GIT: From Beginner to Fearless

GIT status, diff, log, and show Activity: What is the state of my repo?

Brian Gorman, Author/Instructor/Trainer

Introduction

There will be many times we will want to check on the status of our repository. By "status," we are actually looking to find out what has been changed, as well as what is ready to be committed. Don't worry if these terms are still new to you at this point, because as we spend more time working through this activity, they will become quite a bit more familiar.

We've already worked through a few flow operations where we've seen files move from untracked to tracked to staged to committed. Additionally, we've seen the "official" diagram regarding the general flow of files through different phases in GIT. We've learned about the working directory, the Index (or staging area) and the Head, which is essentially a pointer to the last commit on the checked-out branch.

In this activity, we'll go through some of these things in detail, and learn about ways we can use the status command to get us the information we need about the current state of our repo.

Let's gets started!

GFBTF: GIT status, diff, log, and show Activity

Step 1: Start with a working copy of the repository files:

a) Download the default web as a .zip file.
We're going to start fresh for practice and to see the status change for various objects
https://github.com/majorguidancesolutions/DefaultWeb

b) Alternatively, if you already have a copy and want to use what you have
You could just copy the existing directory then delete the .git Folder

c) Alternatively, clone the repo
You could also clone the repo locally, then delete the .git Folder

Step 2: Establish a repository:

a) Check your folder structure
Use [**`ls -al`**] to determine if a repo already exists. If one exists, delete it

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb
$ ls -al
total 52
drwxr-xr-x 1 Brian 197608    0 Jul  5 09:23 ./
drwxr-xr-x 1 Brian 197608    0 Jul  5 09:23 ../
-rw-r--r-- 1 Brian 197608 6946 Jun 30 00:15 About.html
-rw-r--r-- 1 Brian 197608 6392 Jun 30 00:15 ContactUs.html
drwxr-xr-x 1 Brian 197608    0 Jun 29 22:50 css/
-rw-r--r-- 1 Brian 197608 3213 Jun 30 00:18 details.html
drwxr-xr-x 1 Brian 197608    0 Jul 25  2016 fonts/
drwxr-xr-x 1 Brian 197608    0 Jun 30 00:00 images/
-rw-r--r-- 1 Brian 197608 5745 Jun 30 00:15 index.html
drwxr-xr-x 1 Brian 197608    0 Jun 29 22:45 js/
-rw-r--r-- 1 Brian 197608 6933 Jun 30 00:15 portfolio.html
```

Step 3: Check Status:

a) Create a new repository
[**`git init`**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb
$ git init
Initialized empty Git repository in G:/Data/GFBTF/DefaultWeb/.git/
```

Notes

Use [**ls -al**] to see all files, validate a .git folder exists

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb
$ ls -al
total 56
drwxr-xr-x 1 Brian 197608    0 Jul  5 09:23 ./
drwxr-xr-x 1 Brian 197608    0 Jul  5 09:23 ../
drwxr-xr-x 1 Brian 197608    0 Jul  5 09:23 .git/
-rw-r--r-- 1 Brian 197608 6946 Jun 30 00:15 About.html
-rw-r--r-- 1 Brian 197608 6392 Jun 30 00:15 ContactUs.html
drwxr-xr-x 1 Brian 197608    0 Jun 29 22:50 css/
-rw-r--r-- 1 Brian 197608 3213 Jun 30 00:18 details.html
drwxr-xr-x 1 Brian 197608    0 Jul 25  2016 fonts/
drwxr-xr-x 1 Brian 197608    0 Jun 30 00:00 images/
-rw-r--r-- 1 Brian 197608 5745 Jun 30 00:15 index.html
drwxr-xr-x 1 Brian 197608    0 Jun 29 22:45 js/
-rw-r--r-- 1 Brian 197608 6933 Jun 30 00:15 portfolio.html
```

.git folder is present when a local repo exists

b) Take a look at the status of the repo right now:

[**git status**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status
On branch master

Initial commit

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

        About.html
        ContactUs.html
        css/
        details.html
        fonts/
        images/
        index.html
        js/
        portfolio.html

nothing added to commit but untracked files present (use "git add" to track)
```

Note that at this time, all the listed objects are untracked.

