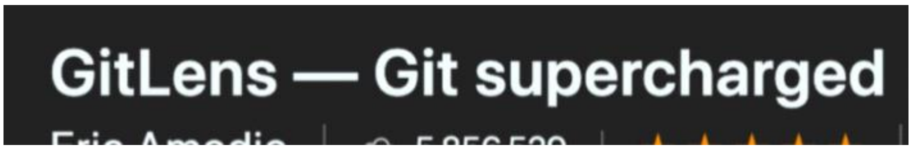
Getting Started

Thursday, April 28, 2022 3:20 PM

4- Using Git

VsCode



GitLens — Git supercharged

git自带的可视化工具Graphical User Interface图形用户界面

GUI Clients

Git comes with built-in GUI tools for committing (git-gui) and browsing (gitk), but there a party tools for users looking for platform-specific experience.

If you want to add another GUI tool to this list, just follow the instructions.

超级推荐

GitKraken Git GUI

Legendary Git client for Windows, Mac & Linux. Free for open source

可以配合使用：issue问题

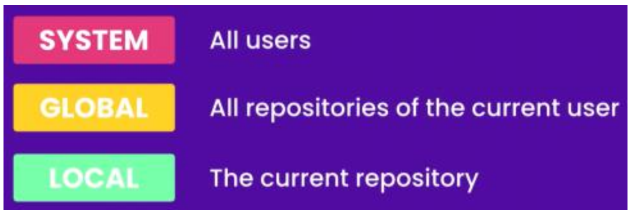
GitKraken boards for issue tracking, and GitKraken timelines for project management.

另一个选择

Sourcetree

6- Configuring Git

三个层



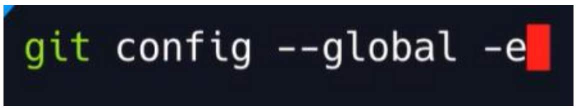
SYSTEM	All users
GLOBAL	All repositories of the current user
LOCAL	The current repository

- Default Editor
- Line Ending

```
git config --global core.editor "code --wait"
```

wait意味着：告诉terminal window，等直到我们关闭vs code instance

所有这个config被保存在一个txt file中，我们可以利用我们的default editor edit这个file

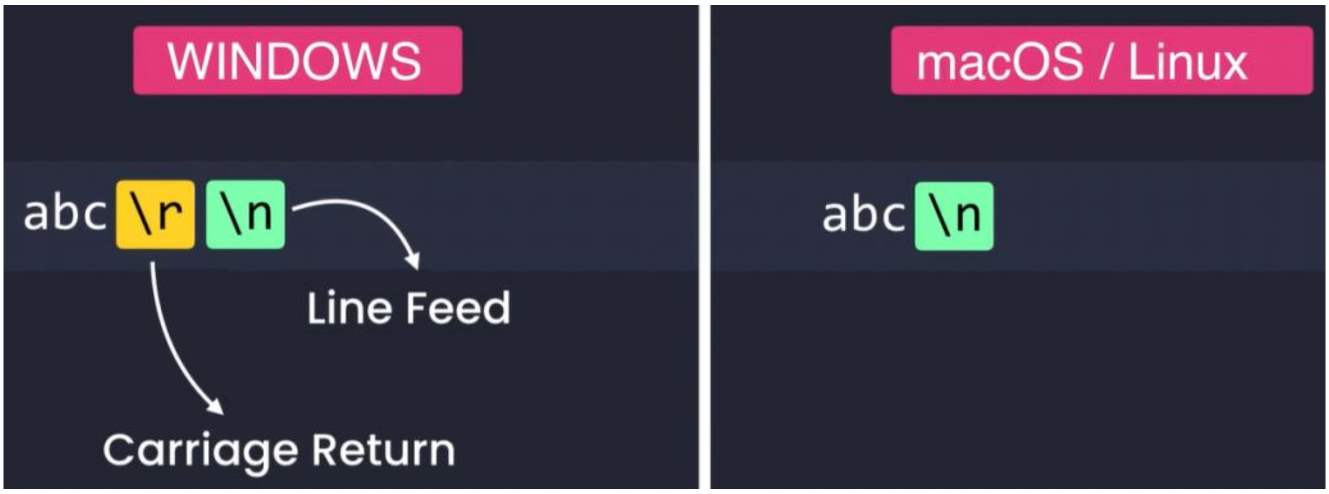


```
git config --global -e
```

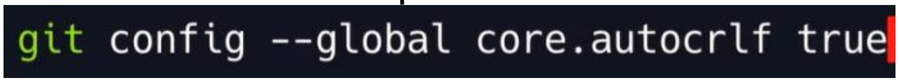
Edit all the global settings

Carriage return 回车，line feed 换行

在windows中，end of line are marked with two special characters,回车和换行



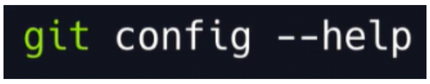
Windows true mac input



```
git config --global core.autocrlf true
```

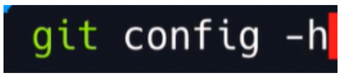
7- Getting Help

Doc可以，以下命令也可以，space是换页，q是exit



```
git config --help
```

如果只需要quick refresher



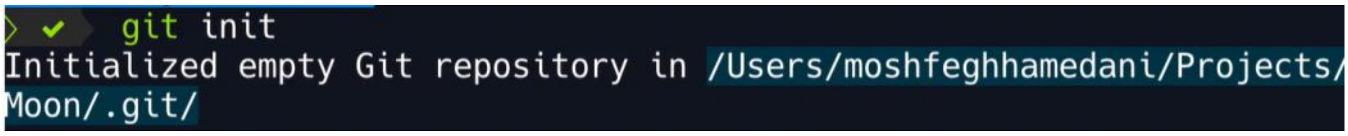
```
git config -h
```

Creating Snapshots

Thursday, April 28, 2022 5:03 PM

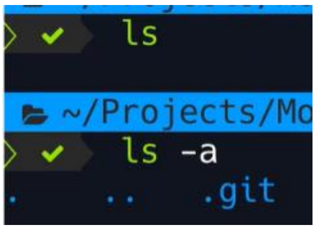
2- Initializing a Repository

在需要的项目folder中，新建一个repo，folder中会出现一个新的folder--->.git/，这是hidden的folder



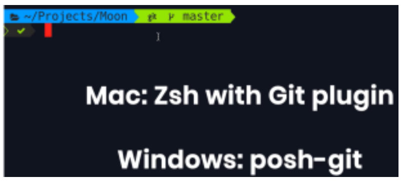
```
git init
Initialized empty Git repository in /Users/moshfeghhamedani/Projects/
Moon/.git/
```

ls是看不到隐藏文件夹的，-a意味着 for all



```
ls

~/Projects/Mo
ls -a
.    ..    .git
```

下图是colorful terminal:

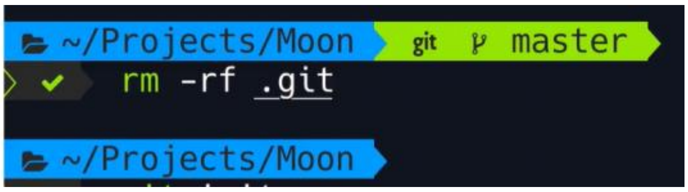


```
~/Projects/Moon  git  master
```

Mac: Zsh with Git plugin

Windows: posh-git

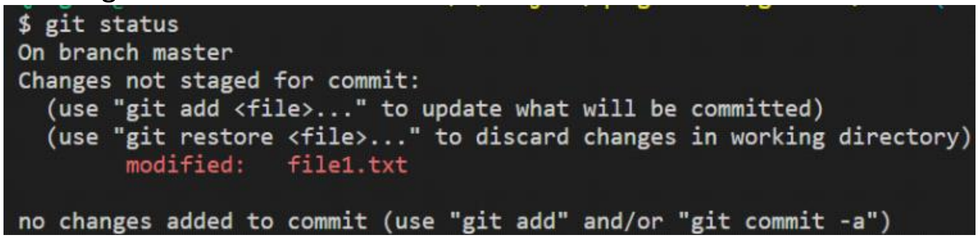
删除后就没有绿色的部分了



```
~/Projects/Moon  git  master
rm -rf .git

~/Projects/Moon
```

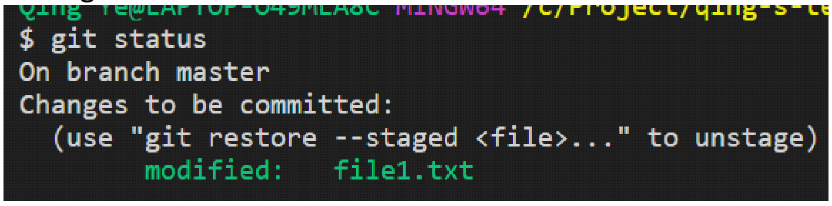
4- Staging Files

Before git add . :



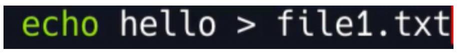
```
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   file1.txt

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

After git add . :

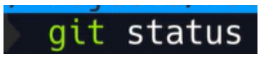


```
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   file1.txt
```

Git status 查看working directory以及staging area的所有情况

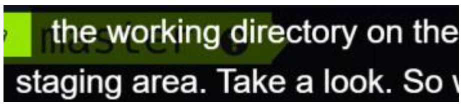


```
echo hello > file1.txt
```

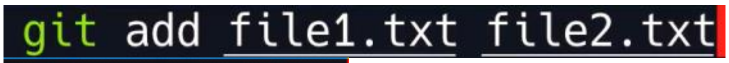
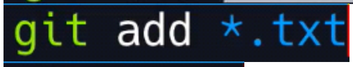
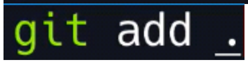


```
git status
```

查看

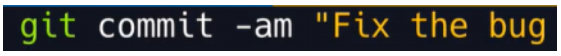

the working directory on the
staging area. Take a look. So v

添加


```
git add file1.txt file2.txt
```

```
git add *.txt
```

```
git add .
```

7- Skipping the Staging Area

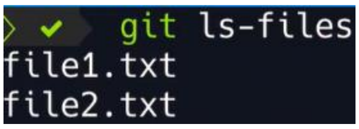
-a -m可以写在一起


```
git commit -am "Fix the bug
```

8- Removing Files

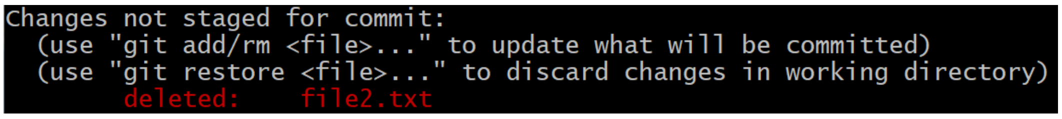
删除file2以后

以下命令列出的是files in staging area

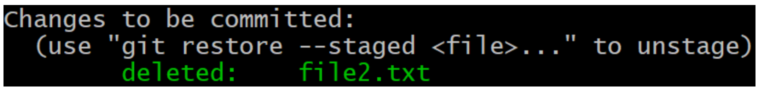

```
 ✔  git ls-files
file1.txt
file2.txt
```

File2依然再staging area是因为，每次commit后，staging area都不会清空，而会与repo保持一致

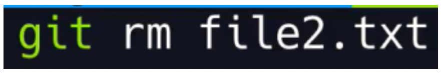
add之前，红色表示，working directory是dirty的


```
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    file2.txt
```

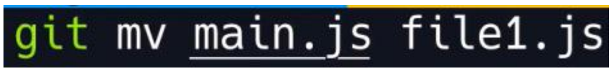
add之后commit之前，绿色表示再staging area中


```
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    file2.txt
```

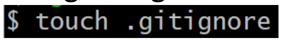
以下命令，会再working directory与staging area中同时删除


```
git rm file2.txt
```

Rename main.js to file1.js，同时再working directory与staging area中rename

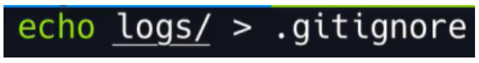

```
git mv main.js file1.js
```

10- Ignoring Files


```
$ touch .gitignore
```

.gitignore 应该在root directory

将logs这个文件夹，写进.gitignore中


```
echo logs/ > .gitignore
```

如果一个file或者folder已经在staging area中了，即使写入gitignore中，进行add的时候，还是会add它，怎么办，利用git rm，但是这会将working directory和staging area中的file都删除，怎么办，利用-h查看一下怎么办

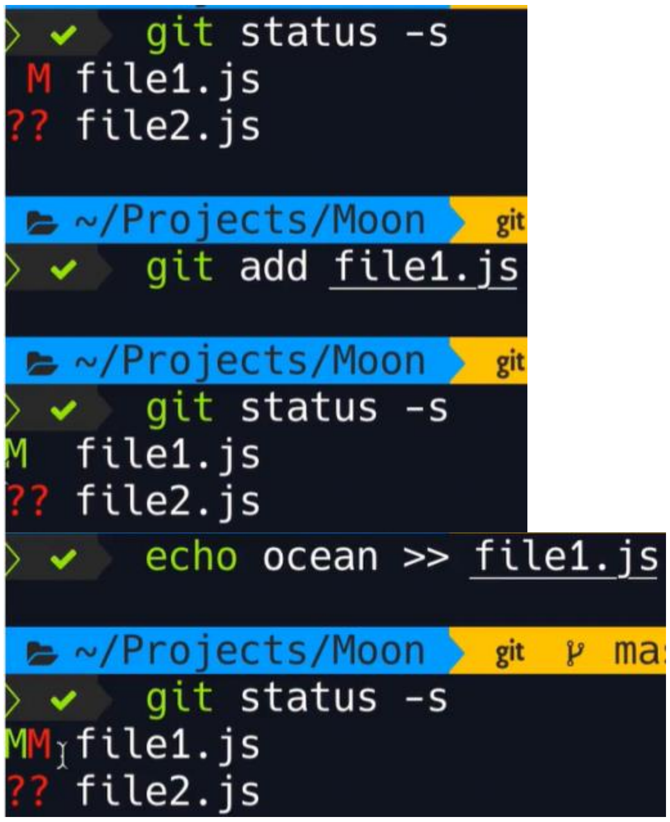
```
git rm -h
```
只删除index或者说staging area中的
```
--cached              only remove from the index
```
删除文件夹
```
-r                    allow recursive removal
```
查看不同语言的常用ignore模板

github.com/github/gitignore

11- Short Status

m是modified，红色是在working directory但是没在staging area的意思

对于一个file，左侧会有两个column：空红m，？？，绿m空。左代表在staging area中的状态，右column代表在working directory中的状态



