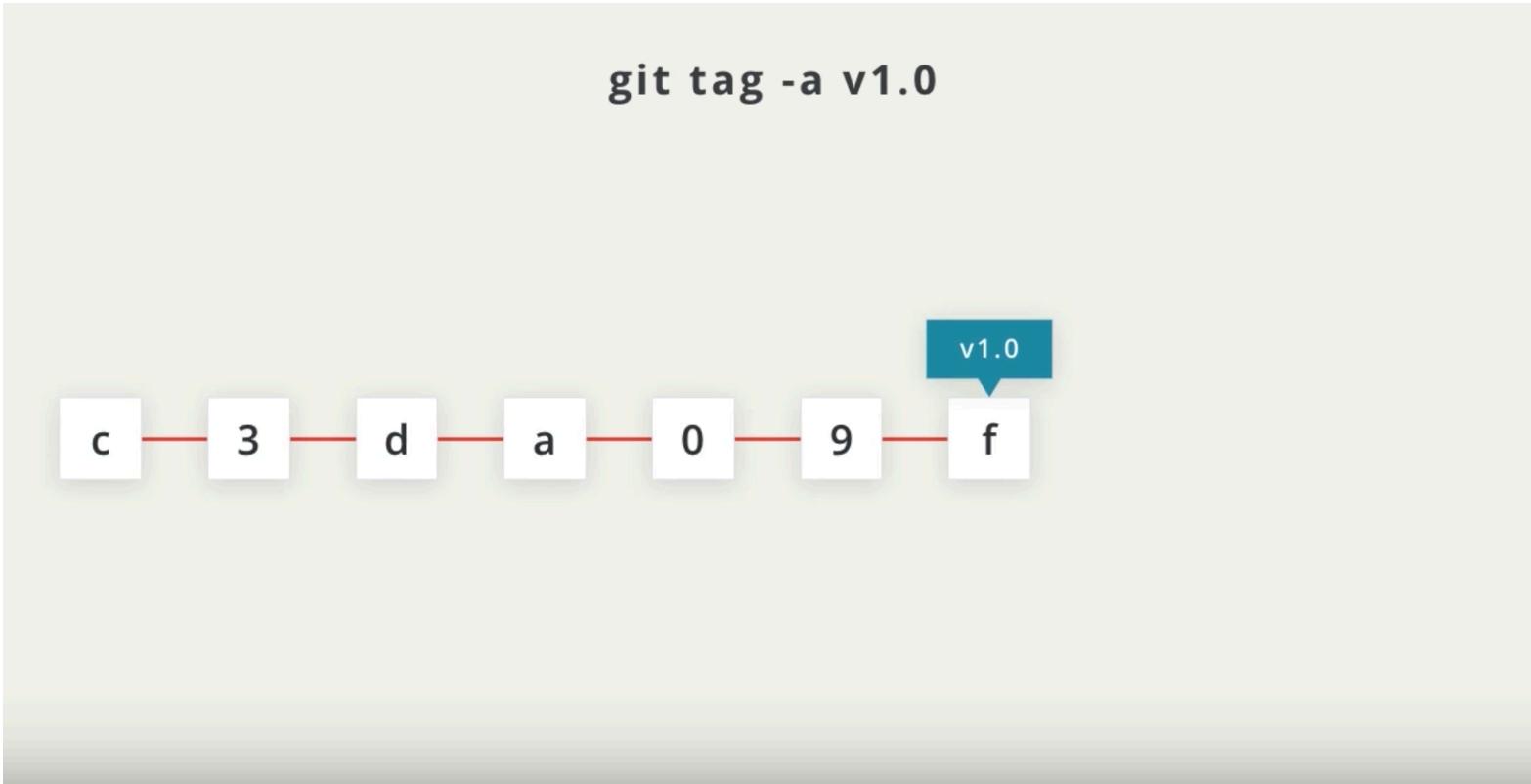
git checkout

Switch between different branches and tags

git merging

Combines changes on different branches

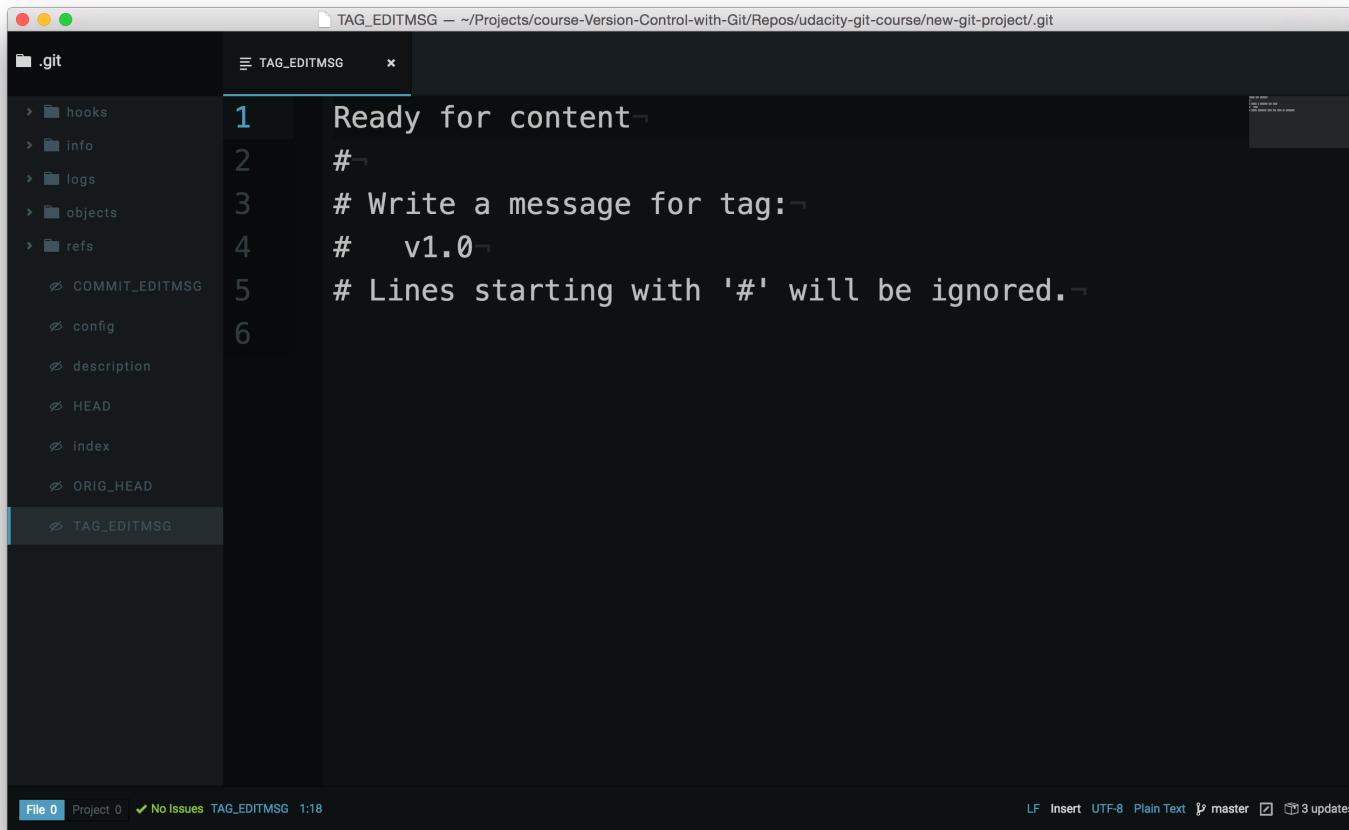
5. Tagging, Branching, and Merging – Tagging



