Version Control with Git

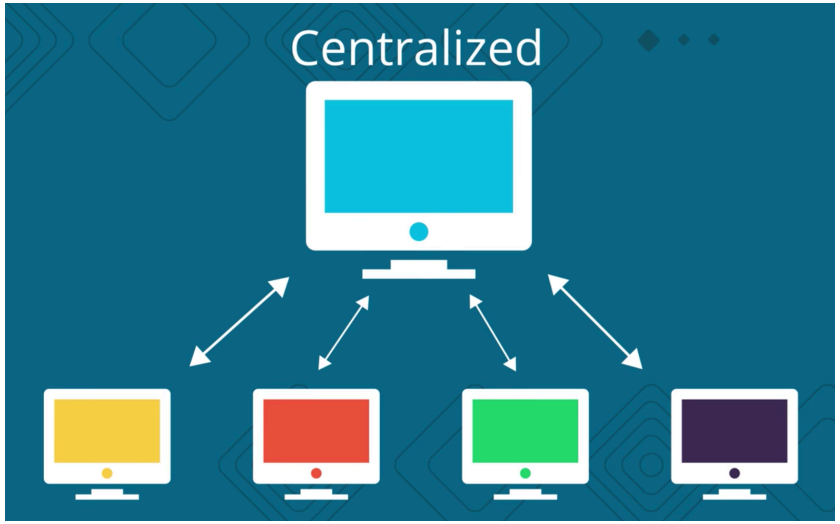
Local Laboratory

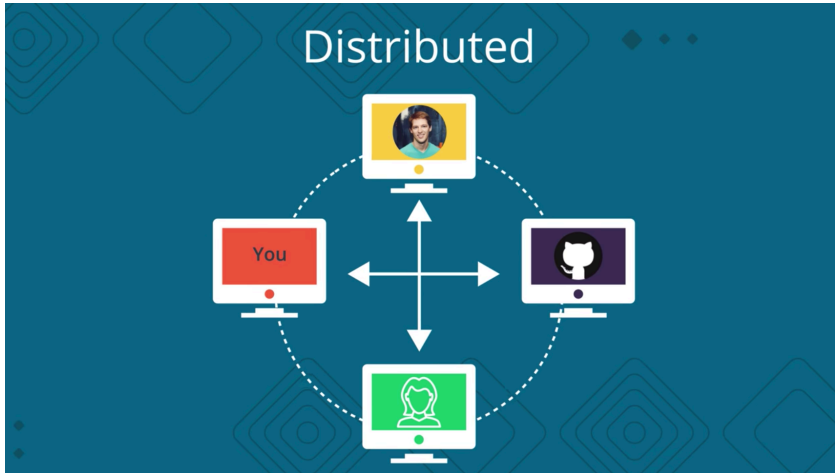
What is Version Control

Git

Subversion

Mercurial





What is Version Control

Version control tool

Service that hosts
Git projects

Git and Version Control Terminology

Version Contorl System / Source Code Manager

Commit

Repository / repo

Working Directory

Checkout

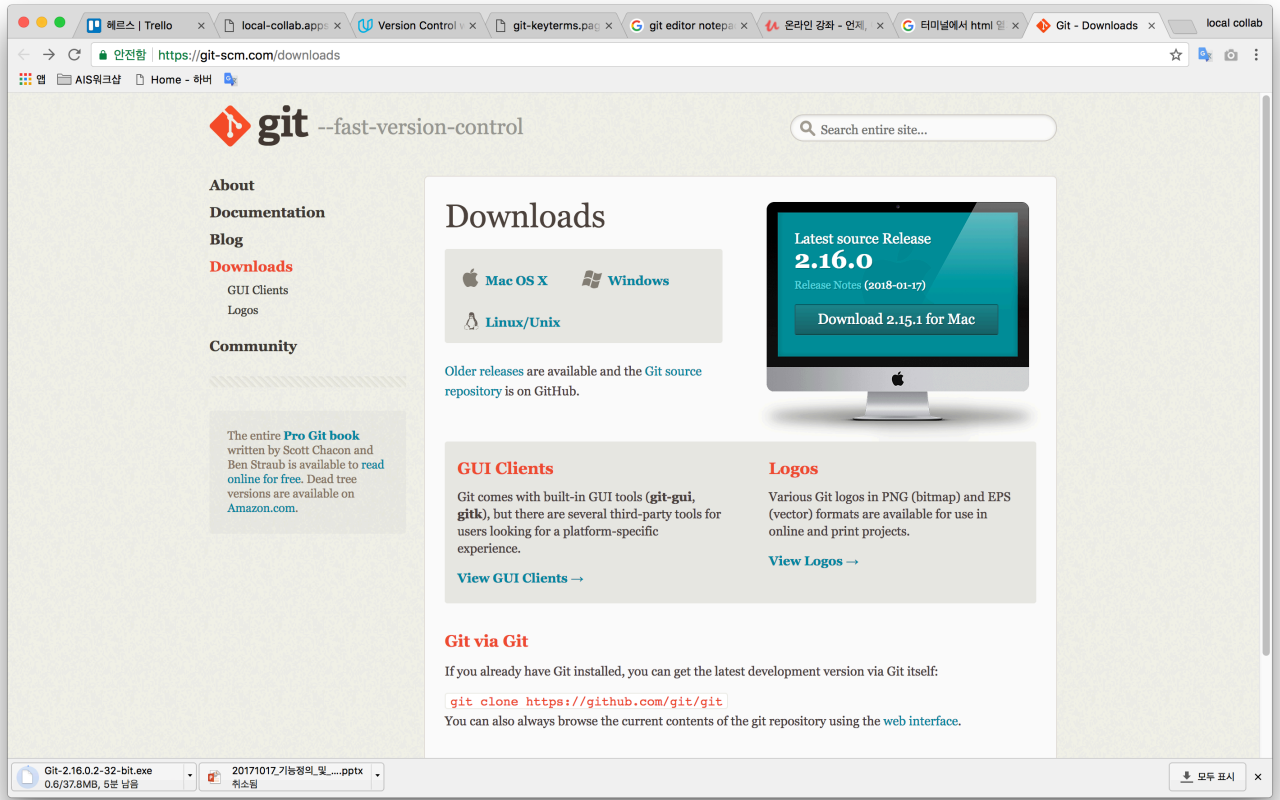
Staging Area

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

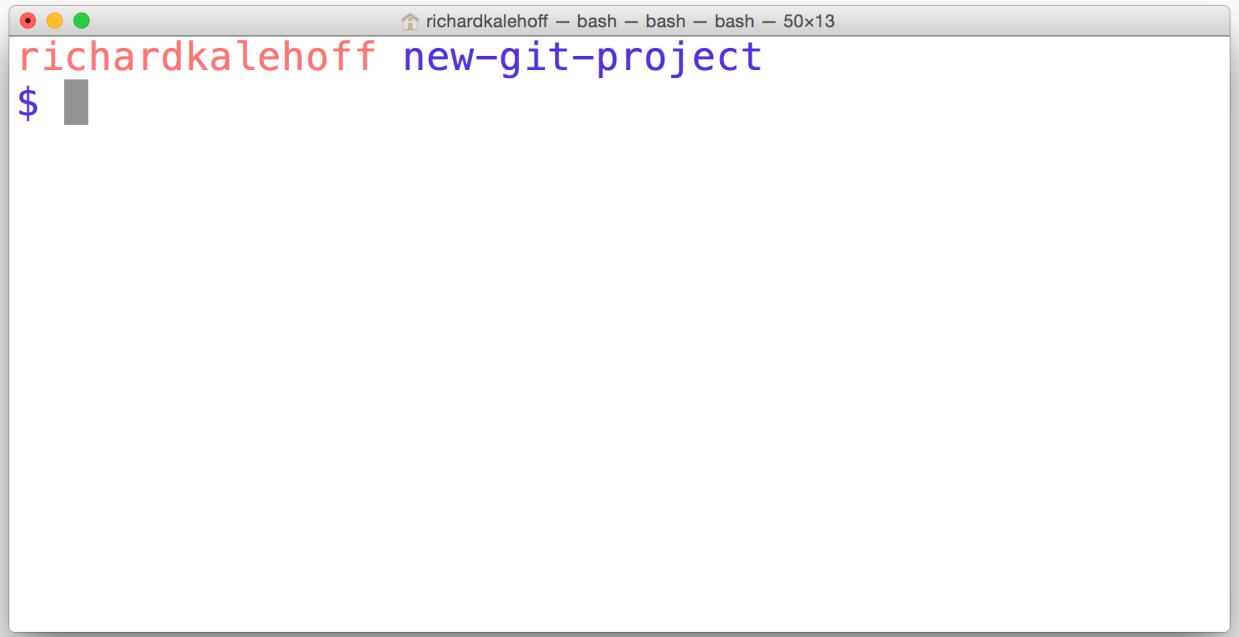
2. Create A Git Repo – Required Commands

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

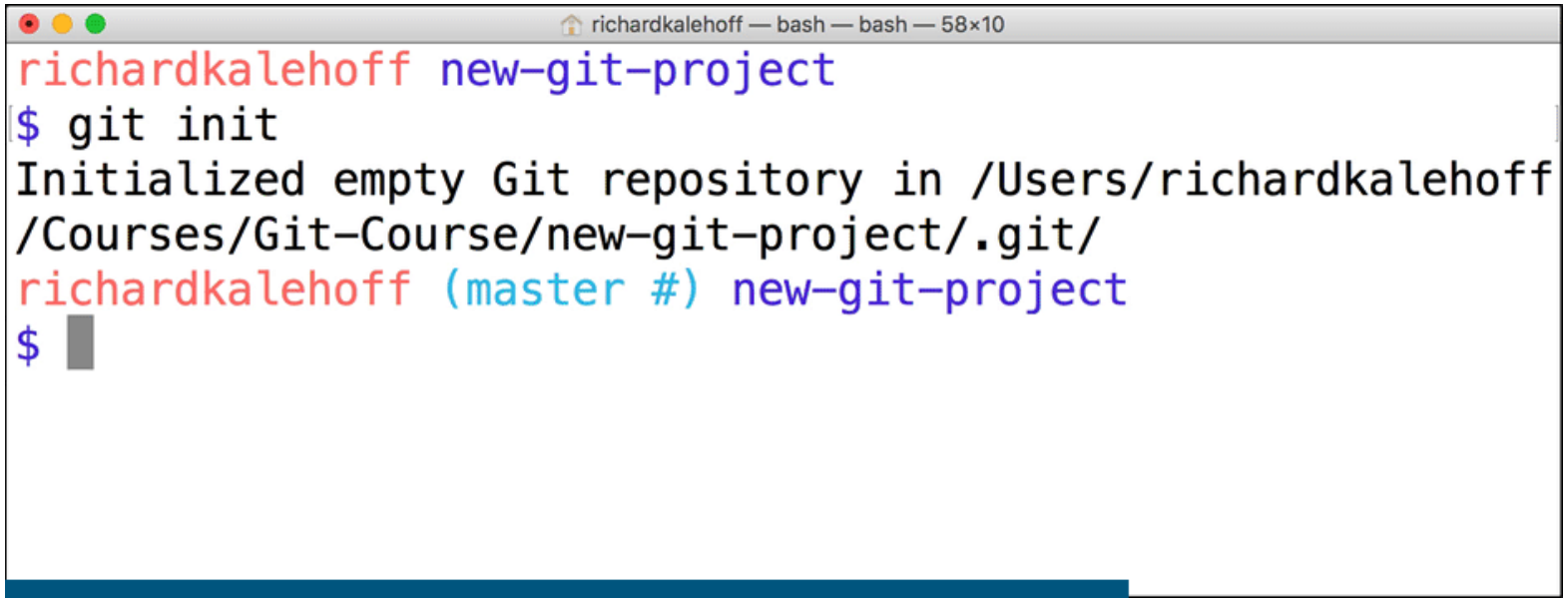
2. Create A Git Repo – Create Course Directory

