Basics of

Git

GitHub

Why this Webinar?

- You registered for Microsoft codefundo++
- Deadline to submit your idea is **OCT 12TH, 23:59 IST**
- Getting you started with Git & GitHub

Agenda

- What is Version Control?
- What is the difference between Git & GitHub?
- How to get Git running on your system
- How to get started with Git
- How to get started with GitHub

Origin

Who created Git?

Linus Torvalds

When?

2005

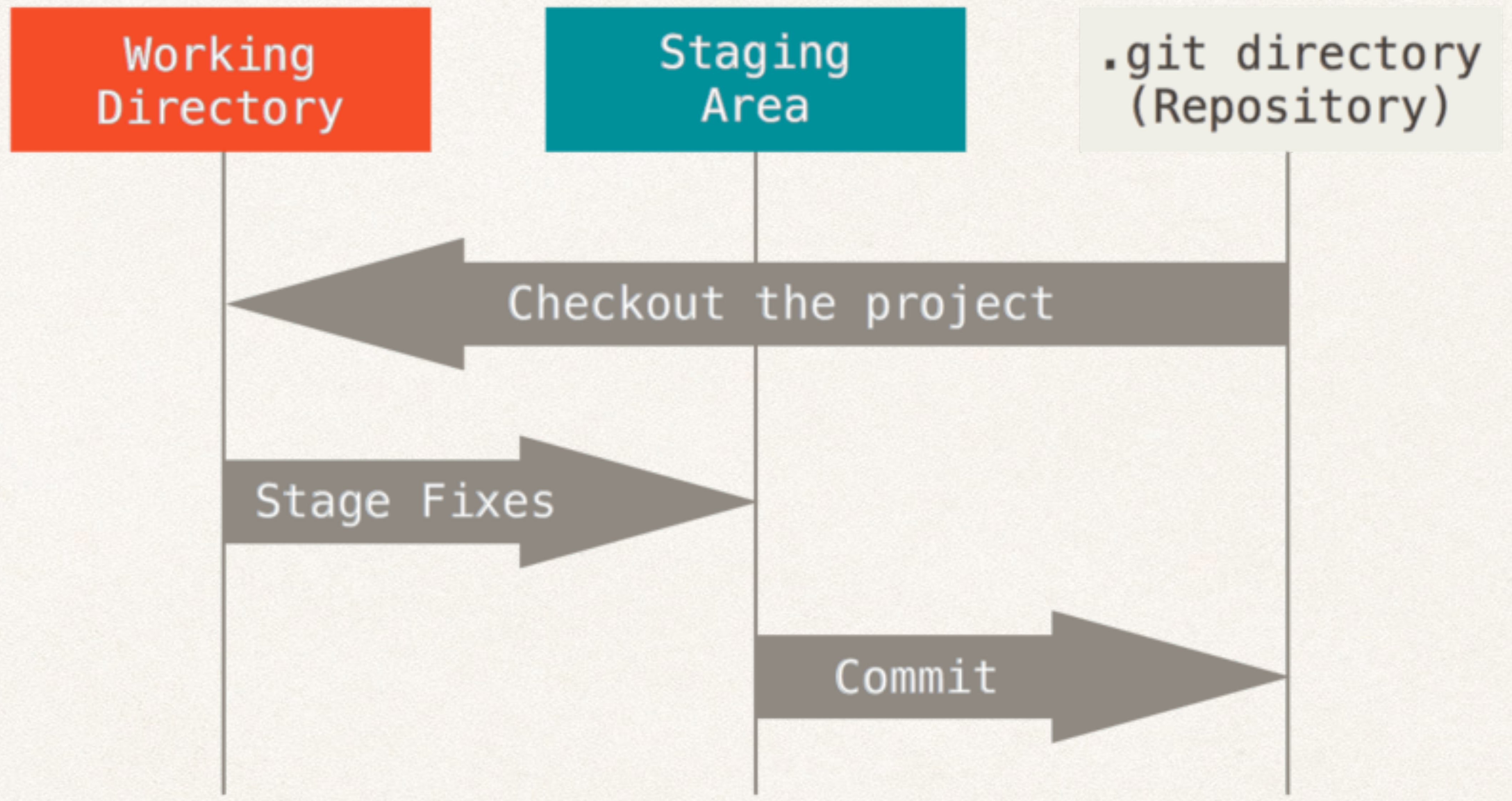
Why?

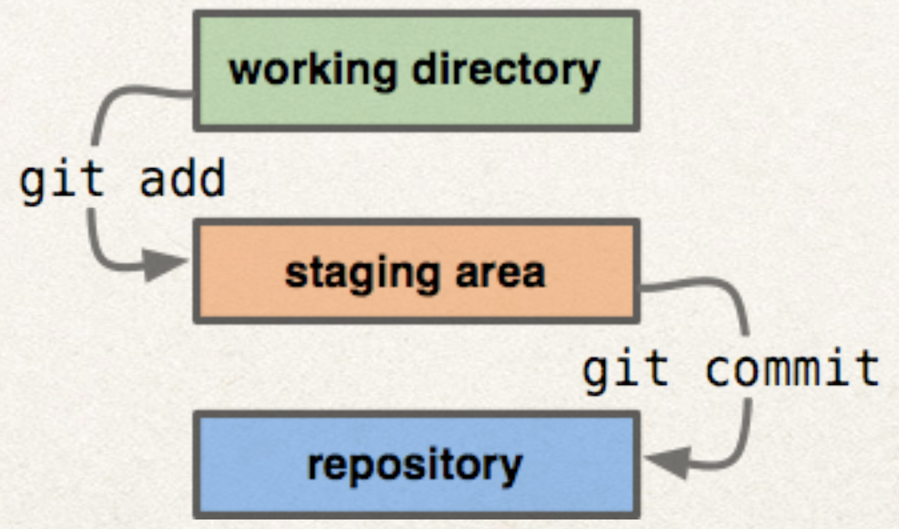
For development of the Linux kernel, with other kernel developers contributing to its initial development.

What does “Git” stand for?

Git in British English slang stands for a stupid or unpleasant person. Torvalds said:
I'm an egotistical bastard, and I name all my projects after myself. First 'Linux', now 'git'.
The man page describes git as "the stupid content tracker".

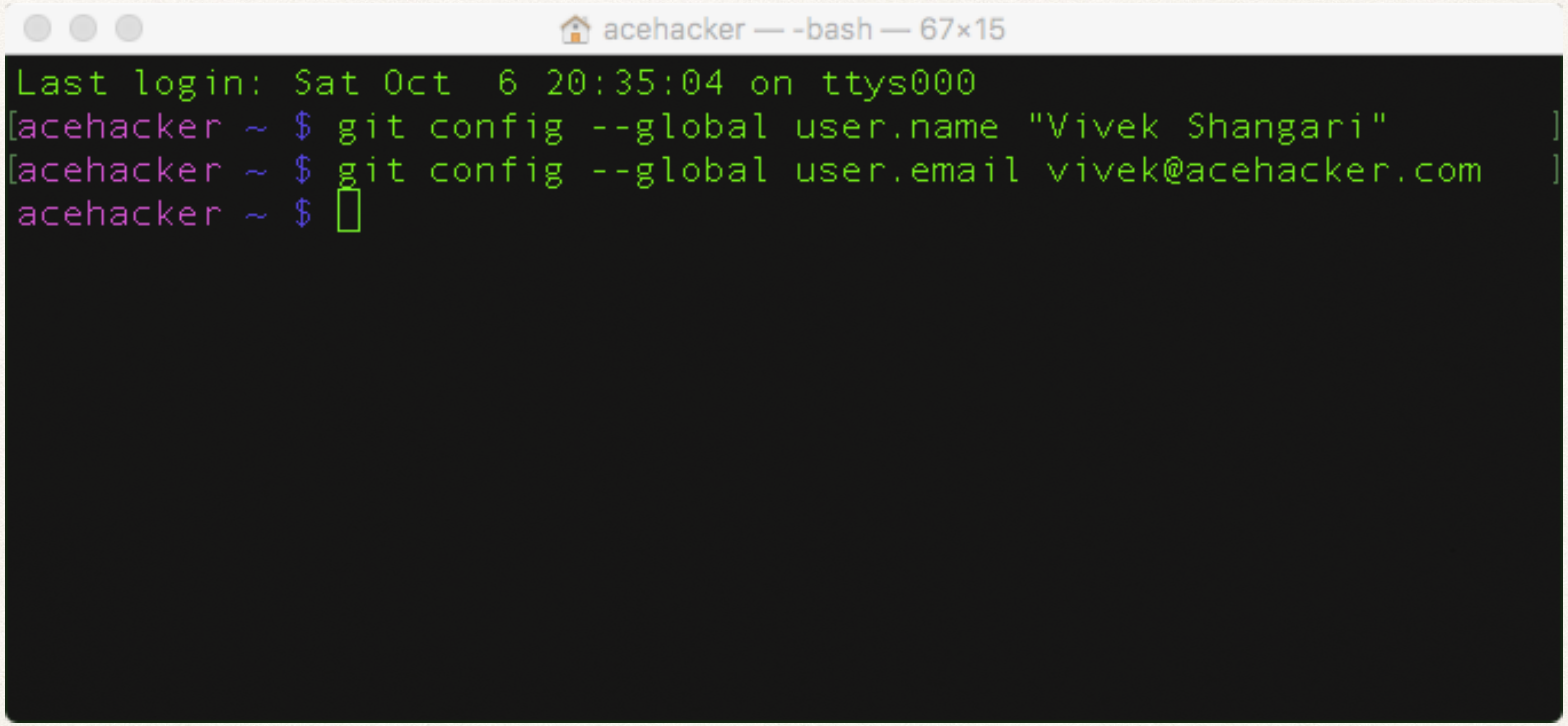
The Three Stages





Your identity

```
git config --global user.name “Vivek Shangari”
```

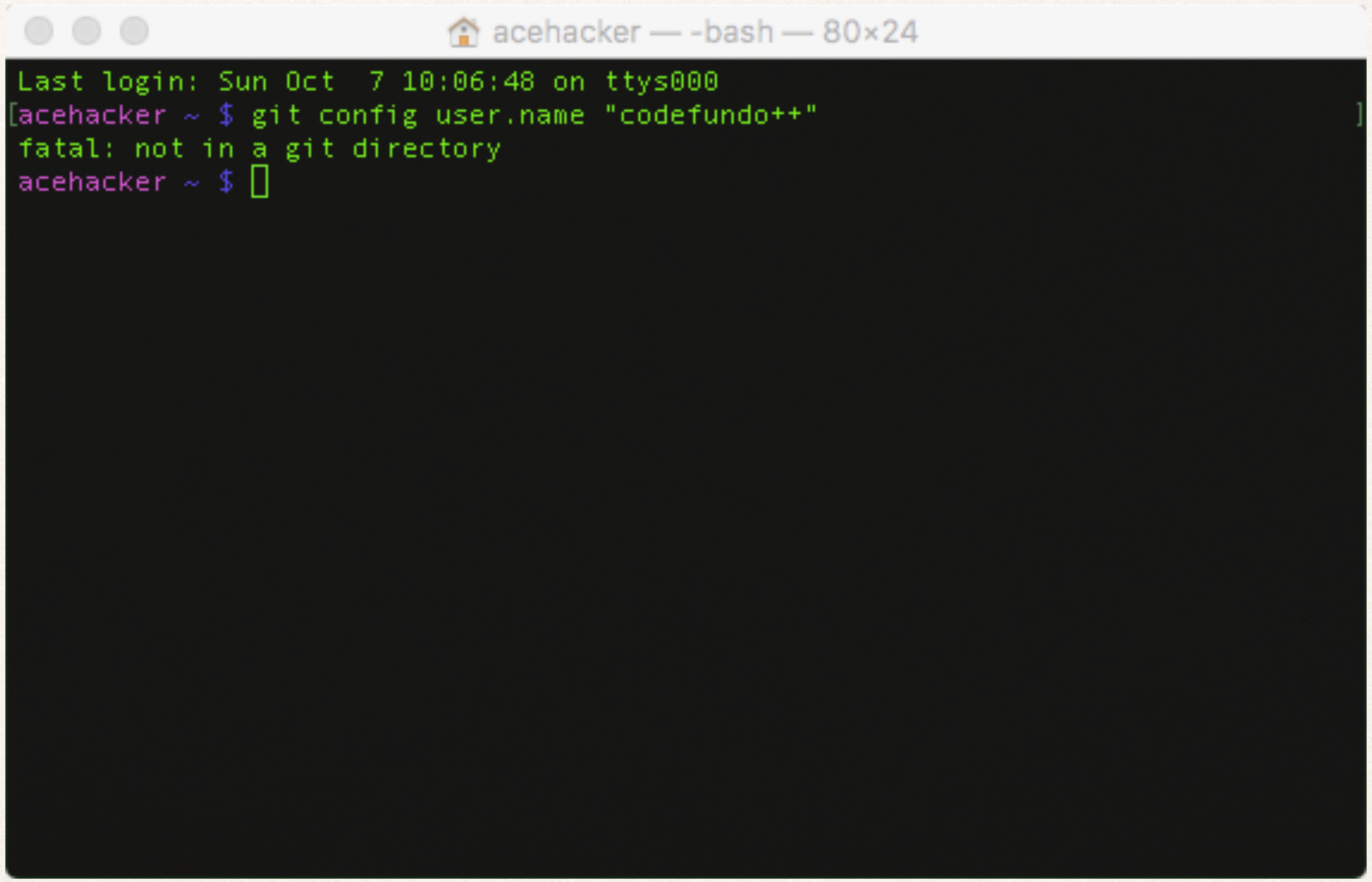


```
Last login: Sat Oct  6 20:35:04 on ttys000
acehacker ~ $ git config --global user.name "Vivek Shangari"
acehacker ~ $ git config --global user.email vivek@acehacker.com
acehacker ~ $ 
```

If you want to override this with a different name or email address for specific projects, you can run the command with the `--global` option when you’re in that project.

Your identity

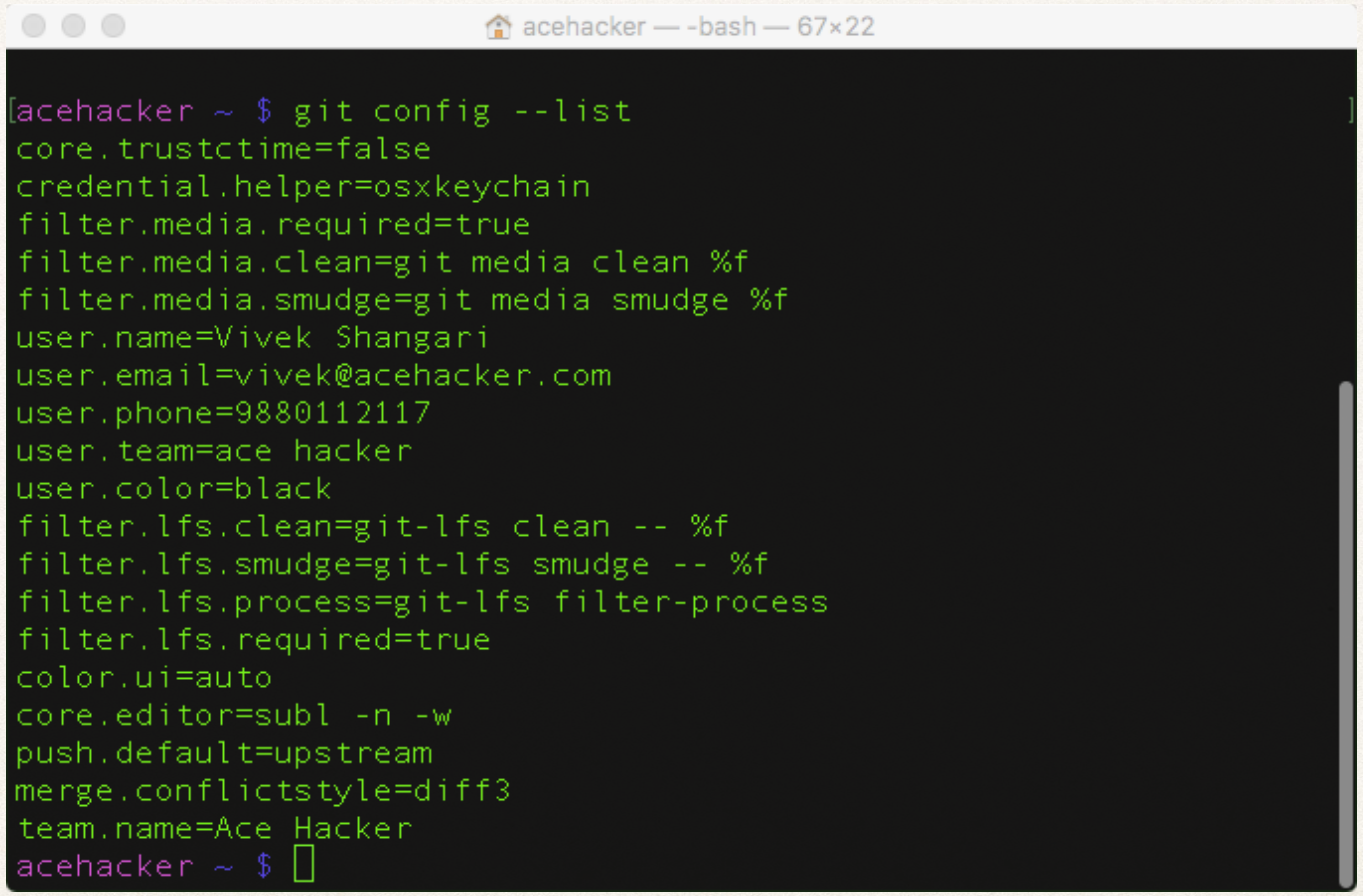
Without `--global` when you are not in a Local Git Repository



```
Last login: Sun Oct  7 10:06:48 on ttys000
acehacker ~ $ git config user.name "codefundo++"
fatal: not in a git directory
acehacker ~ $ 
```