```
git status -s
 M file1.js
?? file2.js

~/Projects/Moon  git
git add file1.js

~/Projects/Moon  git
git status -s
M  file1.js
?? file2.js
echo ocean >> file1.js

~/Projects/Moon  git  ma
git status -s
MM file1.js
?? file2.js
```

M是modify，a是add，？表示untracked file

12- Viewing Staged and Unstaged Changes

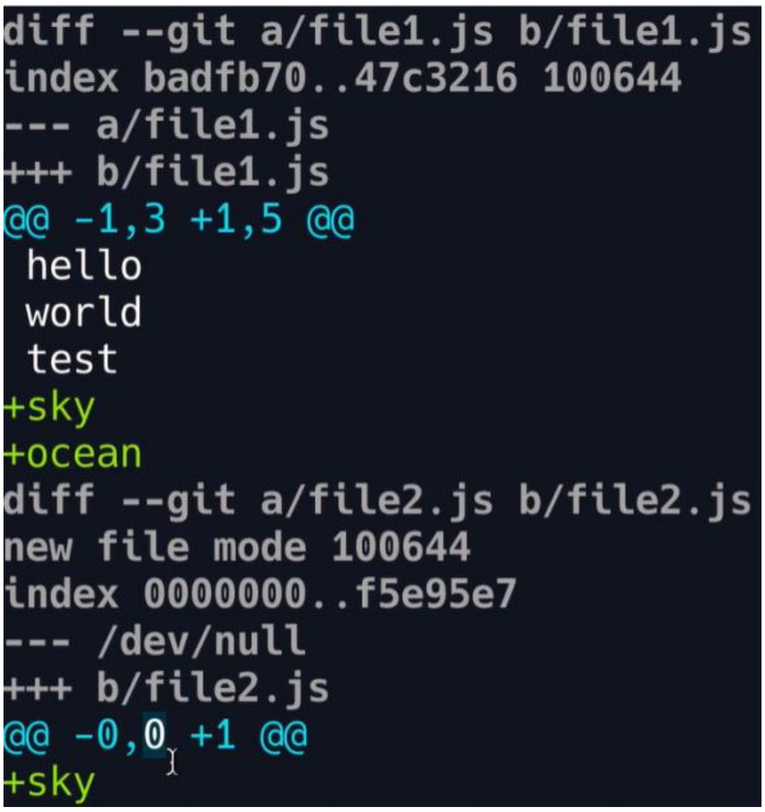
--staged代表，想看在stage中，下一次将被commit的具体内容
```
git diff --staged
```
比较两个file，a代表过去的file，b代表新的file

---代表过去file的变化，+++代表新file的变化

@@是head

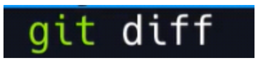
-1，3代表，旧的版本中，从第一行开始，一共三行

/dev/null代表没有就file



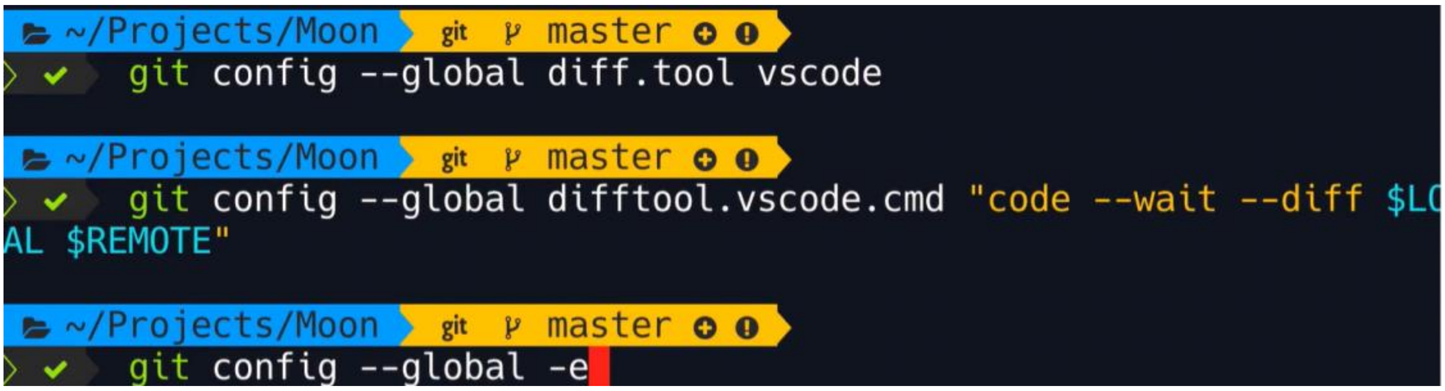
```
diff --git a/file1.js b/file1.js
index badfb70..47c3216 100644
--- a/file1.js
+++ b/file1.js
@@ -1,3 +1,5 @@
 hello
 world
 test
+sky
+ocean
diff --git a/file2.js b/file2.js
new file mode 100644
index 0000000..f5e95e7
--- /dev/null
+++ b/file2.js
@@ -0,0 +1 @@
+sky
```

想要比较staging area和working directory的不同



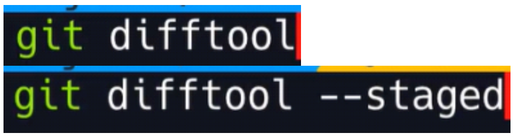
```
git diff
```

13- Visual Diff Tools



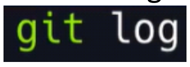
```
~/Projects/Moon  git  master
git config --global diff.tool vscode

~/Projects/Moon  git  master
git config --global difftool.vscode.cmd "code --wait --diff $LC
AL $REMOTE"

~/Projects/Moon  git  master
git config --global -e
```

与diff一样规则

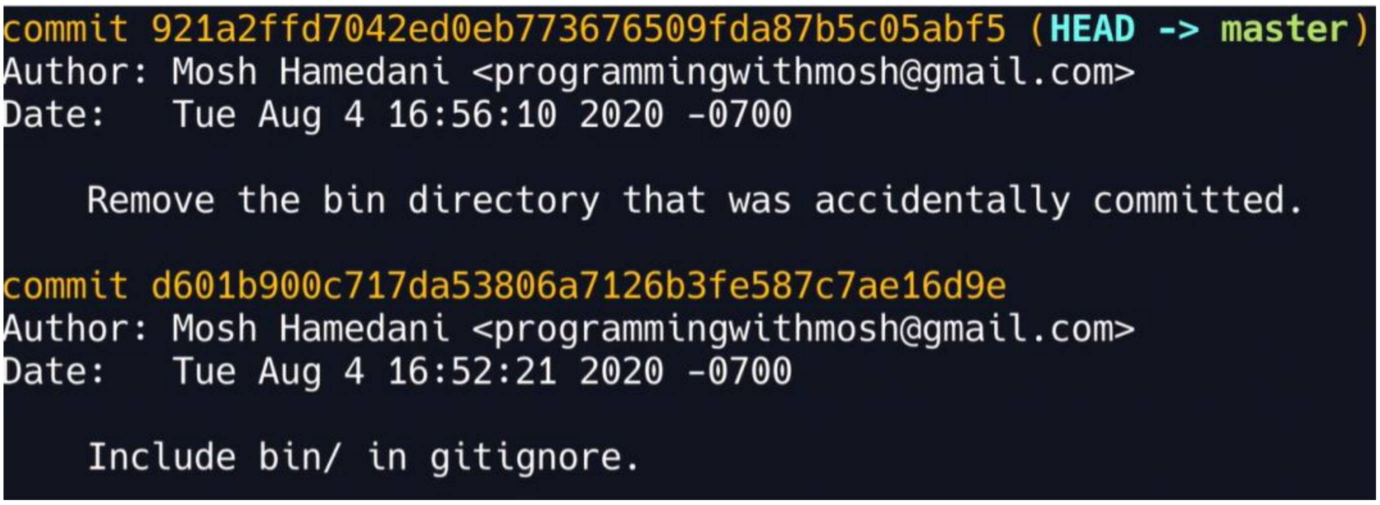


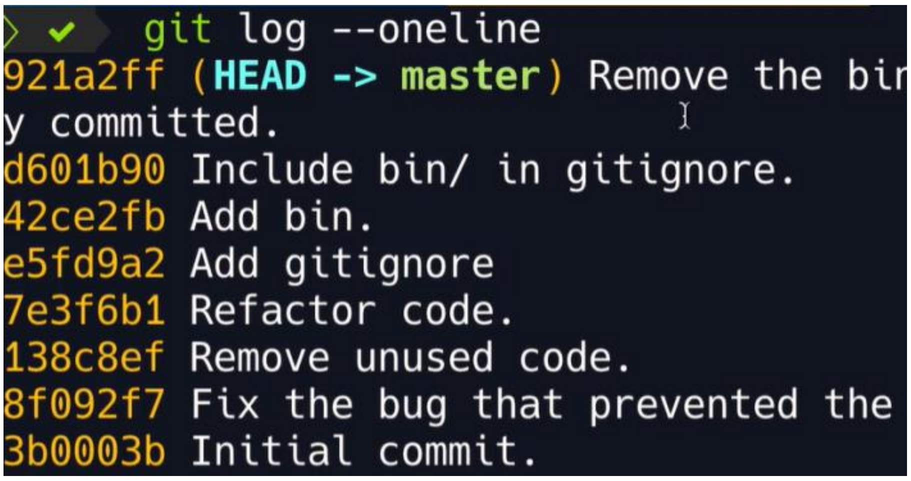
```
git difftool
git difftool --staged
```

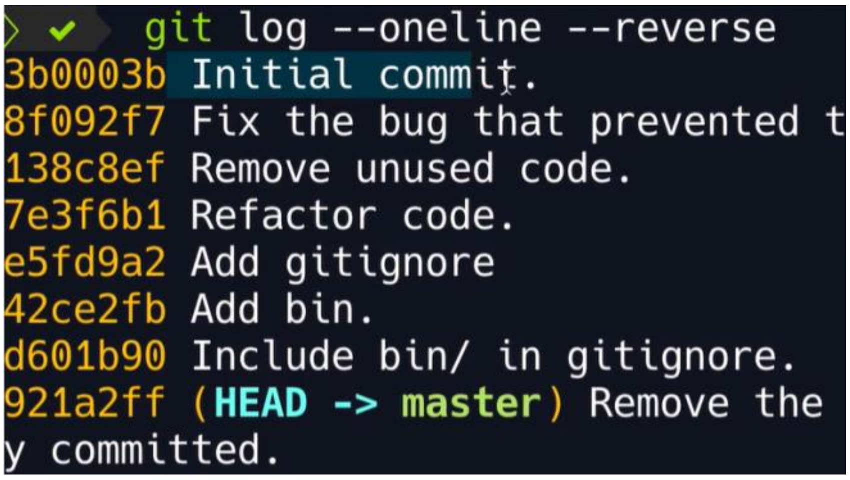
14- Viewing History



```
git log
```


```
commit 921a2ffd7042ed0eb773676509fda87b5c05abf5 (HEAD -> master)
Author: Mosh Hamedani <programmingwithmosh@gmail.com>
Date:   Tue Aug 4 16:56:10 2020 -0700

    Remove the bin directory that was accidentally committed.

commit d601b900c717da53806a7126b3fe587c7ae16d9e
Author: Mosh Hamedani <programmingwithmosh@gmail.com>
Date:   Tue Aug 4 16:52:21 2020 -0700

    Include bin/ in gitignore.
```

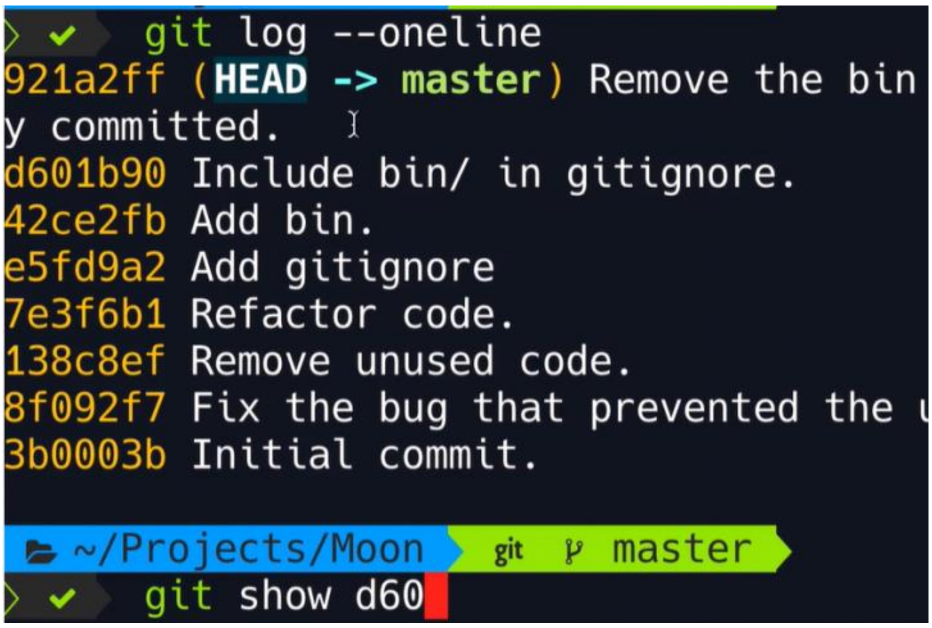

```
> ✔  git log --oneline
921a2ff (HEAD -> master) Remove the bir
y committed.
d601b90 Include bin/ in gitignore.
42ce2fb Add bin.
e5fd9a2 Add gitignore
7e3f6b1 Refactor code.
138c8ef Remove unused code.
8f092f7 Fix the bug that prevented the
3b0003b Initial commit.
```