- 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

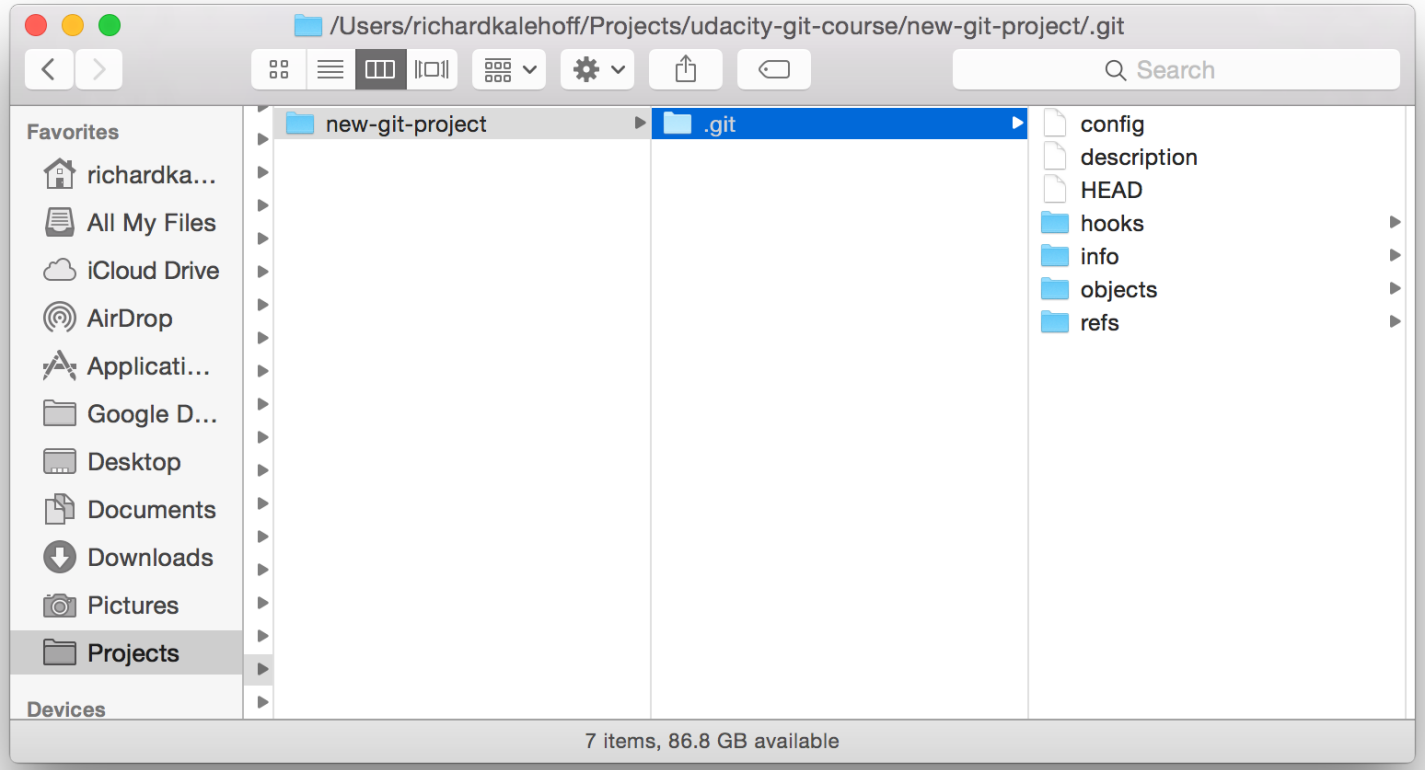


2. Create A Git Repo – Git Init

$ git init



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

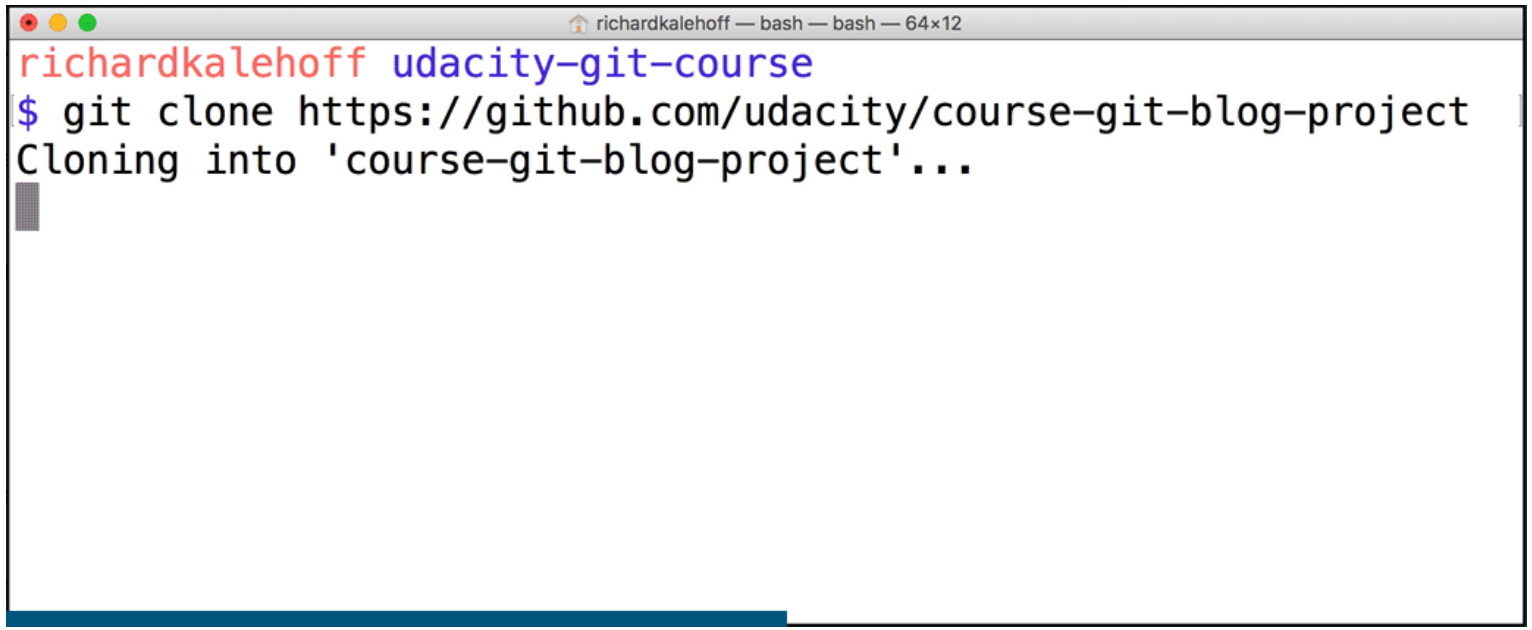


2. Create A Git Repo – Clone An Existing Repo

What is cloning?
to make an identical copy

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

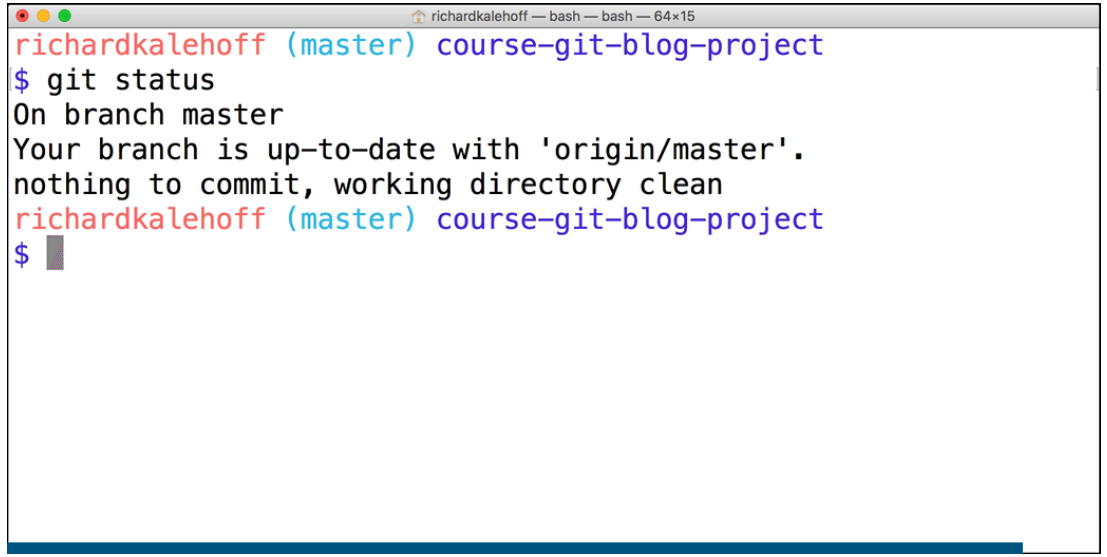
Cloning The Blog Repository

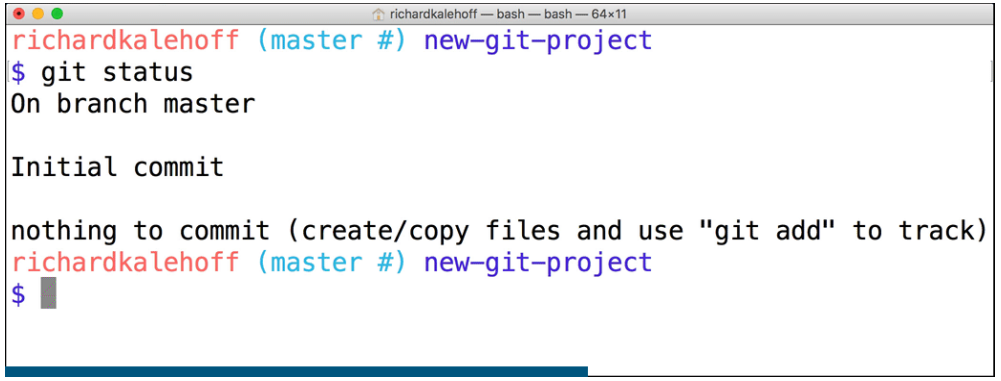


2. Create A Git Repo – Determine A Repo's Status

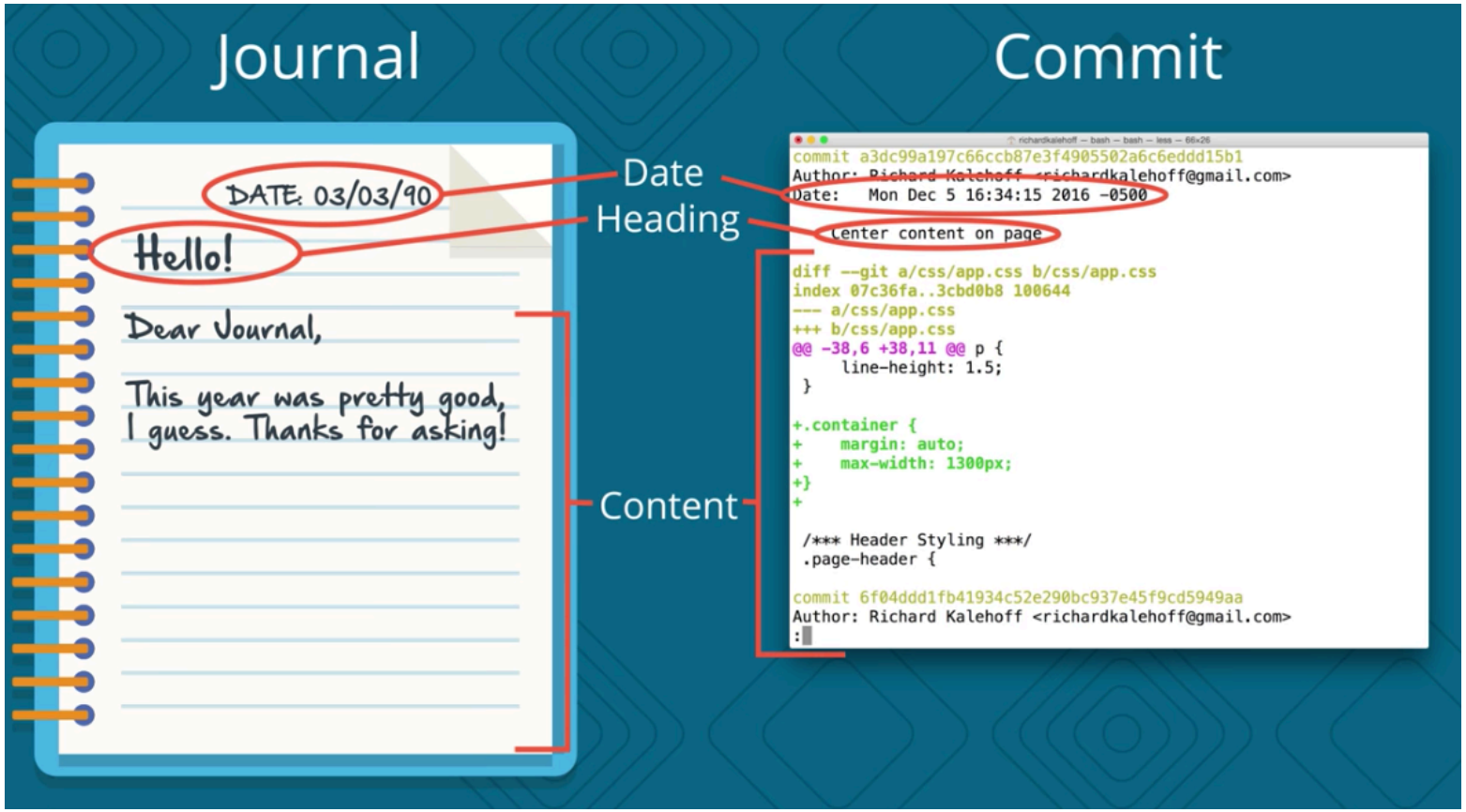
$ git status

Git Status Output

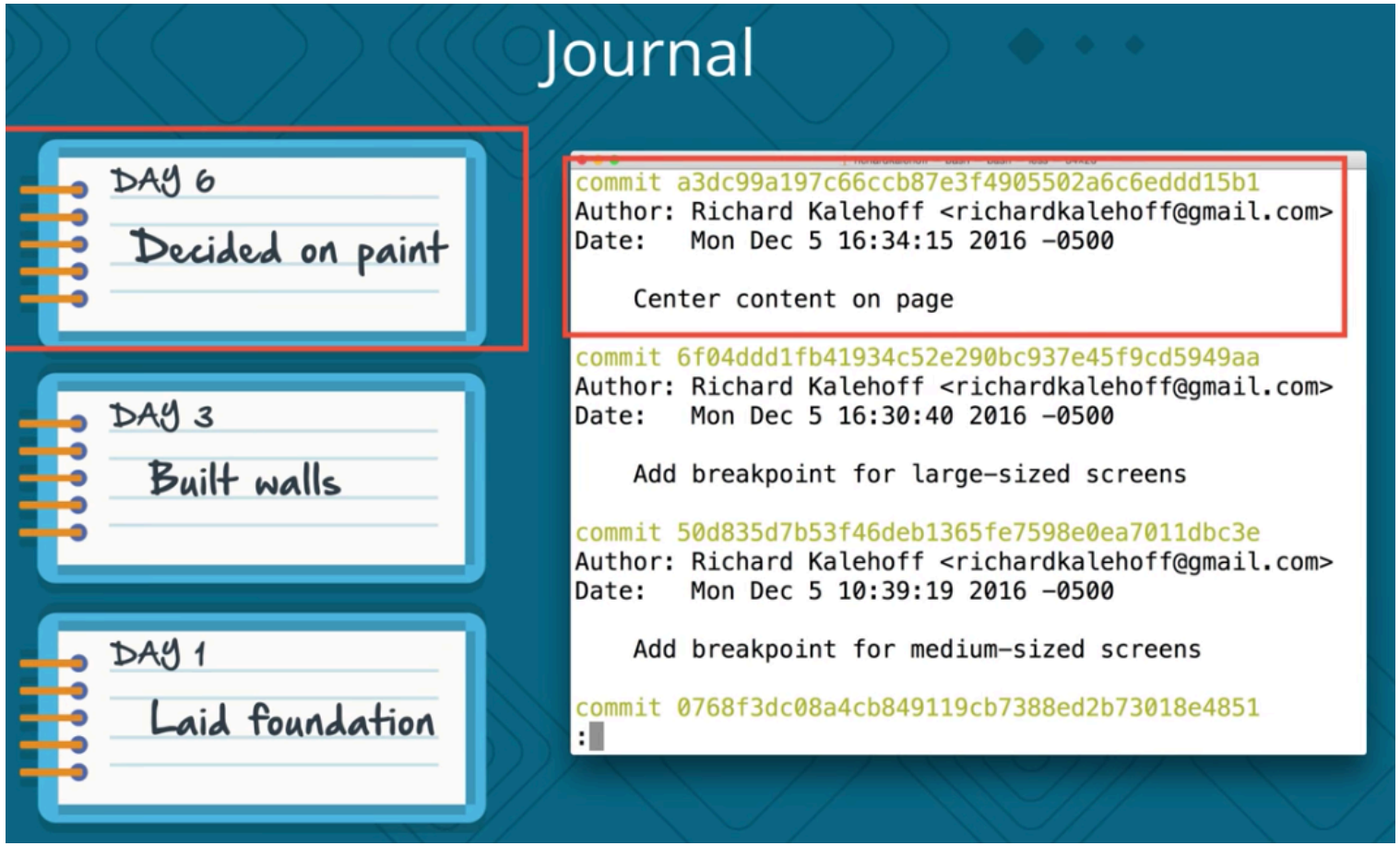




3. Review a Repo's History – Intro



3. Review a Repo's History – Init



3. Review a Repo's History – Init

git log

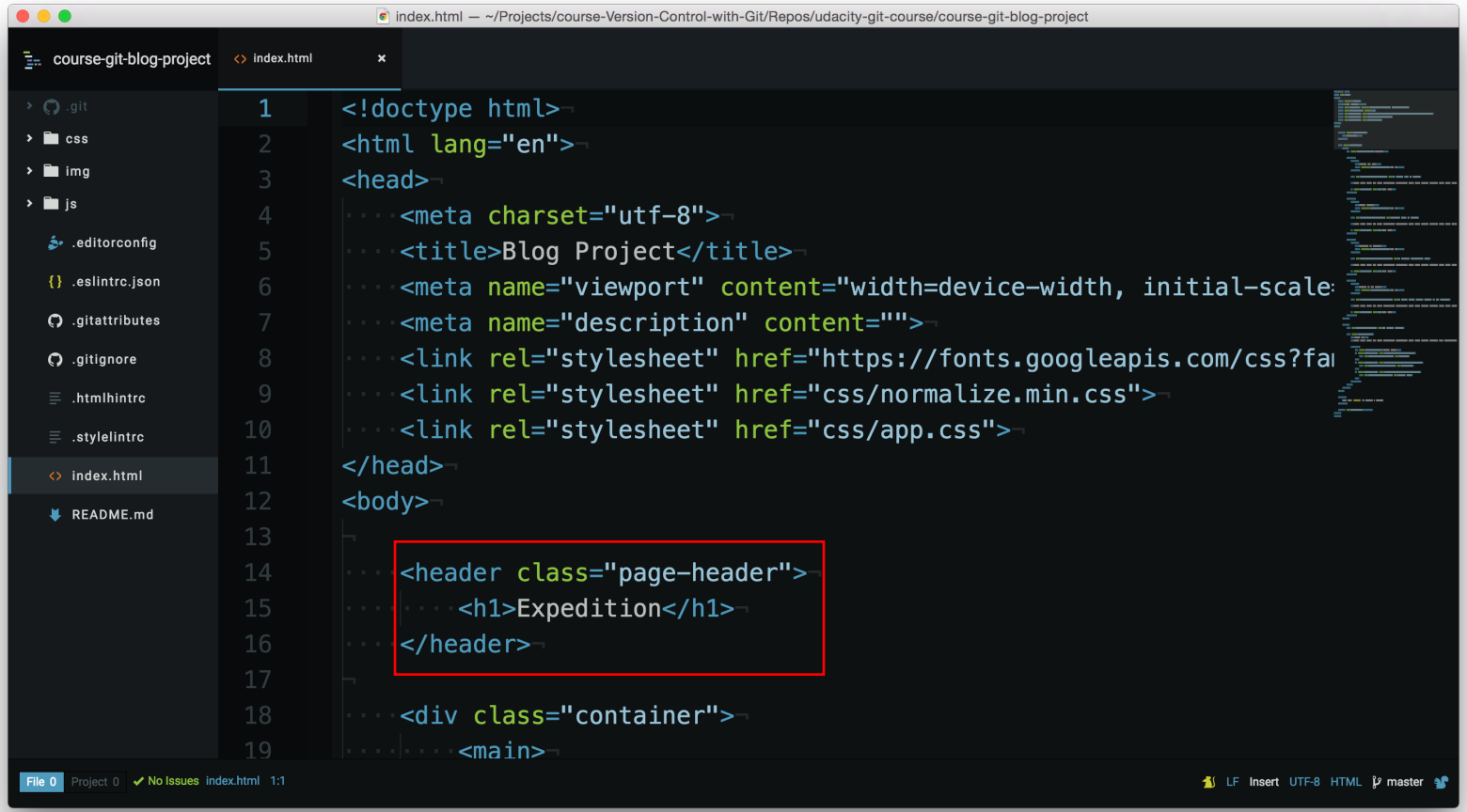
Displays information about the existing commits