Your identity

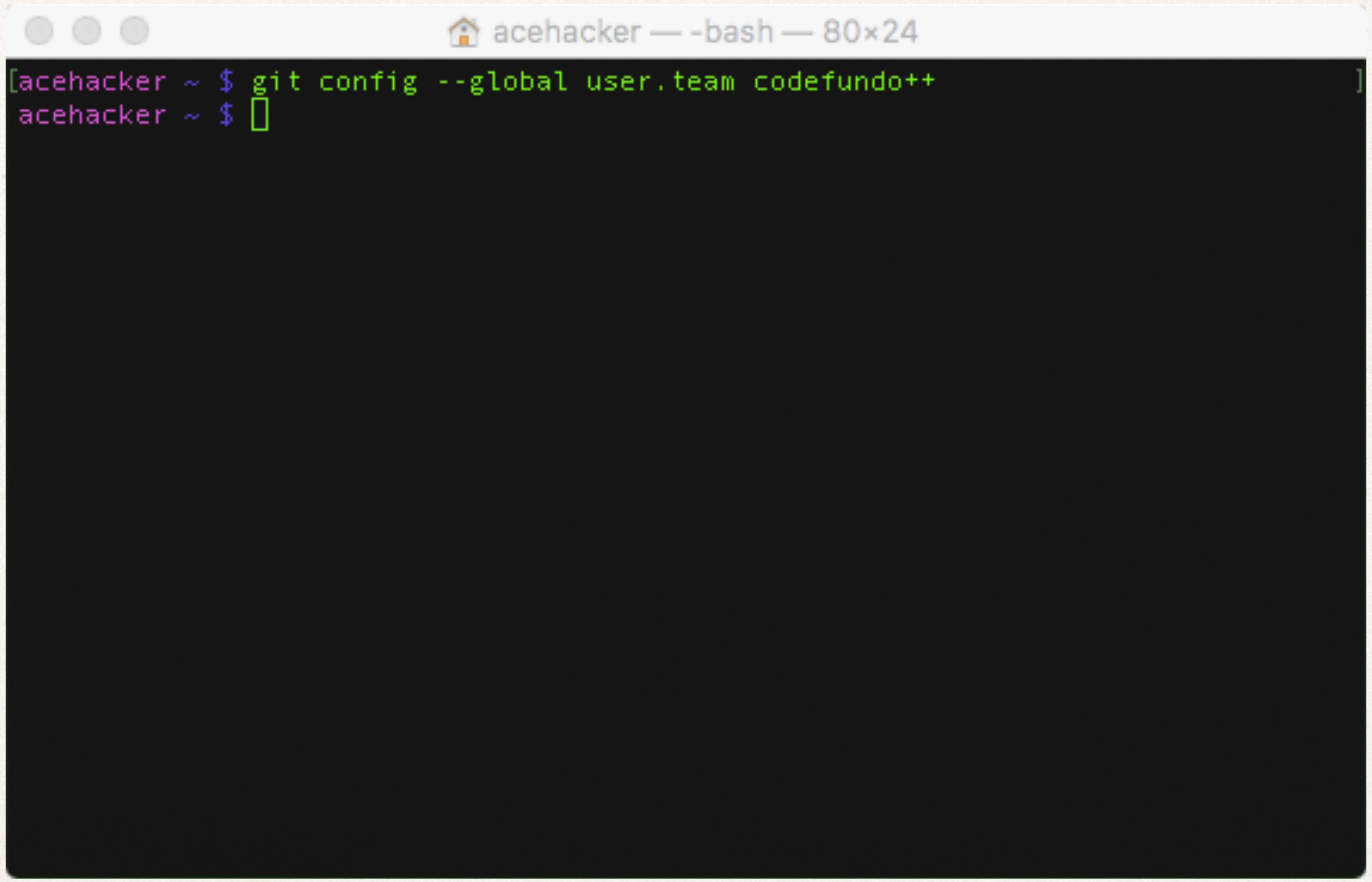
Check Your Settings



```
acehacker ~ $ git config --list
core.trustctime=false
credential.helper=osxkeychain
filter.media.required=true
filter.media.clean=git media clean %f
filter.media.smudge=git media smudge %f
user.name=Vivek Shangari
user.email=vivek@acehacker.com
user.phone=9880112117
user.team=ace hacker
user.color=black
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
color.ui=auto
core.editor=subl -n -w
push.default=upstream
merge.conflictstyle=diff3
team.name=Ace Hacker
acehacker ~ $
```

Your identity

Adding Custom Variables

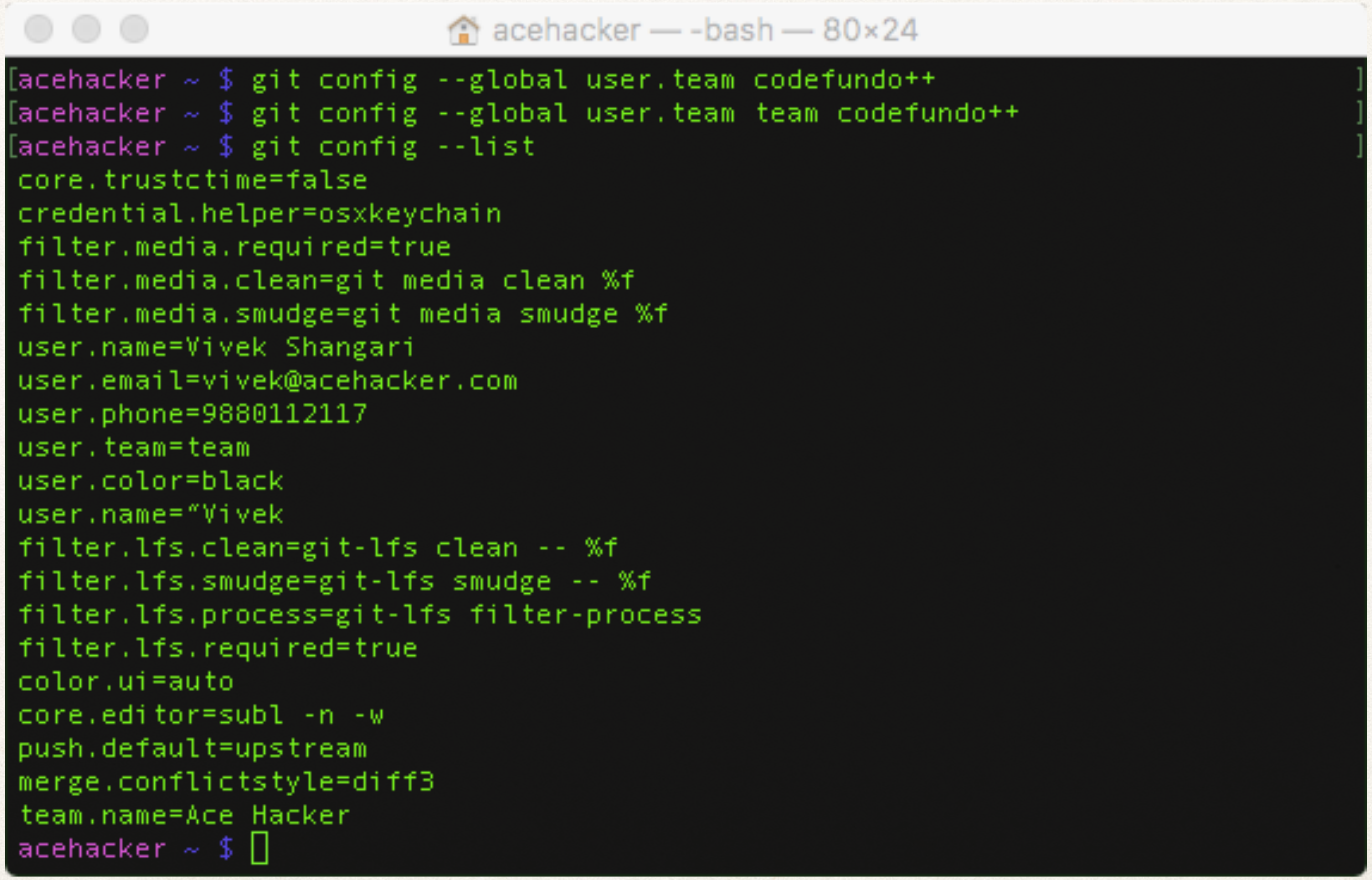


acehacker — -bash — 80×24

```
[acehacker ~ $ git config --global user.team codefundo++
 acehacker ~ $ 
```

Your identity

Adding Custom Variables > When to use Quotes



```
[acehacker ~ $ git config --global user.team codefundo++                        ]
[acehacker ~ $ git config --global user.team team codefundo++                   ]
[acehacker ~ $ git config --list                                                ]
core.trustctime=false
credential.helper=osxkeychain
filter.media.required=true
filter.media.clean=git media clean %f
filter.media.smudge=git media smudge %f
user.name=Vivek Shangari
user.email=vivek@acehacker.com
user.phone=9880112117
user.team=team
user.color=black
user.name="Vivek
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
color.ui=auto
core.editor=subl -n -w
push.default=upstream
merge.conflictstyle=diff3
team.name=Ace Hacker
acehacker ~ $ 
```

Getting Help

```
git help config
```

```
git config --help
```

```
man git-config
```

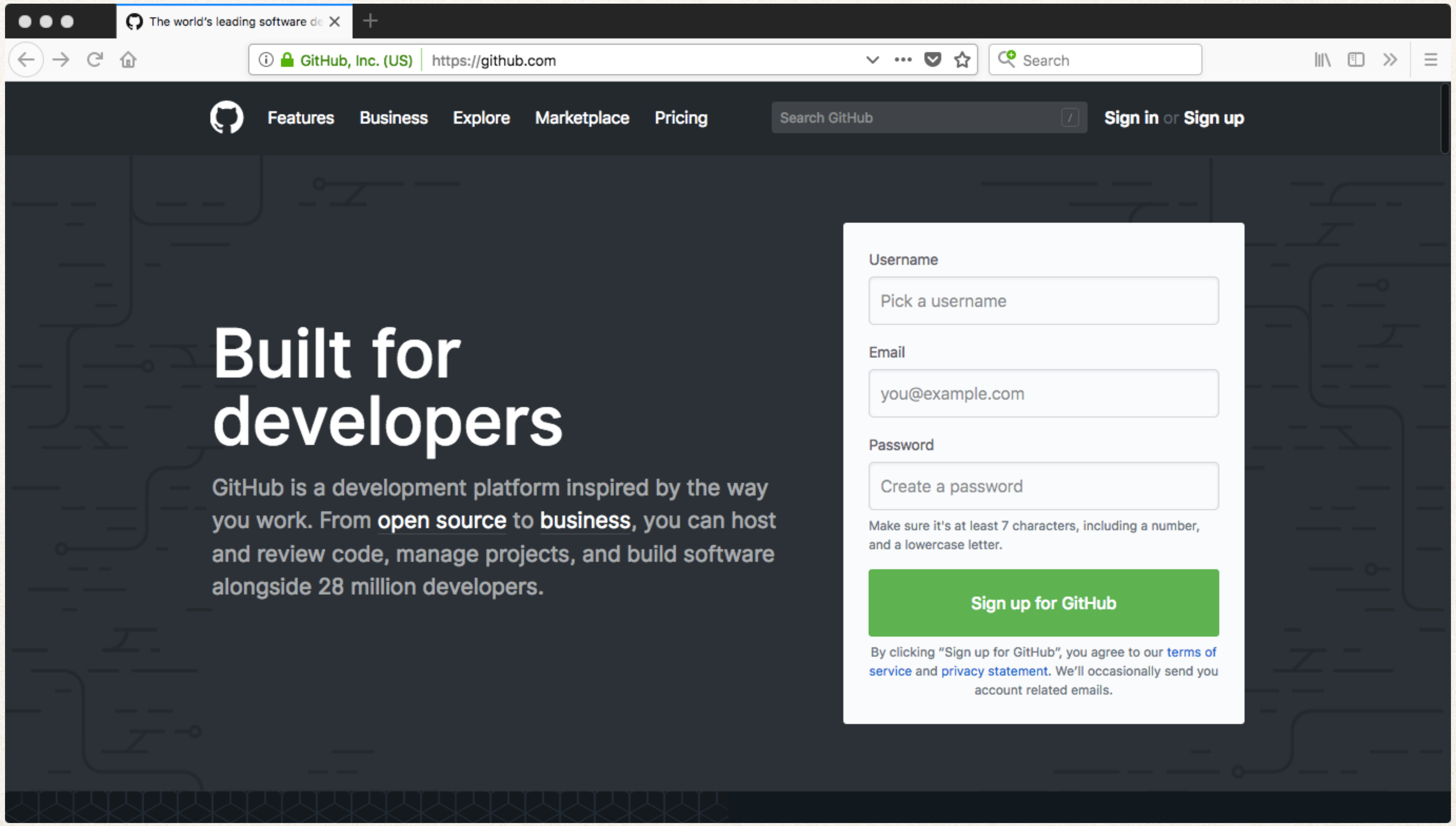
Repository

Local

Remote/Central

GitHub

Create an Account on GitHub.com



The world's leading software de

GitHub, Inc. (US) https://github.com

Search

Features Business Explore Marketplace Pricing Search GitHub Sign in or Sign up

Built for developers

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 28 million developers.