5. Tagging, Branching, and Merging – Tagging

Git Tag Command

```
$ git tag -a v1.0
```



The screenshot shows a terminal window titled "TAG_EDITMSG" with the path "/Projects/course-Version-Control-with-Git/Repos/udacity-git-course/new-git-project/.git". The window displays the following text:

```
1 Ready for content
2 #
3 # Write a message for tag:
4 #   v1.0
5 # Lines starting with '#' will be ignored.
6
```

The terminal interface includes a sidebar on the left showing the project structure, and a status bar at the bottom with "File 0", "Project 0", "No Issues", "TAG_EDITMSG", "1:18", "LF", "Insert", "UTF-8", "Plain Text", "master", and "3 updates".

5. Tagging, Branching, and Merging – Tagging

Verify Tag

```
$ git tag
```



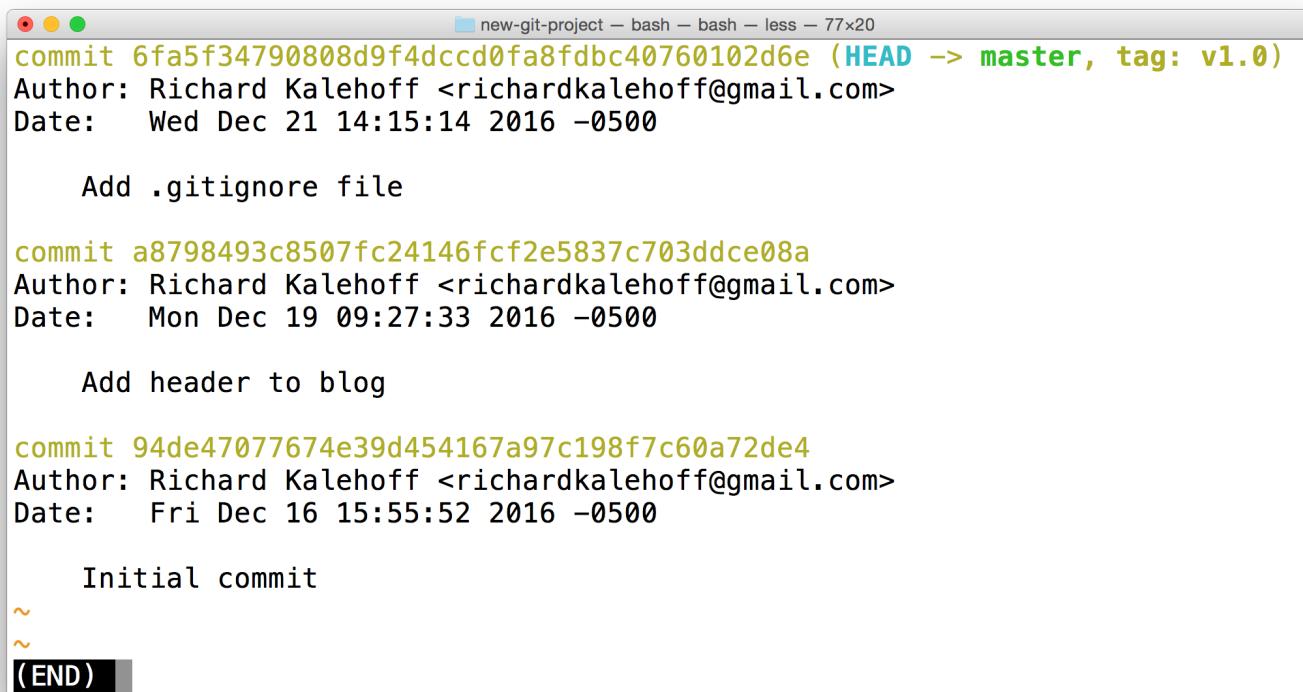
A screenshot of a terminal window titled "new-git-project — bash — bash — bash — 73x20". The window shows the following command and its output:

```
richardkalehoff (master) new-git-project
$ git tag
v1.0
richardkalehoff (master) new-git-project
$
```

5. Tagging, Branching, and Merging – Tagging

Git Log's --decorate Flag (Git 2.13 미만 버전 해당)

```
$ git log --decorate
```



The screenshot shows a terminal window titled "new-git-project" with the command "git log --decorate" running. The output displays three commits from Richard Kalehoff:

```
commit 6fa5f34790808d9f4dccb0fa8fdbca0760102d6e (HEAD -> master, tag: v1.0)
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date:   Wed Dec 21 14:15:14 2016 -0500

    Add .gitignore file

commit a8798493c8507fc24146fcf2e5837c703ddce08a
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date:   Mon Dec 19 09:27:33 2016 -0500

    Add header to blog

commit 94de47077674e39d454167a97c198f7c60a72de4
Author: Richard Kalehoff <richardkalehoff@gmail.com>
Date:   Fri Dec 16 15:55:52 2016 -0500

    Initial commit

~
~
```

(END)

5. Tagging, Branching, and Merging – Tagging

Deleting A Tag

```
$ git tag -d v1.0
```



The screenshot shows a terminal window titled "new-git-project" with three tabs open. The current tab is "bash". The command \$ git tag -d v1.0 is entered, followed by the output: Deleted tag 'v1.0' (was 76631c3).