git Show

Displays information about the given commit

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

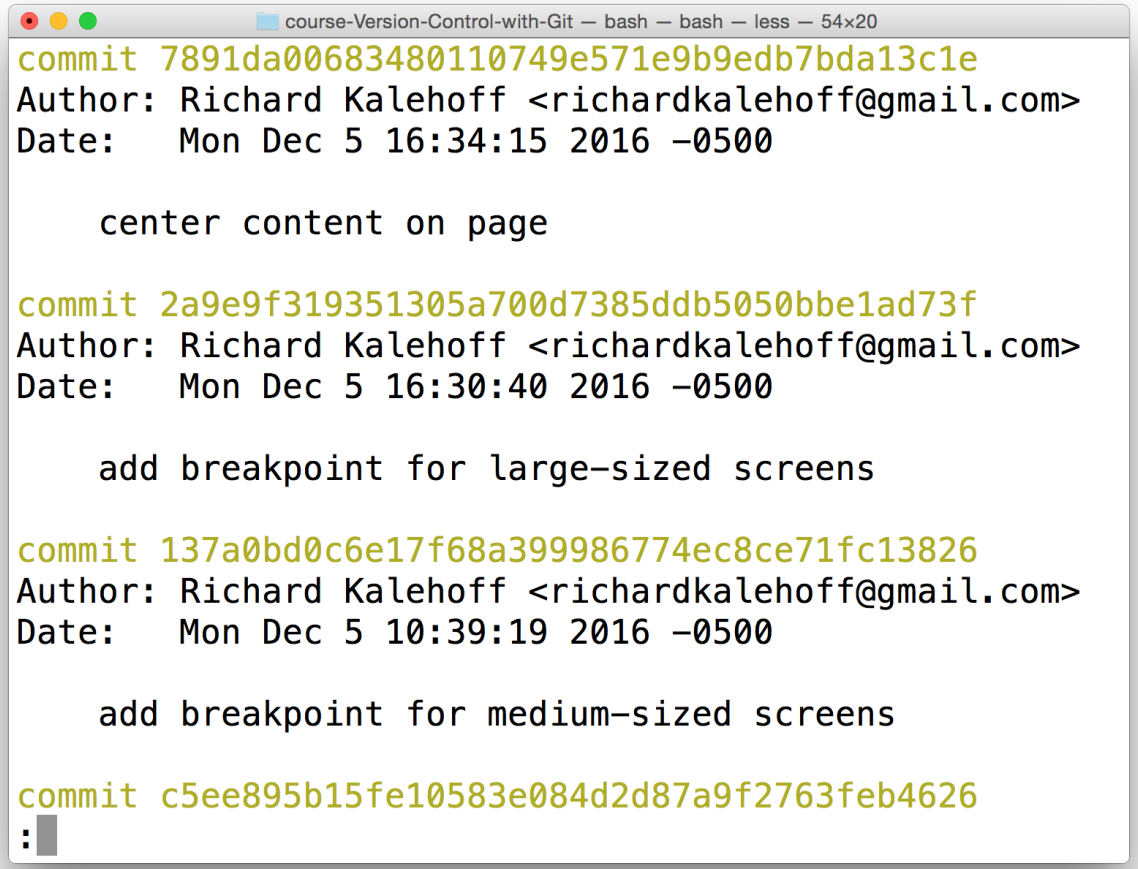
Opening The Project (course-git-blog-project)



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

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

$ git log



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)

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 — 54×20

```
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 — 62×20

```
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

```
course-Version-Control-with-Git — bash — bash — less — 54×20
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 — 62×20
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

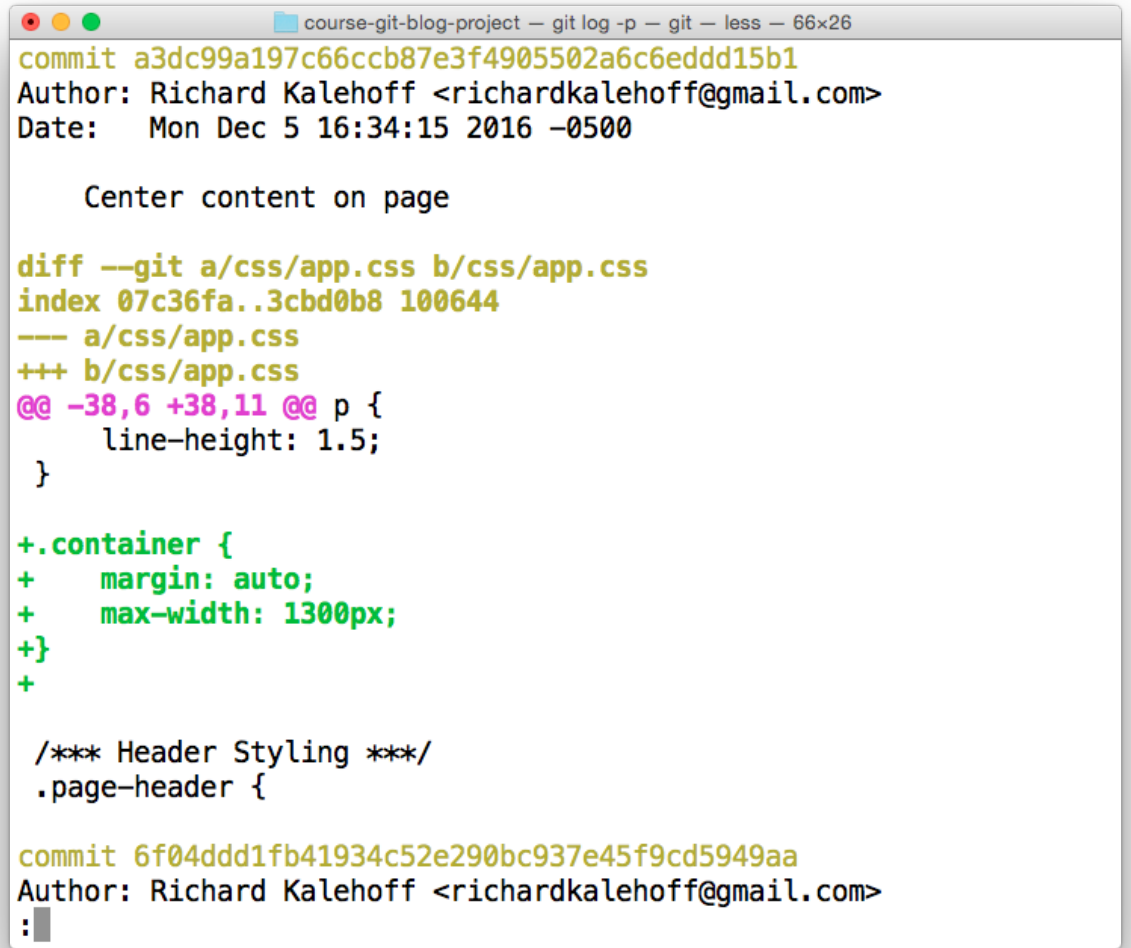
QUESTION

- 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)



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

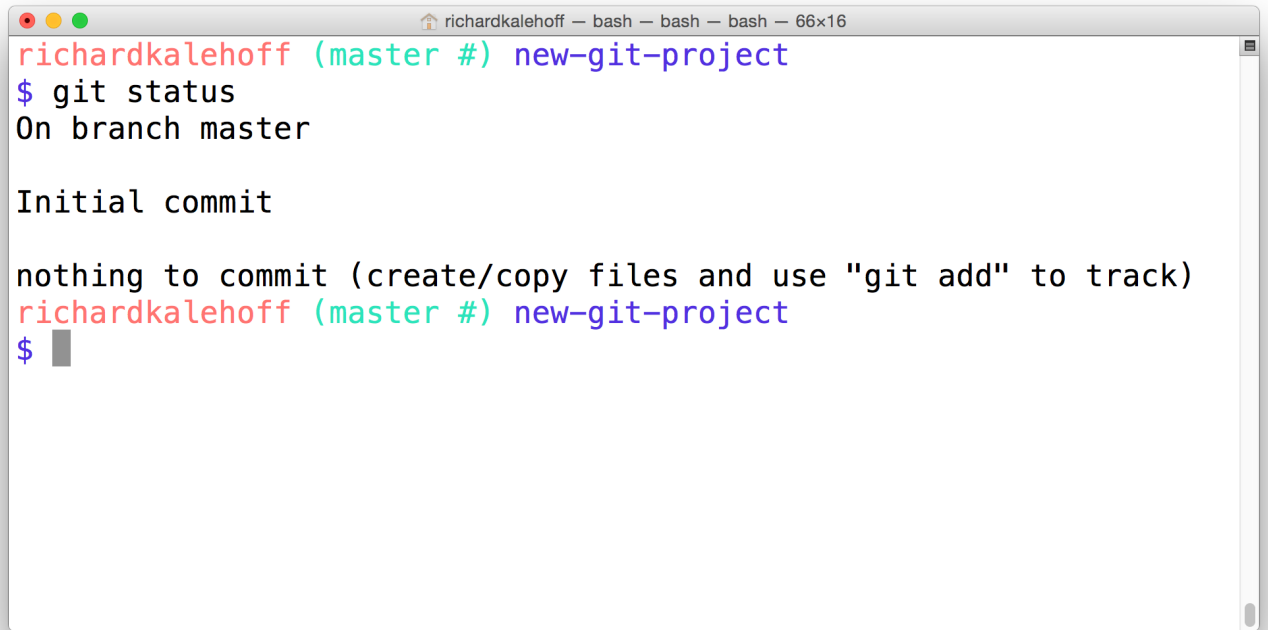
Take files from the staging index and save them in the repository

git diff

Displays the difference between two versions of a file

4. Add Commits To A Repo – Git Add

Status Status Status



4. Add Commits To A Repo – Git Add

Create An HTML File

index.html

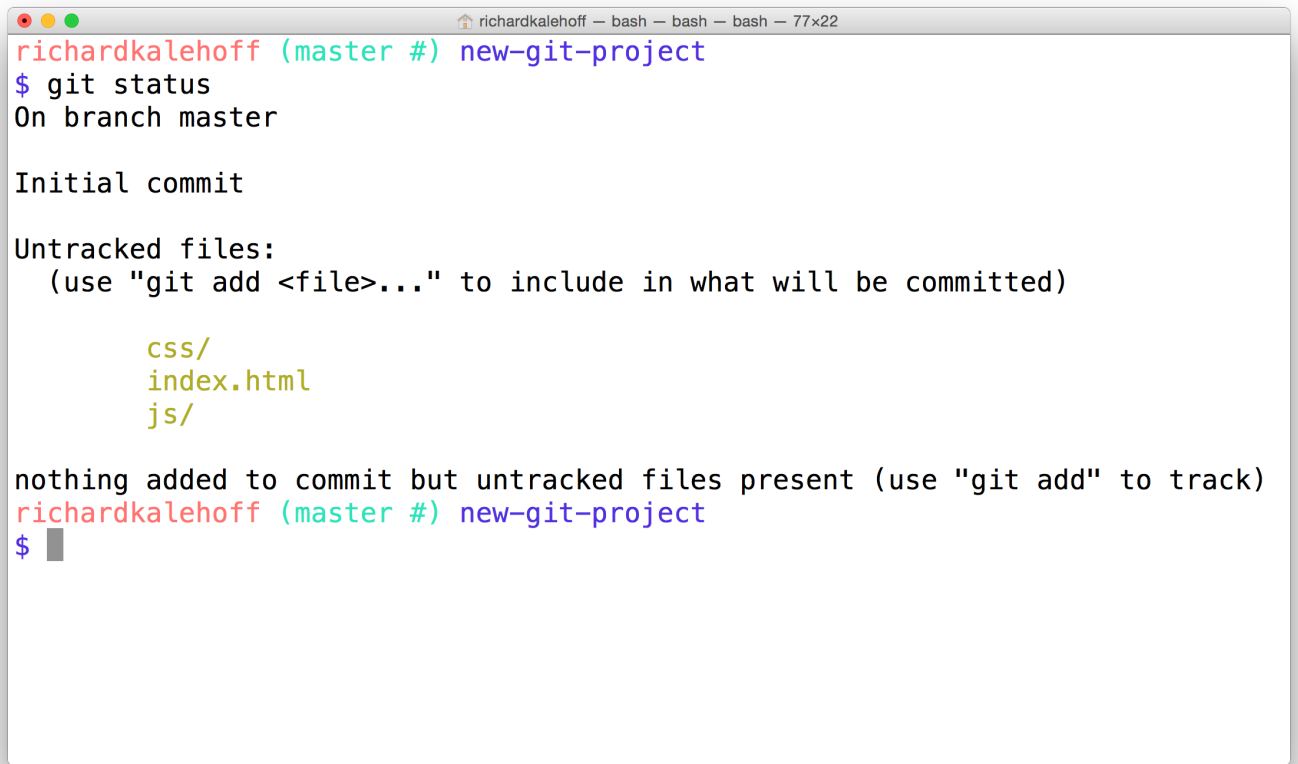
```
<!doctype html>
<html lang="en">
<head>
   <meta charset="utf-8">
   <title>Blog Project</title>
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <meta name="description" content="">
   <link rel="stylesheet" href="css/app.css">
</head>
<body>
   <script src="js/app.js">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