Username

Pick a username

Email

you@example.com

Password

Create a password

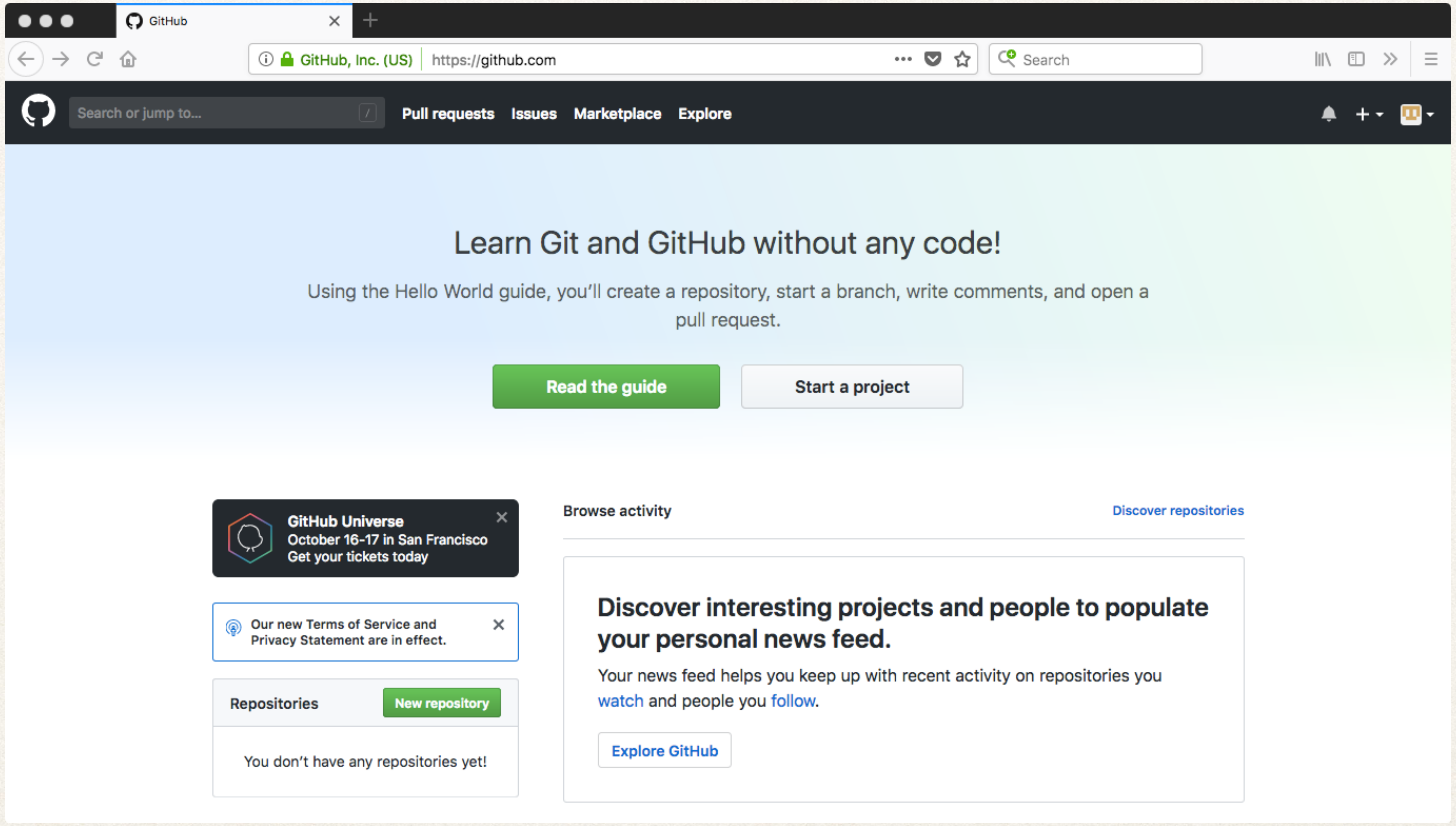
Make sure it's at least 7 characters, including a number, and a lowercase letter.

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our terms of service and privacy statement. We'll occasionally send you account related emails.

GitHub

Start a project



Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

GitHub Universe
October 16-17 in San Francisco
Get your tickets today

Our new Terms of Service and Privacy Statement are in effect.

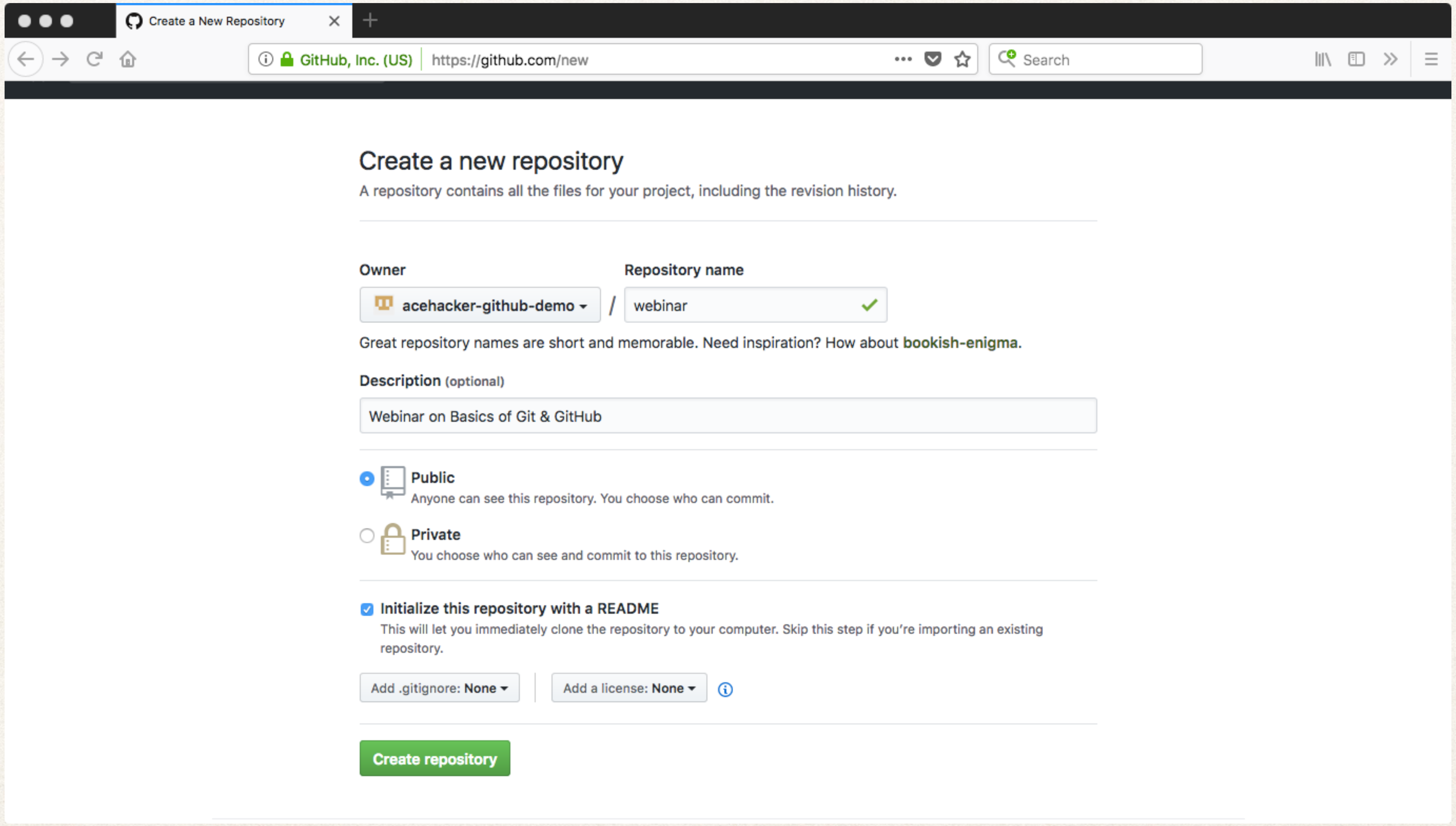
You don't have any repositories yet!

Discover interesting projects and people to populate your personal news feed.

Your news feed helps you keep up with recent activity on repositories you watch and people you follow.

GitHub

Create a new repository



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: acehacker-github-demo / Repository name: webinar ✓

Great repository names are short and memorable. Need inspiration? How about bookish-enigma.

Description (optional): Webinar on Basics of Git & GitHub

● Public
Anyone can see this repository. You choose who can commit.

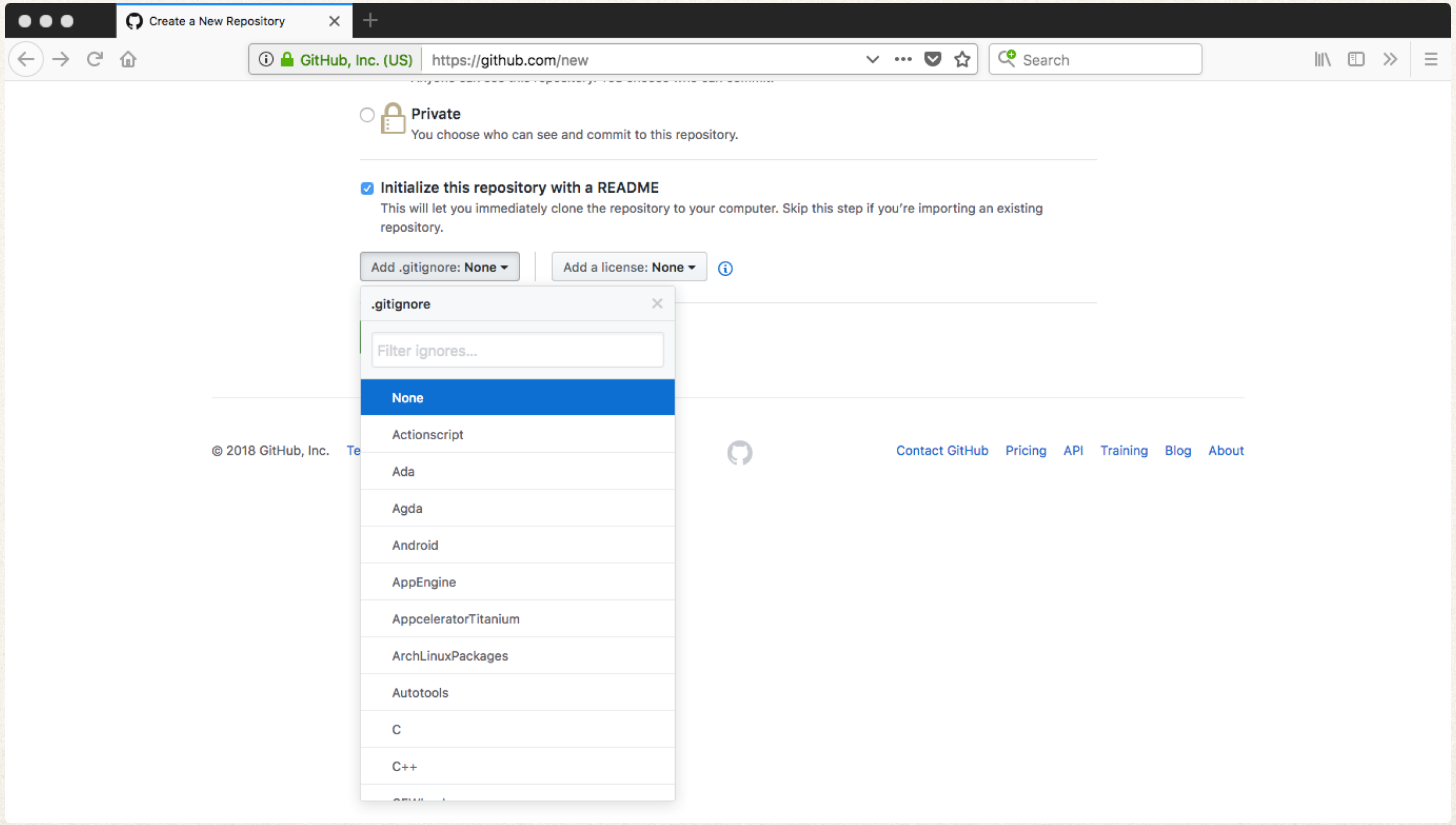
○ Private
You choose who can see and commit to this repository.

☑ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None

GitHub

Add .gitignore



Private
You choose who can see and commit to this repository.

[✓] Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None ▾ Add a license: None ▾

.gitignore

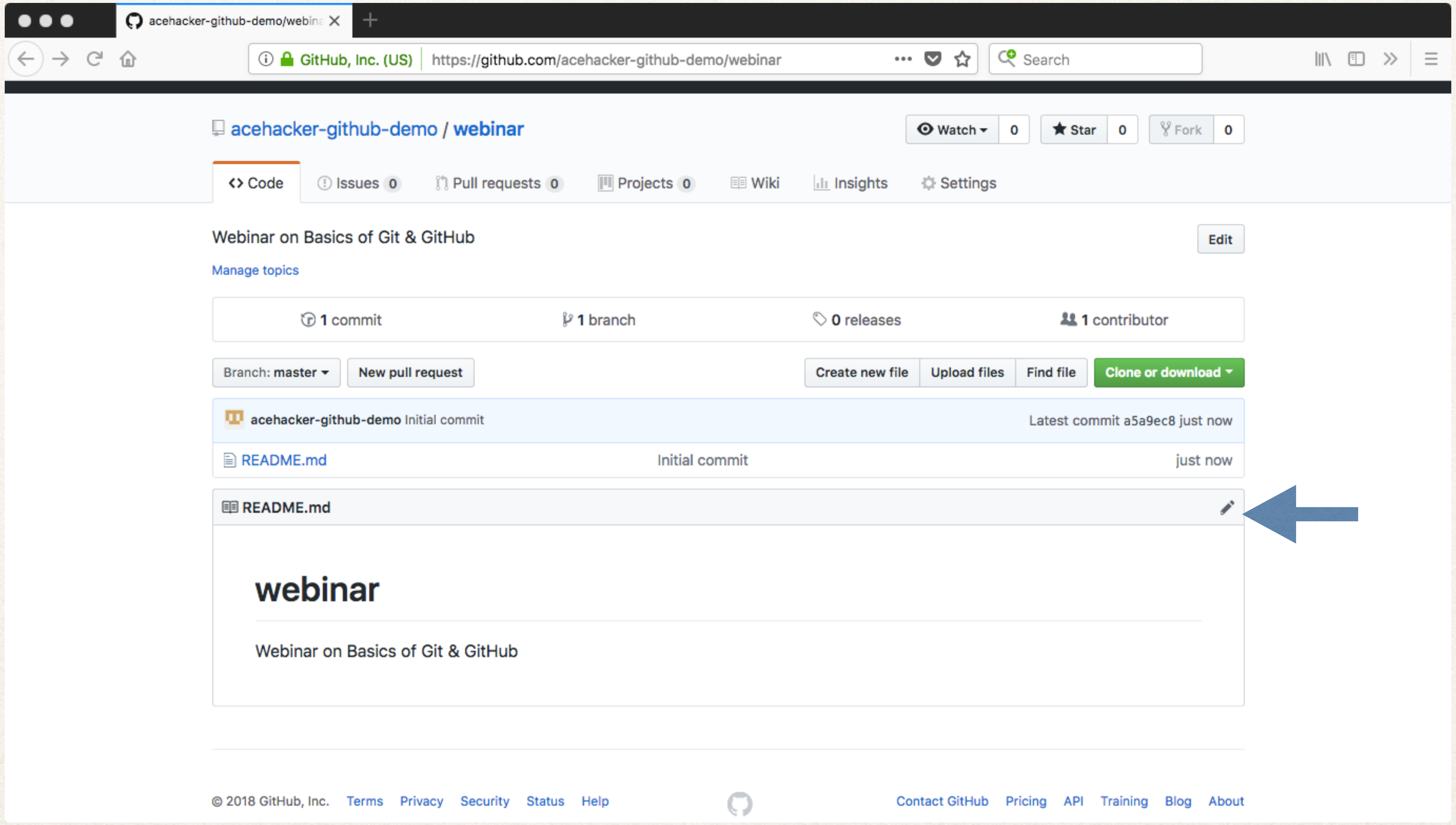
Filter ignores…

- None
- Actionscript
- Ada
- Agda
- Android
- AppEngine
- AppceleratorTitanium
- ArchLinuxPackages
- Autotools
- C
- C++

© 2018 GitHub, Inc.