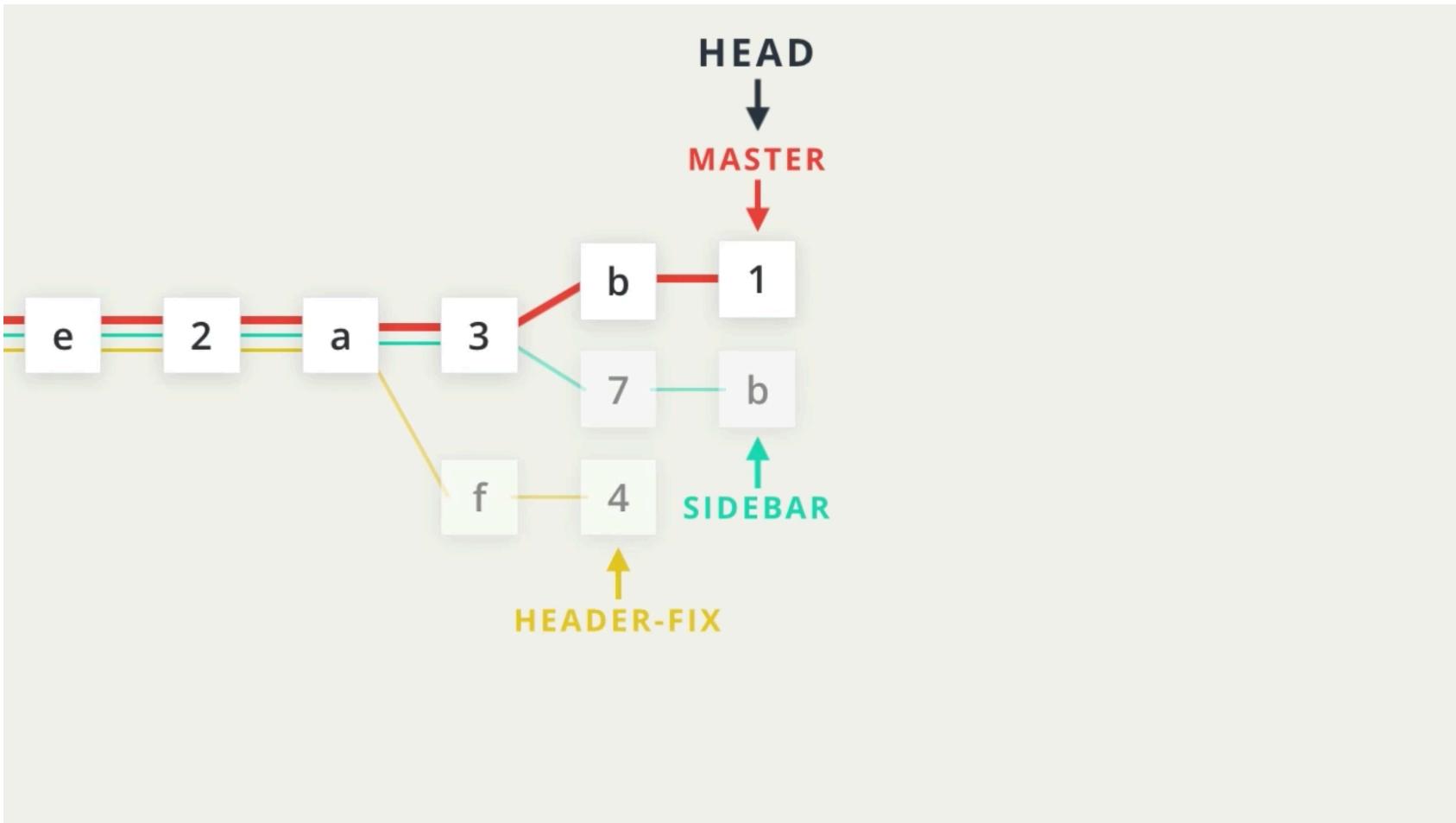
```
richardkalehoff (master) new-git-project
$ git tag
v1.0
richardkalehoff (master) new-git-project
$ git tag -d v1.0
Deleted tag 'v1.0' (was 76631c3)
richardkalehoff (master) new-git-project
$
```

5. Tagging, Branching, and Merging – Tagging

Adding A Tag To A Past Commit

```
$ git tag -a v1.0 a8984
```

5. Tagging, Branching, and Merging – Branching

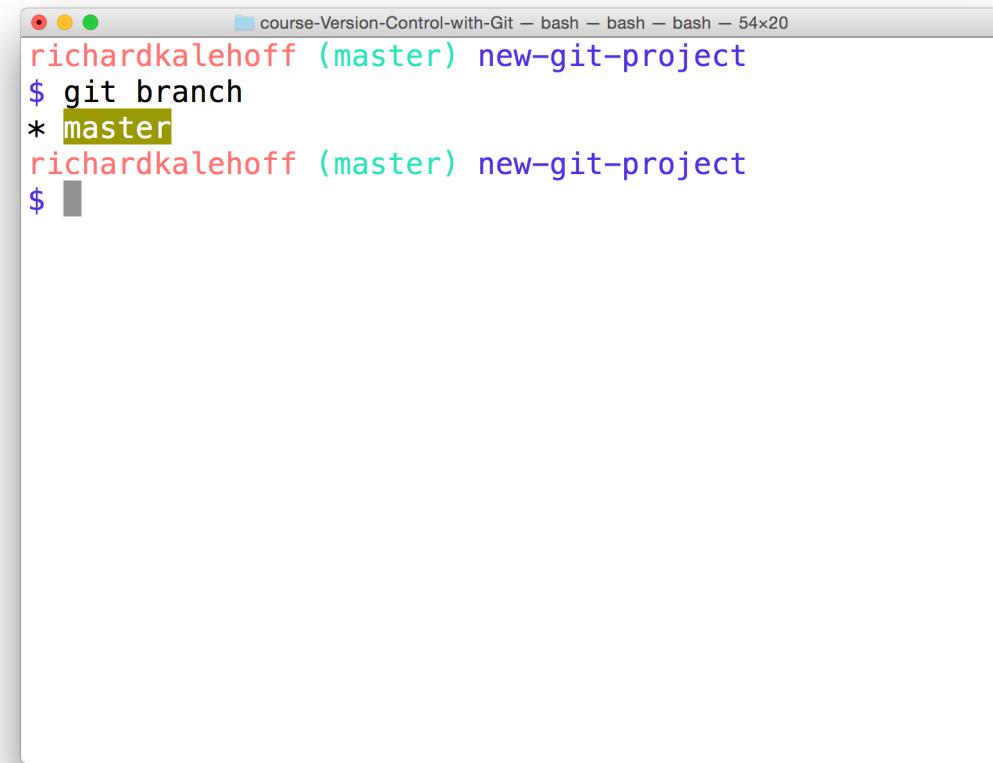


5. Tagging, Branching, and Merging – Branching

git branch command

```
$ git branch
```

- list all branch names in the repository
- create new branches
- delete branches



A screenshot of a terminal window titled "course-Version-Control-with-Git – bash – bash – bash – 54x20". The window shows the command "richardkalehoff (master) new-git-project" followed by the output of the "git branch" command. The output shows a single branch named "master" with an asterisk (*) next to it, indicating it is the current branch. The terminal window has a standard OS X look with red, yellow, and green window control buttons.

```
richardkalehoff (master) new-git-project
$ git branch
* master
richardkalehoff (master) new-git-project
$
```