```
> ✔  git log --oneline --reverse
3b0003b Initial commit.
8f092f7 Fix the bug that prevented t
138c8ef Remove unused code.
7e3f6b1 Refactor code.
e5fd9a2 Add gitignore
42ce2fb Add bin.
d601b90 Include bin/ in gitignore.
921a2ff (HEAD -> master) Remove the
y committed.
```

15- Viewing a Commit

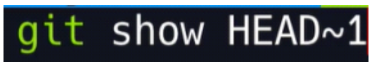
如果想看某一次commit的具体内容：

方法一：只用d60就可以，因为没有其他以d60开头的



```
❯ ✔  git log --oneline
921a2ff (HEAD -> master) Remove the bin
y committed.
d601b90 Include bin/ in gitignore.
42ce2fb Add bin.
e5fd9a2 Add gitignore
7e3f6b1 Refactor code.
138c8ef Remove unused code.
8f092f7 Fix the bug that prevented the u
3b0003b Initial commit.

 ~/Projects/Moon  git  master
❯ ✔  git show d60
```

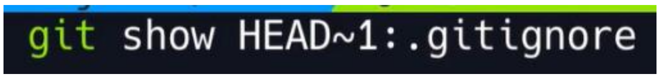
方法二：1代表沿着head往回数几个



```
git show HEAD~1
```

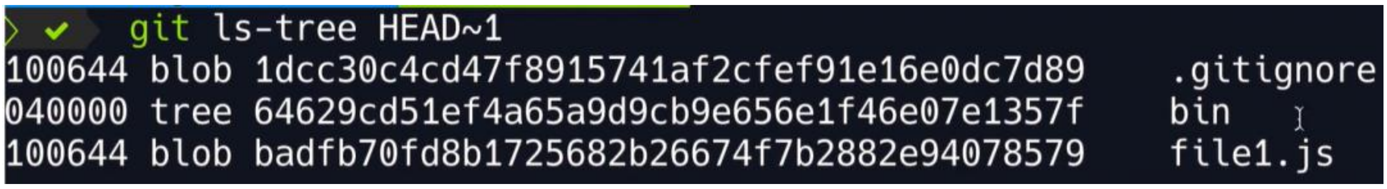
如果我不想看变化diff，想看某次commit中file具体什么样：

看head往前一次中的gitignore这个文件内容



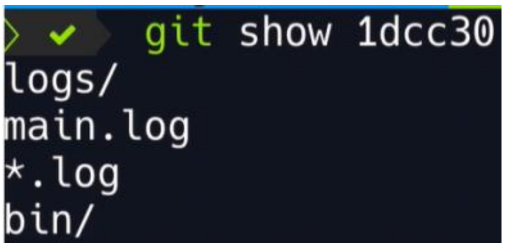
```
git show HEAD~1:.gitignore
```

想查看某次commit中所有file情况：



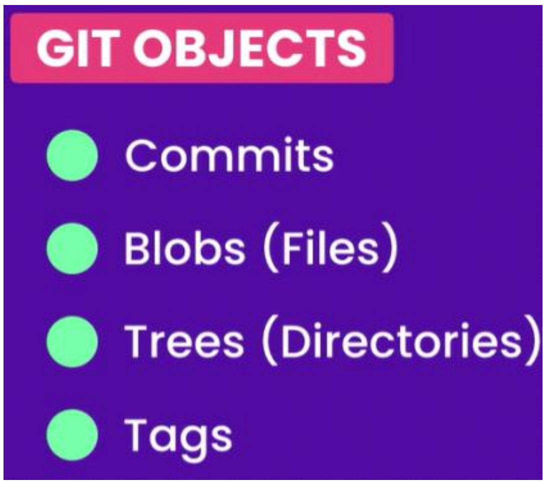
```
❯ ✔  git ls-tree HEAD~1
100644 blob 1dcc30c4cd47f8915741af2cfef91e16e0dc7d89    .gitignore
040000 tree 64629cd51ef4a65a9d9cb9e656e1f46e07e1357f    bin
100644 blob badfb70fd8b1725682b26674f7b2882e94078579    file1.js
```

利用在git数据库中的id，看一个file中的具体内容：



```
❯ ✔  git show 1dcc30
logs/
main.log
*.log
bin/
```

利用show命令可以看几种object：

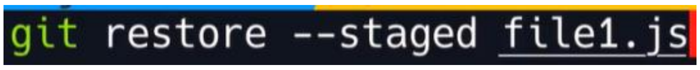


GIT OBJECTS

- Commits
- Blobs (Files)
- Trees (Directories)
- Tags

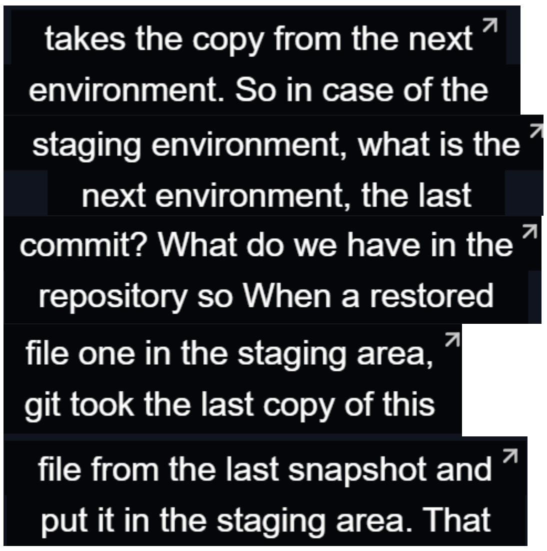
16- Unstaging Files

如果发现某次staged的内容不想commit，可以使用restore命令：



```
git restore --staged file1.js
```

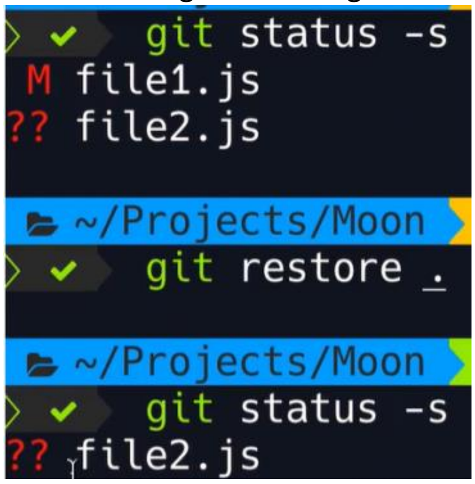
Restore的原理：从next环境中copy



takes the copy from the next environment. So in case of the staging environment, what is the next environment, the last commit? What do we have in the repository so When a restored file one in the staging area, git took the last copy of this file from the last snapshot and put it in the staging area. That

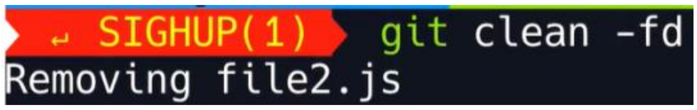
如果是新加的file，repo中没有，使用了restore，restore没法从repo中copy，就会变成？？，untracked file

17- Discarding Local Changes



```
git status -s
 M file1.js
?? file2.js

~/Projects/Moon
git restore .

~/Projects/Moon
git status -s
?? file2.js
```

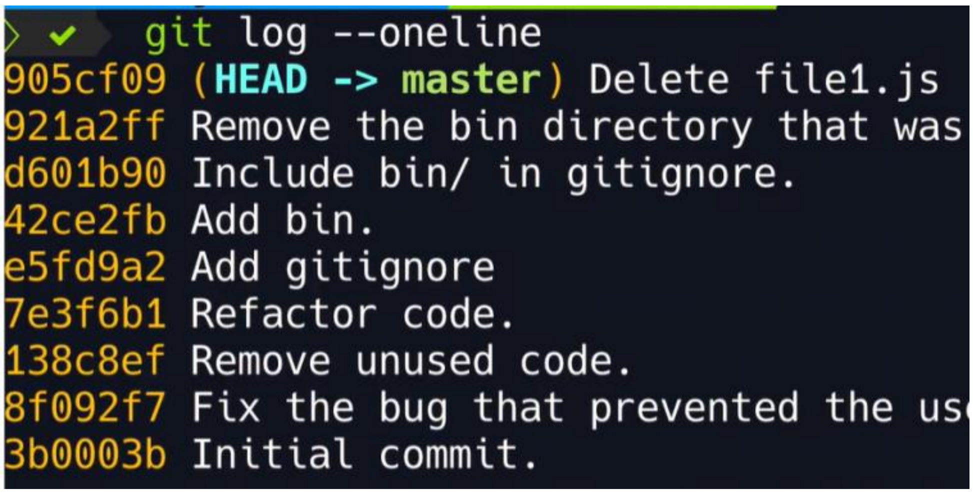
Untracked file依然还在，需要git clean -f fouce -d directory



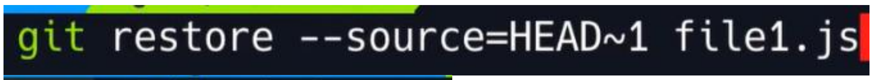
```
SIGHUP(1)  git clean -fd
Removing file2.js
```

18- Restoring a File to an Earlier Version

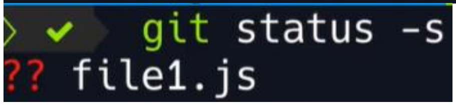
一个已经commit了的deleted的file，想要恢复



```
❯ ✔  git log --oneline
905cf09 (HEAD -> master) Delete file1.js
921a2ff Remove the bin directory that was
d601b90 Include bin/ in gitignore.
42ce2fb Add bin.
e5fd9a2 Add gitignore
7e3f6b1 Refactor code.
138c8ef Remove unused code.
8f092f7 Fix the bug that prevented the us
3b0003b Initial commit.
```

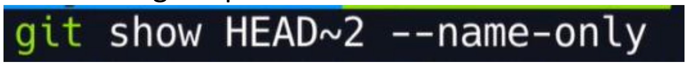


```
git restore --source=HEAD~1 file1.js
```



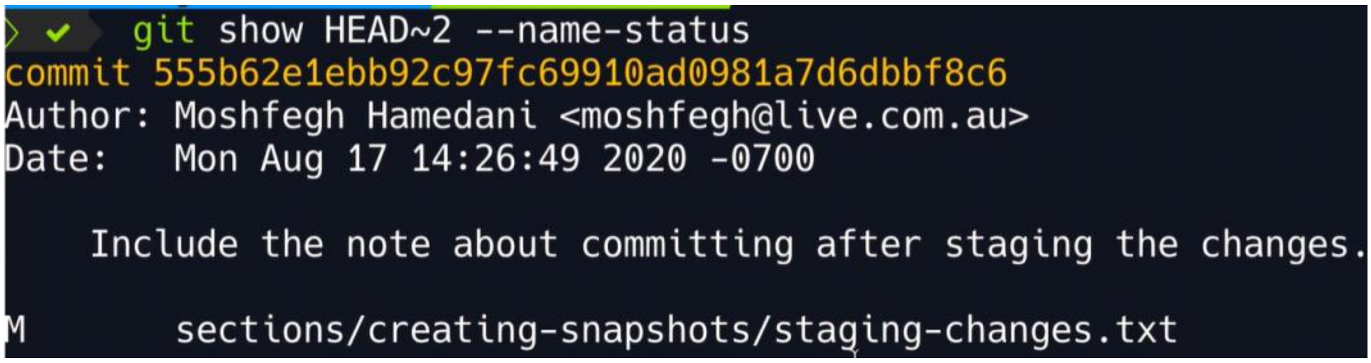
```
❯ ✔  git status -s
?? file1.js
```

19- Creating Snapshots with VSCode



```
git show HEAD~2 --name-only
```

查看文件名only，修改情况，M modified



```
❯ ✔  git show HEAD~2 --name-status
commit 555b62e1ebb92c97fc69910ad0981a7d6dbbf8c6
Author: Moshfegh Hamedani <moshfegh@live.com.au>
Date:   Mon Aug 17 14:26:49 2020 -0700

    Include the note about committing after staging the changes.