What do you think the red color means? _____

c) Review the state of the repo again

[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
?? About.html
?? ContactUs.html
?? css/
?? details.html
?? fonts/
?? images/
?? index.html
?? js/
?? portfolio.html
```

What do you think the red double ??'s mean?: _____

Step 4: Adding and Removing a file from the Index ["Staging Area"]:

a) Add a single file to index

[git add About.html]

[git status]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git add About.html

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status
On branch master

Initial commit

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

        new file:   About.html

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

        ContactUs.html
        css/
        details.html
        fonts/
        images/
        index.html
        js/
        portfolio.html
```

Note the added file is Green and the rest of the files are red. Green indicates the file is "staged" or is in the "Index" ready to be committed. If we were to type 'git commit –m "…"' we would commit just the About.html file. However, we're not going to do that just yet.

b) Review the short status

[git status -s]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
A  About.html
?? ContactUs.html
?? css/
?? details.html
?? fonts/
?? images/
?? index.html
?? js/
?? portfolio.html
```

Here it is very important to note something: look at how the first column is "A " ["A" and a BLANK]. What do you think this means?
The double ??'s indicate that the remaining files are both unstaged and untracked.

c) Remove a file from INDEX

[git rm --cached About.html]

[git status]

[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git rm --cached About.html
rm 'About.html'

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
?? About.html
?? ContactUs.html
?? css/
?? details.html
?? fonts/
?? images/
?? index.html
?? js/
?? portfolio.html

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status
On branch master

Initial commit

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

        About.html
        ContactUs.html
        css/
        details.html
        fonts/
        images/
        index.html
        js/
        portfolio.html

nothing added to commit but untracked files present (use "git add" to track)
```

The files have returned to “UnIndexed” and “UnTracked”

*** IMPORTANT note. This ONLY works without consequences because we’ve never committed the file. If we had committed the file, this would be a terrible idea because it would remove it from the repo as well. If the file is already in the repo and we need to roll back to where it was, we’d use [`git reset`]. We’ll learn about the reset command later in the course.

Step 5: Adding and Removing multiple files:

a) Add all files that are unstaged to the INDEX

[**git add .**]

[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git add .
warning: LF will be replaced by CRLF in fonts/glyphicons-halflings-regular.svg.
The file will have its original line endings in your working directory.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
A  About.html
A  ContactUs.html
A  css/site.css
A  details.html
A  fonts/glyphicons-halflings-regular.eot
A  fonts/glyphicons-halflings-regular.svg
A  fonts/glyphicons-halflings-regular.ttf
A  fonts/glyphicons-halflings-regular.woff
A  fonts/glyphicons-halflings-regular.woff2
A  images/linkedin.png
A  images/twitter.png
A  index.html
A  portfolio.html

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$
```

b) Remove all of the files from the INDEX

[**git rm –cached -r .**]

[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git rm --cached -r .
rm 'About.html'
rm 'ContactUs.html'
rm 'css/site.css'
rm 'details.html'
rm 'fonts/glyphicons-halflings-regular.eot'
rm 'fonts/glyphicons-halflings-regular.svg'
rm 'fonts/glyphicons-halflings-regular.ttf'
rm 'fonts/glyphicons-halflings-regular.woff'
rm 'fonts/glyphicons-halflings-regular.woff2'
rm 'images/linkedin.png'
rm 'images/twitter.png'
rm 'index.html'
rm 'portfolio.html'

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
?? About.html
?? ContactUs.html
?? css/
?? details.html
?? fonts/
?? images/
?? index.html
?? portfolio.html
```

Now we know a way to add and remove all files. Please remember that using rm is something we should be careful with. Once files are being tracked, this actually removes them from the repository, so we'll use a command [**git reset**] when we just want to unstage files and not actually remove them from tracking.

c) Committing our files

Let's add just one file – about.html – to the INDEX. What is the command again to add just one file? _____

After adding the about.html file to the INDEX, commit the file with a commit message:

[**git commit –m 'Added the About.html file'**]

[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git add About.html

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git commit -m 'Added the About.html file'
[master (root-commit) f69f692] Added the About.html file
 1 file changed, 97 insertions(+)
 create mode 100644 About.html