css/app.css
js/app.js

4. Add Commits To A Repo – Git Add

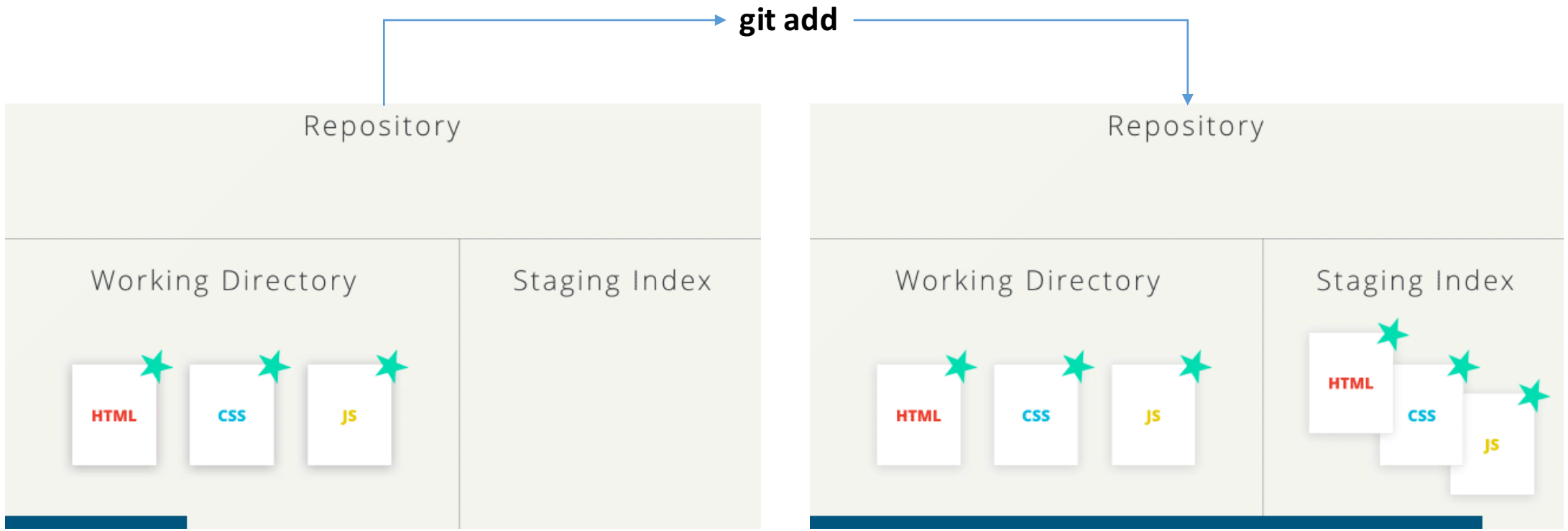
Quick Git Status Check

$ git status



4. Add Commits To A Repo – Git Add

Big Picture Review

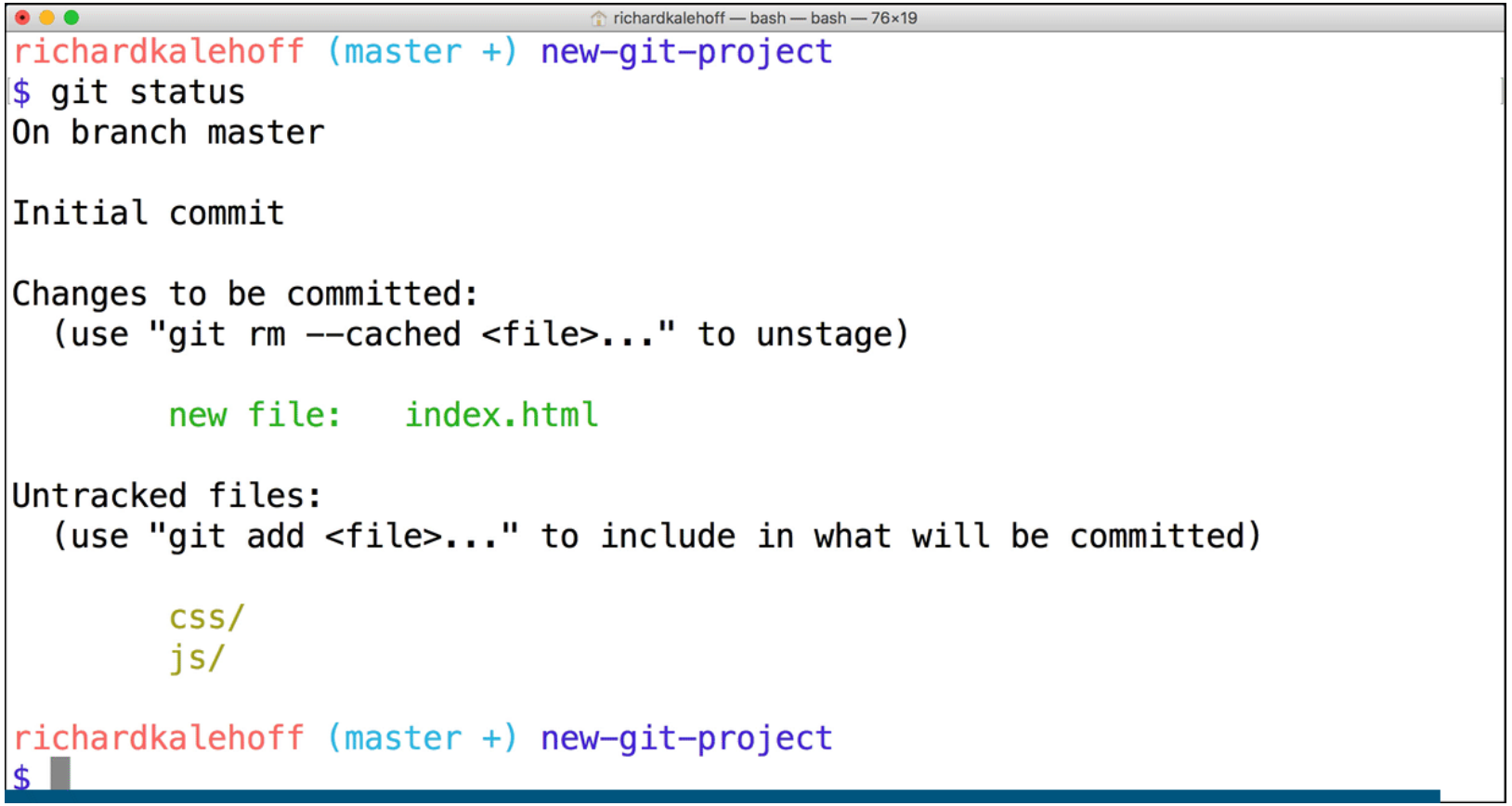


The untracked HTML, CSS, and JavaScript files add from the Working Directory to the Stating Index

4. Add Commits To A Repo – Git Add

Staging Files

$ git add index.html



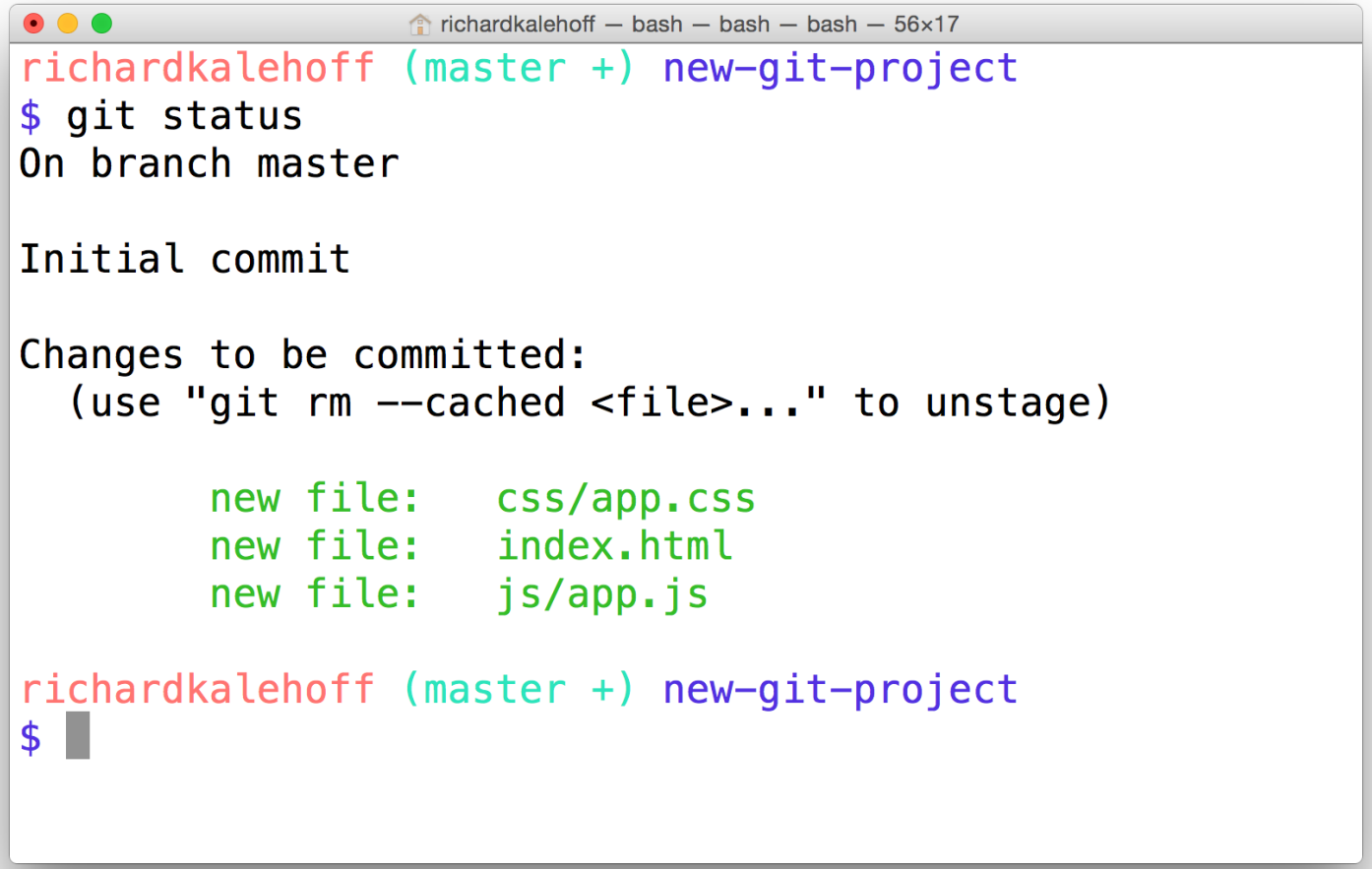
4. Add Commits To A Repo – Git Add

Staging Remaining Files

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

The Priod .

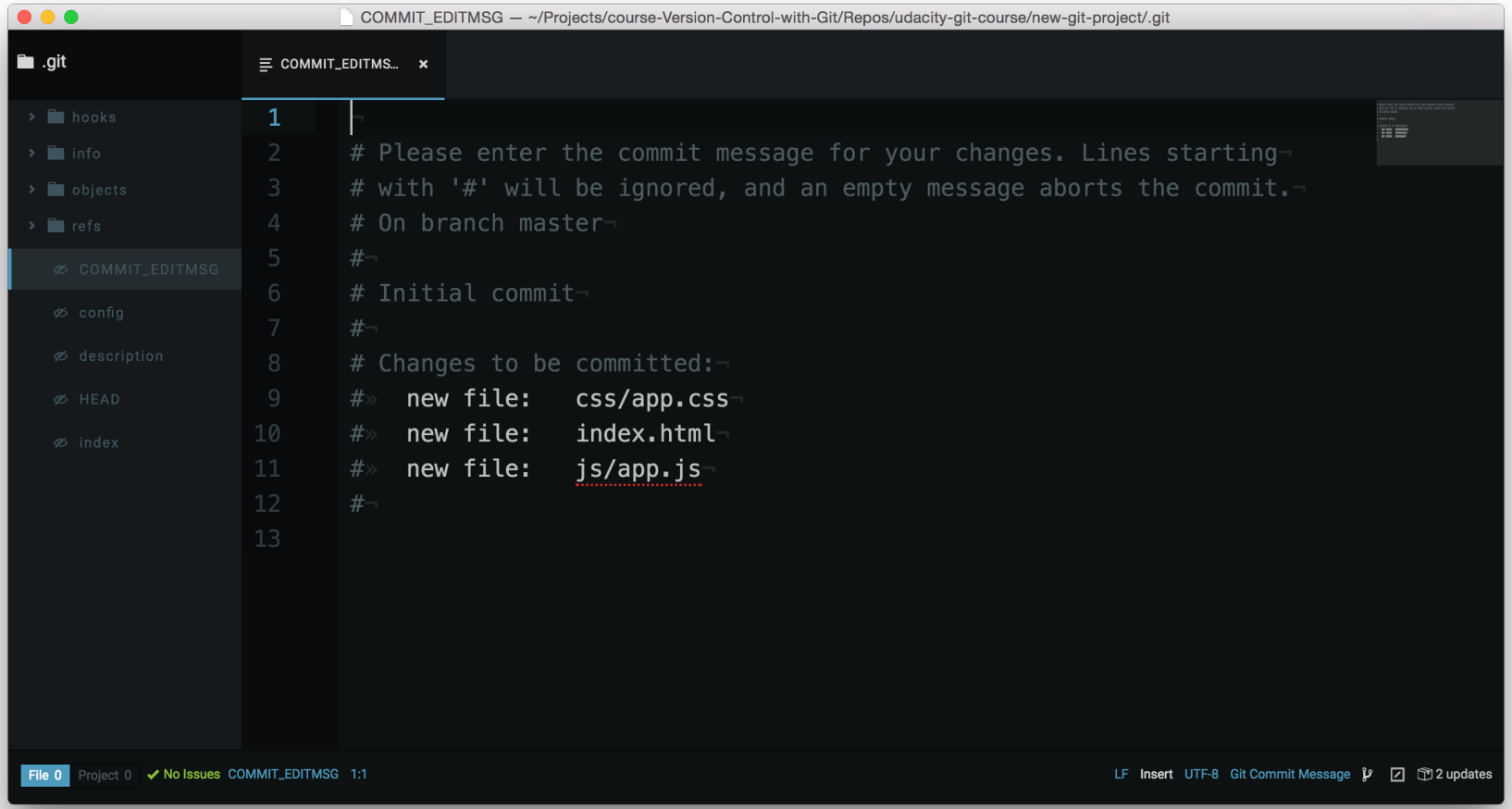
$ git add .



4. Add Commits To A Repo – Git Commit

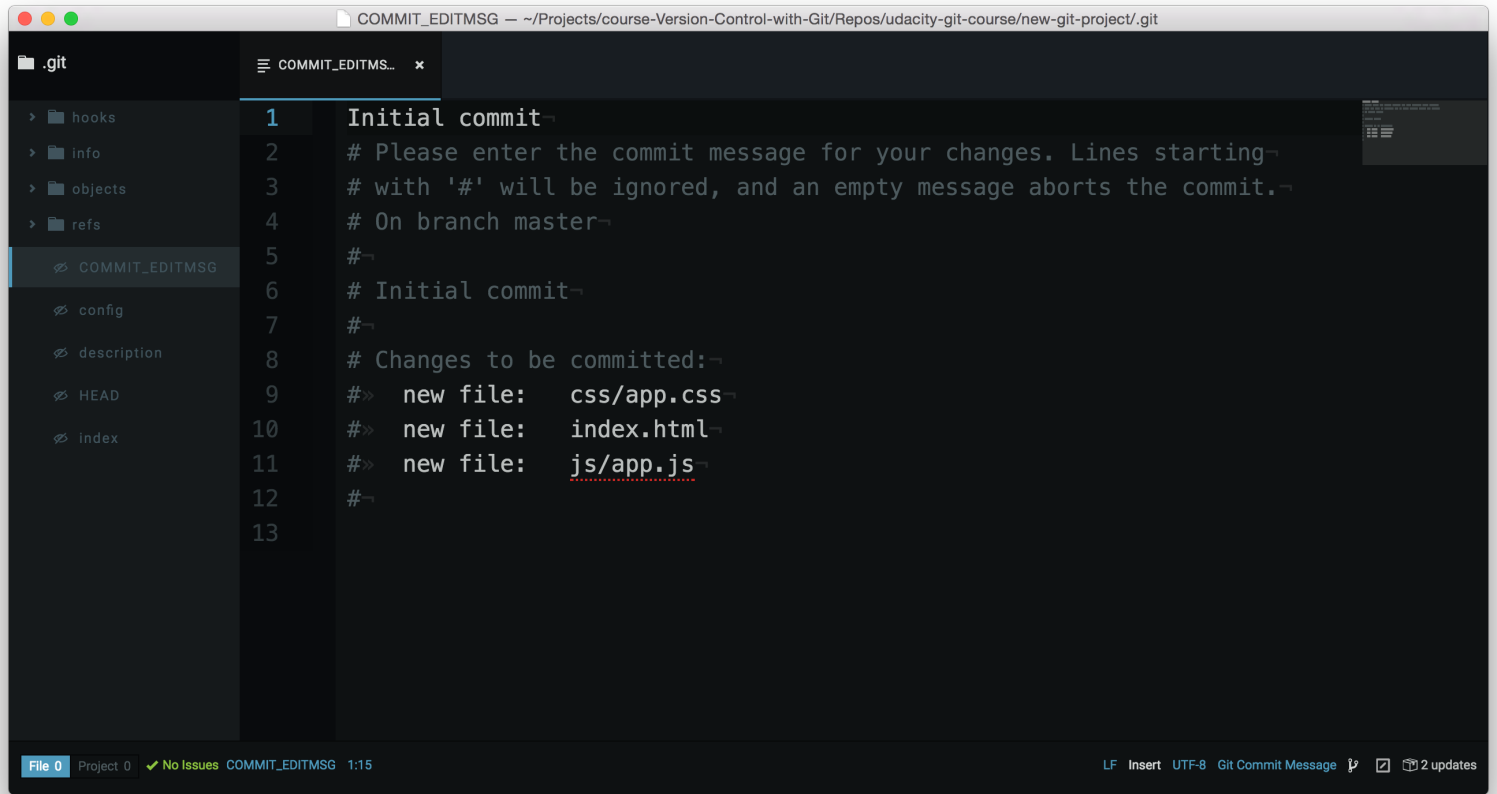
Make A Commit

$ git commit



4. Add Commits To A Repo – Git Commit

Code Editor Commit Message Explanation



Bypass The Editor With The -m Flag

$ git commit -m “Initial commit”