5. Tagging, Branching, and Merging – Branching

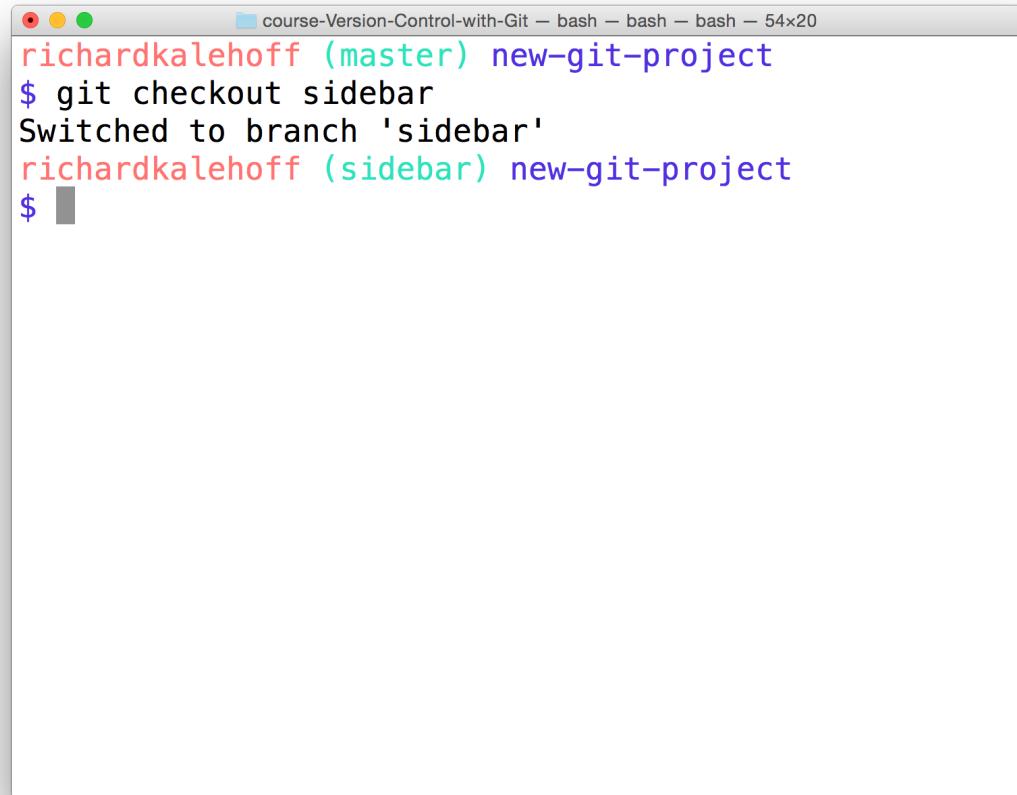
Create A Branch

```
$ git branch sidebar
```

5. Tagging, Branching, and Merging – Branching

The git checkout Command

\$ git checkout sidebar



A screenshot of a terminal window titled "course-Version-Control-with-Git — bash — bash — bash — 54x20". The window shows the following text:

```
richardkalehoff (master) new-git-project
$ git checkout sidebar
Switched to branch 'sidebar'
richardkalehoff (sidebar) new-git-project
$
```

5. Tagging, Branching, and Merging – Branching

Branches In The Log

```
$ git log --oneline --decorate
```



A screenshot of a terminal window titled "course-Version-Control-with-Git — bash — bash — less — 54x20". The window displays a git log output with three commits:

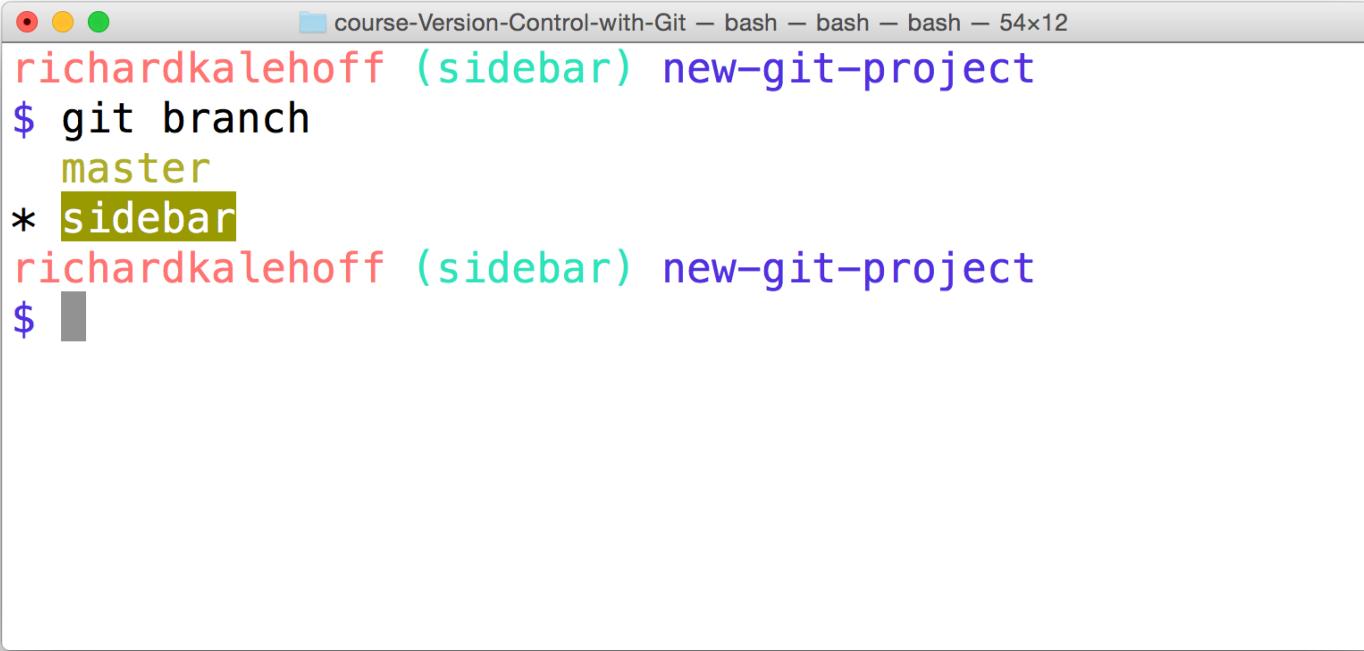
- 6fa5f34 (HEAD -> sidebar, master) Add .gitignore file
- a879849 Add header to blog
- 94de470 Initial commit

The word "(END)" is visible at the bottom of the log output.