GitHub

README.md



Webinar on Basics of Git & GitHub

1 commit | 1 branch | 0 releases | 1 contributor

acehacker-github-demo Initial commit — Latest commit a5a9ec8 just now

README.md	Initial commit	just now

README.md

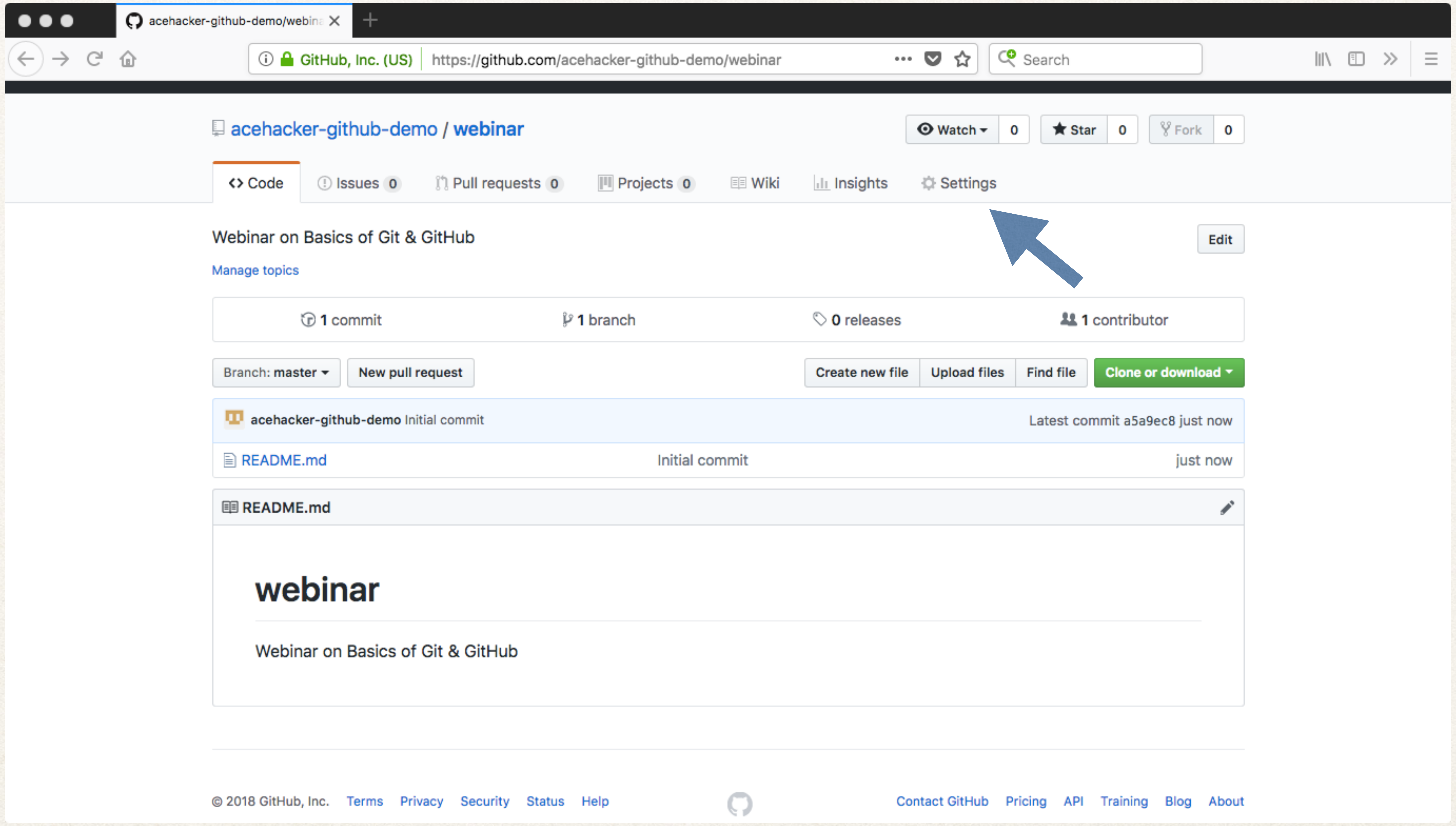
webinar

Webinar on Basics of Git & GitHub

© 2018 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

GitHub

Add `codefundopp` as collaborator



Webinar on Basics of Git & GitHub

1 commit | 1 branch | 0 releases | 1 contributor

acehacker-github-demo Initial commit — Latest commit a5a9ec8 just now

README.md — Initial commit — just now

README.md

webinar

Webinar on Basics of Git & GitHub

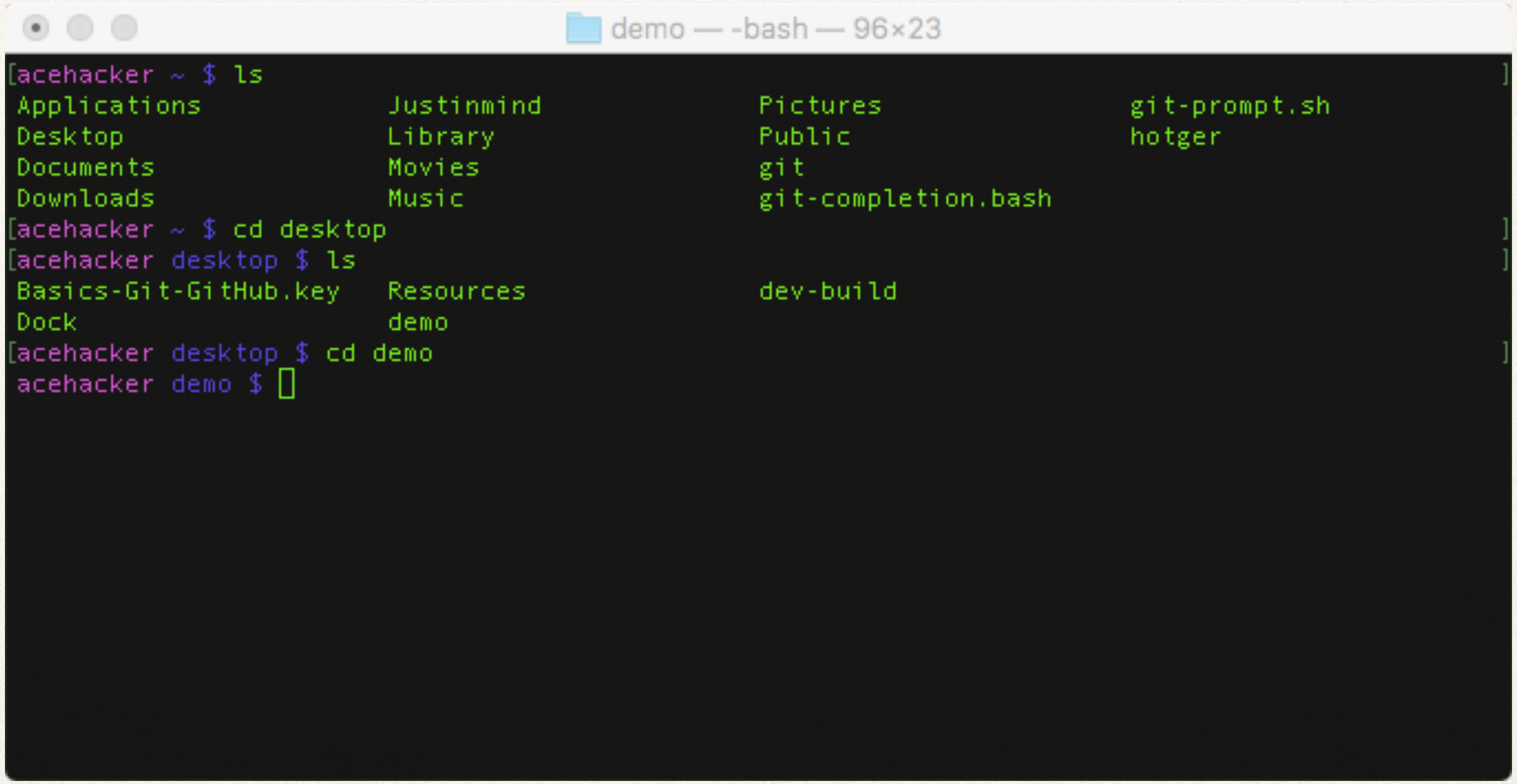
© 2018 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

Git

- Create folder for local repository

Git

Go to local folder



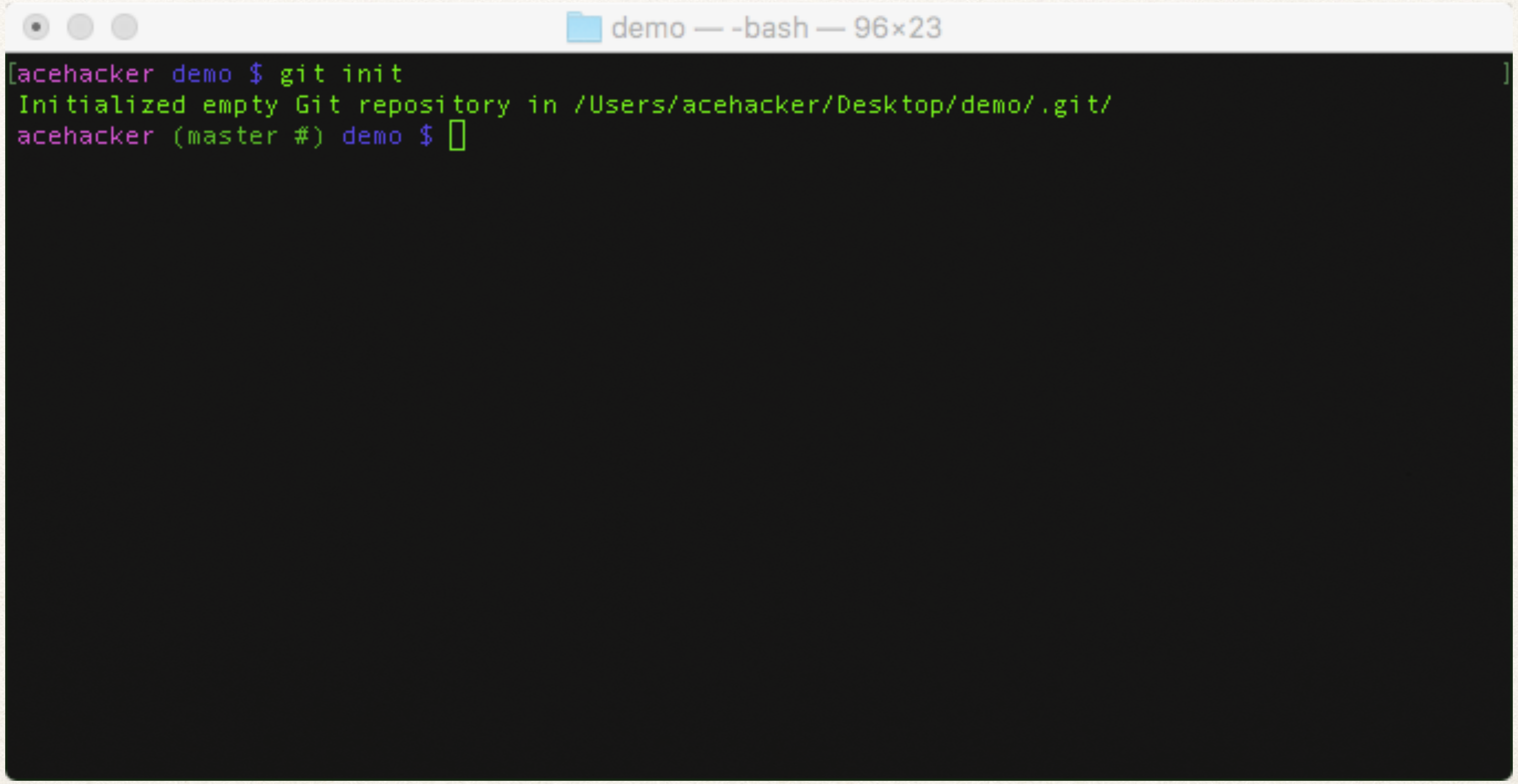
demo — -bash — 96×23

```
[acehacker ~ $ ls
 Applications            Justinmind              Pictures                git-prompt.sh
 Desktop                 Library                 Public                  hotger
 Documents               Movies                  git
 Downloads               Music                   git-completion.bash
[acehacker ~ $ cd desktop
[acehacker desktop $ ls
 Basics-Git-GitHub.key   Resources               dev-build
 Dock                    demo
[acehacker desktop $ cd demo
 acehacker demo $ 
```

Git

Git Local Repository

`git init`



demo — -bash — 96×23

```
[acehacker demo $ git init                                                                      ]
 Initialized empty Git repository in /Users/acehacker/Desktop/demo/.git/
 acehacker (master #) demo $ 
```

Git

Sync Local & Remote Repository

Local

Remote

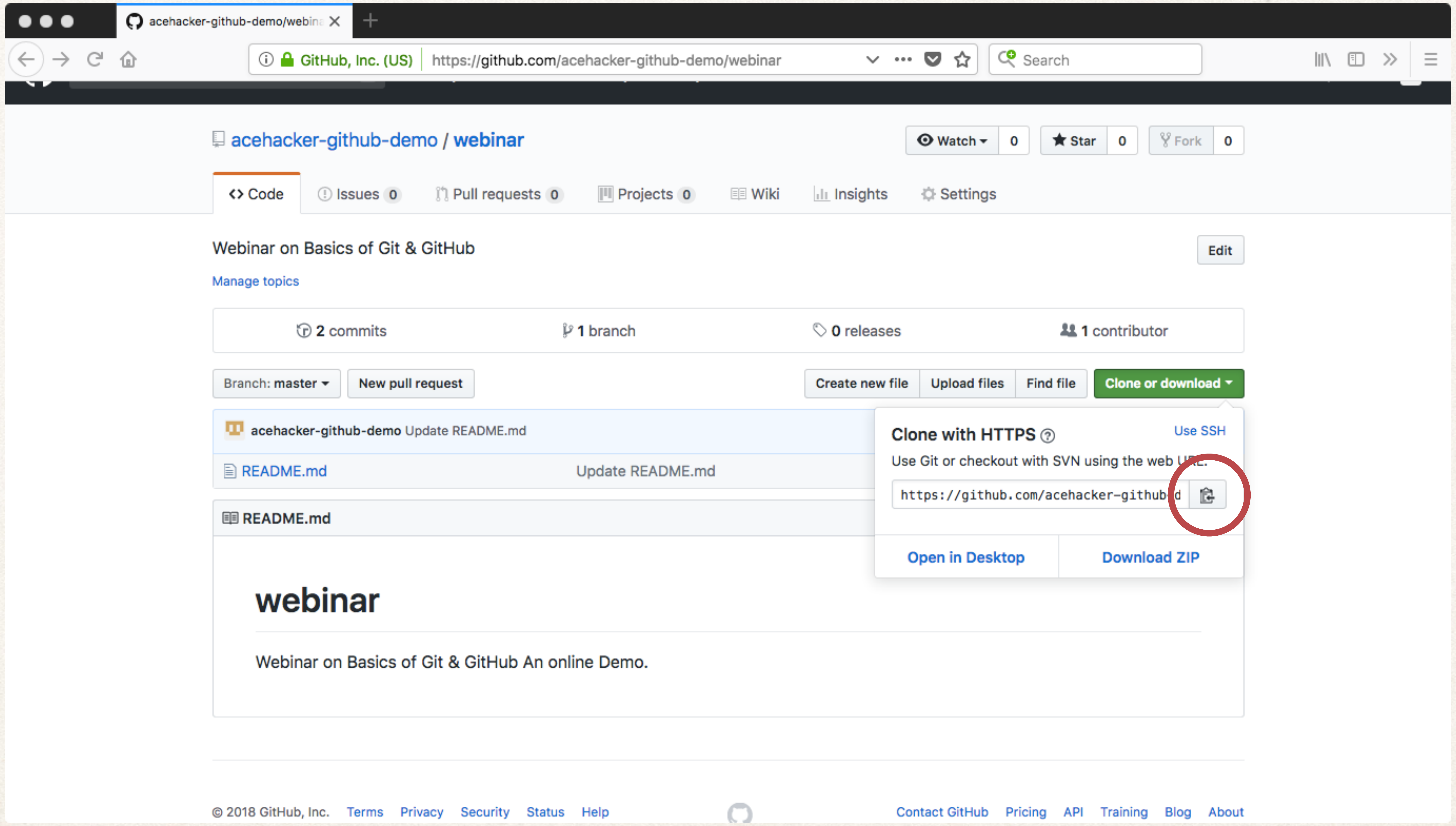
Git

Sync Local & Remote Repository

```
git remote add origin “”
```

GitHub

How to find link



Webinar on Basics of Git & GitHub

acehacker-github-demo Update README.md

README.md Update README.md

Use Git or checkout with SVN using the web URL.

```
https://github.com/acehacker-github-d
```

README.md

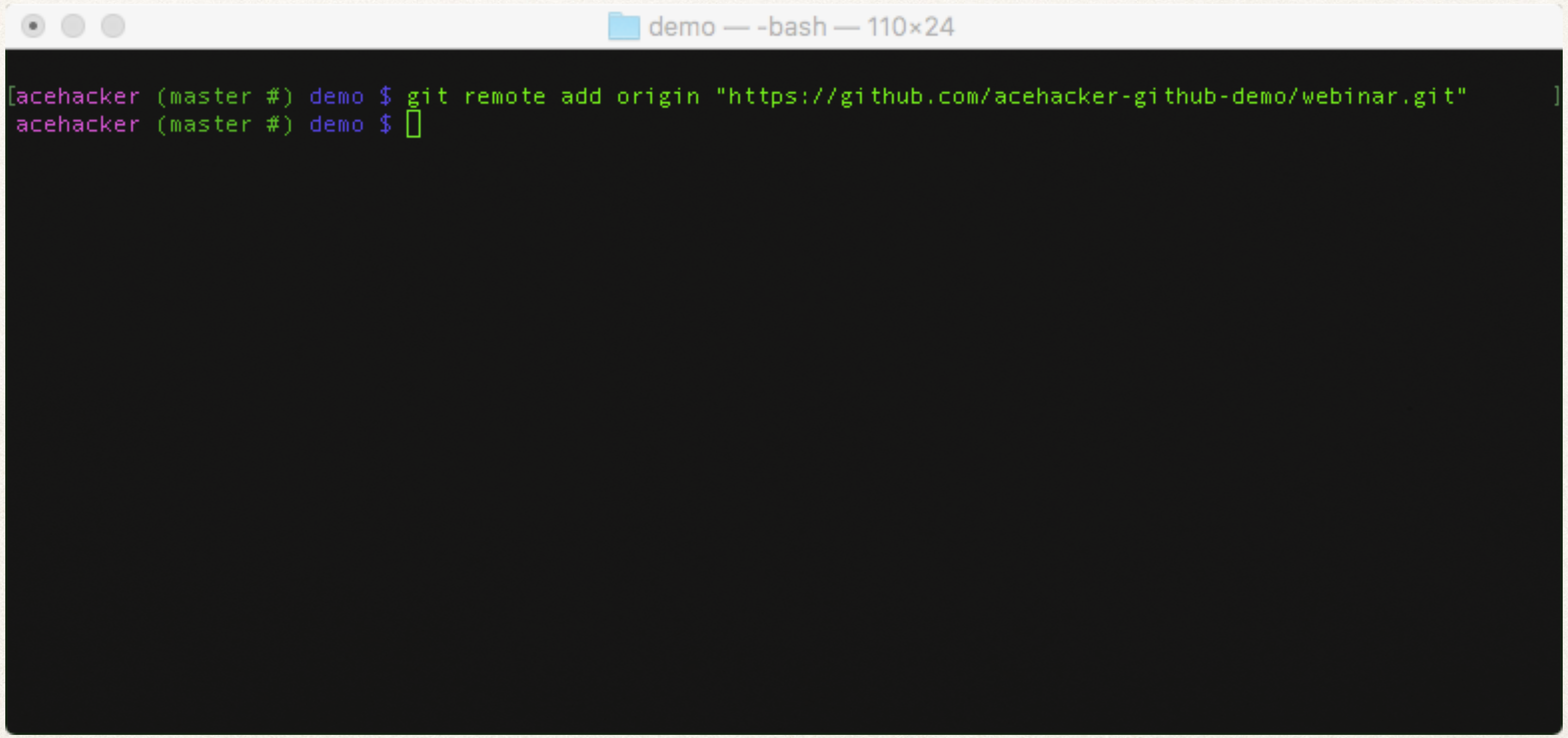
webinar

Webinar on Basics of Git & GitHub An online Demo.