M       sections/creating-snapshots/staging-changes.txt
```

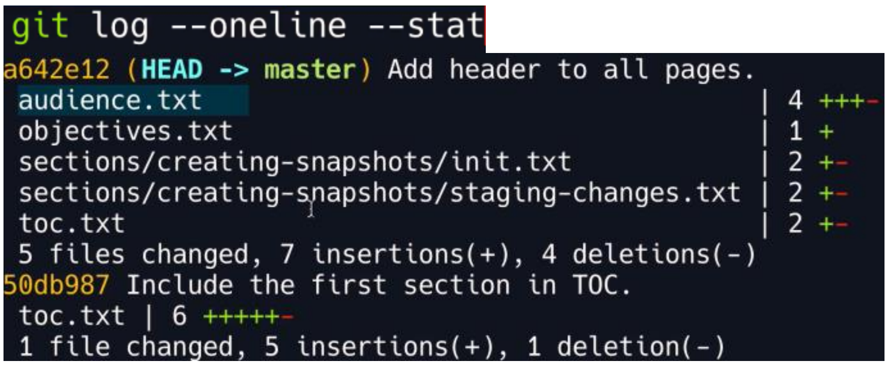
Browsing History

Saturday, April 30, 2022 2:34 PM

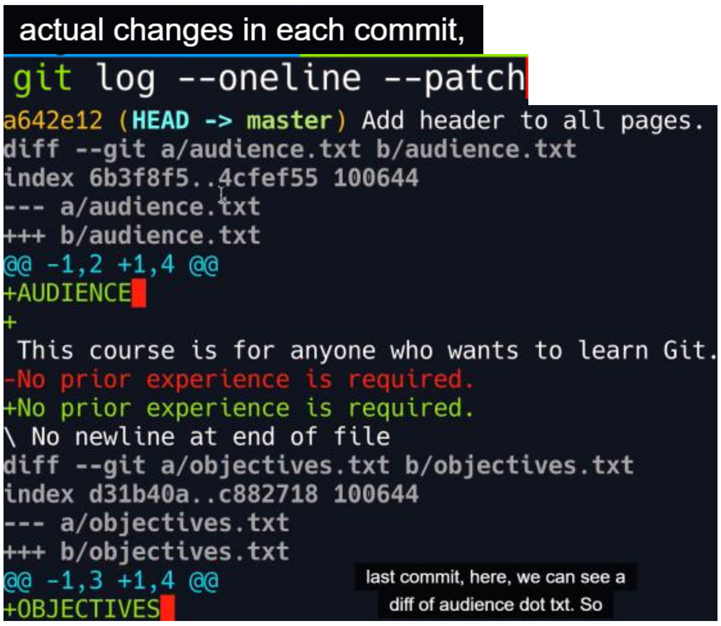
3- Viewing the History

查看所有commit记录：git log，利用space换页，q退出

如果想看每次commit中变化的具体情况



```
git log --oneline --stat
a642e12 (HEAD -> master) Add header to all pages.
 audience.txt                                   | 4 +++-
 objectives.txt                                 | 1 +
 sections/creating-snapshots/init.txt           | 2 +-
 sections/creating-snapshots/staging-changes.txt | 2 +-
 toc.txt                                        | 2 +-
 5 files changed, 7 insertions(+), 4 deletions(-)
50db987 Include the first section in TOC.
 toc.txt | 6 +++++-
 1 file changed, 5 insertions(+), 1 deletion(-)
```



actual changes in each commit,

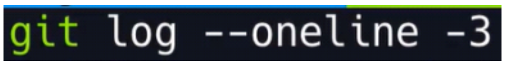
```
git log --oneline --patch
a642e12 (HEAD -> master) Add header to all pages.
diff --git a/audience.txt b/audience.txt
index 6b3f8f5..4cfef55 100644
--- a/audience.txt
+++ b/audience.txt
@@ -1,2 +1,4 @@
+AUDIENCE
+
 This course is for anyone who wants to learn Git.
-No prior experience is required.
+No prior experience is required.
\ No newline at end of file
diff --git a/objectives.txt b/objectives.txt
index d31b40a..c882718 100644
--- a/objectives.txt
+++ b/objectives.txt
@@ -1,3 +1,4 @@
+OBJECTIVES
```

last commit, here, we can see a
diff of audience dot txt. So

4- Filtering the History

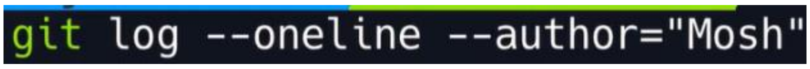
Filter commit

看最近三次



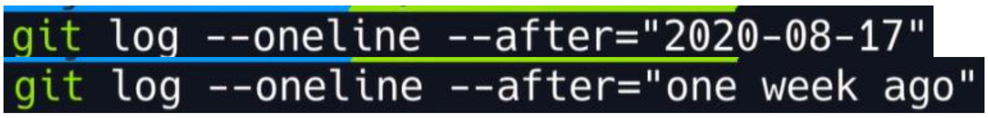
```
git log --oneline -3
```

看特定作者



```
git log --oneline --author="Mosh"
```

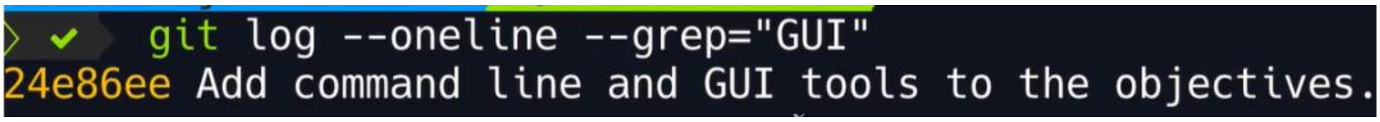
特定时间，yesterday



```
git log --oneline --after="2020-08-17"
git log --oneline --after="one week ago"
```

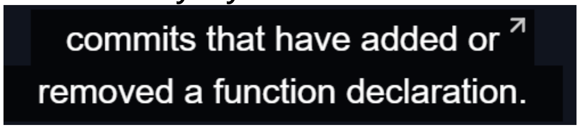
查看commit message包含GUI关键字的，大小写敏感

commits that have the word GUI



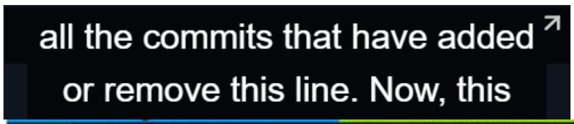
```
git log --oneline --grep="GUI"
24e86ee Add command line and GUI tools to the objectives.
```

Filer history by content

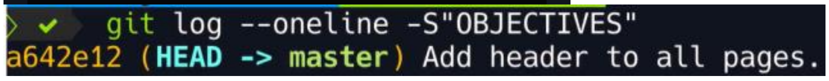


commits that have added or removed a function declaration.

查看添加或者删除了OBJECTIVES这个词的commit

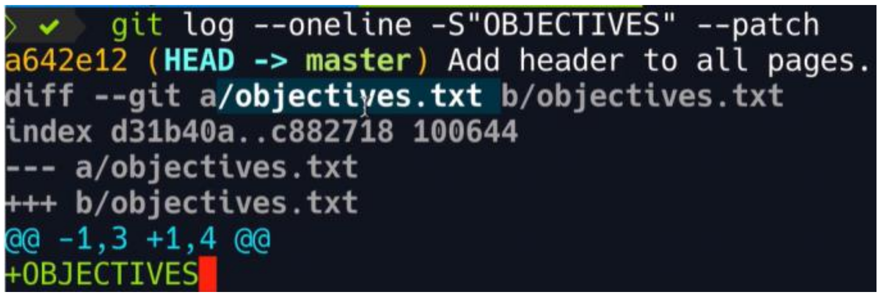


all the commits that have added or remove this line. Now, this



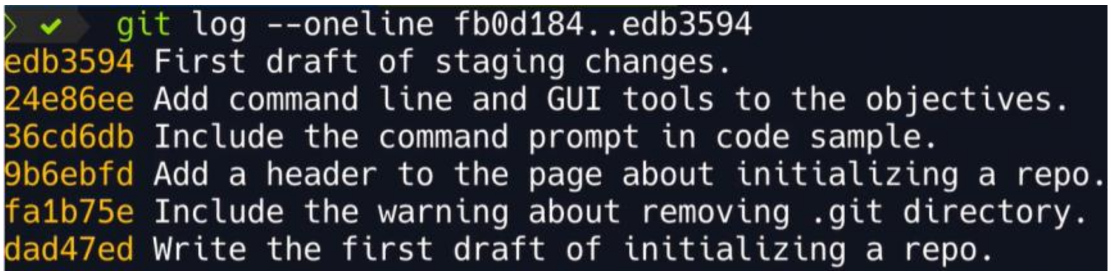
```
git log --oneline -S"OBJECTIVES"
a642e12 (HEAD -> master) Add header to all pages.
```

查看具体内容



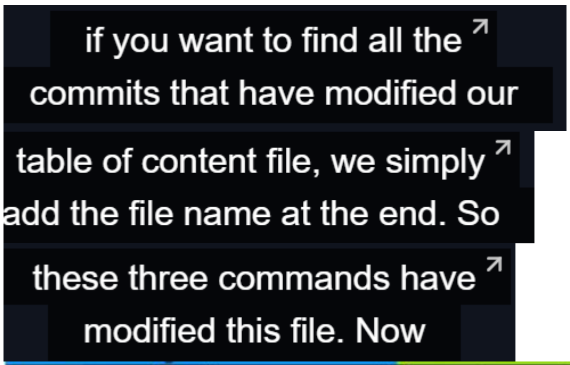
```
git log --oneline -S"OBJECTIVES" --patch
a642e12 (HEAD -> master) Add header to all pages.
diff --git a/objectives.txt b/objectives.txt
index d31b40a..c882718 100644
--- a/objectives.txt
+++ b/objectives.txt
@@ -1,3 +1,4 @@
+OBJECTIVES
```

查看特定的两次commit之间的部分



```
git log --oneline fb0d184..edb3594
edb3594 First draft of staging changes.
24e86ee Add command line and GUI tools to the objectives.
36cd6db Include the command prompt in code sample.
9b6ebfd Add a header to the page about initializing a repo.
fa1b75e Include the warning about removing .git directory.
dad47ed Write the first draft of initializing a repo.
```

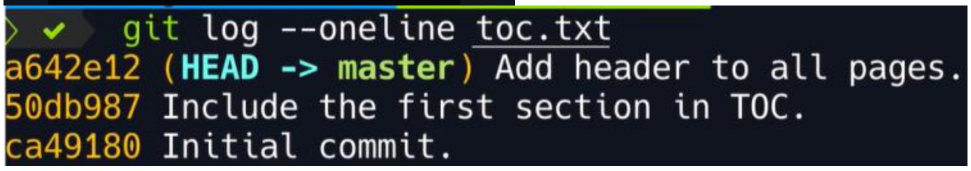
查看哪些commit变化了特定文件



if you want to find all the commits that have modified our

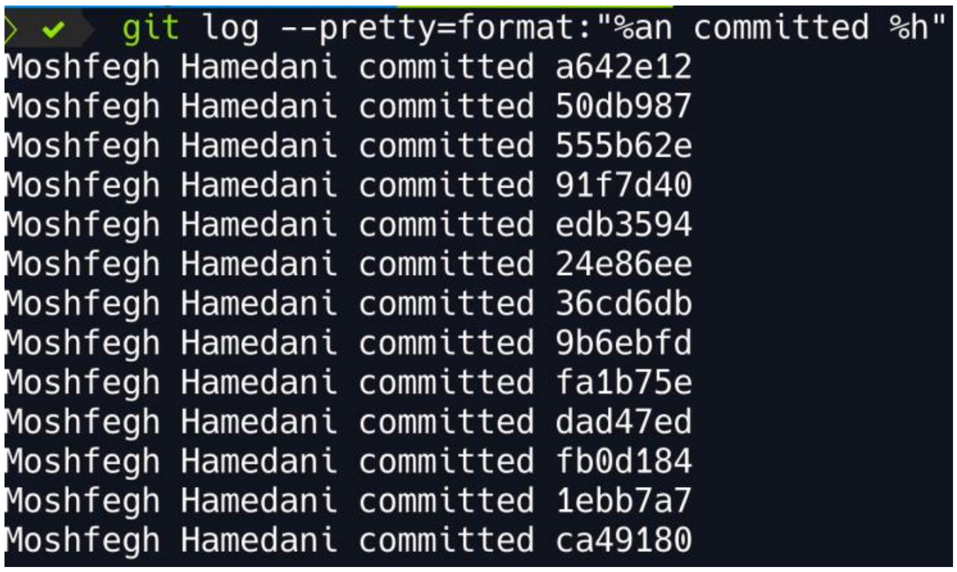
table of content file, we simply add the file name at the end. So

these three commands have modified this file. Now



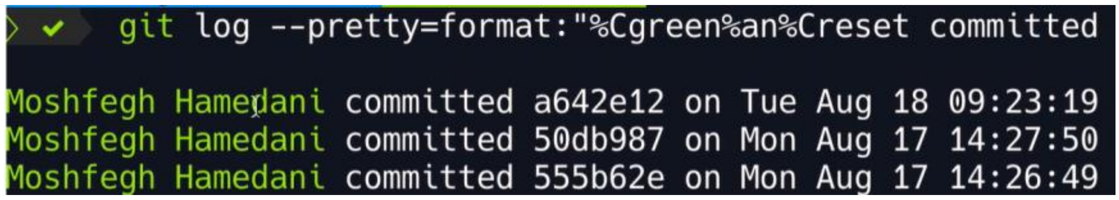
```
git log --oneline toc.txt
a642e12 (HEAD -> master) Add header to all pages.
50db987 Include the first section in TOC.
ca49180 Initial commit.
```

5- Formatting the Log Output
Cd: commit date



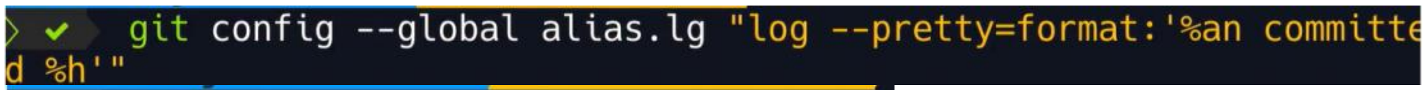
```
git log --pretty=format:"%an committed %h"
Moshfegh Hamedani committed a642e12
Moshfegh Hamedani committed 50db987
Moshfegh Hamedani committed 555b62e
Moshfegh Hamedani committed 91f7d40
Moshfegh Hamedani committed edb3594
Moshfegh Hamedani committed 24e86ee
Moshfegh Hamedani committed 36cd6db
Moshfegh Hamedani committed 9b6ebfd
Moshfegh Hamedani committed fa1b75e
Moshfegh Hamedani committed dad47ed
Moshfegh Hamedani committed fb0d184
Moshfegh Hamedani committed 1ebb7a7
Moshfegh Hamedani committed ca49180
```

变颜色

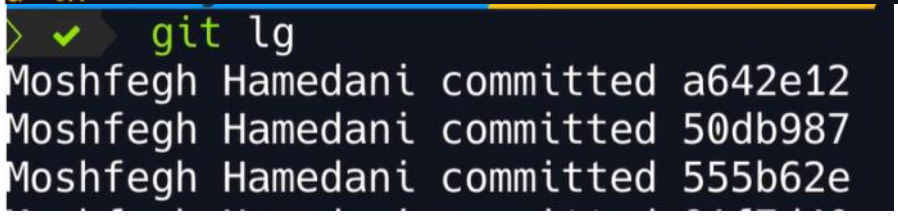