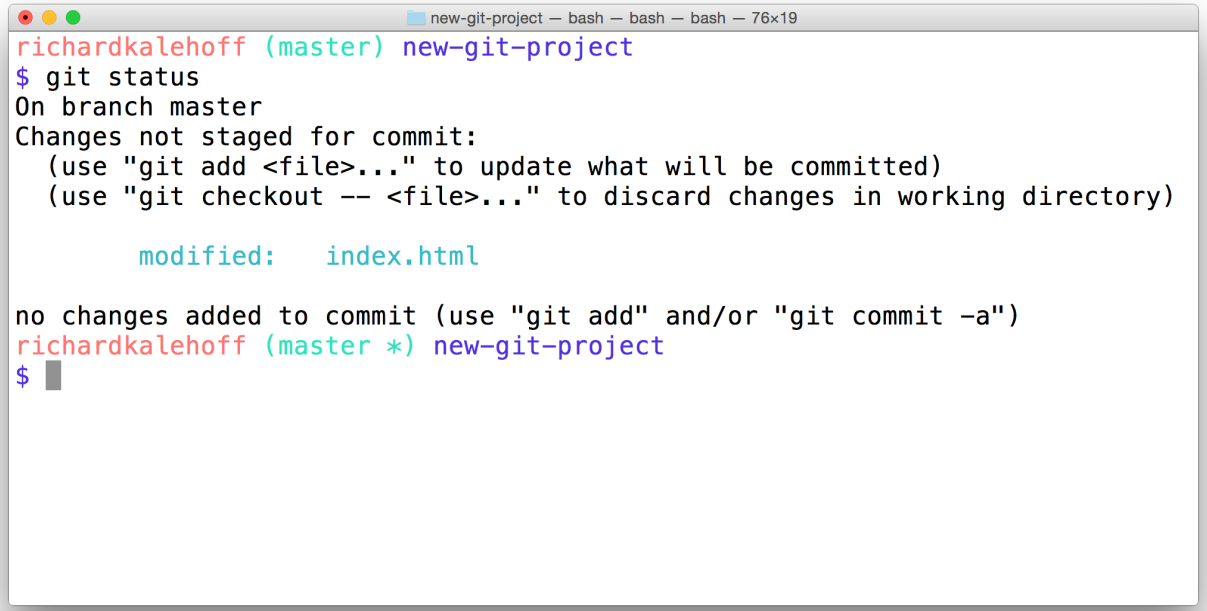
4. Add Commits To A Repo – Git Commit

2nd Commit – Add Changes

index.html의 body tag내에 아래 코드 추가

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

$ git status



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

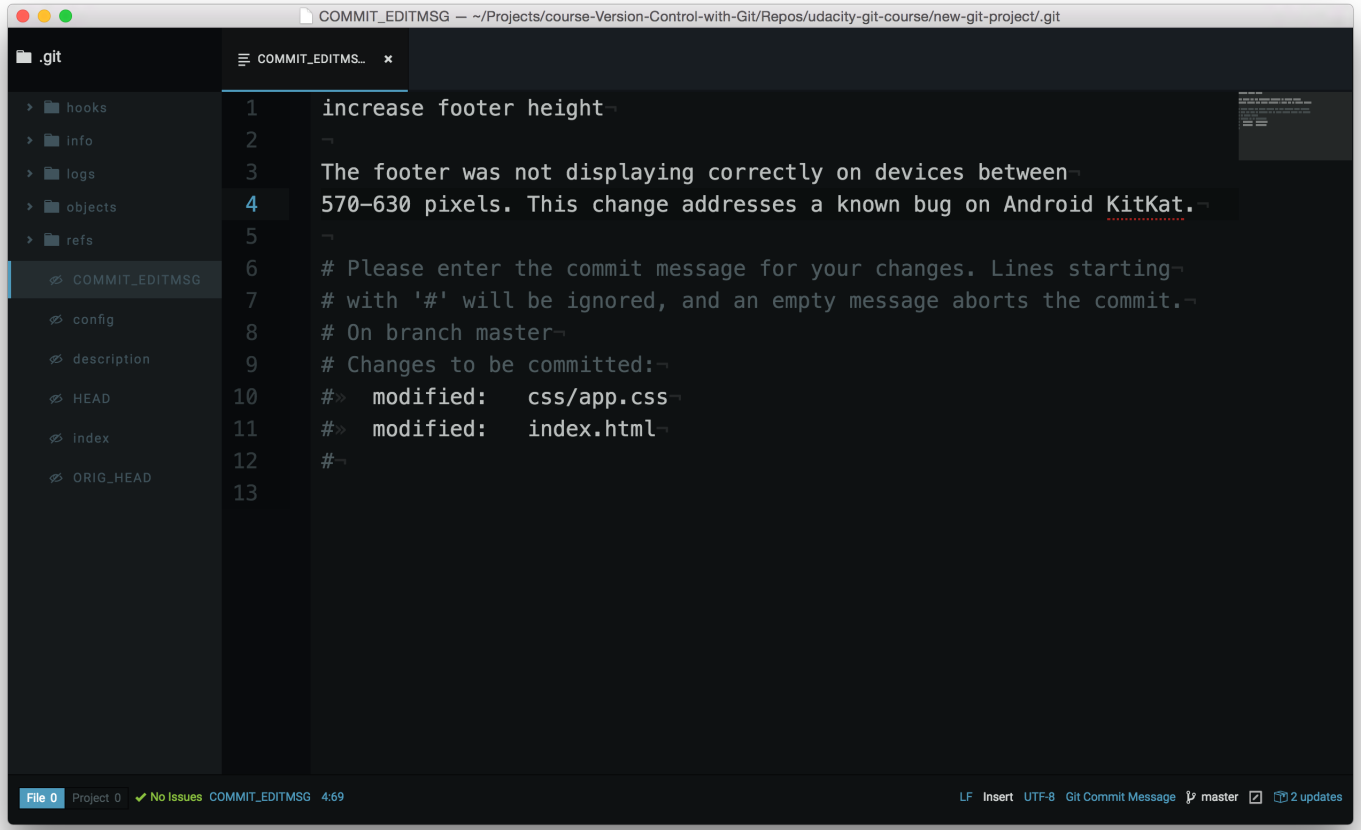
- do keep the **message short** (less than 60-ish characters)
- do **explain what the commit does** (not how or why!)