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
?? ContactUs.html
?? css/
?? details.html
?? fonts/
?? images/
?? index.html
?? portfolio.html

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$
```

Wait! Where is my About.html file? It's not listed!!!

Maybe if I use full git status I can see it?

[**git status**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        ContactUs.html
        css/
        details.html
        fonts/
        images/
        index.html
        portfolio.html

nothing added to commit but untracked files present (use "git add" to track)
```

Oh no! I lost my file! --- or not? Right -It's simply not shown because there are no changes to the file and it's already committed as-is. Want to see it?

d) View files using the log and show commands

[**git log**]

```
$ git log
commit 1b193f68356f88852ccb03afaa4497160114f737 (HEAD -> master)
Author: Brian L. Gorman <blgorman@gmail.com>
Date:   Sat Jul 13 16:15:30 2019 -0500

    Added the About.html file
```

[*note – this is an updated image, so the commit id is not correct for the original tree]

Note "Added the About.html file" is the commit message I typed. If you typed something different, you'd see that there.

[**git show --summary**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git show --summary
commit f69f6921daa4a7f758e0bd849c98501ade961a2c (HEAD -> master)
Author: Brian L. Gorman <blgorman@gmail.com>
Date:   Wed Jul 5 09:56:05 2017 -0500

    Added the About.html file

 create mode 100644 About.html
```

This tells me that the file "About.html" was "created"

We can also see this with other versions of the show command:

[**git show --stat**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git show --stat
commit f69f6921daa4a7f758e0bd849c98501ade961a2c (HEAD -> master)
Author: Brian L. Gorman <blgorman@gmail.com>
Date:   Wed Jul 5 09:56:05 2017 -0500

    Added the About.html file

 About.html | 97 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 1 file changed, 97 insertions(+)
```

[**git show --name-status**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git show --name-status
commit f69f6921daa4a7f758e0bd849c98501ade961a2c (HEAD -> master)
Author: Brian L. Gorman <blgorman@gmail.com>
Date:   Wed Jul 5 09:56:05 2017 -0500

    Added the About.html file

A       About.html
```

e) Add the remaining files

Right now, we have just the About.html file committed (in HEAD). And the rest of the files are Untracked, with nothing in Index (Staging).

Add the remaining files to the Index. What is the command again to add all the untracked or modified file changes to Index?

_____

[**git commit -m "Added the rest of the files"**]

[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git commit -m "Added the rest of the files"
[master 874d595] Added the rest of the files
 12 files changed, 711 insertions(+)
 create mode 100644 ContactUs.html
 create mode 100644 css/site.css
 create mode 100644 details.html
 create mode 100644 fonts/glyphicons-halflings-regular.eot
 create mode 100644 fonts/glyphicons-halflings-regular.svg
 create mode 100644 fonts/glyphicons-halflings-regular.ttf
 create mode 100644 fonts/glyphicons-halflings-regular.woff
 create mode 100644 fonts/glyphicons-halflings-regular.woff2
 create mode 100644 images/linkedin.png
 create mode 100644 images/twitter.png
 create mode 100644 index.html
 create mode 100644 portfolio.html

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$
```

So all of our files are added, and note that the short version of status shows nothing!

[**git status**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status
On branch master
nothing to commit, working tree clean
```

[**git show --name-status**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git show --name-status
commit 874d5950c848e6bee50db22c6bb0b62458f1ca0d (HEAD -> master)
Author: Brian L. Gorman <blgorman@gmail.com>
Date:   Wed Jul 5 10:07:51 2017 -0500

    Added the rest of the files

A       ContactUs.html
A       css/site.css
A       details.html
A       fonts/glyphicons-halflings-regular.eot
A       fonts/glyphicons-halflings-regular.svg
A       fonts/glyphicons-halflings-regular.ttf
A       fonts/glyphicons-halflings-regular.woff
A       fonts/glyphicons-halflings-regular.woff2
A       images/linkedin.png
A       images/twitter.png
A       index.html
A       portfolio.html
```

Note: About.html is not shown. Why do you think that is?

_____

Is About.html sill in the repo?

_____

How can I know for sure?