```
git log --pretty=format:"%Cgreen%an%Creset committed

Moshfegh Hamedani committed a642e12 on Tue Aug 18 09:23:19
Moshfegh Hamedani committed 50db987 on Mon Aug 17 14:27:50
Moshfegh Hamedani committed 555b62e on Mon Aug 17 14:26:49
```

6- Aliases

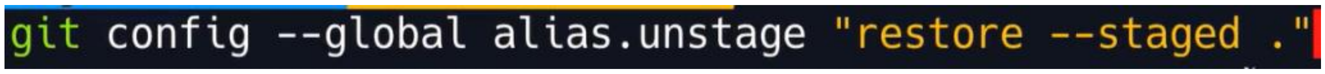


```
git config --global alias.lg "log --pretty=format:'%an committe
d %h'"
```



```
git lg
Moshfegh Hamedani committed a642e12
Moshfegh Hamedani committed 50db987
Moshfegh Hamedani committed 555b62e
```

非常有用的别名：



```
git config --global alias.unstage "restore --staged ."
```

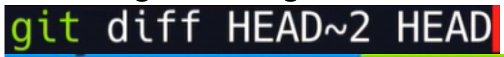
7- Viewing a Commit

Log vs. show vs. diff

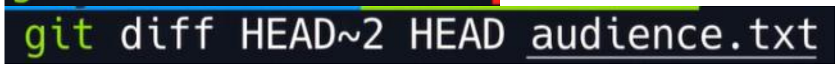
https://www.cnblogs.com/little-ab/p/11406321.html

https://blog.csdn.net/weixin_40633131/article/details/78242210

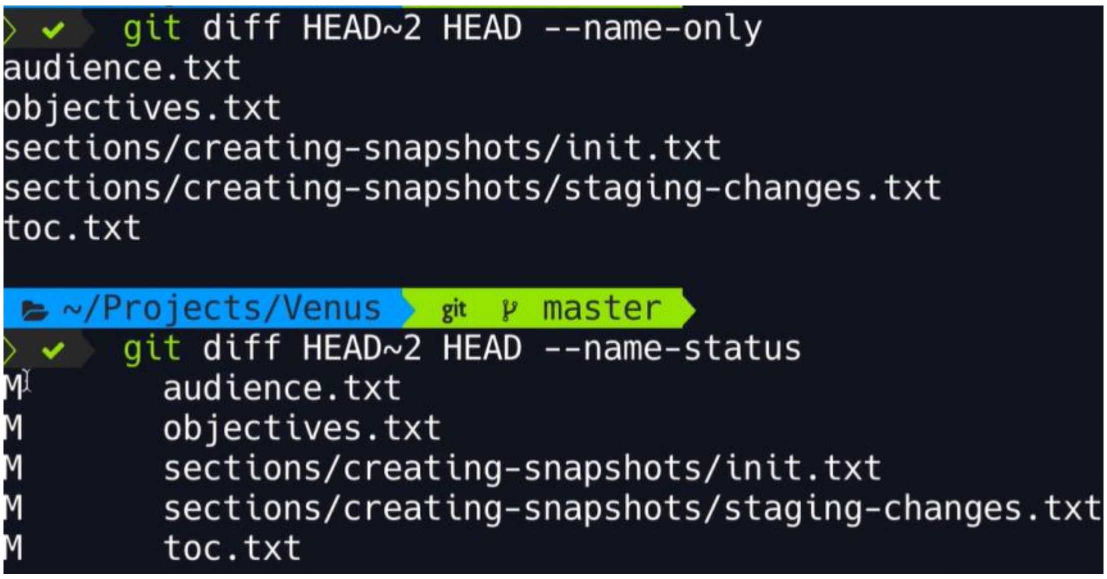
8- Viewing the Changes Across Commits



```
git diff HEAD~2 HEAD
```



```
git diff HEAD~2 HEAD audience.txt
```



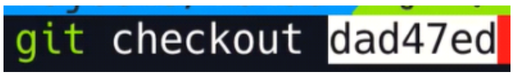
```
❯ ✓  git diff HEAD~2 HEAD --name-only
audience.txt
objectives.txt
sections/creating-snapshots/init.txt
sections/creating-snapshots/staging-changes.txt
toc.txt

 ~/Projects/Venus  git  master
❯ ✓  git diff HEAD~2 HEAD --name-status
M       audience.txt
M       objectives.txt
M       sections/creating-snapshots/init.txt
M       sections/creating-snapshots/staging-changes.txt
M       toc.txt
```

9- Checking Out a Commit

checkout命令

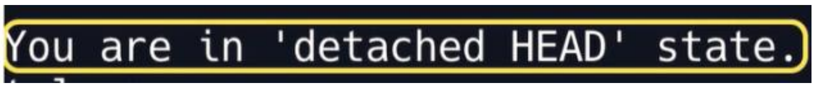
https://juejin.cn/post/6978436693485420575

有时我们想把working directory与之前的一次commit链接，看看有什么diff，需要移动head。



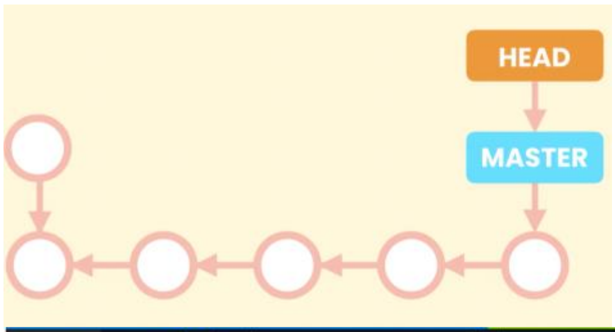
```
git checkout dad47ed
```

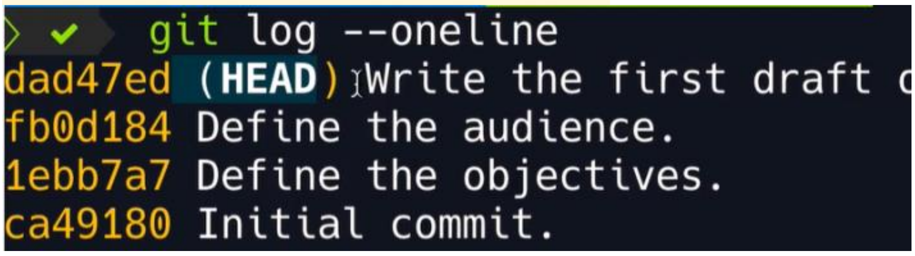
通常head指向master，master指向最后一次commit。



```
You are in 'detached HEAD' state.
```

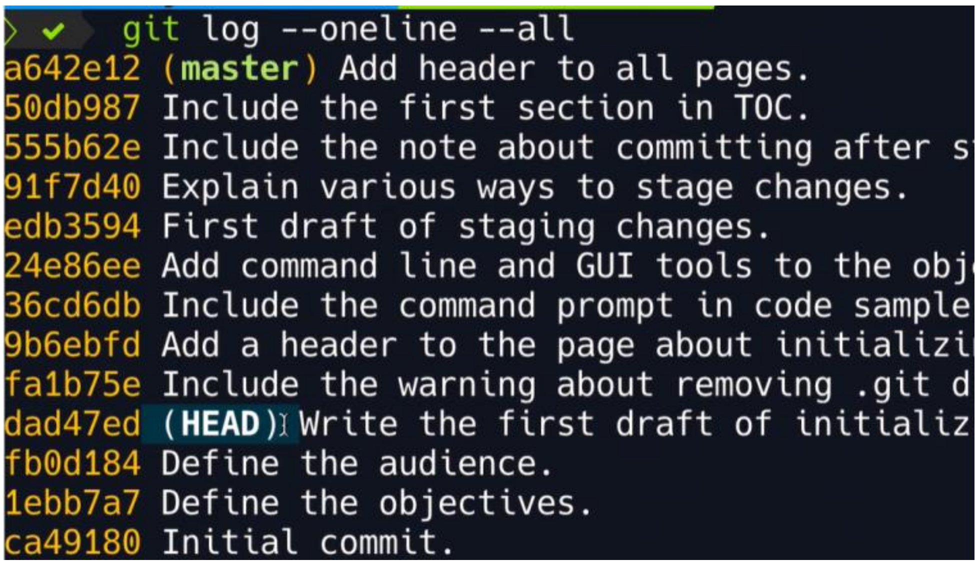
detached意味着，你可以查看，但是这时候commit，这次的commit的内容将无法被point到





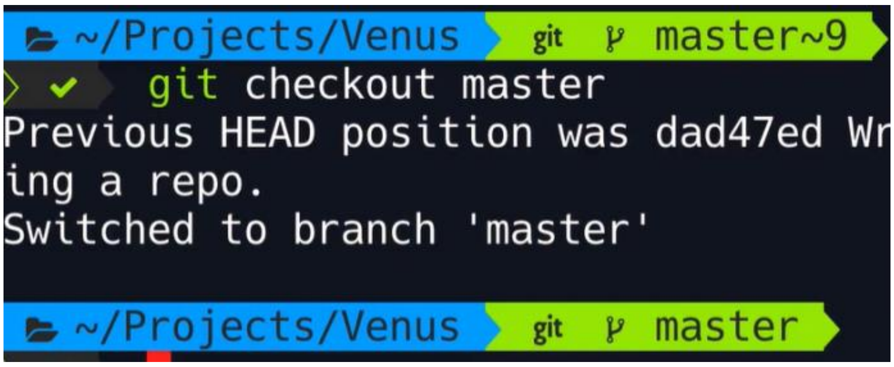
```
❯ ✓  git log --oneline
dad47ed (HEAD) Write the first draft o
fb0d184 Define the audience.
1ebb7a7 Define the objectives.
ca49180 Initial commit.
```

主意下图中的master和head的位置



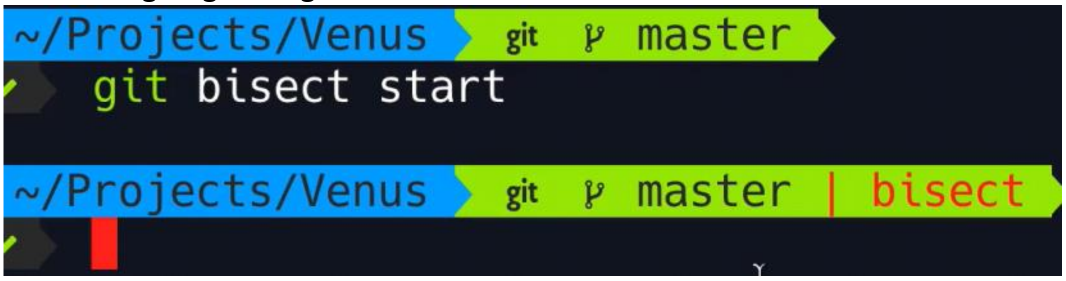
```
git log --oneline --all
a642e12 (master) Add header to all pages.
50db987 Include the first section in TOC.
555b62e Include the note about committing after s
91f7d40 Explain various ways to stage changes.
edb3594 First draft of staging changes.
24e86ee Add command line and GUI tools to the obj
36cd6db Include the command prompt in code sample
9b6ebfd Add a header to the page about initializi
fa1b75e Include the warning about removing .git d
dad47ed (HEAD) Write the first draft of initializ
fb0d184 Define the audience.
1ebb7a7 Define the objectives.
ca49180 Initial commit.
```

返回master



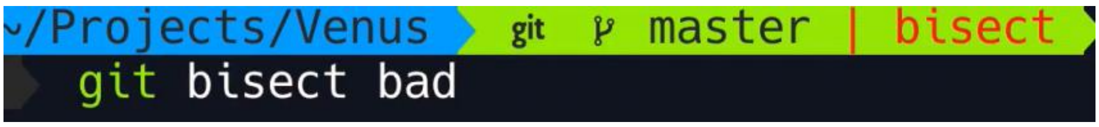
```
~/Projects/Venus  git  master~9
git checkout master
Previous HEAD position was dad47ed Wr
ing a repo.
Switched to branch 'master'