5. Tagging, Branching, and Merging – Branching

The Active Branch

```
$ git branch
```



```
richardkalehoff (sidebar) new-git-project
$ git branch
  master
* sidebar
richardkalehoff (sidebar) new-git-project
$
```

5. Tagging, Branching, and Merging – Branching

Specific Branch

```
$ git branch alt-sidebar-loc 42a69f
```

5. Tagging, Branching, and Merging – Branching

Delete A Branch

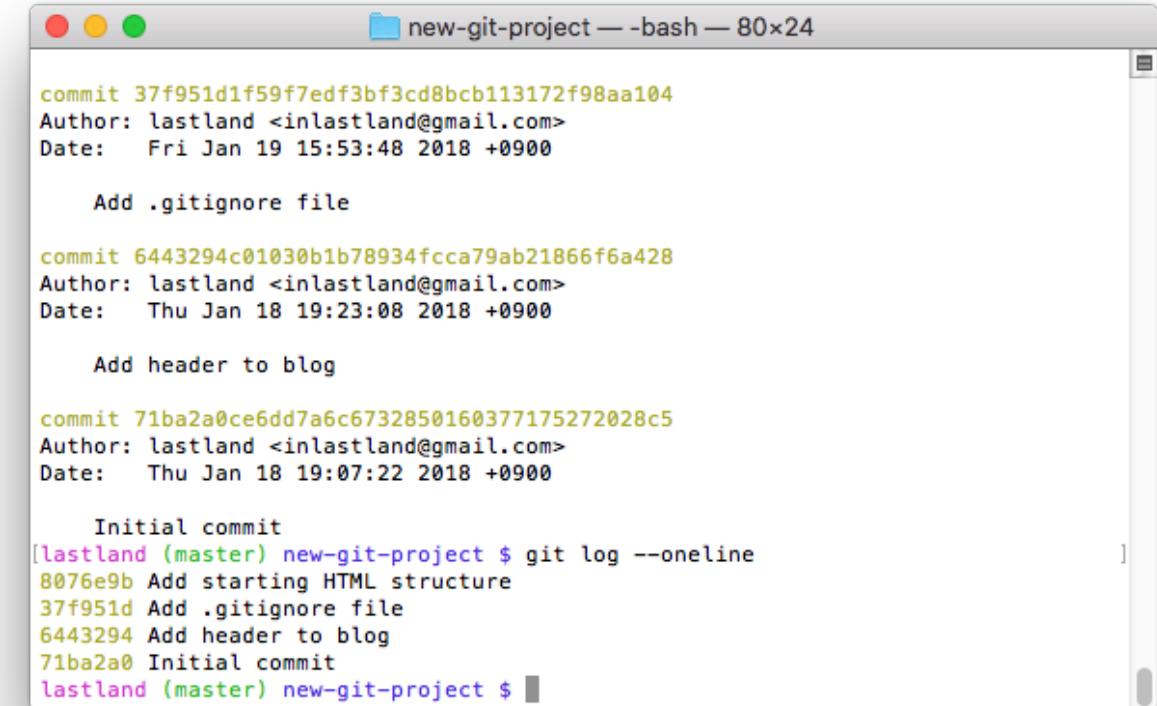
```
$ git branch -d sidebar
```

```
$ git branch -D sidebar
```

5. Tagging, Branching, and Merging – Branching Effectively

index.html 수정

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Blog Project</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="description" content="">
  <link rel="stylesheet" href="css/app.css">
</head>
<body>
  <header>
    <h1>Expedition</h1>
  </header>
  <div class="container">
    <main>
    </main>
  </div>
  <footer>
    Made with ❤ @ Udacity
  </footer>
  <script src="js/app.js">
</script>
</body>
</html>
```



A screenshot of a terminal window titled "new-git-project — bash — 80x24". The window displays a history of four commits:

- commit 37f951d1f59f7edf3bf3cd8bcb113172f98aa104
Author: lastland <inlastland@gmail.com>
Date: Fri Jan 19 15:53:48 2018 +0900
Add .gitignore file
- commit 6443294c01030b1b78934fccca79ab21866f6a428
Author: lastland <inlastland@gmail.com>
Date: Thu Jan 18 19:23:08 2018 +0900
Add header to blog
- commit 71ba2a0ce6dd7a6c6732850160377175272028c5
Author: lastland <inlastland@gmail.com>
Date: Thu Jan 18 19:07:22 2018 +0900
Initial commit
- [lastland (master) new-git-project \$ git log --oneline
8076e9b Add starting HTML structure
37f951d Add .gitignore file
6443294 Add header to blog
71ba2a0 Initial commit
lastland (master) new-git-project \$]

\$ git commit -m "Add starting HTML structure"

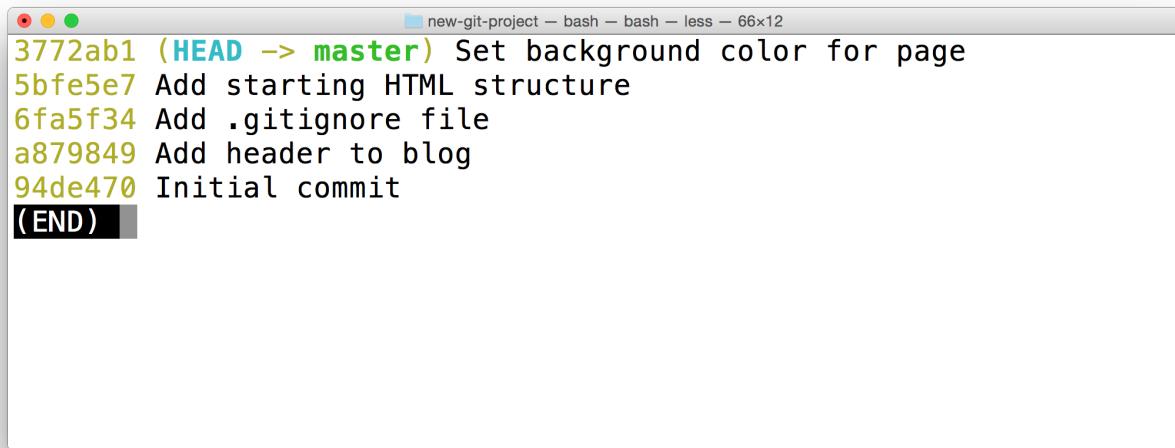
5. Tagging, Branching, and Merging – Branching Effectively

Change 1 - Add Page Color

css/app.css 수정 후 커밋