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*



$ git log
$ git log --oneline

8a11b3f

4. Add Commits To A Repo – Git Diff

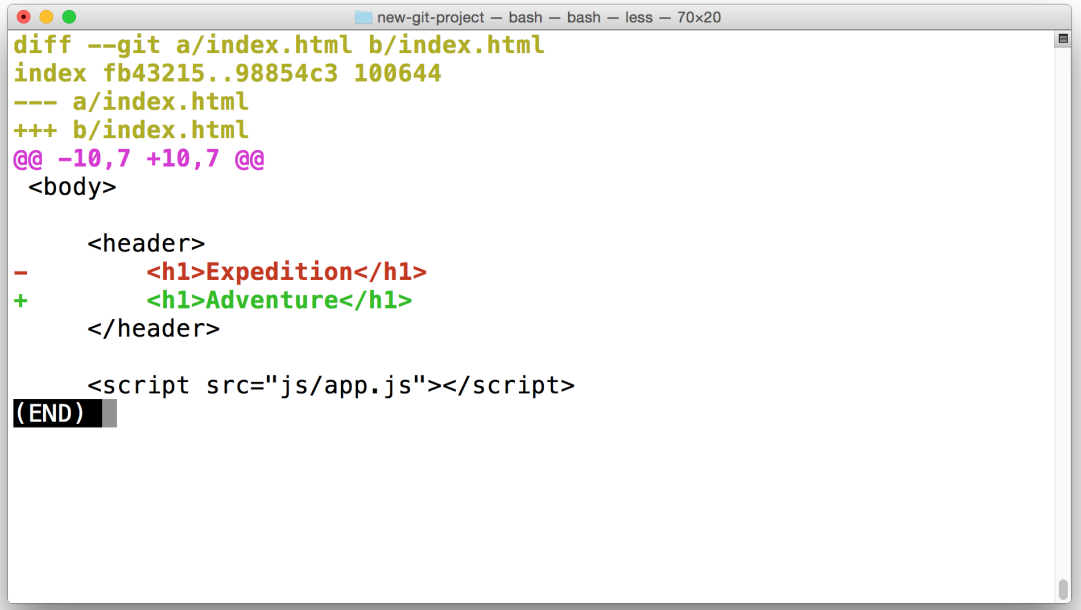
Why Do We Need This

git diff

index.html의 h1태그 내용 변경

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

$ git diff

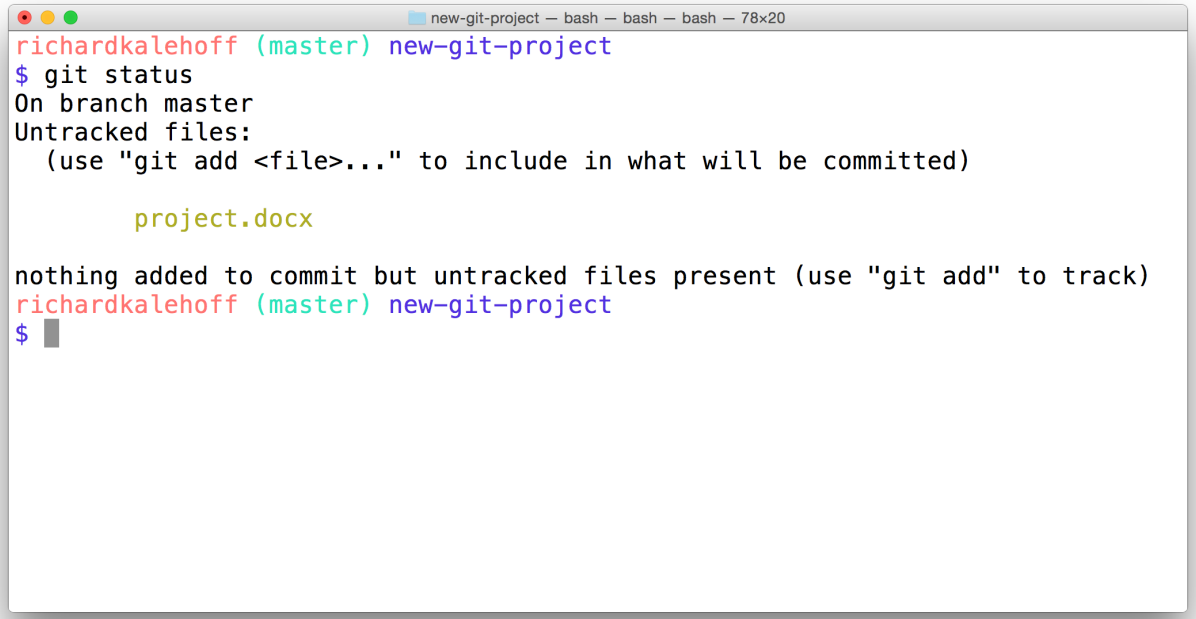


4. Add Commits To A Repo – Having Git Ignore Files

Why Shoud Files Be Ignored

The Problem

$ touch project.docx
$ git status



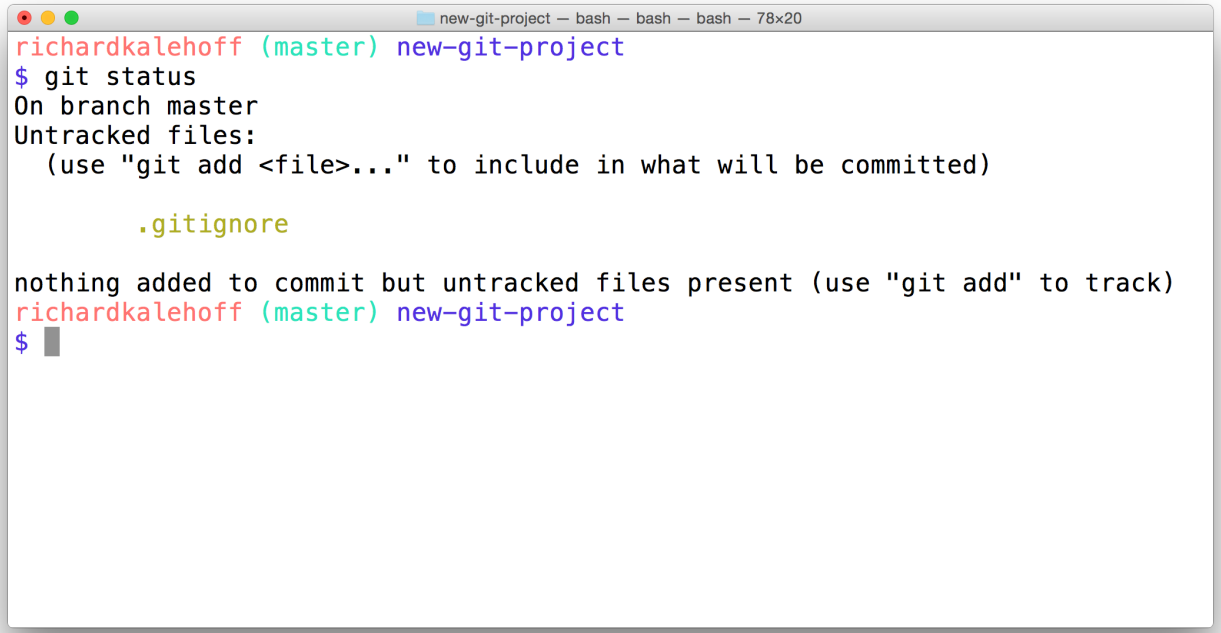
git add . 실행시 문제 발생

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



$ 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

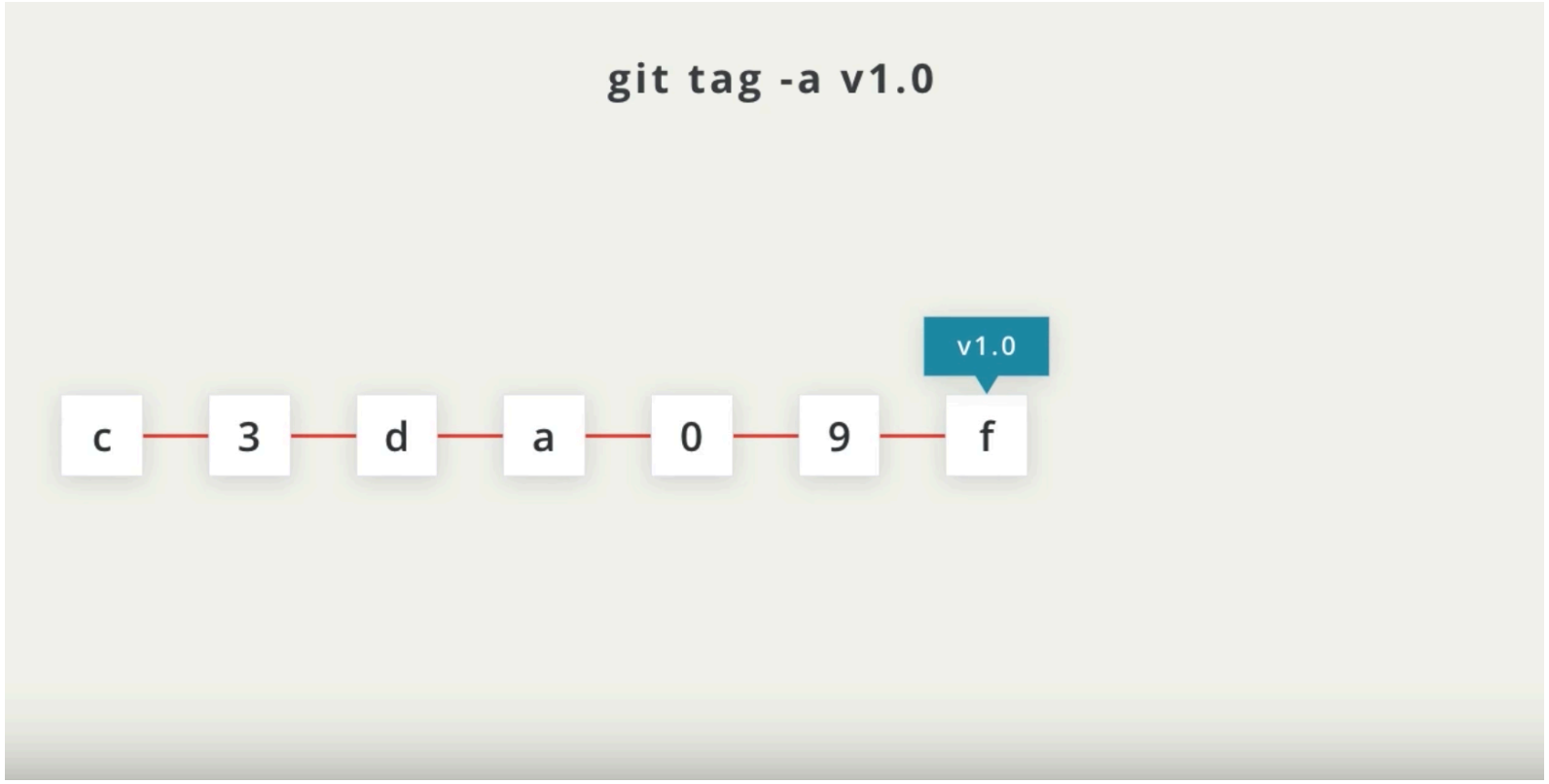
git tag
Add tags to specific commits

git branch
Allows multiple lines of development

git checkout
Switch between different branches and tags

git merging
Combines changes on different branches

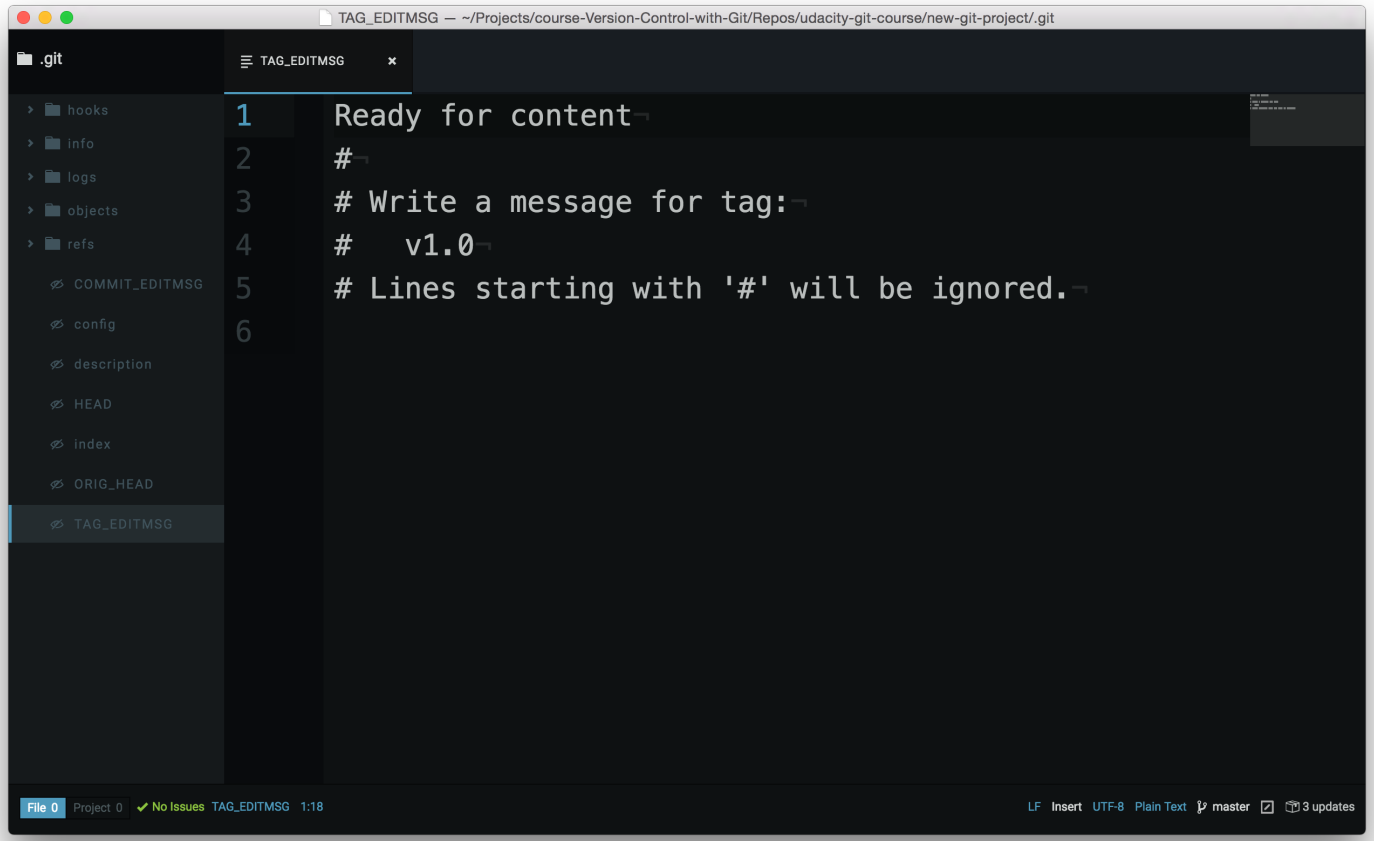
5. Tagging, Branching, and Merging – Tagging