~/Projects/Venus  git  master
```

10- Finding Bugs Using Bisect

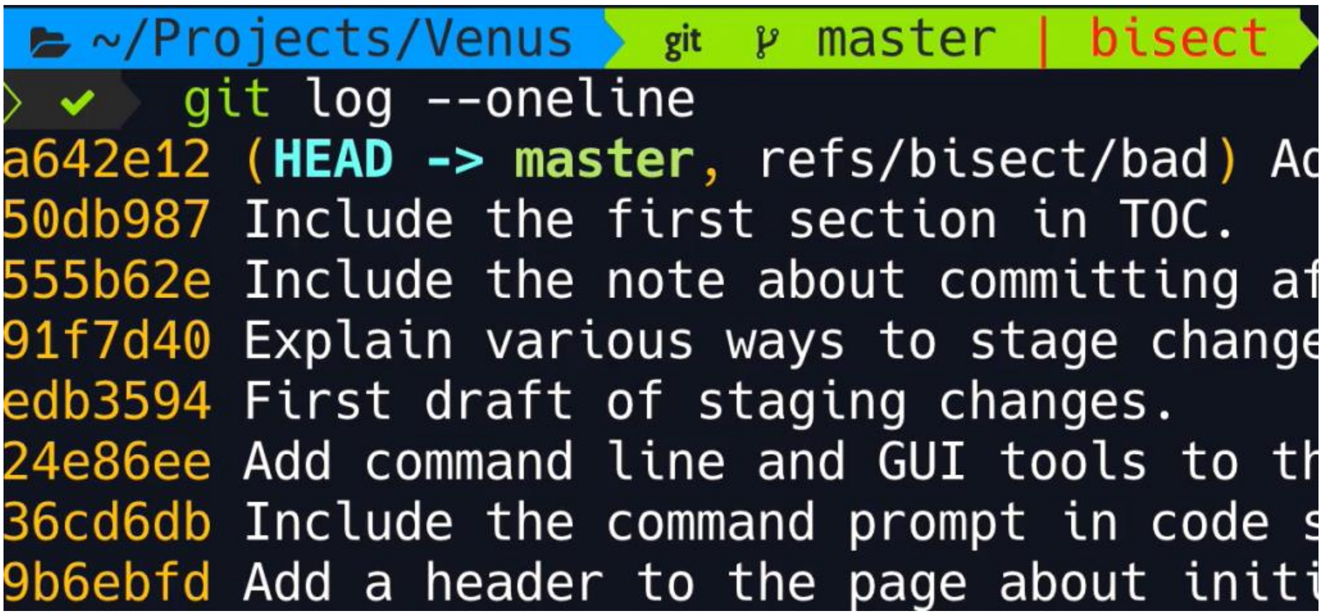


```
~/Projects/Venus  git  master
git bisect start

~/Projects/Venus  git  master | bisect
```

master是bad的

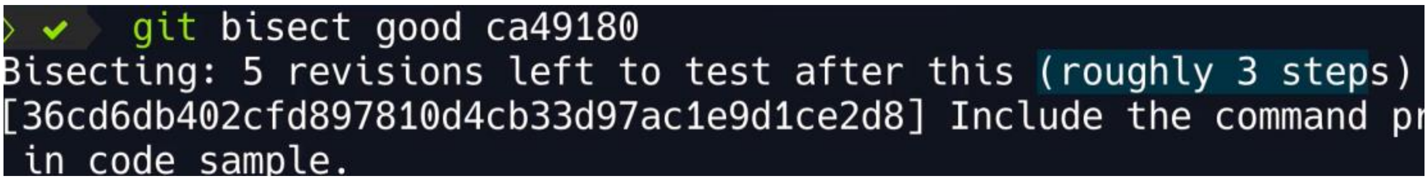


```
~/Projects/Venus  git  master | bisect
git bisect bad
```

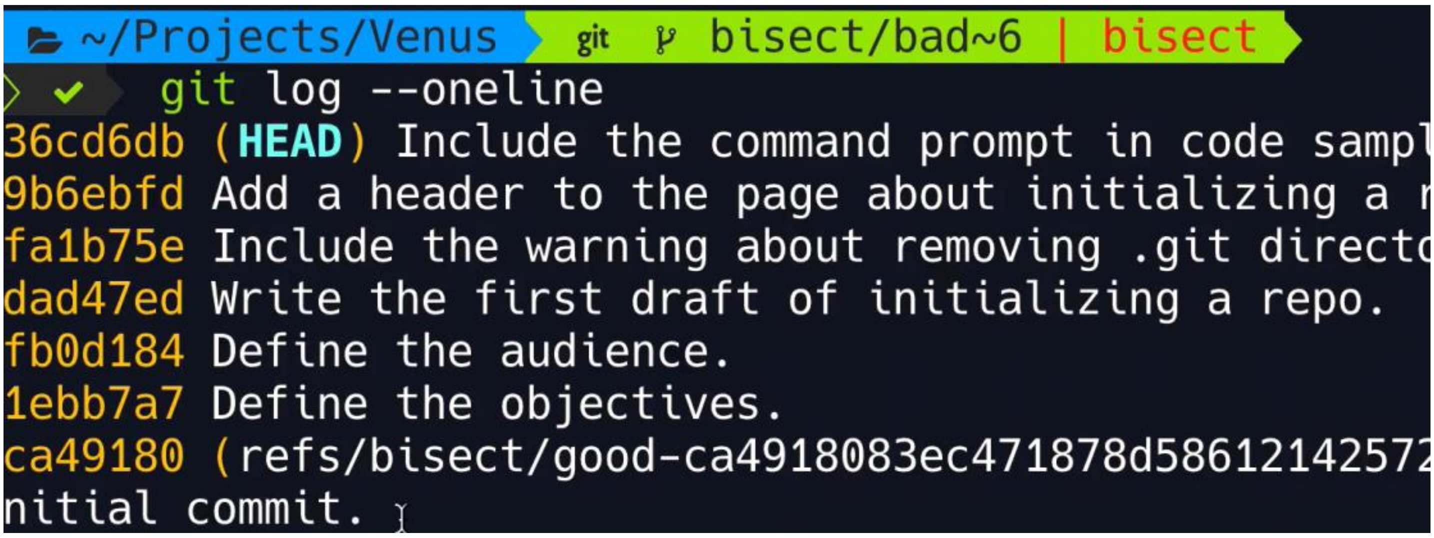


```
~/Projects/Venus  git  master | bisect
›  ✔   git log --oneline
a642e12 (HEAD -> master, refs/bisect/bad) Ac
50db987 Include the first section in TOC.
555b62e Include the note about committing af
91f7d40 Explain various ways to stage change
edb3594 First draft of staging changes.
24e86ee Add command line and GUI tools to th
36cd6db Include the command prompt in code s
9b6ebfd Add a header to the page about initi
```

某一次定为good的

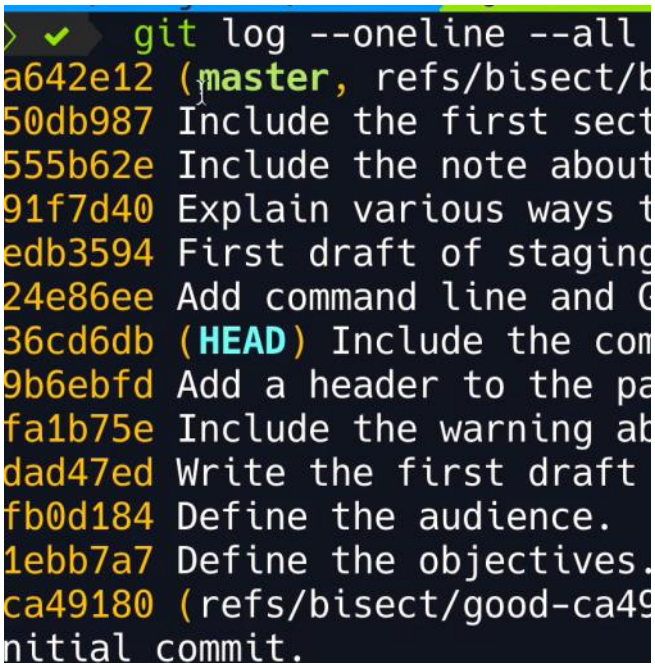


```
›  ✔   git bisect good ca49180
Bisecting: 5 revisions left to test after this (roughly 3 steps)
[36cd6db402cfd897810d4cb33d97ac1e9d1ce2d8] Include the command pr
 in code sample.
```



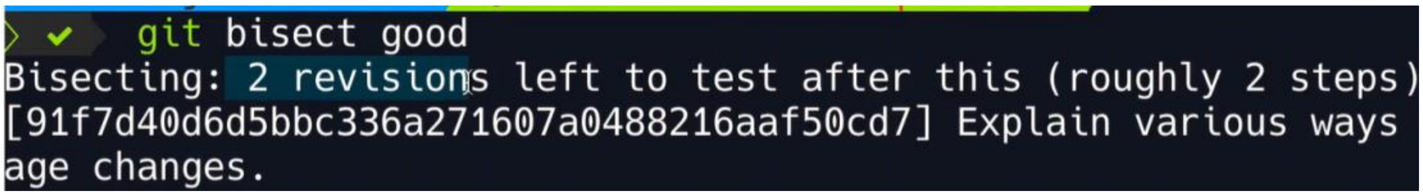
```
~/Projects/Venus  git  bisect/bad~6 | bisect
›  ✔   git log --oneline
36cd6db (HEAD) Include the command prompt in code sampl
9b6ebfd Add a header to the page about initializing a r
fa1b75e Include the warning about removing .git directo
dad47ed Write the first draft of initializing a repo.
fb0d184 Define the audience.
1ebb7a7 Define the objectives.
ca49180 (refs/bisect/good-ca4918083ec471878d58612142572
nitial commit.
```

Master 是bad，head在中间，最下面是good，head会自动挪到正中间。



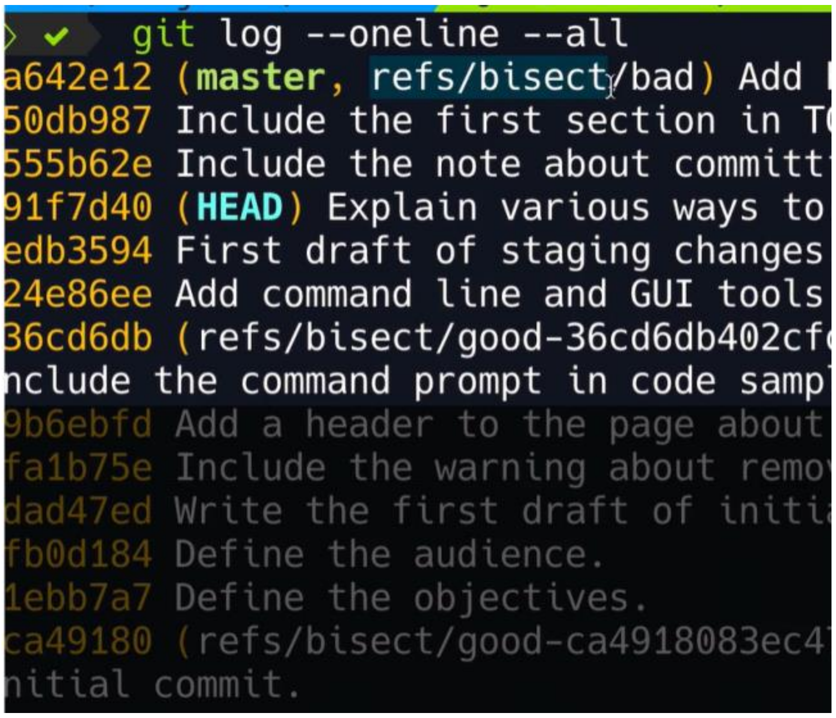
```
❯ ✔  git log --oneline --all
a642e12 (master, refs/bisect/b
50db987 Include the first sect
555b62e Include the note about
91f7d40 Explain various ways t
edb3594 First draft of staging
24e86ee Add command line and G
36cd6db (HEAD) Include the com
9b6ebfd Add a header to the pa
fa1b75e Include the warning ab
dad47ed Write the first draft
fb0d184 Define the audience.
1ebb7a7 Define the objectives.
ca49180 (refs/bisect/good-ca49
nitial commit.
```

Run head的代码，如果good，就下图：



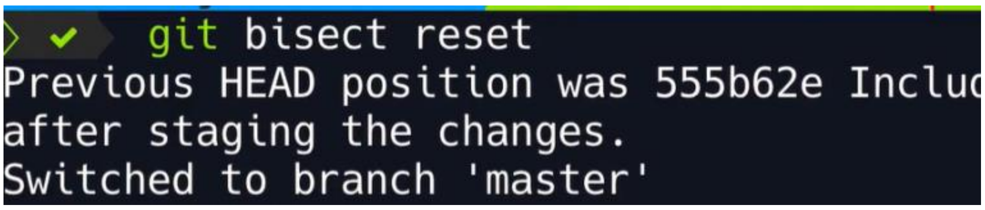
```
❯ ✔  git bisect good
Bisecting: 2 revisions left to test after this (roughly 2 steps)
[91f7d40d6d5bbc336a271607a0488216aaf50cd7] Explain various ways
age changes.
```

这个时候，head就会上移