```
body {  
    background-color: #00cae4;  
}
```

\$ git commit -m "Set background color for page"



A screenshot of a terminal window titled "new-git-project". The window shows a list of commits:

- 3772ab1 (HEAD -> master) Set background color for page
- 5bfe5e7 Add starting HTML structure
- 6fa5f34 Add .gitignore file
- a879849 Add header to blog
- 94de470 Initial commit

The word "(END)" is visible at the bottom of the list.

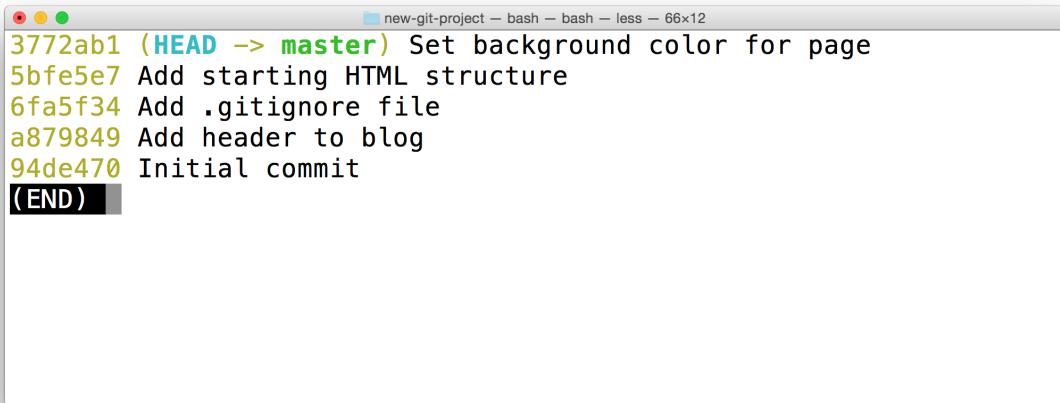
5. Tagging, Branching, and Merging – Branching Effectively

Change 2 – Add Sidebar

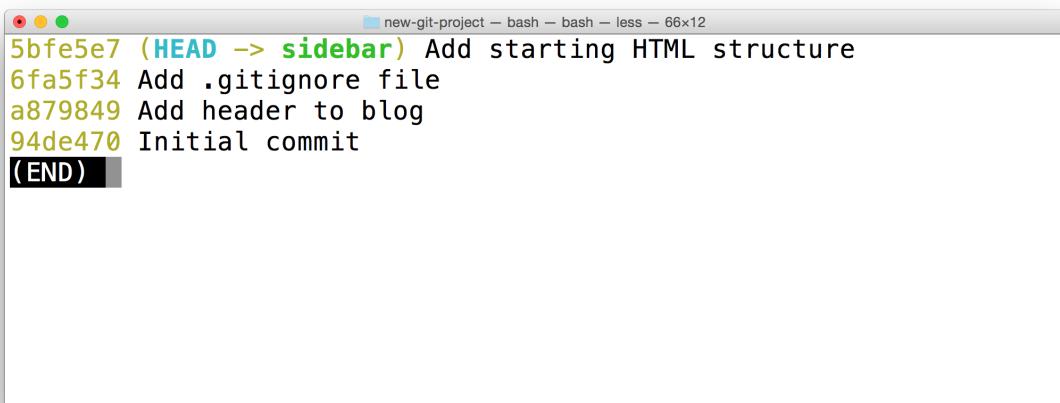
```
$ git branch sidebar 5bfe5e7
```

```
$ git checkout sidebar
```

```
$ git log --oneline
```



```
3772ab1 (HEAD -> master) Set background color for page
5bfe5e7 Add starting HTML structure
6fa5f34 Add .gitignore file
a879849 Add header to blog
94de470 Initial commit
(END)
```



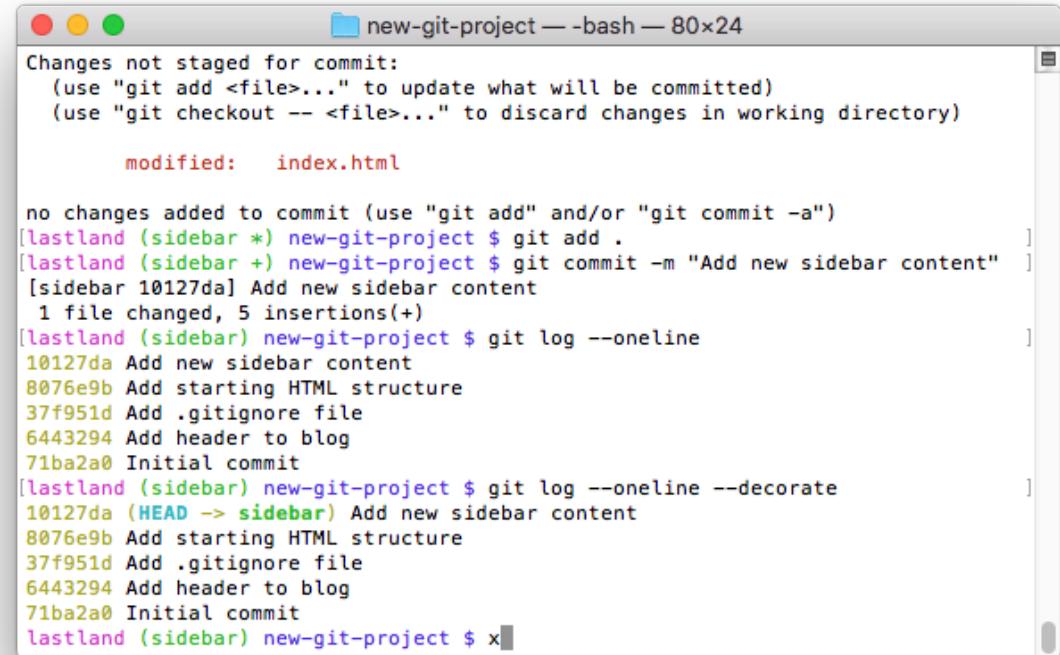
```
5bfe5e7 (HEAD -> sidebar) Add starting HTML structure
6fa5f34 Add .gitignore file
a879849 Add header to blog
94de470 Initial commit
(END)
```

5. Tagging, Branching, and Merging – Branching Effectively

index.html 수정

```
<div class="container">
  <main>
    <aside>
      <h2>About Me</h2>
      <p>Lorem ipsum dolor sit amet, consectetur
      adipisicing elit. Eos, debitis earum molestias veniam suscipit
      aliquam totam exercitationem tempore neque vitae. Minima,
      corporis pariatur facere at quo porro beatae similique!
      Odit.</p>
    </aside>
  </main>
</div>
<footer>
  Made with ❤ @ Udacity
</footer>
```

\$ git commit -m "Add new sidebar content"



The terminal window shows the following git log output:

```
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

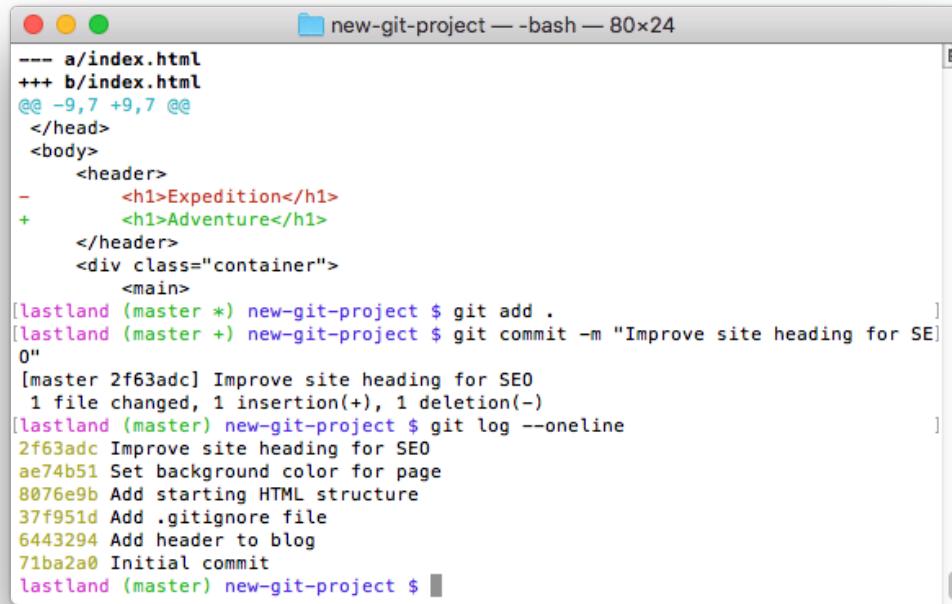
no changes added to commit (use "git add" and/or "git commit -a")
[lastland (sidebar *) new-git-project $ git add .]
[lastland (sidebar +) new-git-project $ git commit -m "Add new sidebar content"]
[sidebar 10127da] Add new sidebar content
  1 file changed, 5 insertions(+)
[lastland (sidebar) new-git-project $ git log --oneline]
10127da Add new sidebar content
8076e9b Add starting HTML structure
37f951d Add .gitignore file
6443294 Add header to blog
71ba2a0 Initial commit
[lastland (sidebar) new-git-project $ git log --oneline --decorate]
10127da (HEAD -> sidebar) Add new sidebar content
8076e9b Add starting HTML structure
37f951d Add .gitignore file
6443294 Add header to blog
71ba2a0 Initial commit
lastland (sidebar) new-git-project $ x
```

5. Tagging, Branching, and Merging – Branching Effectively

Change 3 – Change Heading On Master

```
$ git checkout master
```

1. <h1> 태그 안의 “Expedition”을 변경(“Adventure”)
2. 커밋!



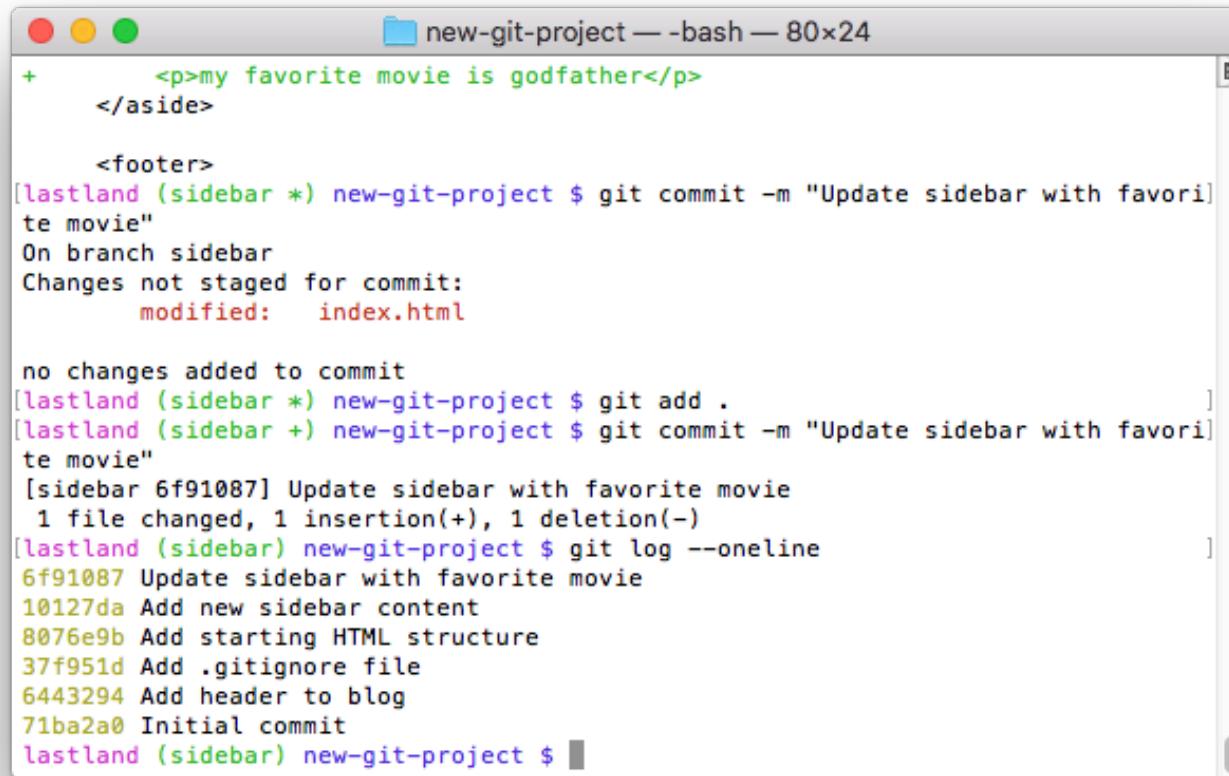
```
new-git-project — bash — 80x24
--- a/index.html
+++ b/index.html
@@ -9,7 +9,7 @@
</head>
<body>
  <header>
-    <h1>Expedition</h1>
+    <h1>Adventure</h1>
  </header>
  <div class="container">
    <main>
[lastland (master *) new-git-project $ git add .
[lastland (master +) new-git-project $ git commit -m "Improve site heading for SEO"
0"
[master 2f63adc] Improve site heading for SEO
 1 file changed, 1 insertion(+), 1 deletion(-)
[lastland (master) new-git-project $ git log --oneline
2f63adc Improve site heading for SEO
ae74b51 Set background color for page
8076e9b Add starting HTML structure
37f951d Add .gitignore file
6443294 Add header to blog
71ba2a0 Initial commit
lastland (master) new-git-project $ ]
```

```
$ git commit -m "Improve site heading for SEO"
```

5. Tagging, Branching, and Merging – Branching Effectively

Change 4 – Add More Content To Sidebar

1. <aside> 태그 안의 내용 변경 – 좋아하는 영화, 책 등
2. 커밋!