5. Tagging, Branching, and Merging – Tagging

Git Tag Command

$ git tag -a v1.0



5. Tagging, Branching, and Merging – Tagging

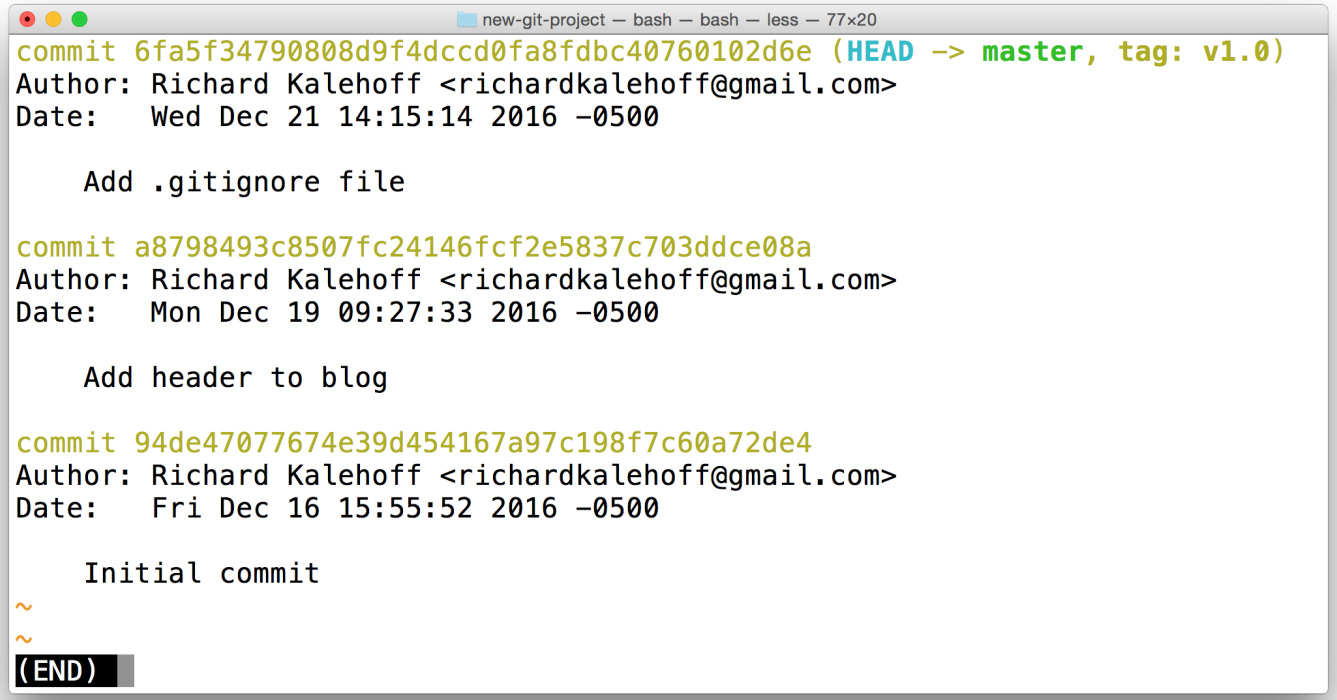
Verify Tag

$ git tag

5. Tagging, Branching, and Merging – Tagging

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

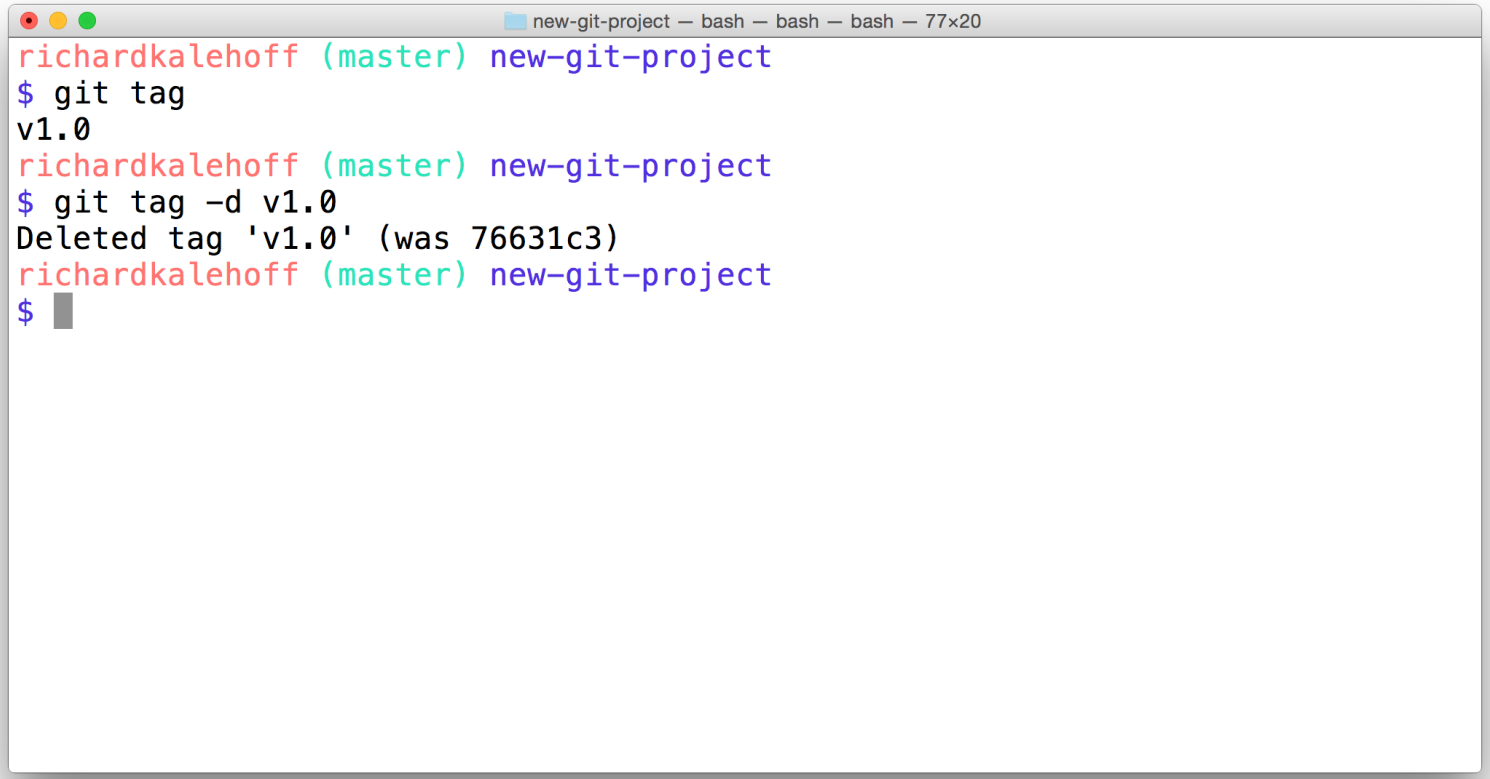
$ git log --decorate



5. Tagging, Branching, and Merging – Tagging

Deleting A Tag

$ git tag -d v1.0

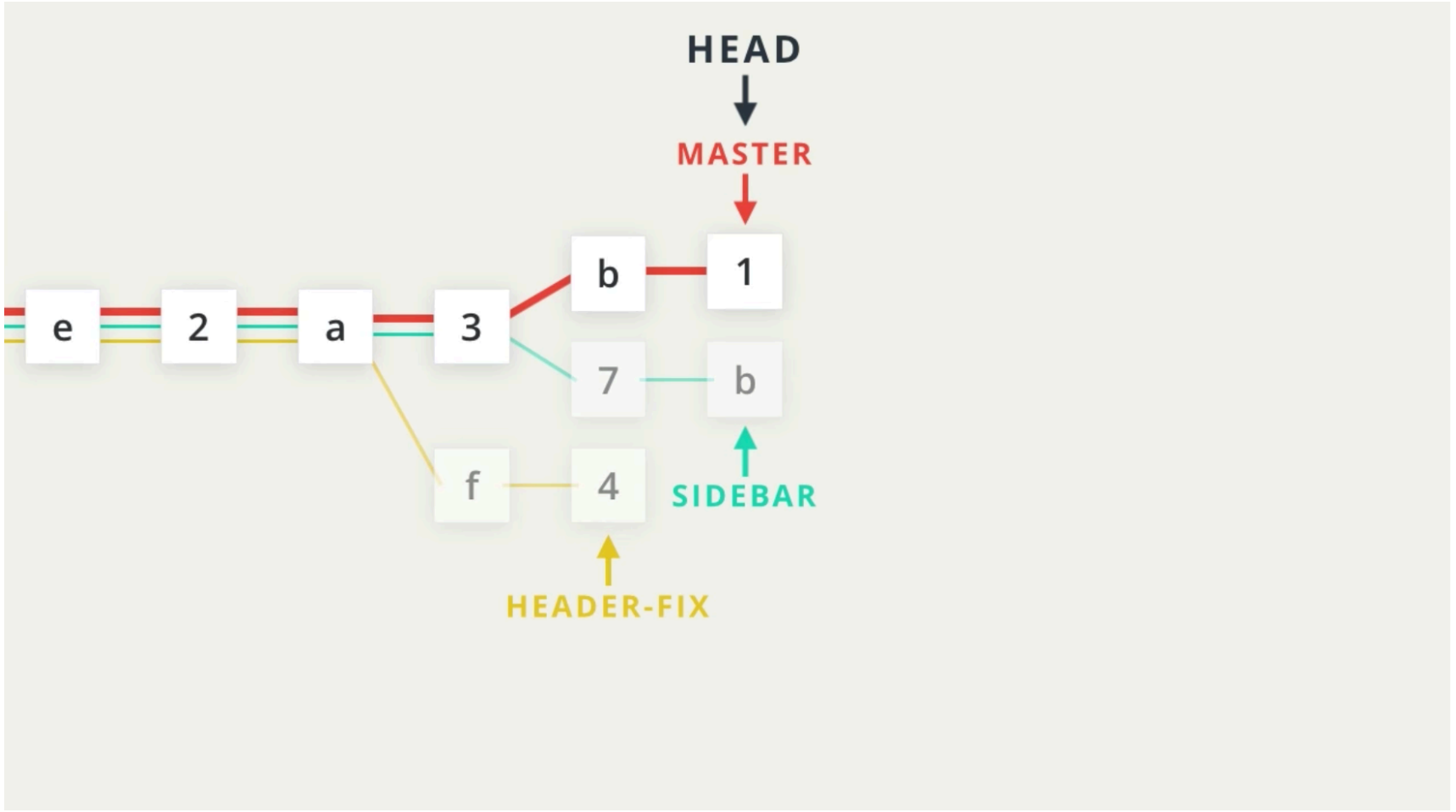


5. Tagging, Branching, and Merging – Tagging

Adding A Tag To A Past Commit

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

5. Tagging, Branching, and Merging – Branching

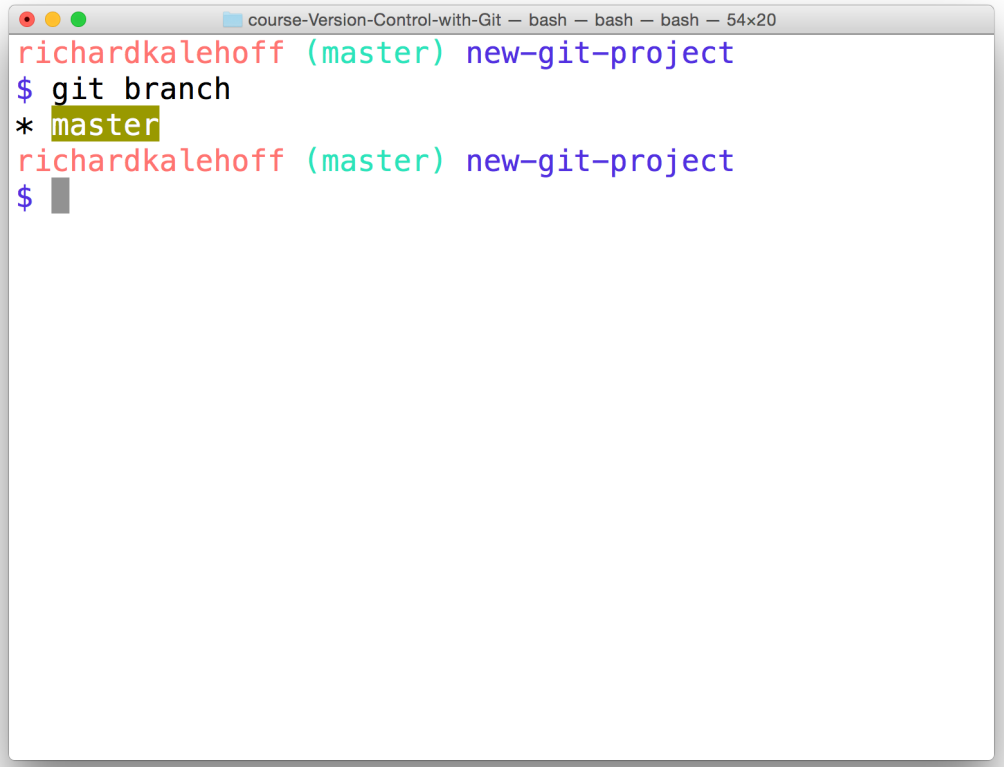


5. Tagging, Branching, and Merging – Branching

git branch command

$ git branch

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



5. Tagging, Branching, and Merging – Branching

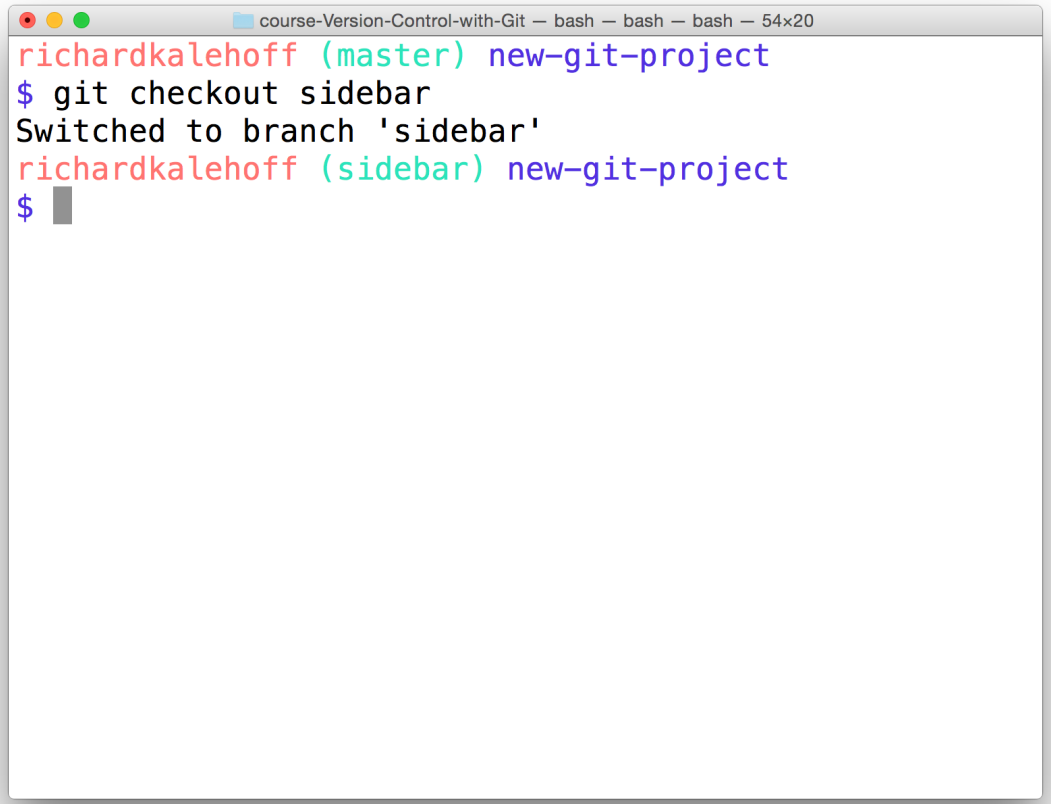
Create A Branch

$ git branch sidebar

5. Tagging, Branching, and Merging – Branching

The git checkout Command

$ git checkout sidebar



5. Tagging, Branching, and Merging – Branching

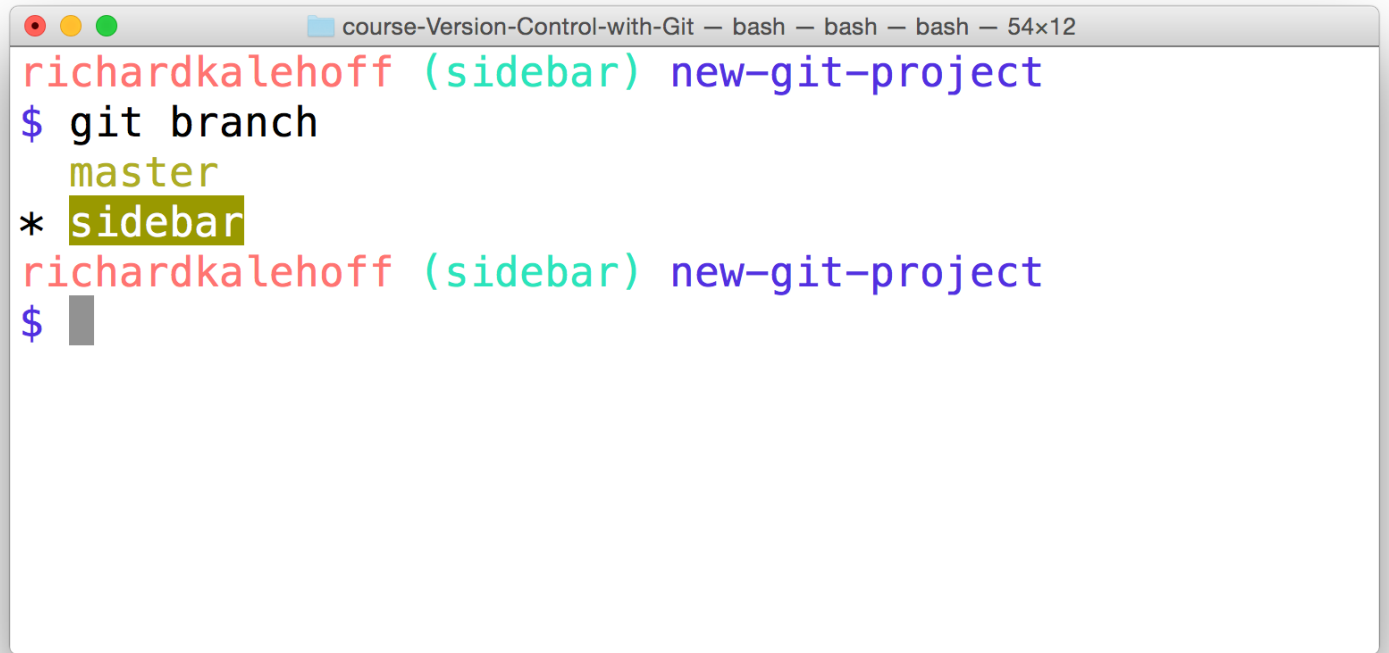
Branches In The Log

$ git log --oneline --decorate

5. Tagging, Branching, and Merging – Branching

The Active Branch

$ git branch



5. Tagging, Branching, and Merging – Branching

Specific Branch

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

5. Tagging, Branching, and Merging – Branching

Deletea A Branch

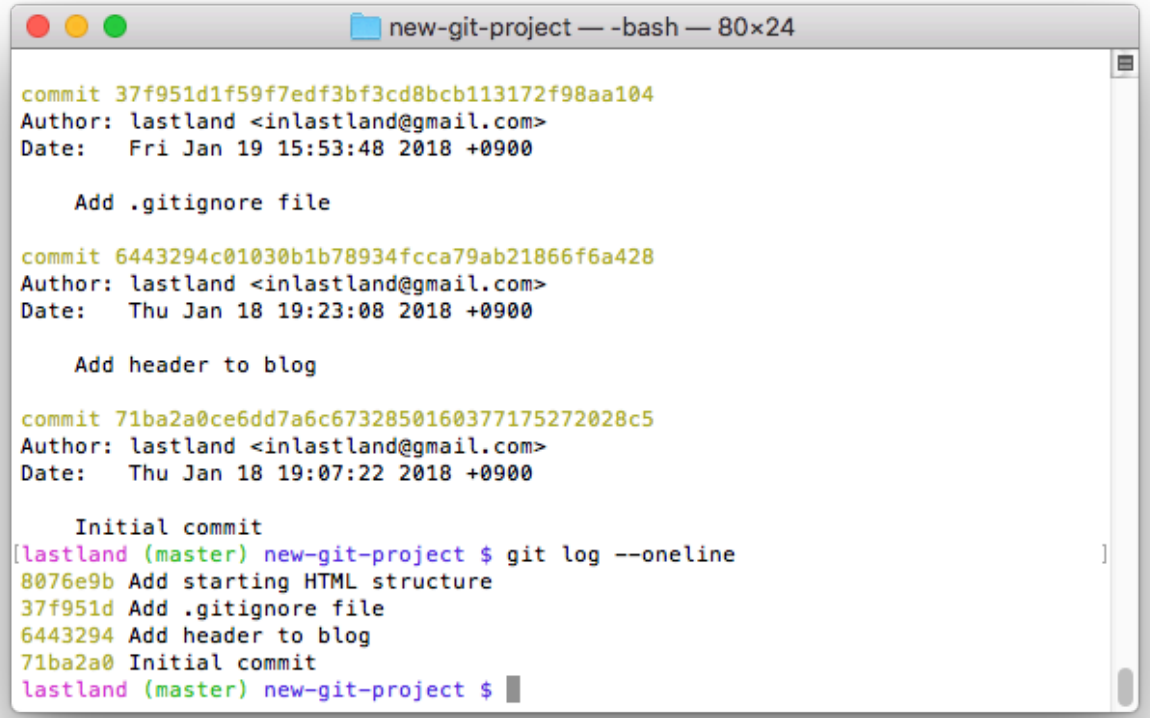
$ git branch –d sidebar

$ 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>
```



$ git commit –m “Add starting HTML structure”

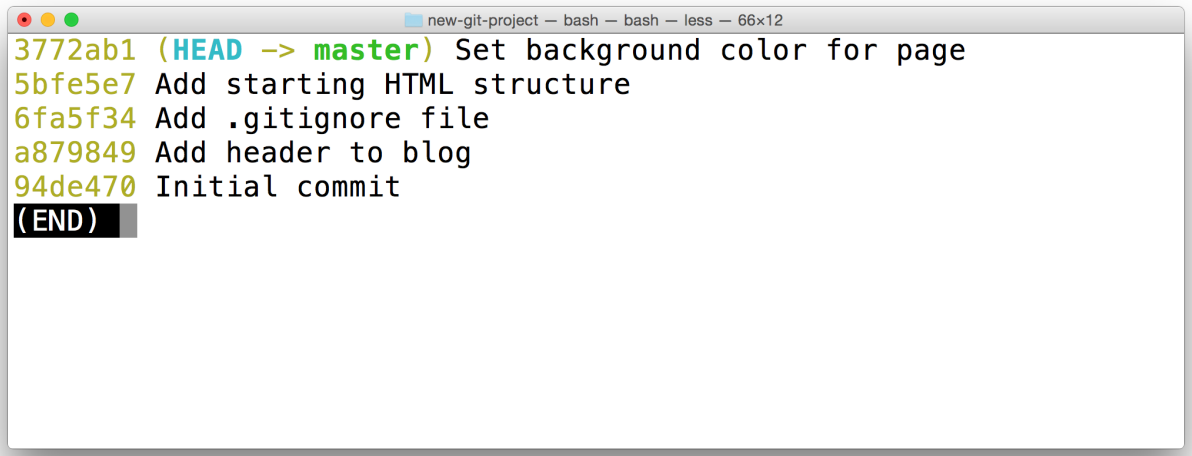
5. Tagging, Branching, and Merging – Branching Effectively