```
❯ ✔  git log --oneline --all
a642e12 (master, refs/bisect/bad) Add
50db987 Include the first section in T
555b62e Include the note about committ
91f7d40 (HEAD) Explain various ways to
edb3594 First draft of staging changes
24e86ee Add command line and GUI tools
36cd6db (refs/bisect/good-36cd6db402cf
nclude the command prompt in code samp
9b6ebfd Add a header to the page about
fa1b75e Include the warning about remo
dad47ed Write the first draft of initi
fb0d184 Define the audience.
1ebb7a7 Define the objectives.
ca49180 (refs/bisect/good-ca4918083ec4
nitial commit.
```

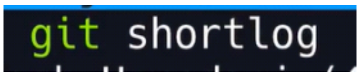
找到了之后：



```
❯ ✔  git bisect reset
Previous HEAD position was 555b62e Incluc
after staging the changes.
Switched to branch 'master'
```

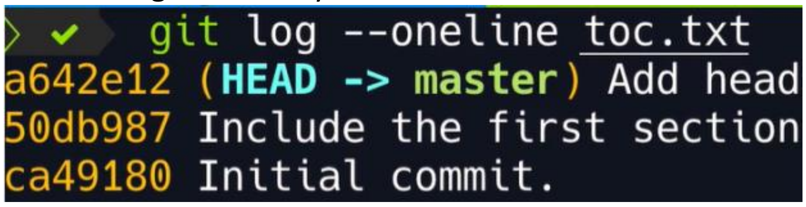
11- Finding Contributors Using Shortlog

每个用户commit的次数和每次commit的注释



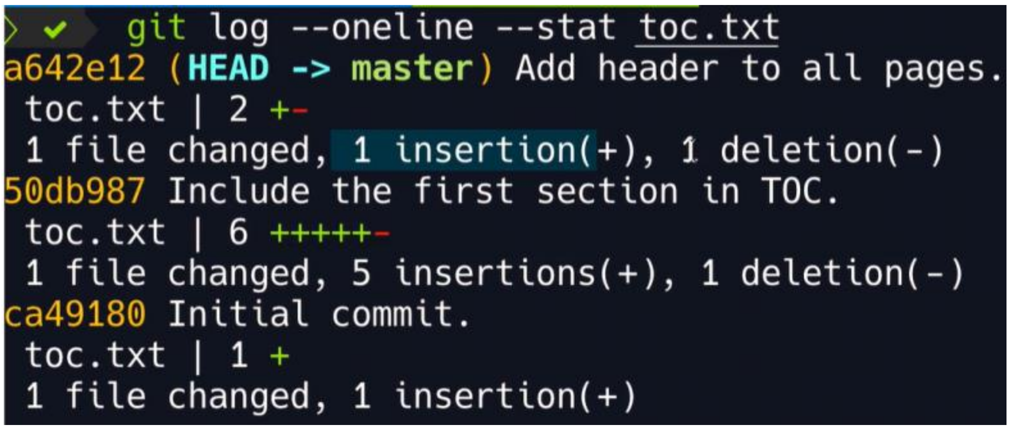
```
git shortlog
```

12- Viewing the History of a File



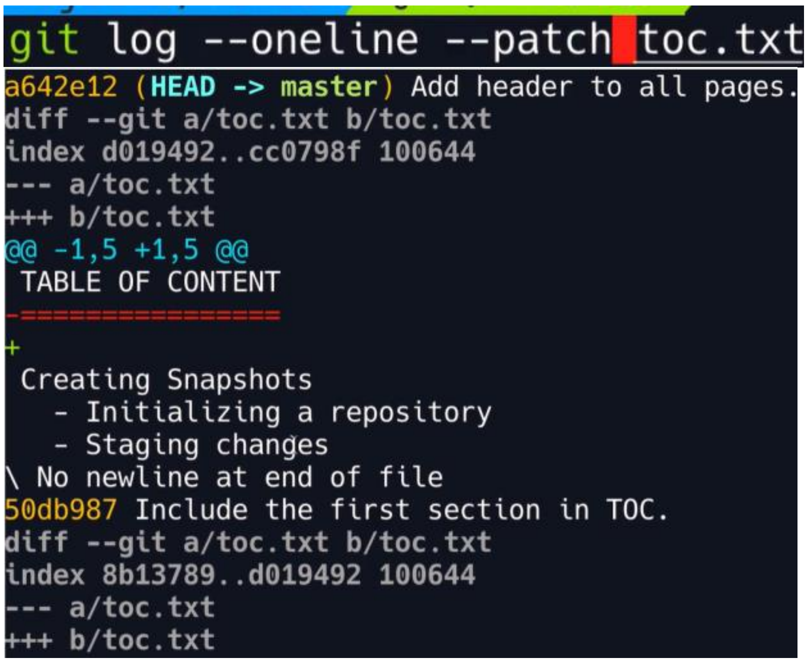
```
git log --oneline toc.txt
a642e12 (HEAD -> master) Add head
50db987 Include the first section
ca49180 Initial commit.
```

--stat具体情况



```
git log --oneline --stat toc.txt
a642e12 (HEAD -> master) Add header to all pages.
 toc.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
50db987 Include the first section in TOC.
 toc.txt | 6 +++++-
 1 file changed, 5 insertions(+), 1 deletion(-)
ca49180 Initial commit.
 toc.txt | 1 +
 1 file changed, 1 insertion(+)
```

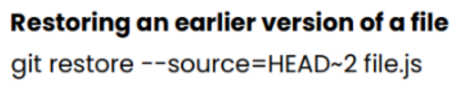
更具体：patch



```
git log --oneline --patch toc.txt
a642e12 (HEAD -> master) Add header to all pages.
diff --git a/toc.txt b/toc.txt
index d019492..cc0798f 100644
--- a/toc.txt
+++ b/toc.txt
@@ -1,5 +1,5 @@
 TABLE OF CONTENT
-================
+
 Creating Snapshots
   - Initializing a repository
   - Staging changes
\ No newline at end of file
50db987 Include the first section in TOC.
diff --git a/toc.txt b/toc.txt
index 8b13789..d019492 100644
--- a/toc.txt
+++ b/toc.txt
```

13- Restoring a Deleting File

使用restore不仅可以恢复一个已经删除的文件，还可以恢复当时文件的内容，那个commit的时候的那个文件就会出现在working directory

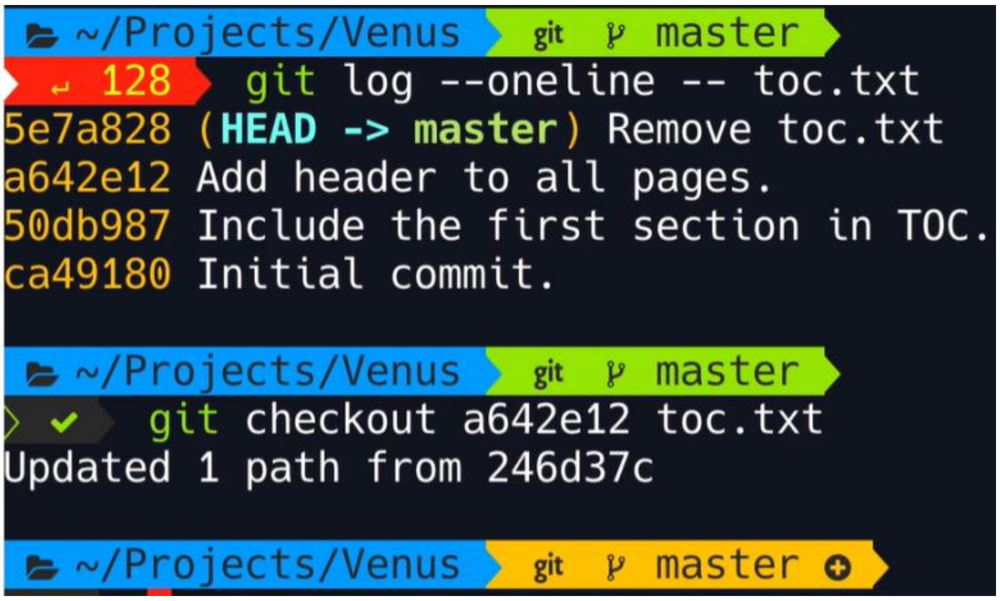


Restoring an earlier version of a file

```
git restore --source=HEAD~2 file.js
```

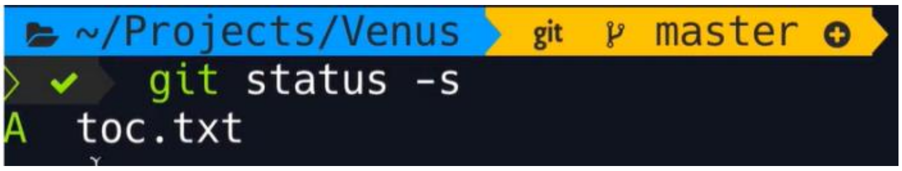
想要回复一个已经删除的文件，方法二：

文件名前面要用--隔开，最后一次commit删除了toc.txt文件，所以定位到a642e12。执行了checkout特定文件后绿色会变成黄色



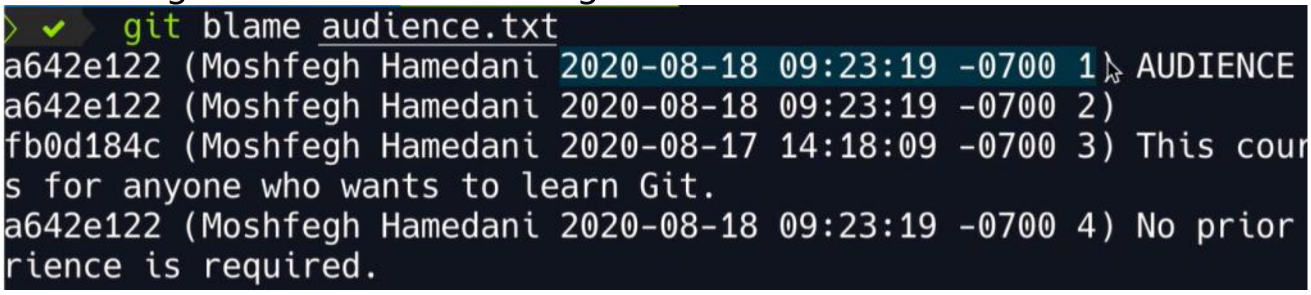
```
~/Projects/Venus  git  master
↵ 128  git log --oneline -- toc.txt
5e7a828 (HEAD -> master) Remove toc.txt
a642e12 Add header to all pages.
50db987 Include the first section in TOC.
ca49180 Initial commit.

~/Projects/Venus  git  master
✔  git checkout a642e12 toc.txt
Updated 1 path from 246d37c

~/Projects/Venus  git  master ⊕
```

以上会令working directory变dirty，查看会发现多了一个file add



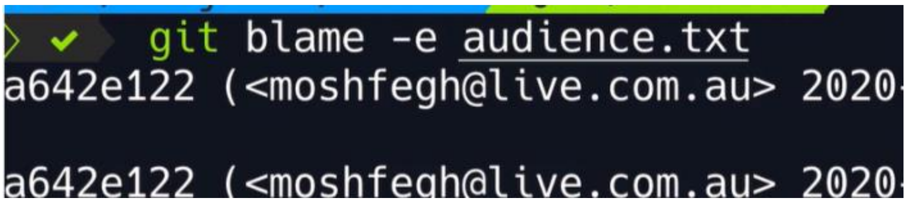
```
~/Projects/Venus  git  master ⊕
✔  git status -s
A  toc.txt
```

14- Finding the Author of Line Using Blame



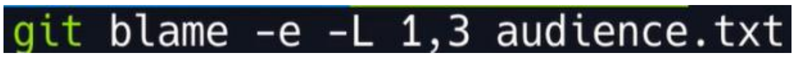
```
✔  git blame audience.txt
a642e122 (Moshfegh Hamedani 2020-08-18 09:23:19 -0700 1) AUDIENCE
a642e122 (Moshfegh Hamedani 2020-08-18 09:23:19 -0700 2)
fb0d184c (Moshfegh Hamedani 2020-08-17 14:18:09 -0700 3) This cour
s for anyone who wants to learn Git.
a642e122 (Moshfegh Hamedani 2020-08-18 09:23:19 -0700 4) No prior
rience is required.
```

email



```
✔  git blame -e audience.txt
a642e122 (<moshfegh@live.com.au> 2020

a642e122 (<moshfegh@live.com.au> 2020
```

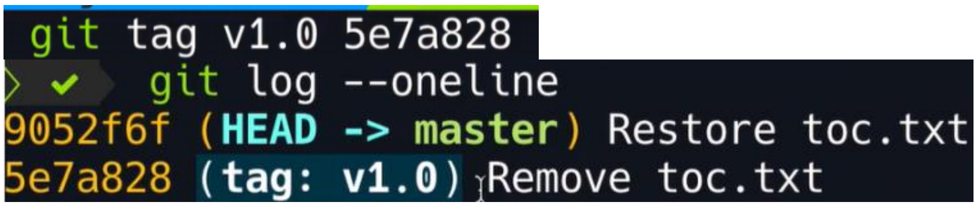
前三条



```
git blame -e -L 1,3 audience.txt
```

15- Tagging

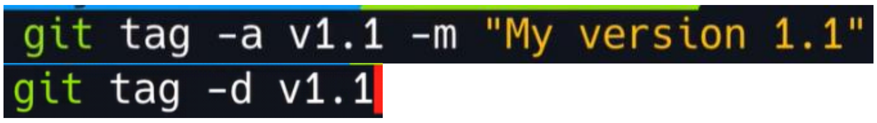
一个tag只能用于一个commit，主要用于标志记录，一眼就可以看到一个特定的commit