_____

[**git log**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git log
commit 874d5950c848e6bee50db22c6bb0b62458f1ca0d (HEAD -> master)
Author: Brian L. Gorman <blgorman@gmail.com>
Date:   Wed Jul 5 10:07:51 2017 -0500

    Added the rest of the files

commit f69f6921daa4a7f758e0bd849c98501ade961a2c
Author: Brian L. Gorman <blgorman@gmail.com>
Date:   Wed Jul 5 09:56:05 2017 -0500

    Added the About.html file
```

Additionally, we can always list the names of the files and their statuses from any commit using the sha1 first six characters. For example,

[**git show –name-status 874d59**]

```
$ git show --name-status 1b193f
commit 1b193f68356f88852ccb03afaa4497160114f737
Author: Brian L. Gorman <blgorman@gmail.com>
Date:   Sat Jul 13 16:15:30 2019 -0500

    Added the About.html file

A       About.html
```

*reminder: I updated the image so commit changed. Yours will also be different.

Step 6: More Show and Log examples:

a) Make some changes

[**vim Details.html**]

When the file comes up, make modifications.

[**i**] //hit ‘i’ to insert in VIM

Add an h2 tag under the <h1> that says ‘details page with more content:

[**<h2>Git is awesome and I’m learning so much cool stuff!”</h2>**]

[**{esc}**] //hit the escape key to exit insert mode in VIM

[**:wq**] //to write and quit VIM

[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ vim Details.html

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
 M details.html
```

Note the file has a red “M”. The red “M” is in the second column. This means the file is “Modified” against the Index version of the file {which is yet another way to know we are tracking the file}. Note the “M” is RED because the file is not yet added to index.

[**git status**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ 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:   details.html

no changes added to commit (use "git add" and/or "git commit -a")
```

b) Add the changes

Add the details.html file to the index. What is the command for adding a single file to Index again?

_____

What if I had just typed [**git add .**]? [all modified files would be staged – and since only one file was modified, it would be staged just the same!]

[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git add .

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
M  details.html
```

Note: Now the file has a Green “M” in the Index column. We now know the file is Modified and Added to Index [staged] for commit.

c) Commit the changes

Let’s commit the changes:

[**git commit –m “Added an h2 tag to the details page”**]

[**git status -s**]

[**git log --oneline**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git commit -m 'Added an h2 tag to the details page'
[master 331b291] Added an h2 tag to the details page
 1 file changed, 2 insertions(+), 2 deletions(-)

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status
On branch master
nothing to commit, working tree clean

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git log --oneline
331b291 (HEAD -> master) Added an h2 tag to the details page
874d595 Added the rest of the files
f69f692 Added the About.html file
```

Note the three commits, and note HEAD -> master. This means the current HEAD is at commit 331b29 and the branch we are on is master. Don't worry if that's a little unclear still. We'll be learning much more about this as we progress through the course.

Step 7: Modified files that are already tracked, only commit the things we want to commit:

a) Repeat the modification steps as above to make another change in details.html.

[**vim details.html**]
[**i**]
//under the <h2> tag we entered above, place the following line
[**<h3>Multiple changes are coming</h3>**]
[**{esc}**]
[**:wq**]
[**git add details.html**] //add the change for the <h3> tag to INDEX

b) Create another change

[**vim details.html**]
[**i**]
//under the <h3> tag we entered above, place the following line
[**<h4>Yet another change</h4>**]
[**{esc}**]
[**:wq**]
//DO NOT add this change to staging (INDEX).
[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
MM details.html
```

Whoa! Now we have one modification ready to add [Red, modified against HEAD but not in index] and one modification ready to commit [Green, modified against HEAD and Staged in INDEX]. That's cool. But what will I commit if I commit from this state? [we know it's just the h3 tag, but this is an easy activity. Think about hundreds of changes in a system with only some of them added to Index and some not added, or a practical situation where you want certain changes but don't want others…]

c) See what is added and ready to commit

[**git diff --cached details.html**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git diff --cached details.html
diff --git a/details.html b/details.html
index b92420b..19ce9f9 100644
--- a/details.html
+++ b/details.html
@@ -44,6 +44,7 @@