Change 1 - Add Page Color

css/app.css 수정 후 커밋

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

$ git commit –m “Set background color for page”



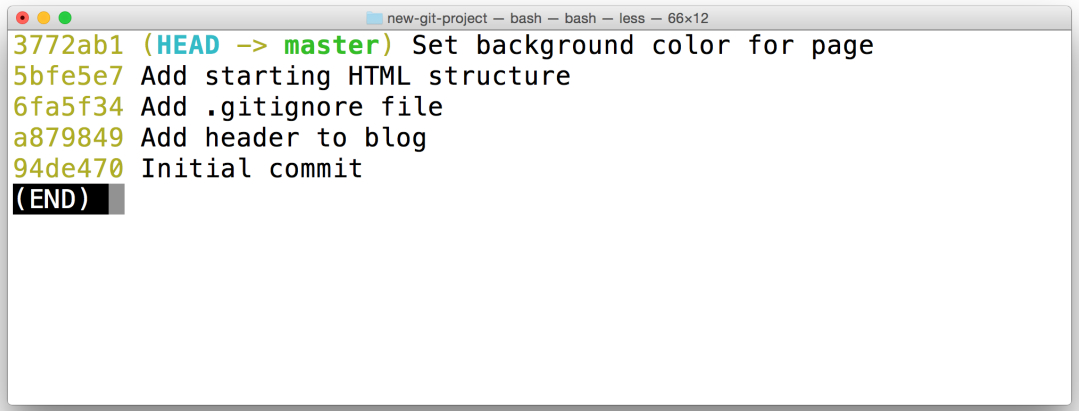
5. Tagging, Branching, and Merging – Branching Effectively

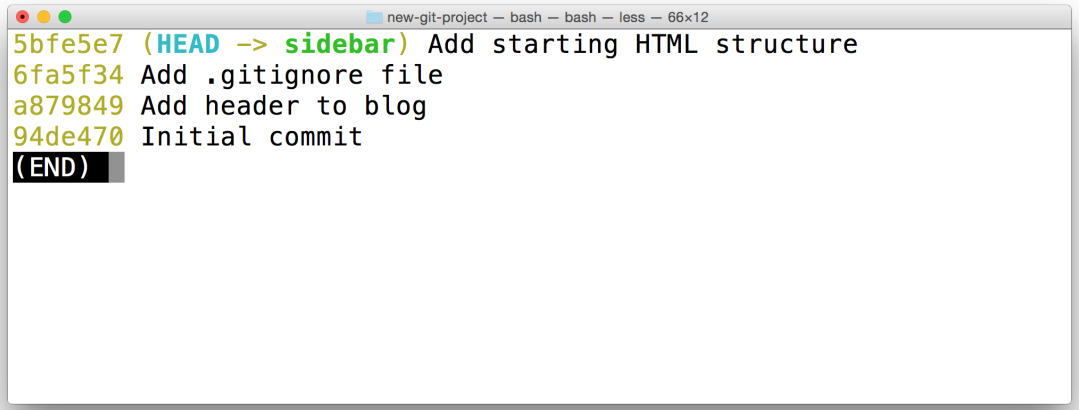
Change 2 – Add Sidebar

$ git branch sidebar 5bfe5e7

$ git checkout sidebar

$ git log --oneline

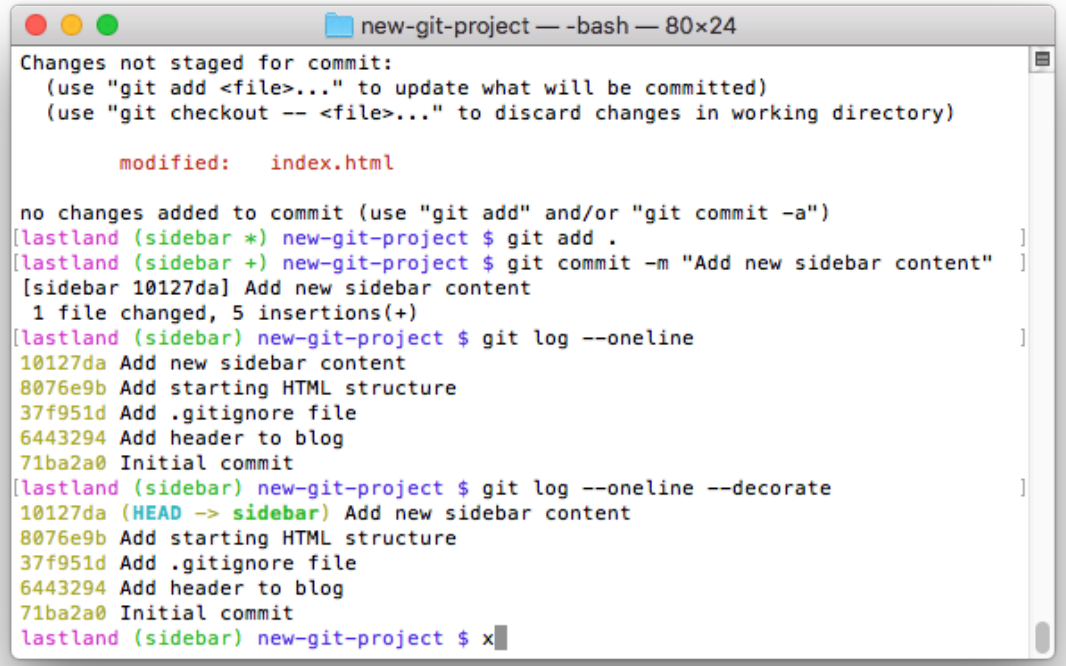




5. Tagging, Branching, and Merging – Branching Effectively

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>
```



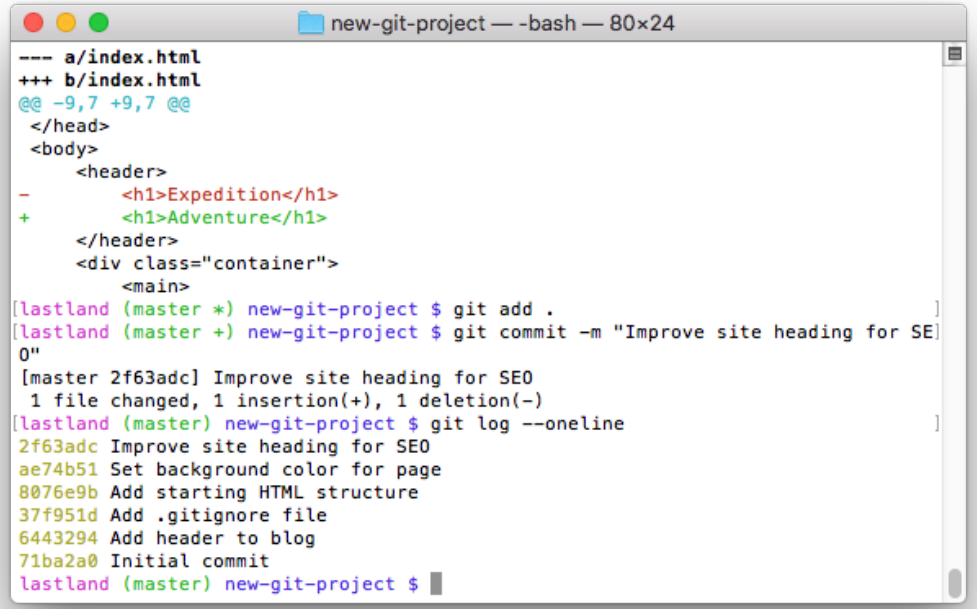
$ git commit –m “Add new sidebar content”

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!

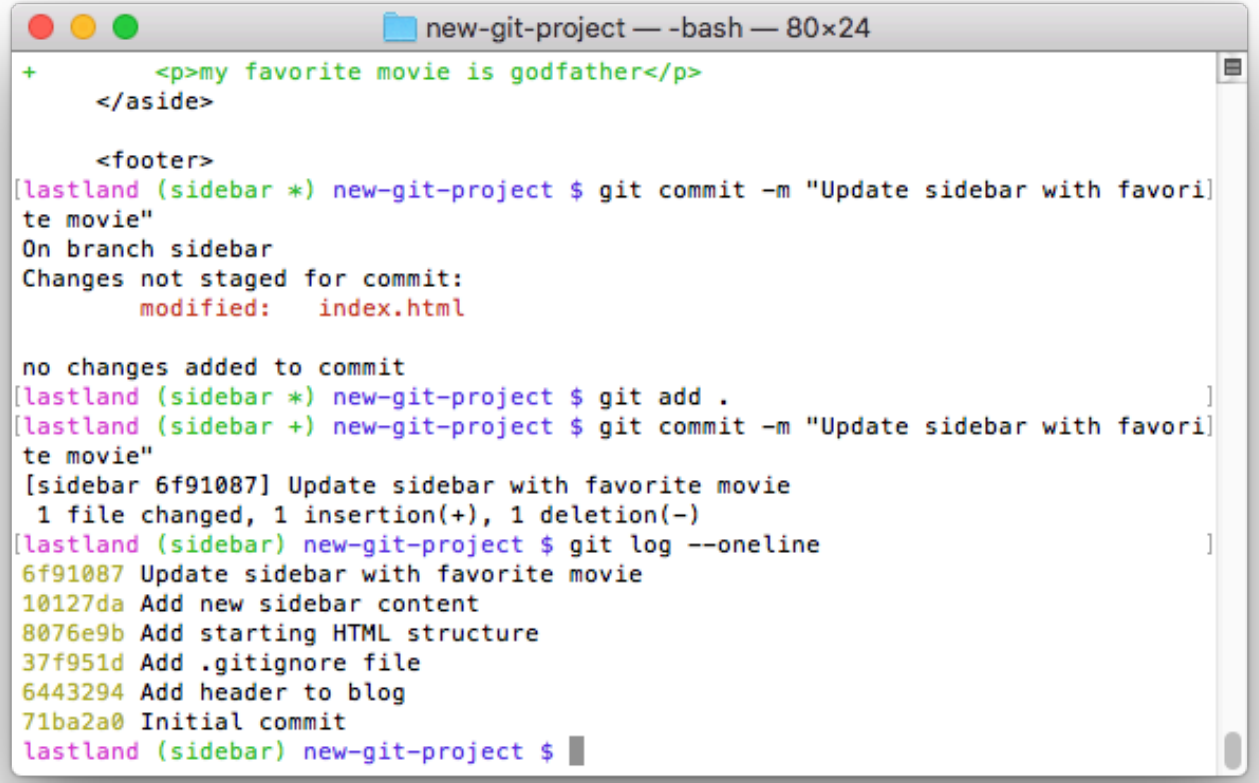


$ git commit –m “Improve site heading for SEO”

5. Tagging, Branching, and Merging – Branching Effectively

Change 4 – Add More Content To Sidebar

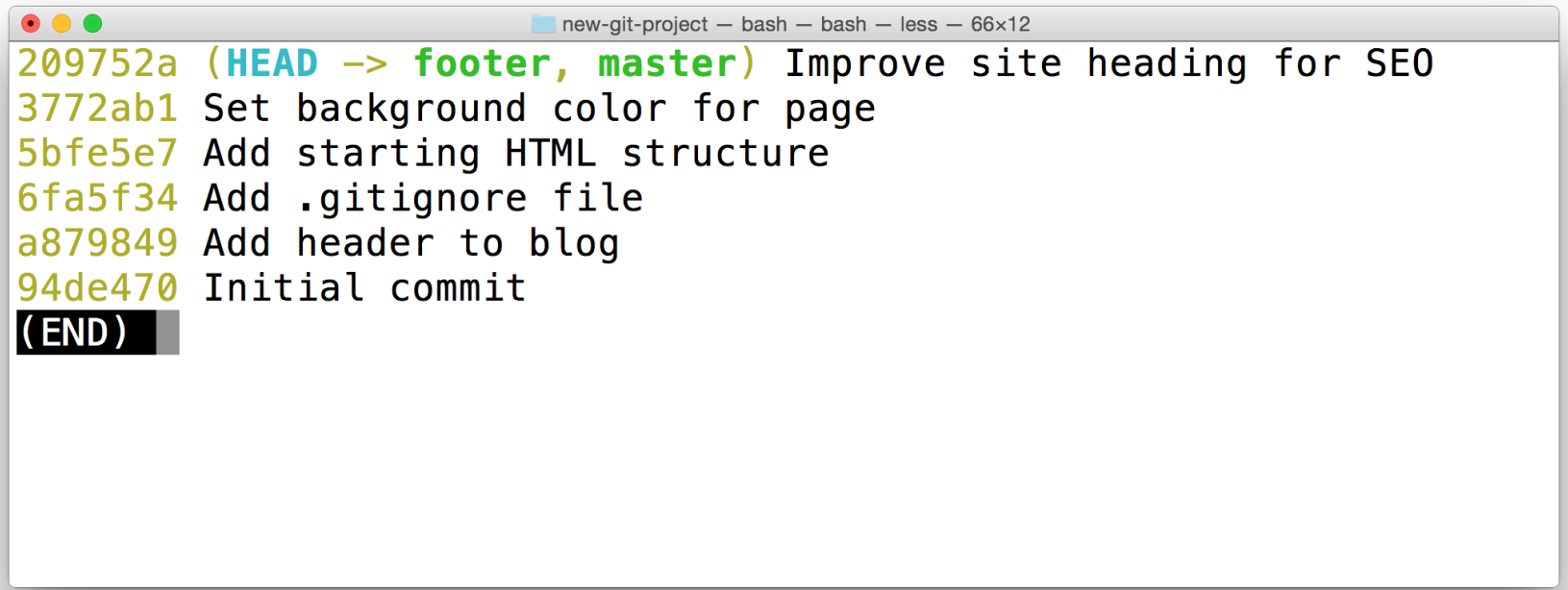
1. <aside> 태그 안의 내용 변경 – 좋아하는 영화, 책 등
2. 커밋!



5. Tagging, Branching, and Merging – Branching Effectively

Change 5 – Add Social Links To Footer

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



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">
      <img src="img/social-twitter.png" alt="Twitter">
    </a>
    <a class="social-link"  href="https://www.instagram.com/udacity/">
      <img src="img/social-instagram.png" alt="Instagram">
    </a>      <a class="social-link"  href="https://plus.google.com/+Udacity">
      <img src="img/social-google.png" alt="Google Plus">
    </a>
  </section>
</footer>
```

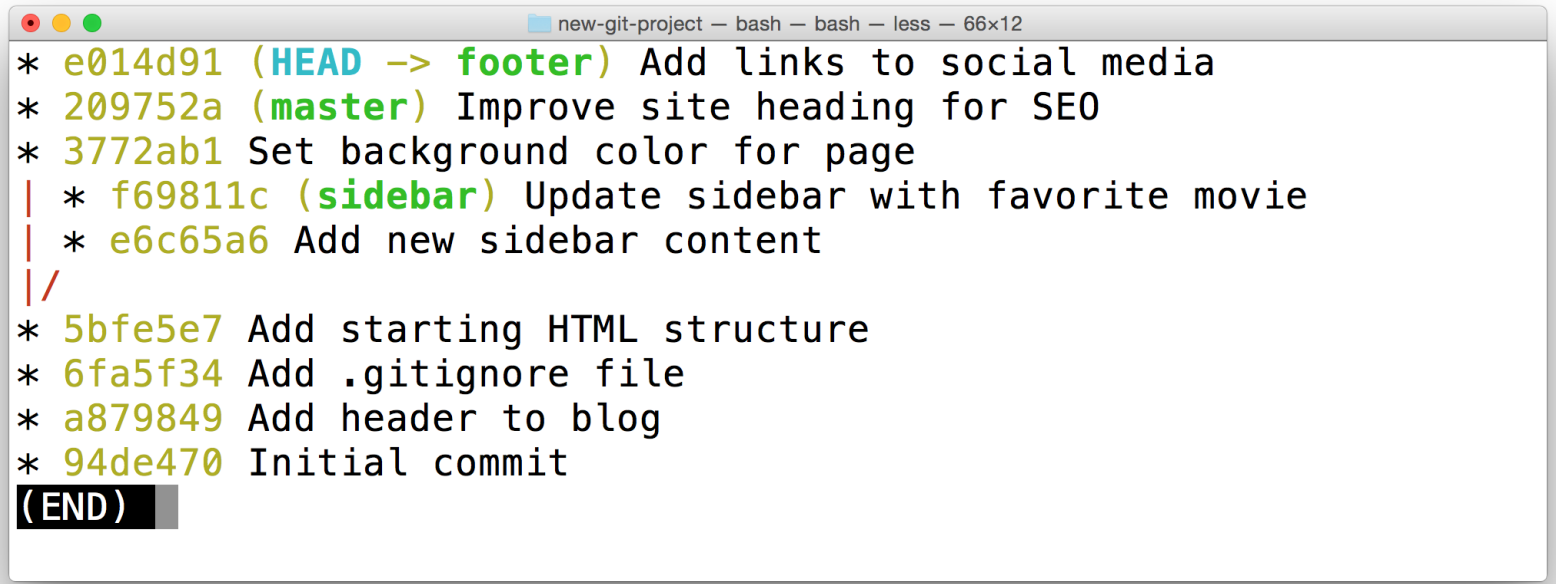
커밋!

$ 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



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