```
git tag v1.0 5e7a828
✔  git log --oneline
9052f6f (HEAD -> master) Restore toc.txt
5e7a828 (tag: v1.0) Remove toc.txt
```

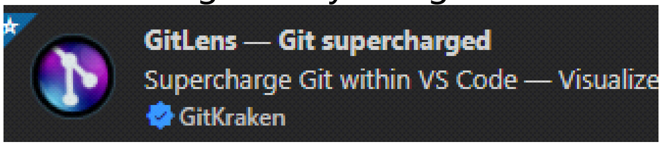
Annotated tag

加-m很难看出message的具体内容



```
git tag -a v1.1 -m "My version 1.1"
git tag -d v1.1
```

16- Browsing History Using VSCode



GitLens — Git supercharged
Supercharge Git within VS Code — Visualize
GitKraken

Branching

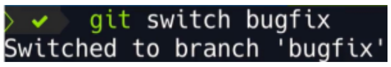
Sunday, May 1, 2022 9:09 PM

2- What are Branches

每个branch指向一个commit，切换branch，就是切换那个commit的内容，到当前的working directory。Current branch利用head指向。

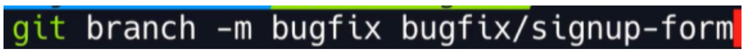
4- Working with Branches

checkout用法过多，容易产生歧义，使用switch转换branch



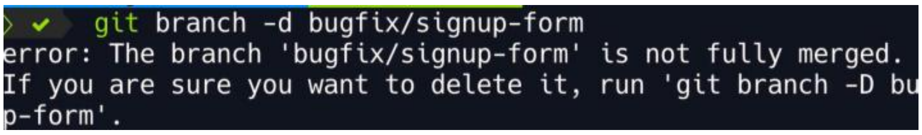
```
> ✔  git switch bugfix
Switched to branch 'bugfix'
```

Rename branch

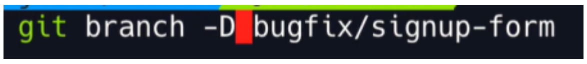


```
git branch -m bugfix bugfix/signup-form
```

删除一个branch，如果没merge会禁止删除，可以使用-D大写强制删除



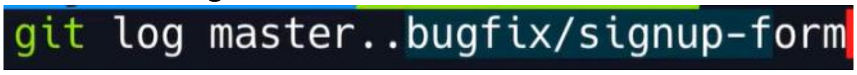
```
> ✔  git branch -d bugfix/signup-form
error: The branch 'bugfix/signup-form' is not fully merged.
If you are sure you want to delete it, run 'git branch -D bu
p-form'.
```



```
git branch -D bugfix/signup-form
```

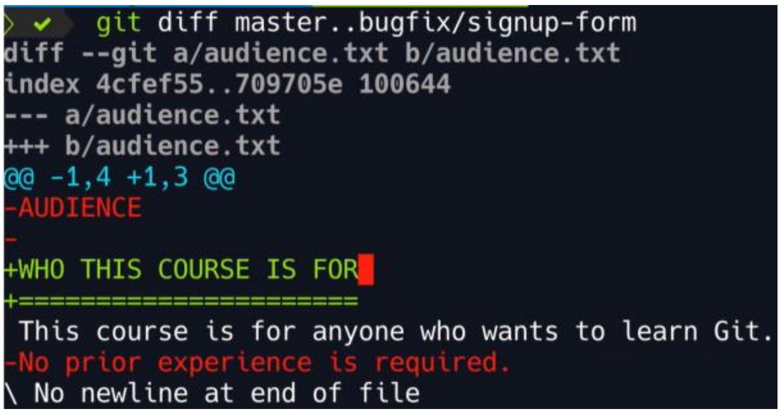
5- Comparing Branches

Commits in bug...branch and not in master branch



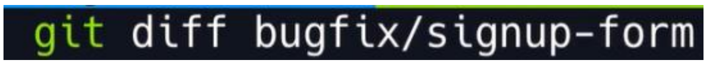
```
git log master..bugfix/signup-form
```

具体不同之处



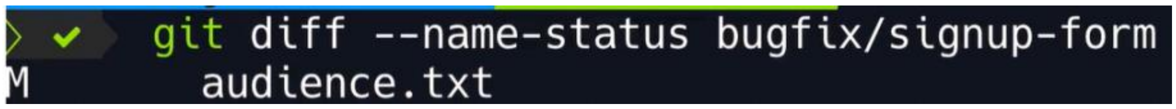
```
> ✔  git diff master..bugfix/signup-form
diff --git a/audience.txt b/audience.txt
index 4cfef55..709705e 100644
--- a/audience.txt
+++ b/audience.txt
@@ -1,4 +1,3 @@
-AUDIENCE
-
+WHO THIS COURSE IS FOR
+======================
 This course is for anyone who wants to learn Git.
-No prior experience is required.
\ No newline at end of file
```

在master branch中，所以可以直接简写成一下：



```
git diff bugfix/signup-form
```

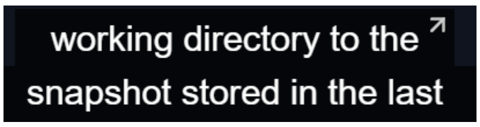
只想看文件名--name-only，想看文件名和修改状态--name-status



```
> ✔  git diff --name-status bugfix/signup-form
M       audience.txt
```

6- Stashing

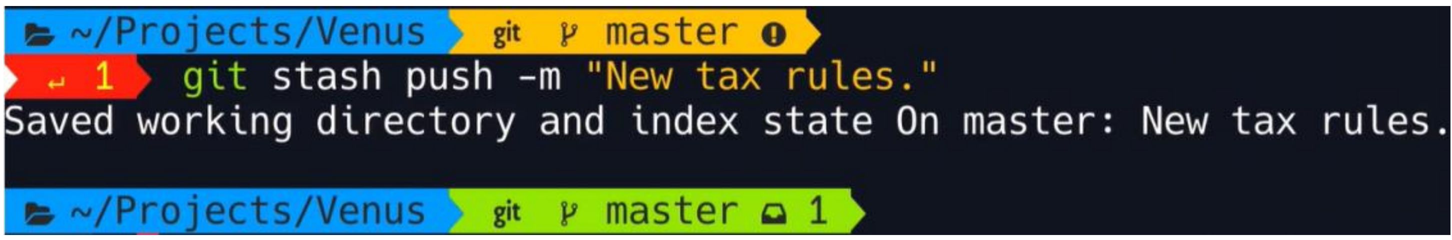
Switch branch是将我们的working directory 变到



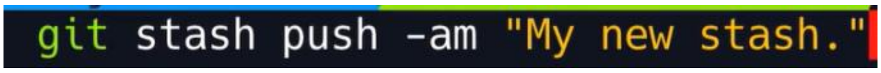
working directory to the
snapshot stored in the last

commit of the target branch. ↗

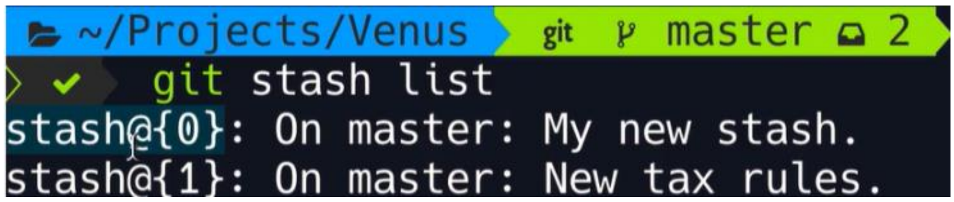
如果我们当前的working directory还灭有commit，系统不会允许我们switch branch。如果我们不想commit changes，我们就需要找另一个地方保存这些变化。使用stash，


```
~/Projects/Venus  git  master
↵ 1  git stash push -m "New tax rules."
Saved working directory and index state On master: New tax rules.

~/Projects/Venus  git  master  1
```

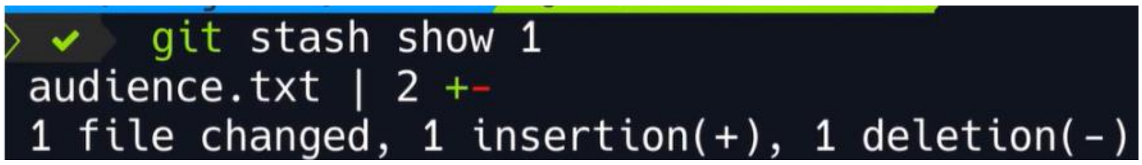
但是untracked的file不会包括在里面，需要使用--all或者-a


```
git stash push -am "My new stash."
```

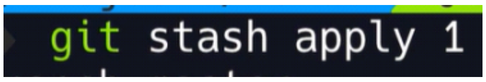
一共有多少stash

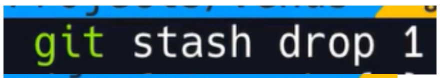

```
~/Projects/Venus  git  master  2
 ✔  git stash list
stash@{0}: On master: My new stash.
stash@{1}: On master: New tax rules.
```

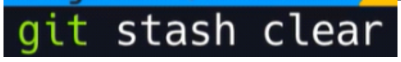
查看index是1 的stash的具体变化


```
 ✔  git stash show 1
 audience.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

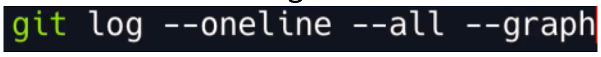
使用index是1的stash


```
 git stash apply 1
```

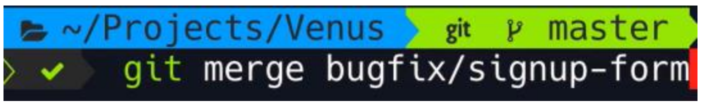

```
git stash drop 1
```


```
git stash clear
```

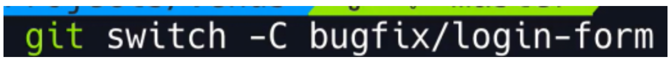
8- Fast-forward Merges

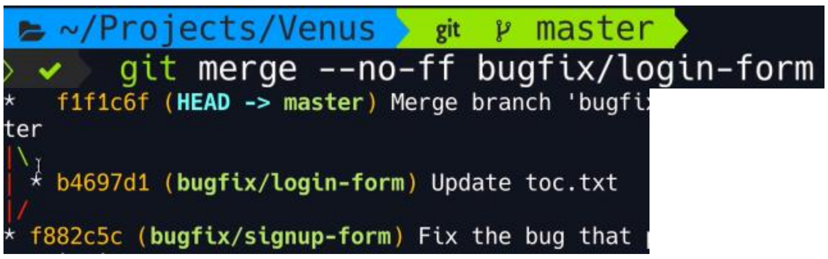

```
git log --oneline --all --graph
```

master上发展出来bugfix branch，之后master没有commit，现在想做merge，目前正处于master，利用以下命令做fast forward merge

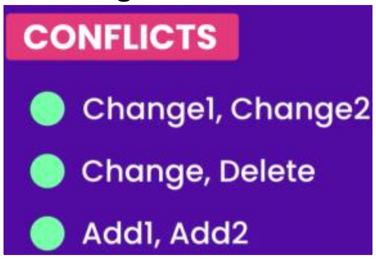

```
~/Projects/Venus  git  master
 ✔  git merge bugfix/signup-form
```

Create and switch branch -C


```
git switch -C bugfix/login-form
```


```
~/Projects/Venus  git  master
 ✔  git merge --no-ff bugfix/login-form
*   f1f1c6f (HEAD -> master) Merge branch 'bugfi
ter
|\
| * b4697d1 (bugfix/login-form) Update toc.txt
|/
* f882c5c (bugfix/signup-form) Fix the bug that
```

11- Merge Conflicts



CONFLICTS

- Change1, Change2
- Change, Delete
- Add1, Add2

merge后会显示conflict，显示在merge中间状态

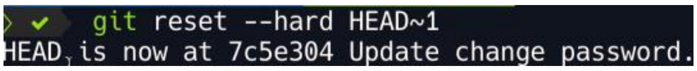
需要找到相应的文件，修改后，add，commit，就merge成功了

14- Undoing a Faulty Merge

Remove commit，如果是local可以，但是如果是shared，就会影响其他人

Revert commit，

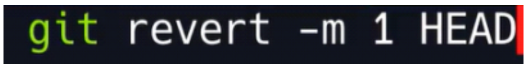
Remove commit：



```
> ✔  git reset --hard HEAD~1
HEAD is now at 7c5e304 Update change password.
```

Revert commit

-m后面的参数1是指恢复到master的head（最后一次）的这个commit，如果是2就是恢复到另一个branch的某个commit



```
git revert -m 1 HEAD
```

15- Squash Merging

git merge --squash bugfix

进行了squash merge后必须大写d强制删除这个branch，否则下次查看branch的时候会confuse，‘怎么有一个branch没有merge过’，实际是已经merge过了，但是squash merge不被记录。这种merge可以使主线是liner的

Git branch -D bugfix

16- Rebasing

rebase就是将一个非master branch（bugfix）的起点往后挪几个（这几个是master中不同于bugfix的commit），这可以使merge变为liner的。这能再local使用，因为rebase会改写历史

具体操作是，到bugfix中，git rebase master，这个命令会将master的最后一个commit设置为bugfix的第一个

17- Cherry Picking

master想要bugfix中的其中一次commit

git cherry-pick dad47ed

Cherry pick 可以进行两次，不受指针影响

Collaboration

Friday, August 19, 2022 3:07 PM

2- Workflows

对于公共项目，不能让每个人都有权限去修改。个人需要先fork（clone一个project on cloud），再修改，再push到clone 的project上。然后pull request，让给有权限的人，这个人再pull clone project，check后，再push到原项目中