Git

Sync Local & Remote Repository

`git remote add origin “https://github.com/acehacker-github-demo/webinar.git”`

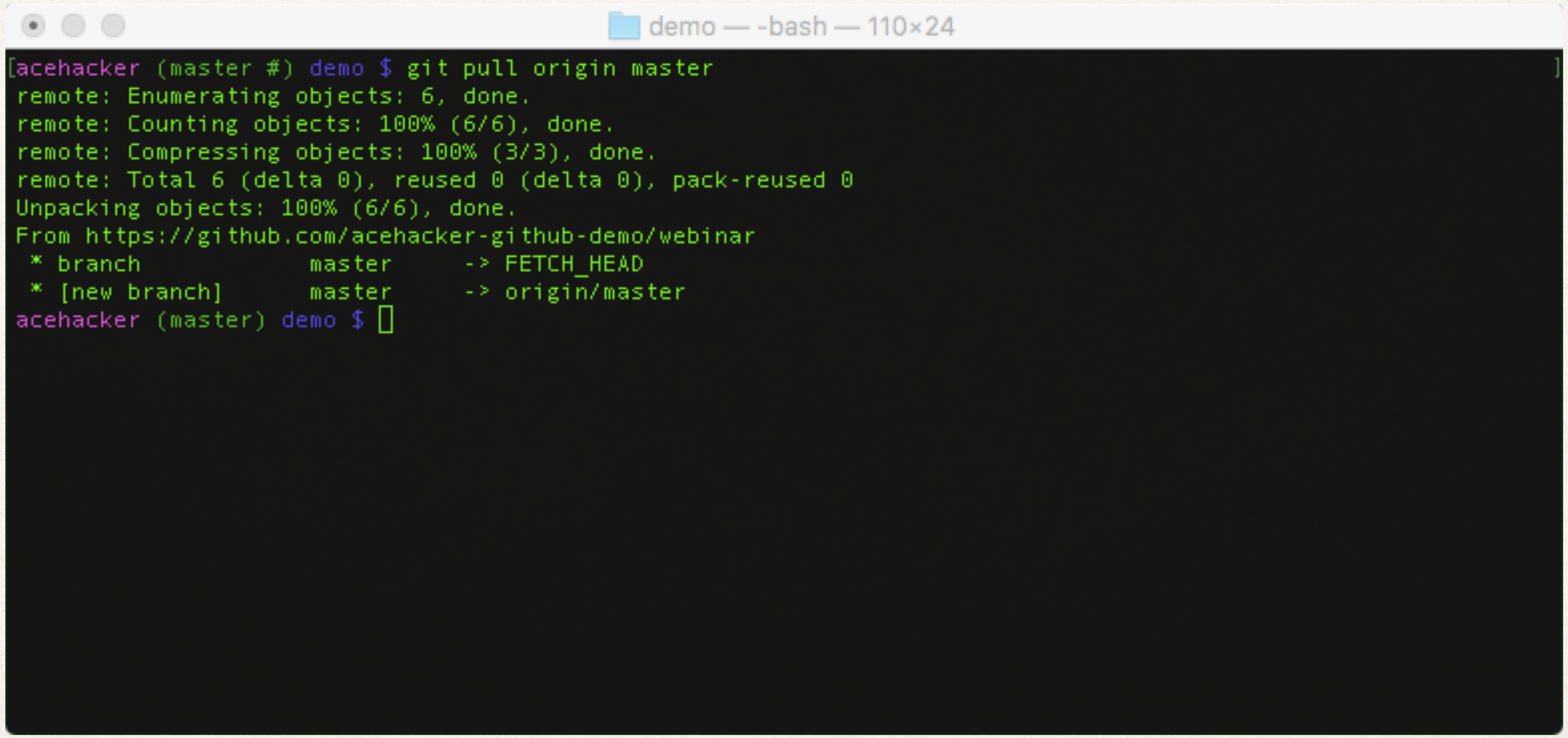

demo — -bash — 110×24

```
[acehacker (master #) demo $ git remote add origin "https://github.com/acehacker-github-demo/webinar.git"      ]
 acehacker (master #) demo $ 
```

Git

Pull files from Central Repository to Local Repository

`git pull origin master`

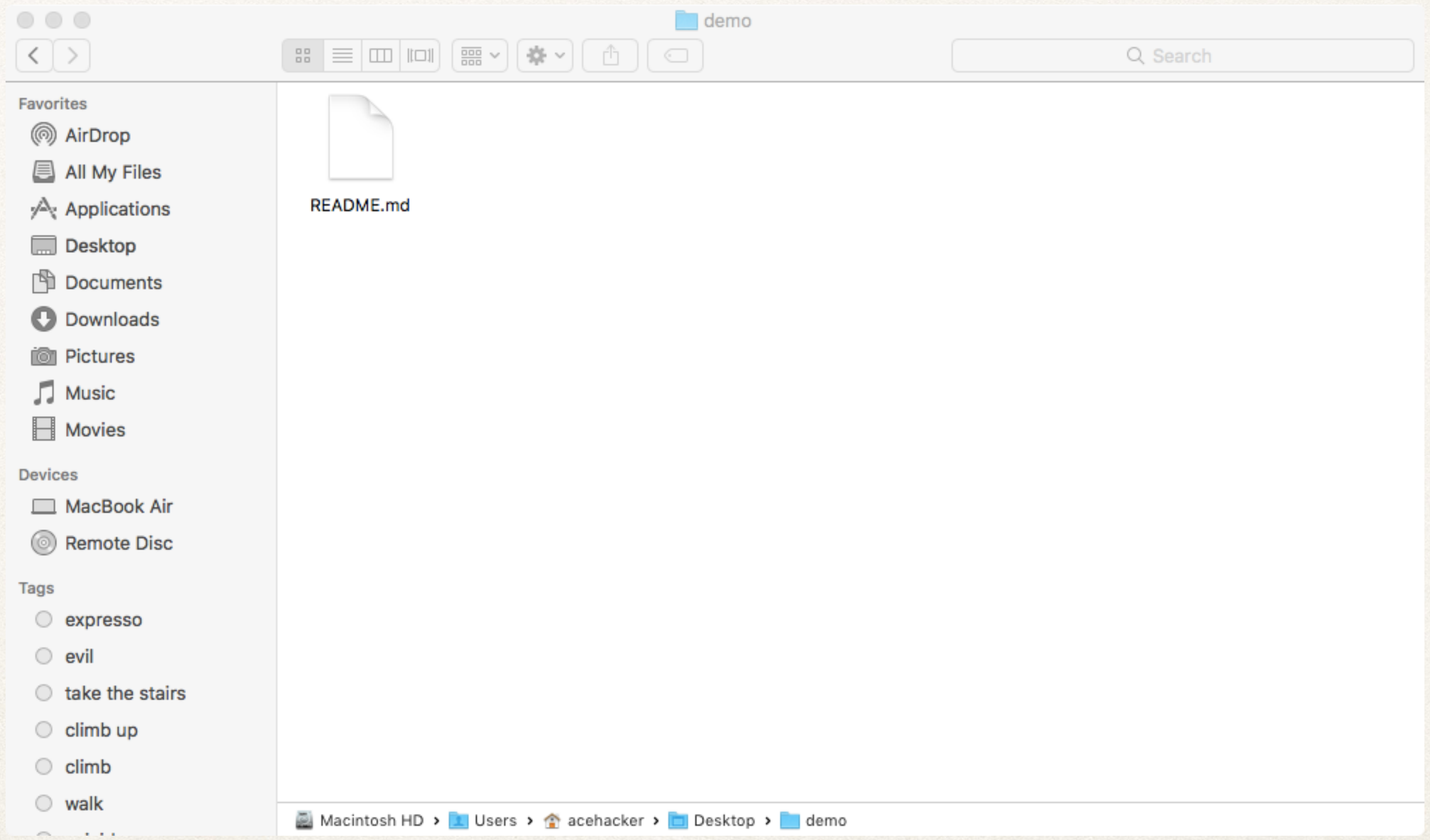


```
acehacker (master #) demo $ git pull origin master
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), done.
From https://github.com/acehacker-github-demo/webinar
 * branch            master     -> FETCH_HEAD
 * [new branch]      master     -> origin/master
acehacker (master) demo $
```

Git

Check your local repository

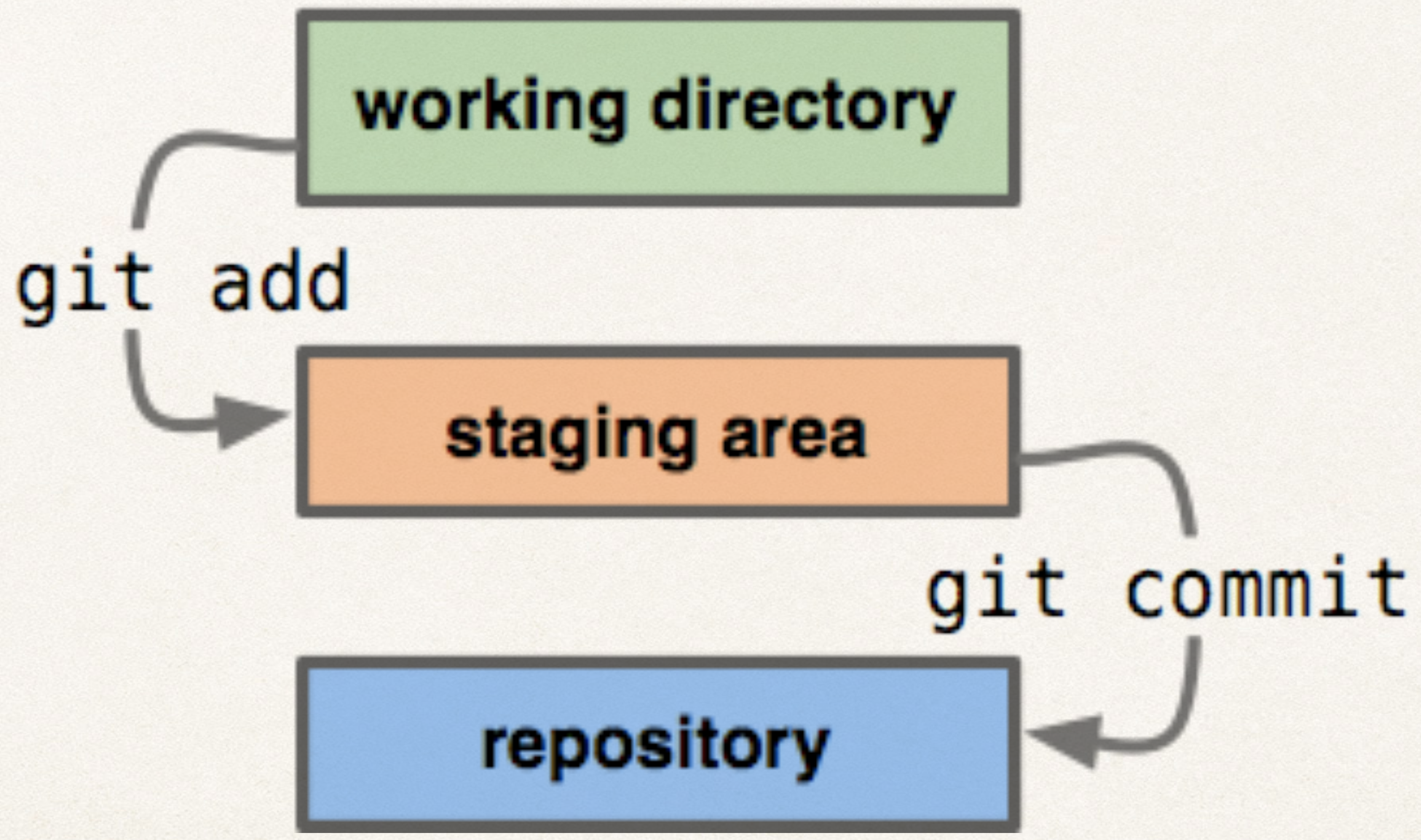
On your machine



Git

How to make changes

Add changes to Staging Area (or Index) first.



Git

How to add to the Staging Area/Index

```
git add
```

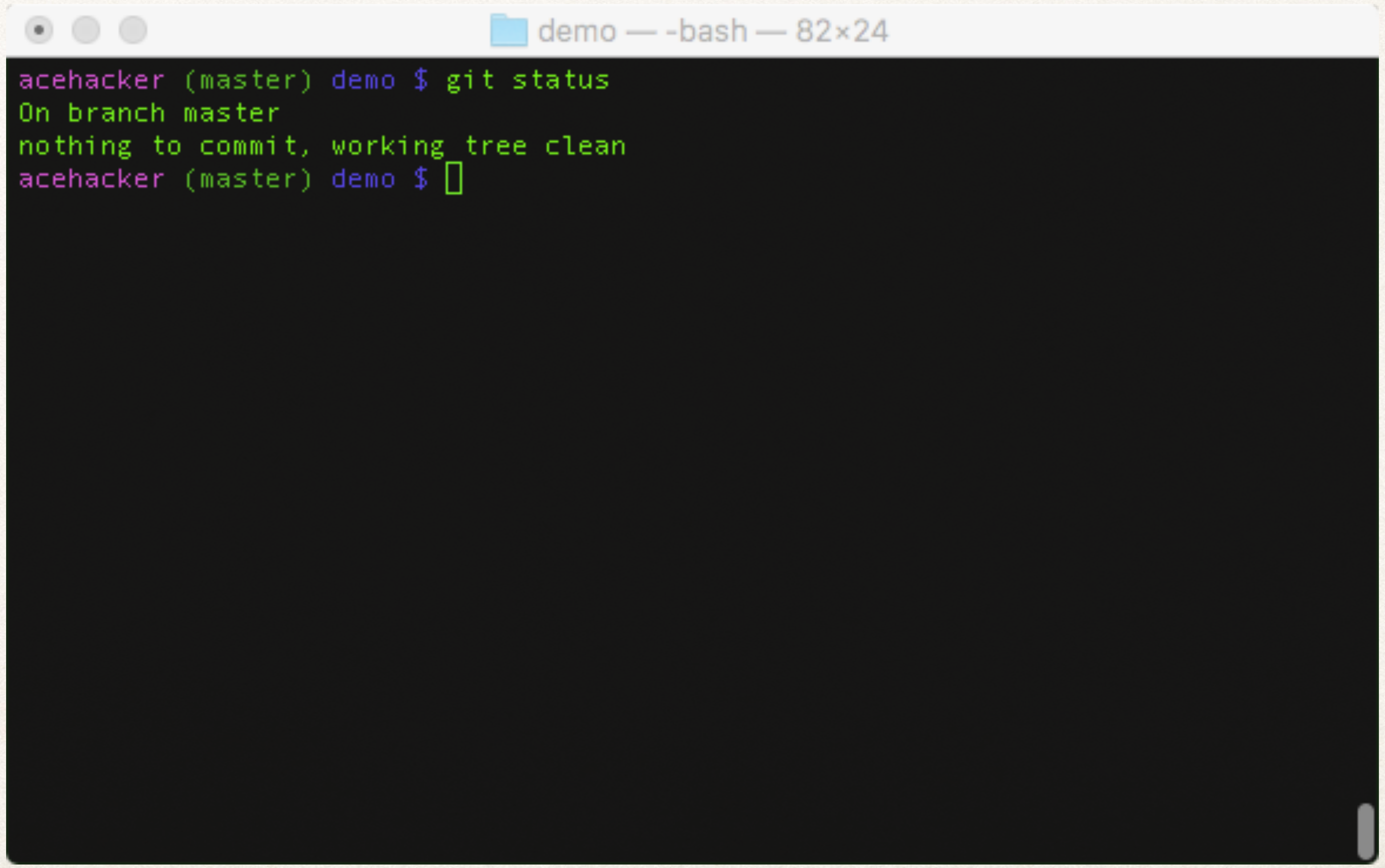
Which files are there in the Index and which are not

```
git status
```

Git

When there is nothing in the Staging Area to commit.