```
+      <p>my favorite movie is godfather</p>
+      </aside>

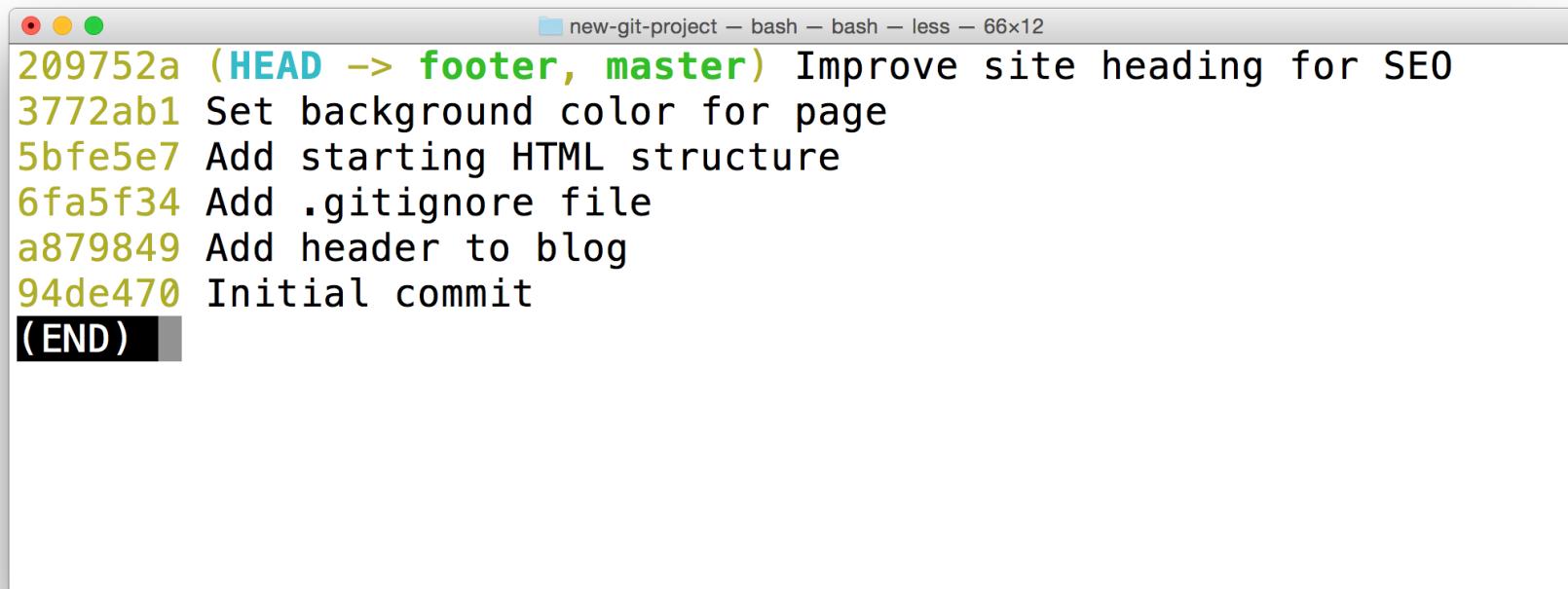
      <footer>
[lastland (sidebar *) new-git-project $ git commit -m "Update sidebar with favorite movie"
On branch sidebar
Changes not staged for commit:
  modified:   index.html

no changes added to commit
[lastland (sidebar *) new-git-project $ git add .]
[lastland (sidebar +) new-git-project $ git commit -m "Update sidebar with favorite movie"
[sidebar 6f91087] Update sidebar with favorite movie
  1 file changed, 1 insertion(+), 1 deletion(-)
[lastland (sidebar) new-git-project $ git log --oneline]
  6f91087 Update sidebar with favorite movie
  10127da Add new sidebar content
  8076e9b Add starting HTML structure
  37f951d Add .gitignore file
  6443294 Add header to blog
  71ba2a0 Initial commit
lastland (sidebar) new-git-project $ ]
```

5. Tagging, Branching, and Merging – Branching Effectively

Change 5 – Add Social Links To Footer

```
$ git checkout -b footer master  
$ git log --oneline --decorate
```



A screenshot of a terminal window titled "new-git-project — bash — bash — less — 66x12". The window displays a list of commits from a repository. The commits are color-coded: the first commit (HEAD) is blue, and subsequent commits are yellow. The commits are listed as follows:

- 209752a (**HEAD -> footer, master**) Improve site heading for SEO
- 3772ab1 Set background color for page
- 5bfe5e7 Add starting HTML structure
- 6fa5f34 Add .gitignore file
- a879849 Add header to blog
- 94de470 Initial commit

At the bottom of the list, there is a black rectangular button labeled "(END)".

5. Tagging, Branching, and Merging – Branching Effectively

Change 5 – Add Social Links To Footer

index.html 수정

```
<footer>
  <section>
    <h3 class="visuallyhidden">Social Links</h3>
    <a class="social-link" href="https://twitter.com/udacity">
      
    </a>
    <a class="social-link" href="https://www.instagram.com/udacity/">
      
    </a>    <a class="social-link" href="https://plus.google.com/+Udacity">
      
    </a>
  </section>
</footer>
```

커밋!

```
$ git commit -m "Add links to social media"
```

5. Tagging, Branching, and Merging – Branching Effectively

See All Branches At Once

```
$ git log --oneline --decorate --graph --all
```

```
* e014d91 (HEAD -> footer) Add links to social media
* 209752a (master) Improve site heading for SEO
* 3772ab1 Set background color for page
| * f69811c (sidebar) Update sidebar with favorite movie
| * e6c65a6 Add new sidebar content
|/
* 5bfe5e7 Add starting HTML structure
* 6fa5f34 Add .gitignore file
* a879849 Add header to blog
* 94de470 Initial commit
(END)
```

5. Tagging, Branching, and Merging – Branching Effectively

The Merge Command

```
$ git merge footer
```

Fast-forward Merge

```
$ git merge sidebar
```

* merge message “Merge branch sidebar”

5. Tagging, Branching, and Merging – Branching Effectively

Perform A Regular Merge

```
$ git merge sidebar
```

6. Undoing Changes - Modifying The Last Commit

Changing The Last Commit & Add Forgotten Files To Commit

```
$ git commit --amend
```

6. Undoing Changes - Reverting A Commit

The git revert Command

```
$ git revert <SHA-of-commit-to-revert>
```

6. Undoing Changes - Reverting A Commit

Resetting Commits

```
$ git reset<reference-to-commit>
```

- **the parent commit** – the following indicate the parent commit of the current commit
 - HEAD^
 - HEAD~
 - HEAD~1
- **the grandparent commit** – the following indicate the grandparent commit of the current commit
 - HEAD^^
 - HEAD~2
- **the great-grandparent commit** – the following indicate the great-grandparent commit of the current commit
 - HEAD^^^
 - HEAD~3