             <h1> Details Page with more content... </h1>
            <h2> Git is awesome and I'm learning so much cool stuff!</h2>
+           <h3> More changes are coming </h3>
             <hr />
             <footer>
                 <p>&copy; 2017 - <a href="http://www.majorguidancesolutions.com
">Major Guidance Solutions</a></p>
```

If we commit now, we'd see the green line added to the file in the new HEAD. Exactly what we expect.

To see what changes would NOT be committed [they would NOT be lost, just not committed to HEAD]:

[**git diff details.html**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git diff details.html
diff --git a/details.html b/details.html
index 19ce9f9..5812aae 100644
--- a/details.html
+++ b/details.html
@@ -45,7 +45,8 @@
             <h1> Details Page with more content... </h1>
            <h2> Git is awesome and I'm learning so much cool stuff!</h2>
            <h3> More changes are coming </h3>
-            <hr />
+           <h4> Yet another change </h4>
+           <hr />
             <footer>
                 <p>&copy; 2017 - <a href="http://www.majorguidancesolutions.com
">Major Guidance Solutions</a></p>
             </footer>
```

Here we can see that the changes for the <h4> tag are waiting to be staged.

It's important to remember that ANYTHING that is not added to INDEX [staged] at time of commit will be left out of the commit.

It is equally as important to remember that anything that is modified but not staged is NOT deleted by a commit action. It's just not included, and your file remains altered with changes ready to be staged and committed.

d) Commit the partial changes

Let's see this in action to lose any remaining fear we have:

[**git commit -m "Added the h3 tag for upcoming changes"**]

[**git status -s**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git commit -m "Added the h3 tag for upcoming changes"
[master cb5204a] Added the h3 tag for upcoming changes
 1 file changed, 1 insertion(+)

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
 M details.html
```

Again, here we see that our h3 tag was added, and now we have ONLY the remaining modification for the h4 tag. Since nothing is added to INDEX for staging, [**`git diff –cached details.html`**] yields 0 results, but [**`git diff details.html`**] still shows our remaining changes!

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git diff --cached details.html

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git diff details.html
diff --git a/details.html b/details.html
index 19ce9f9..5812aae 100644
--- a/details.html
+++ b/details.html
@@ -45,7 +45,8 @@
            <h1> Details Page with more content... </h1>
            <h2> Git is awesome and I'm learning so much cool stuff!</h2>
            <h3> More changes are coming </h3>
-           <hr />
+           <h4> Yet another change </h4>
+           <hr />
            <footer>
                <p>&copy; 2017 - <a href="http://www.majorguidancesolutions.com
">Major Guidance Solutions</a></p>
            </footer>
```

e) Final commit for the activity

Let's do one final commit to wrap up the activity with the add and commit command:

[**`git commit -am "Added the h4 tag change"`**]

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git commit -am "Added the h4 tag change"
[master c8672c8] Added the h4 tag change
 1 file changed, 2 insertions(+), 1 deletion(-)
```

Note: We can add and commit on the same line, in one command. Also note this only works on files that are already tracked, so using '`-am`' requires a modification to a previously committed file in order to work as expected.

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git status -s
```

Try it. Create a new file, but do not stage to index (don't add). Try to run the command [**`git commit -am "my new file added"`**]. What happens?

Running 'git log --oneline' reveals our final history:

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb (master)
$ git log --oneline
c8672c8 (HEAD -> master) Added the h4 tag change
cb5204a Added the h3 tag for upcoming changes
331b291 Added an h2 tag to the details page
874d595 Added the rest of the files
f69f692 Added the About.html file
```

Yours should be similar – but you commit ids will be different than mine, of course.

Closing Thoughts

In this activity, we've worked through making changes and seeing the state of our repo with the status command. We saw that status has both a long version and a short version. This allows us flexibility in how we want to show the status.

Additionally, we learned about how we can use the log and show commands to be able to list out our commit history, see details about commits, and view the contents of a commit.

We also learned that not everything has to be committed at one time, and also how important it is to make sure that everything we want to commit is actually staged into the INDEX. We can always see what is going to be added or review what is not going to be added using the diff command. We'll also learn in the future that the diff command will be highly useful for comparing changes for entire commits, as well as how we've used it here to view changes in the working directory and index.

Take a few minutes to make some notes about the various commands we've learned about in this activity, and practice using them.

Notes