`git status`

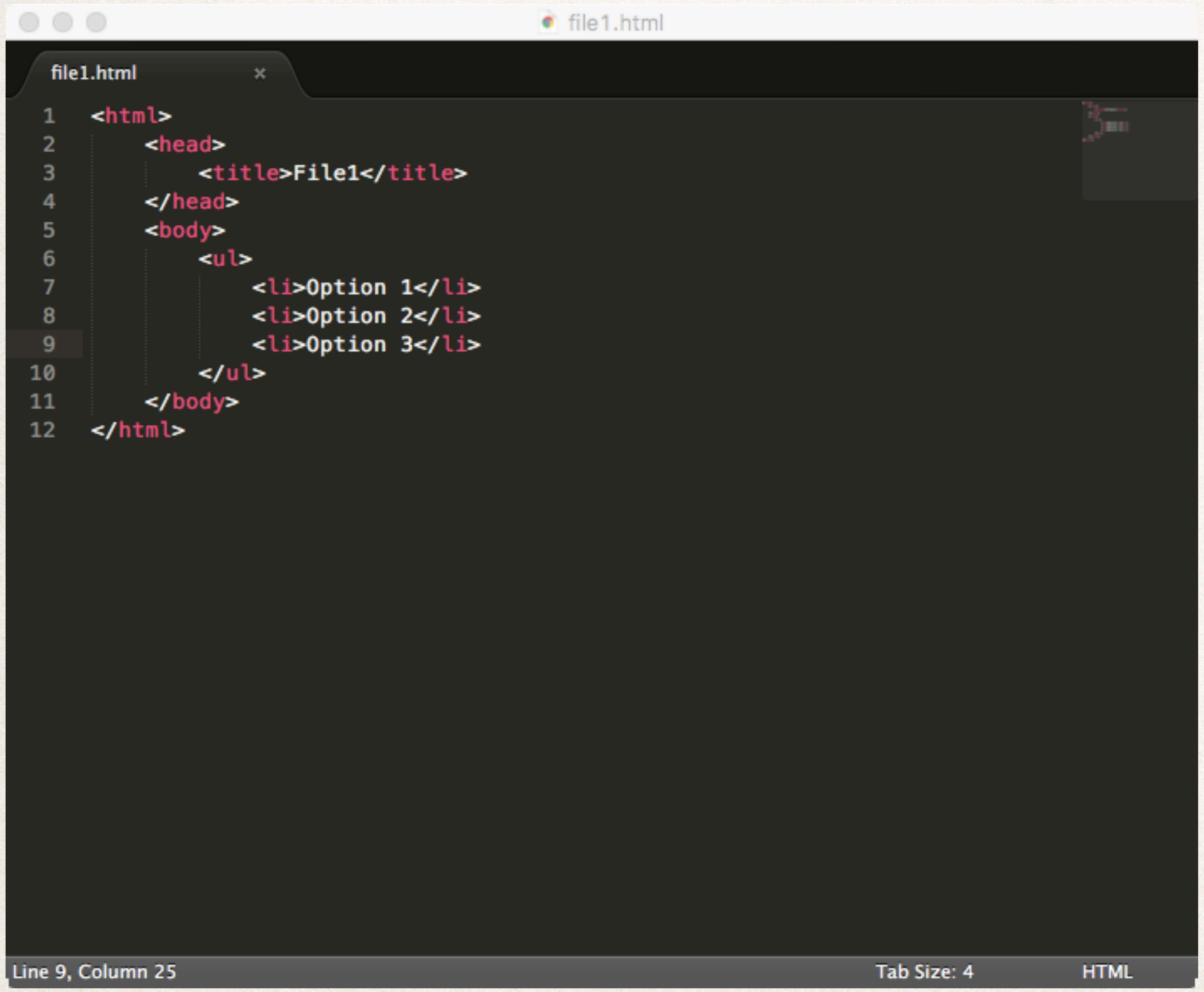


demo — -bash — 82×24

```
acehacker (master) demo $ git status
On branch master
nothing to commit, working tree clean
acehacker (master) demo $ 
```

Git

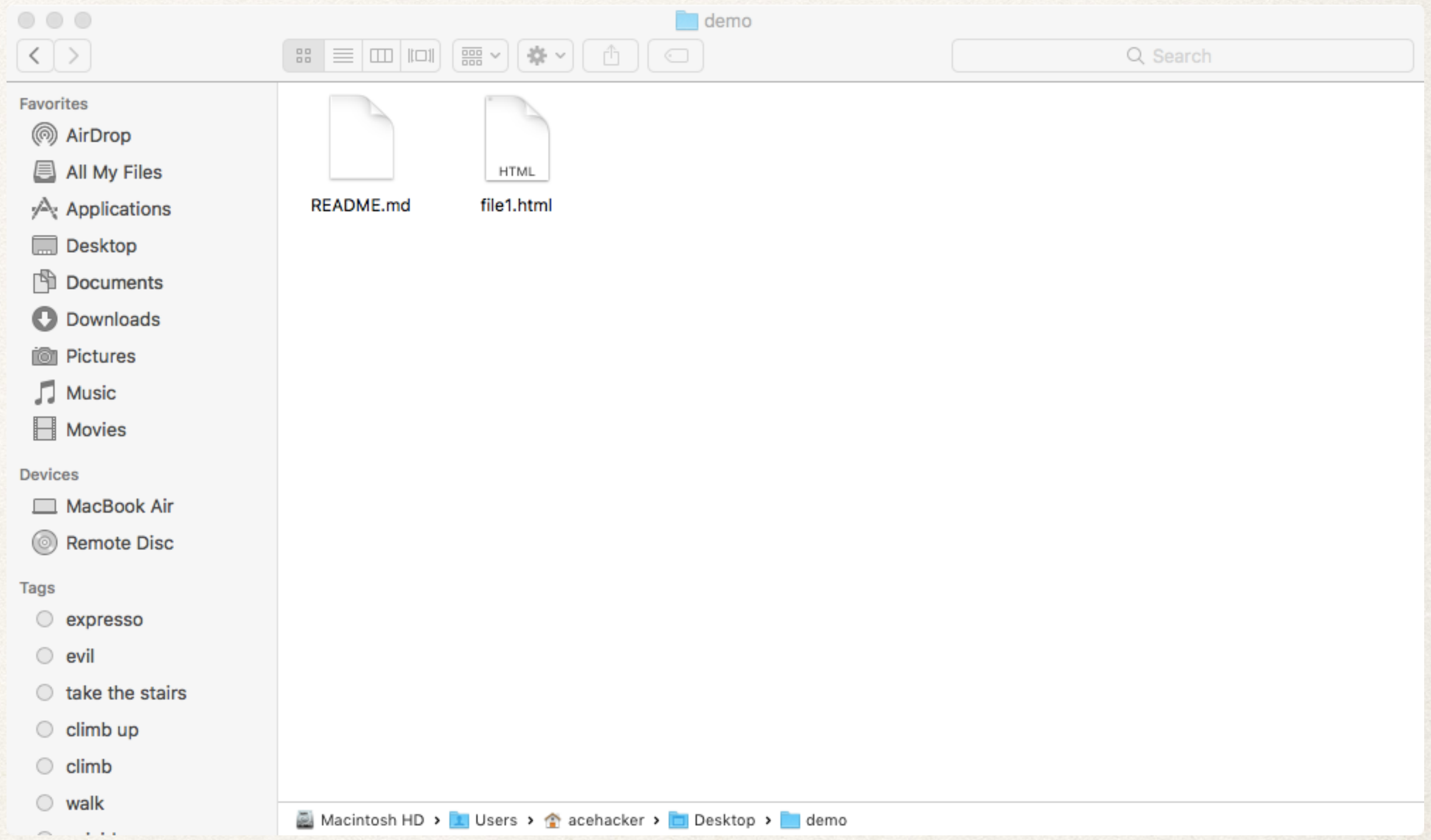
Create some files in your local repository



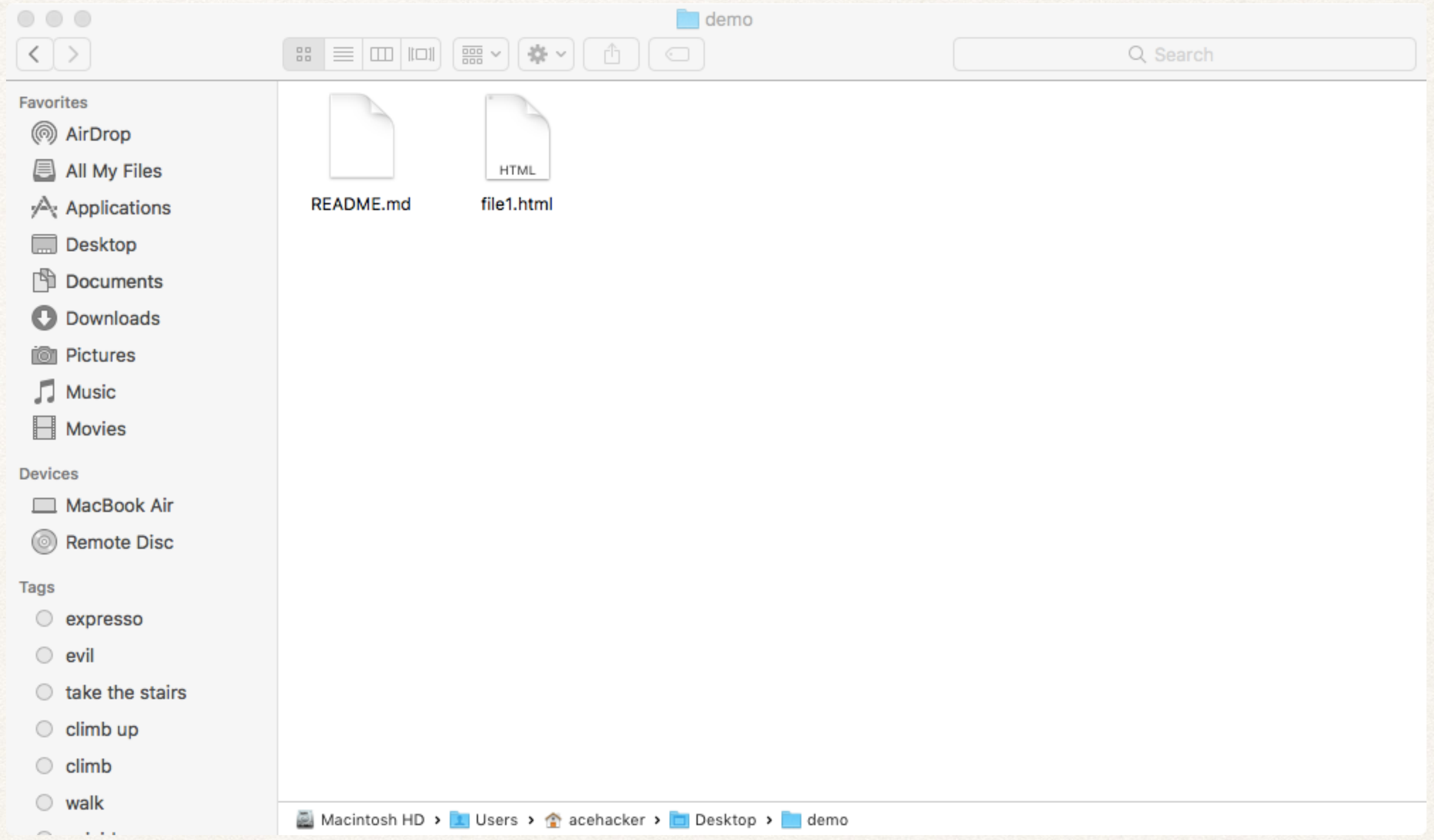
file1.html

```
<html>
    <head>
        <title>File1</title>
    </head>
    <body>
        <ul>
            <li>Option 1</li>
            <li>Option 2</li>
            <li>Option 3</li>
        </ul>
    </body>
</html>
```

Line 9, Column 25 Tab Size: 4 HTML



Git

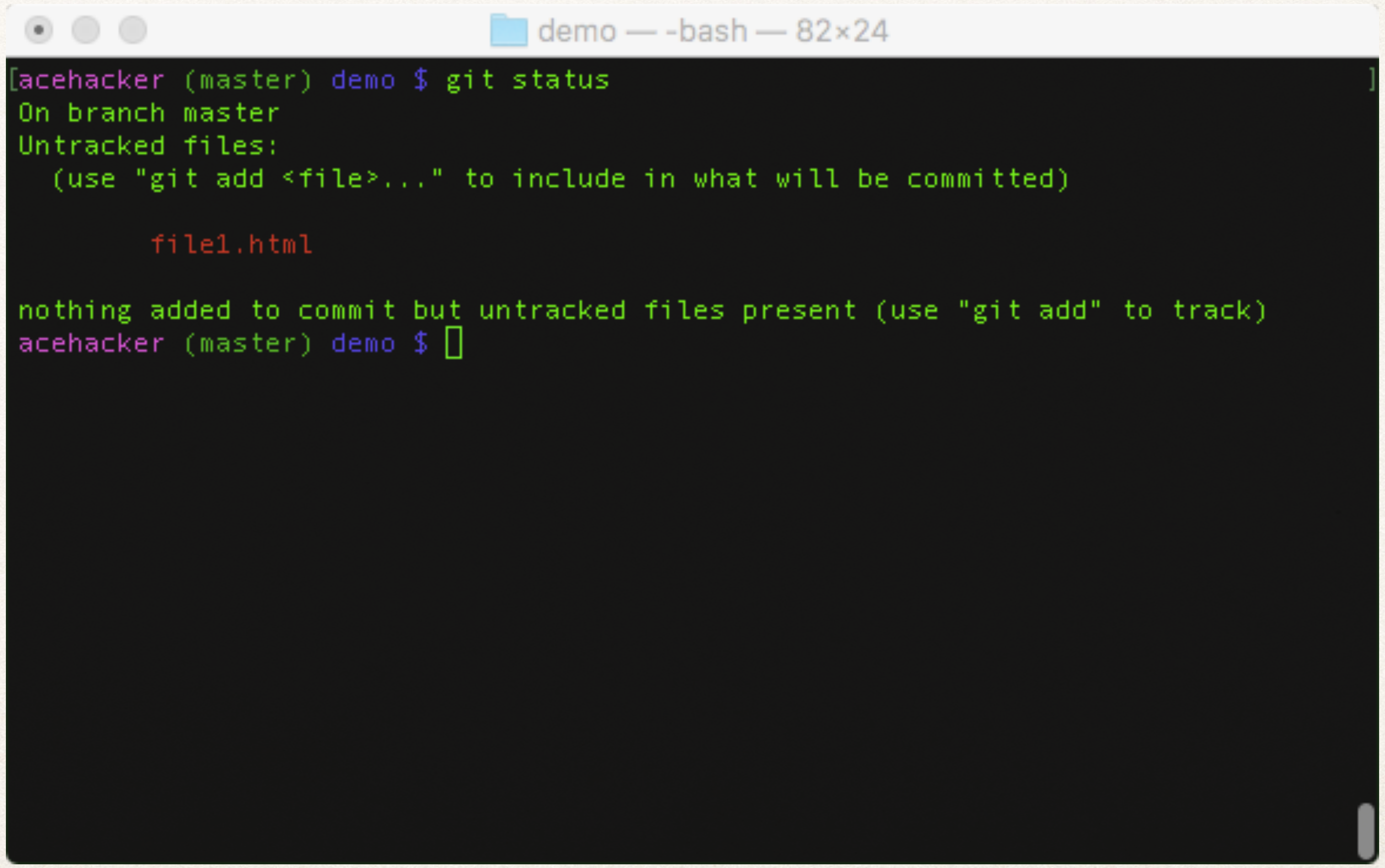


- The file you just created is in your workspace.
- It is not in the Staging Area/Index yet because you.

Git

See for yourself

`git status`

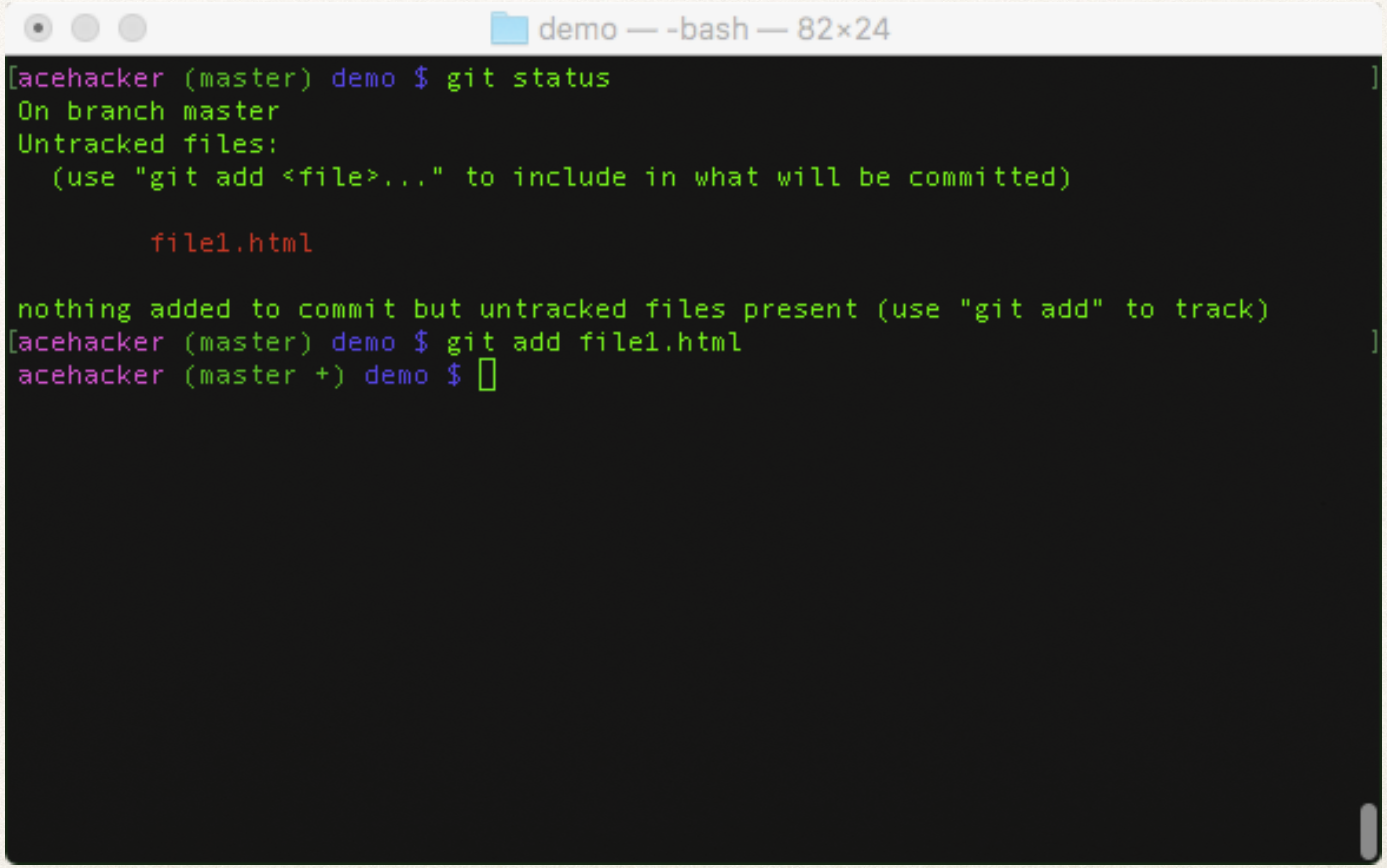


```
[acehacker (master) demo $ git status                                          ]
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be commited)

        file1.html

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

Git

Add this file to the Staging Area

`git add`



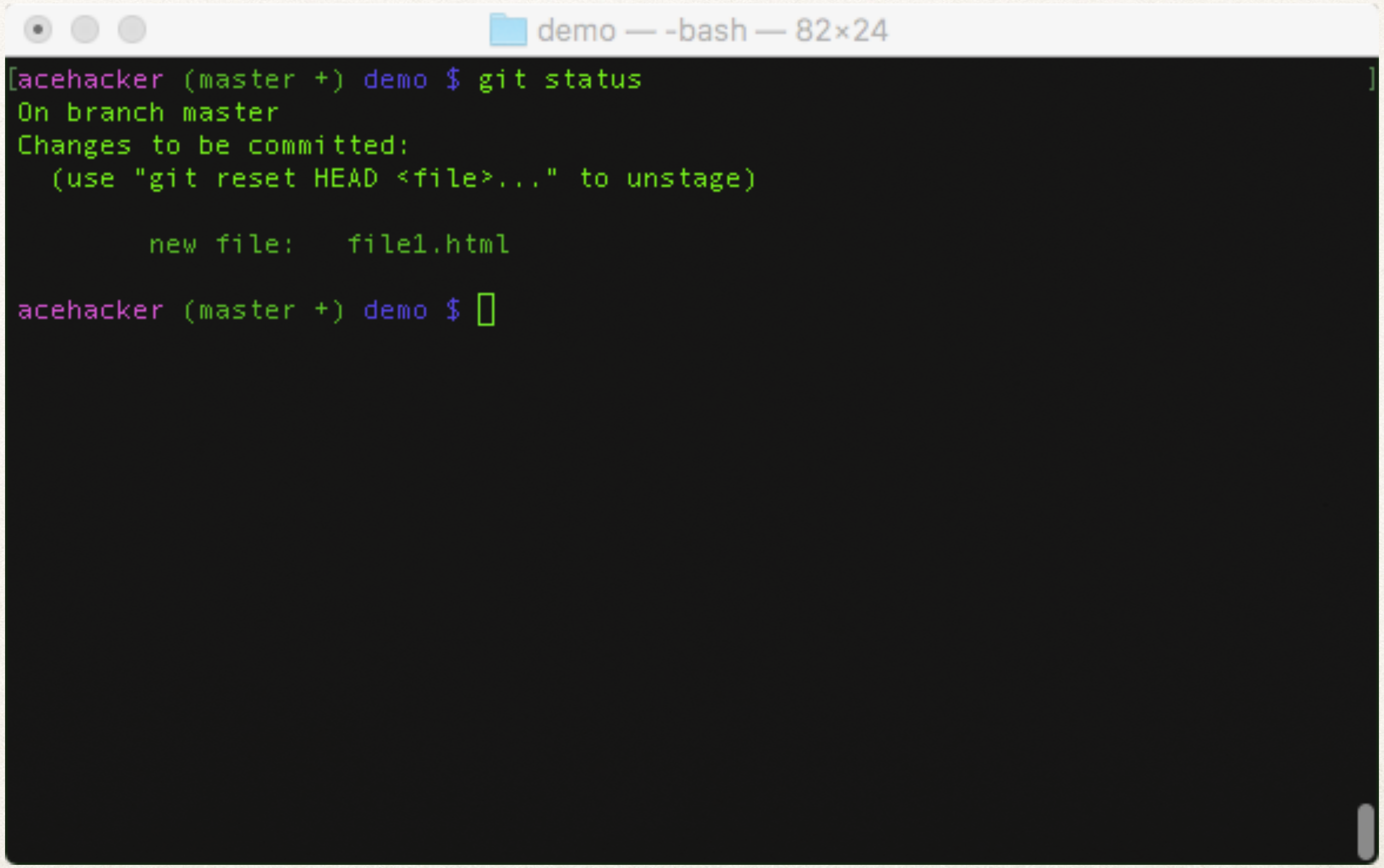
demo — -bash — 82×24

```
[acehacker (master) demo $ git status                                            ]
 On branch master
 Untracked files:
   (use "git add <file>..." to include in what will be committed)

         file1.html

 nothing added to commit but untracked files present (use "git add" to track)
[acehacker (master) demo $ git add file1.html                                    ]
 acehacker (master +) demo $ 
```

Git

Check the Status again

`git status`



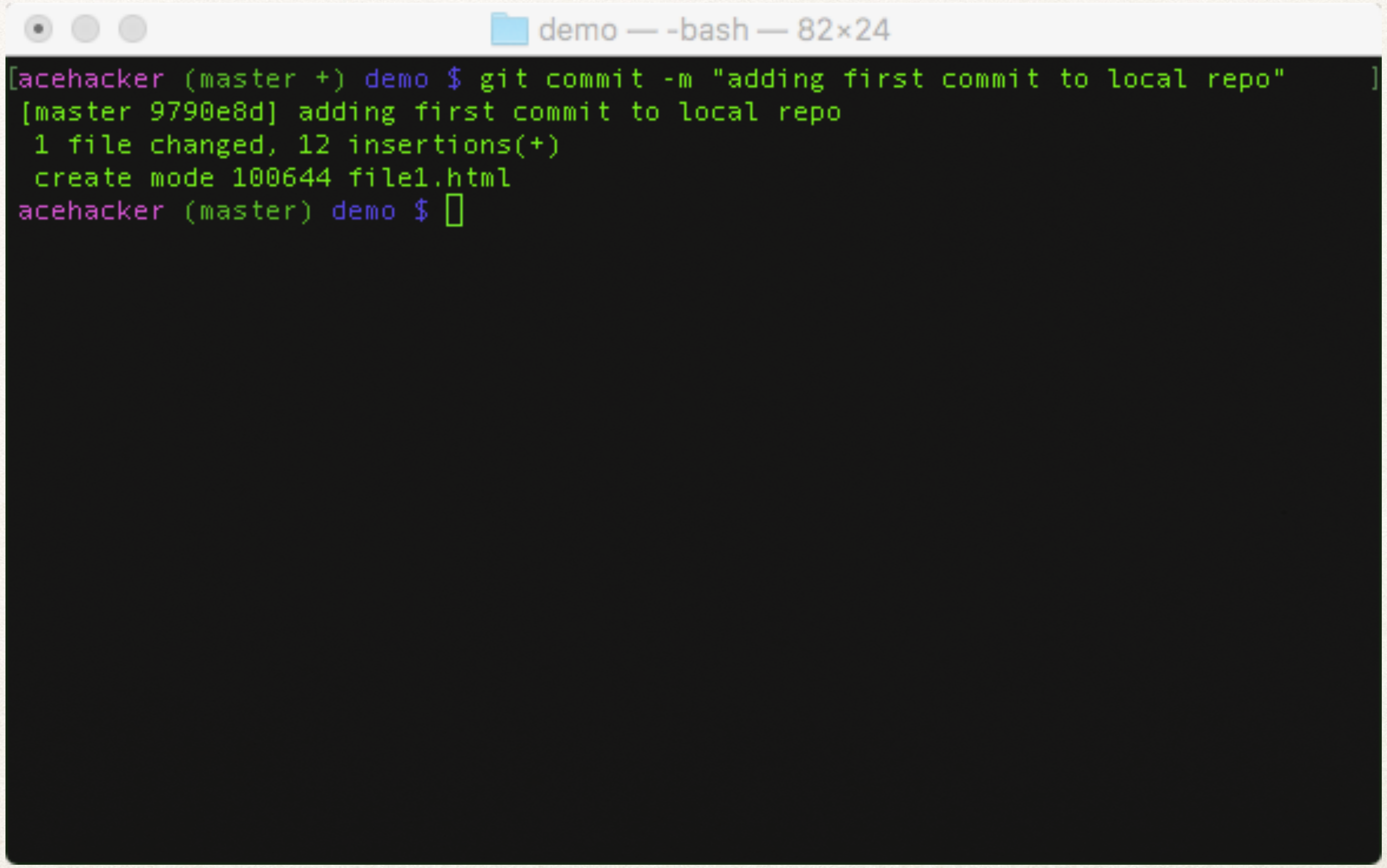
demo — -bash — 82×24

```
[acehacker (master +) demo $ git status                                          ]
 On branch master
 Changes to be committed:
   (use "git reset HEAD <file>..." to unstage)

         new file:   file1.html

 acehacker (master +) demo $ 
```

Git

Now you can commit to Git local repository

`git commit -m "<message>"`



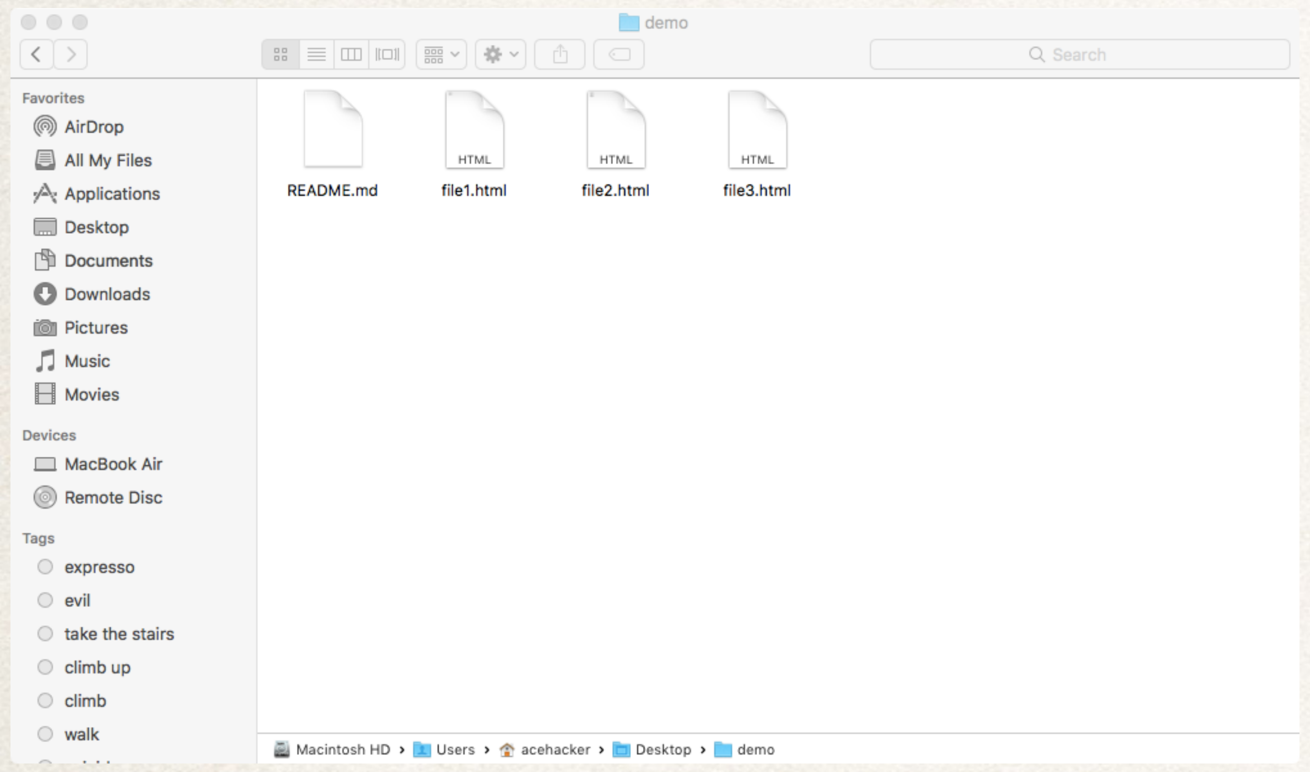
demo — -bash — 82×24

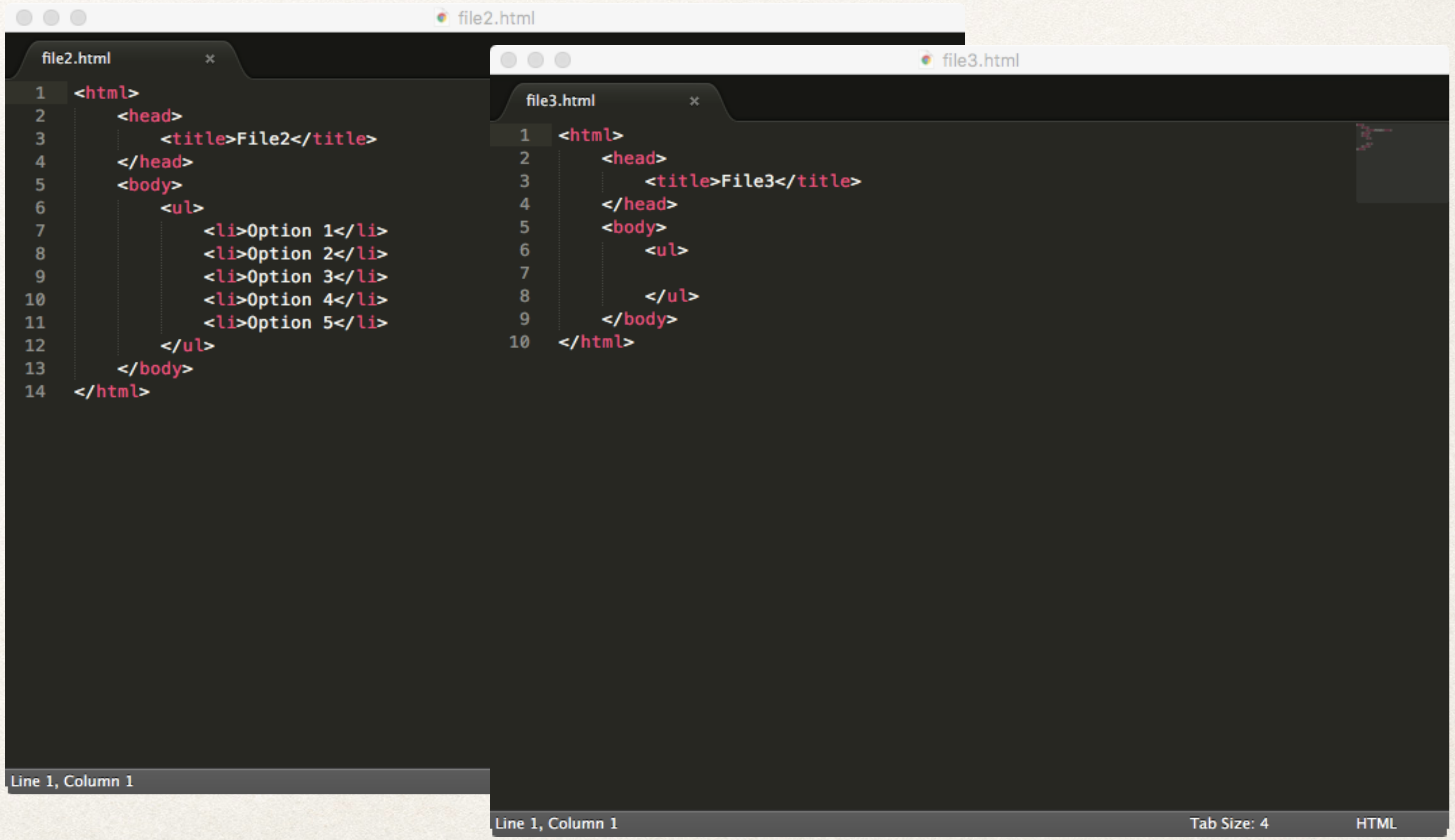
```
[acehacker (master +) demo $ git commit -m "adding first commit to local repo"     ]
 [master 9790e8d] adding first commit to local repo
  1 file changed, 12 insertions(+)
  create mode 100644 file1.html
 acehacker (master) demo $ 
```

Git

How to commit multiples at once

Add some more files to your local folder





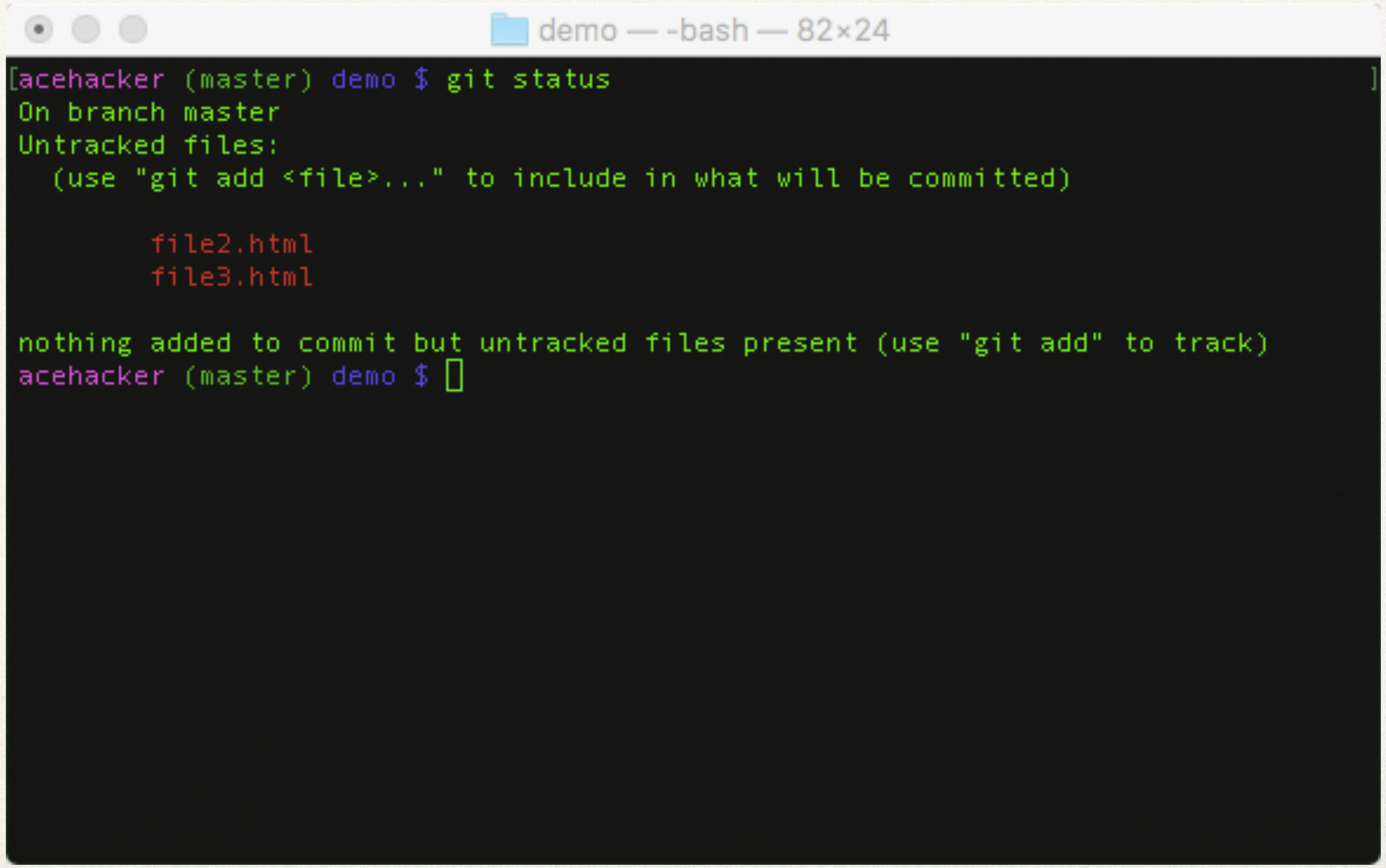
```html
<html>
    <head>
        <title>File2</title>
    </head>
    <body>
        <ul>
            <li>Option 1</li>
            <li>Option 2</li>
            <li>Option 3</li>
            <li>Option 4</li>
            <li>Option 5</li>
        </ul>
    </body>
</html>
```

```html
<html>
    <head>
        <title>File3</title>
    </head>
    <body>
        <ul>

        </ul>
    </body>
</html>
```

Git

How to commit multiples at once

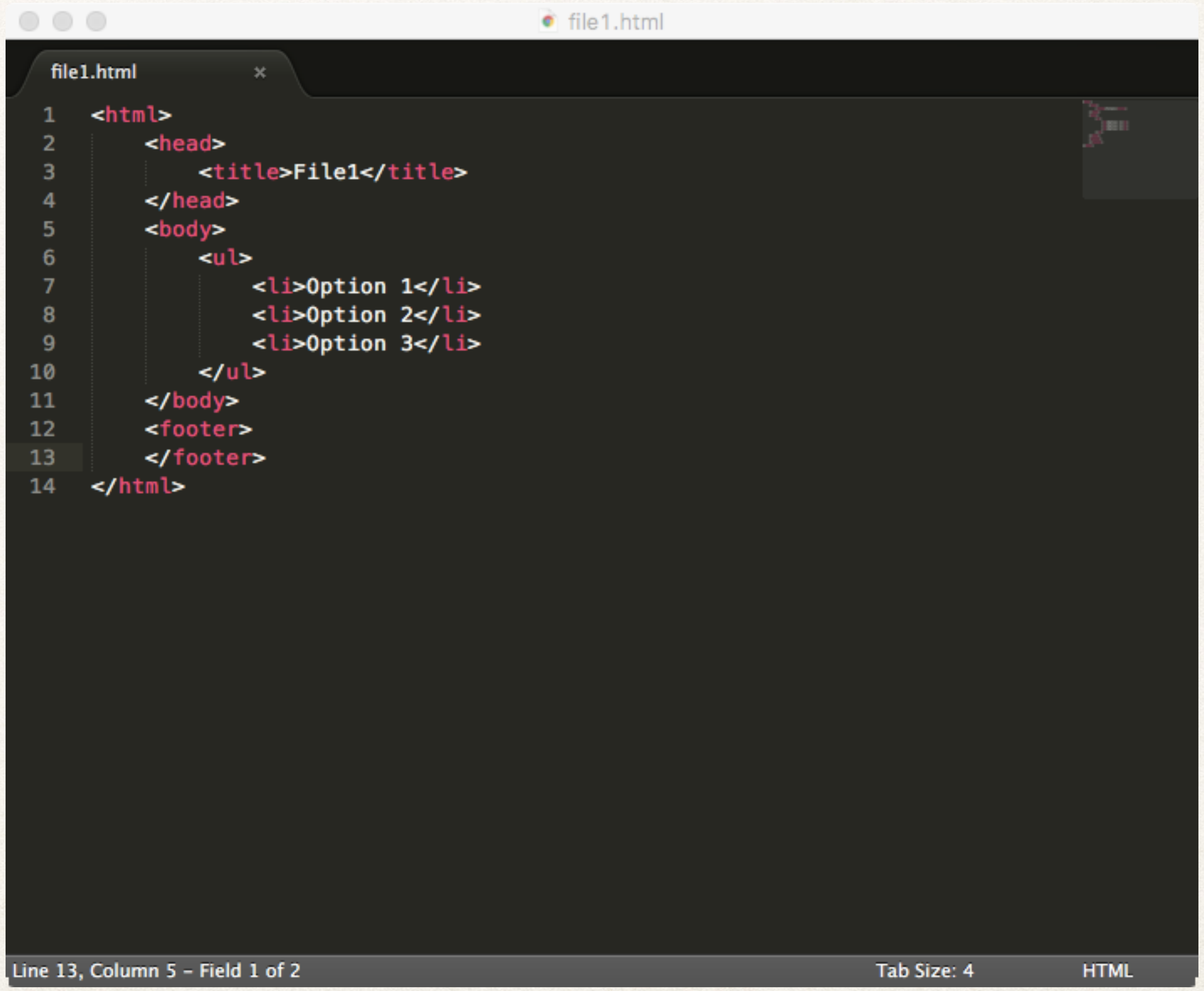
See Git Status with `git status`



```
[acehacker (master) demo $ git status                                            ]
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        file2.html
        file3.html

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

Git

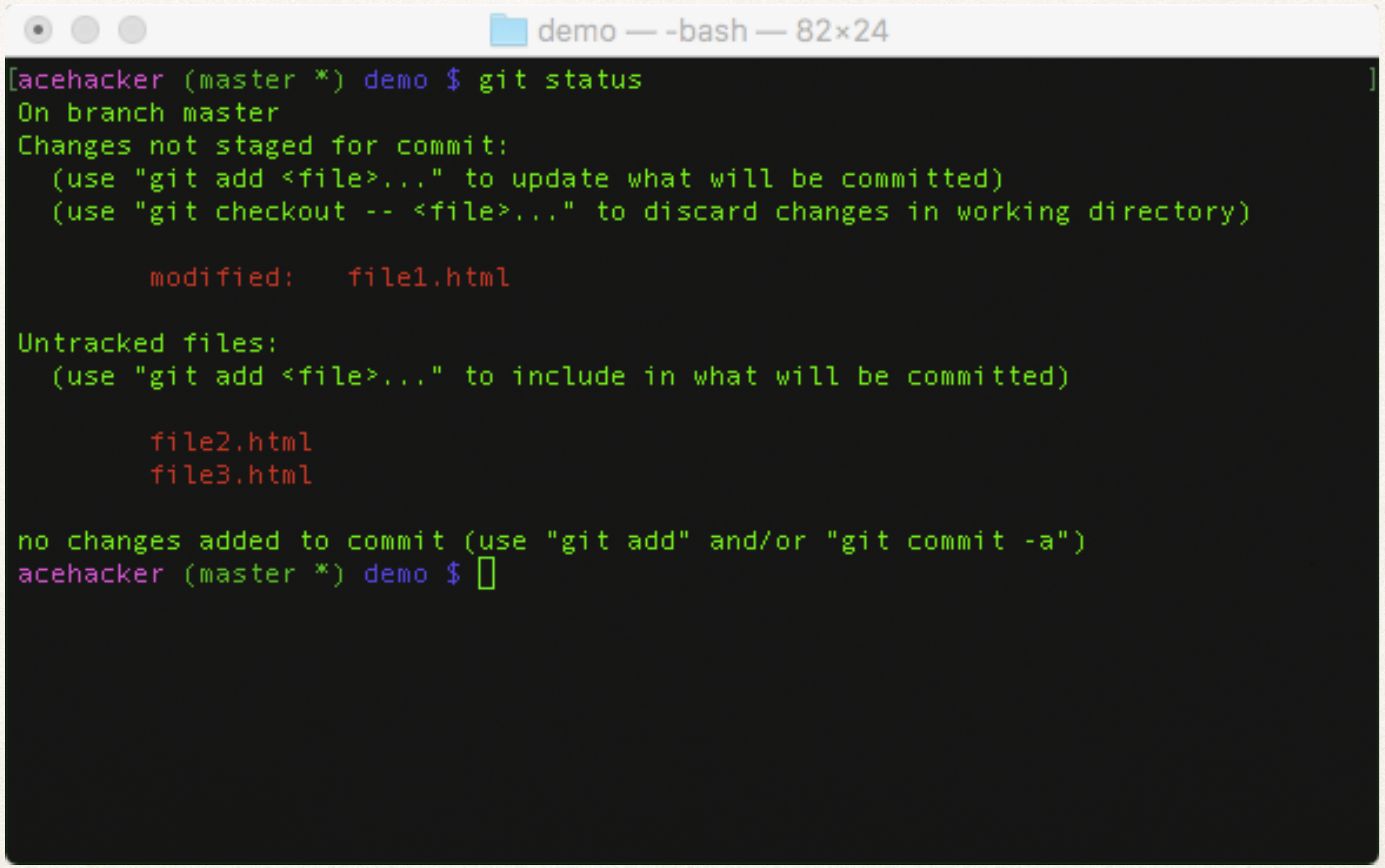
What happens if you modify existing file?



```html
<html>
    <head>
        <title>File1</title>
    </head>
    <body>
        <ul>
            <li>Option 1</li>
            <li>Option 2</li>
            <li>Option 3</li>
        </ul>
    </body>
    <footer>
    </footer>
</html>
```

Git

Check with `git status`



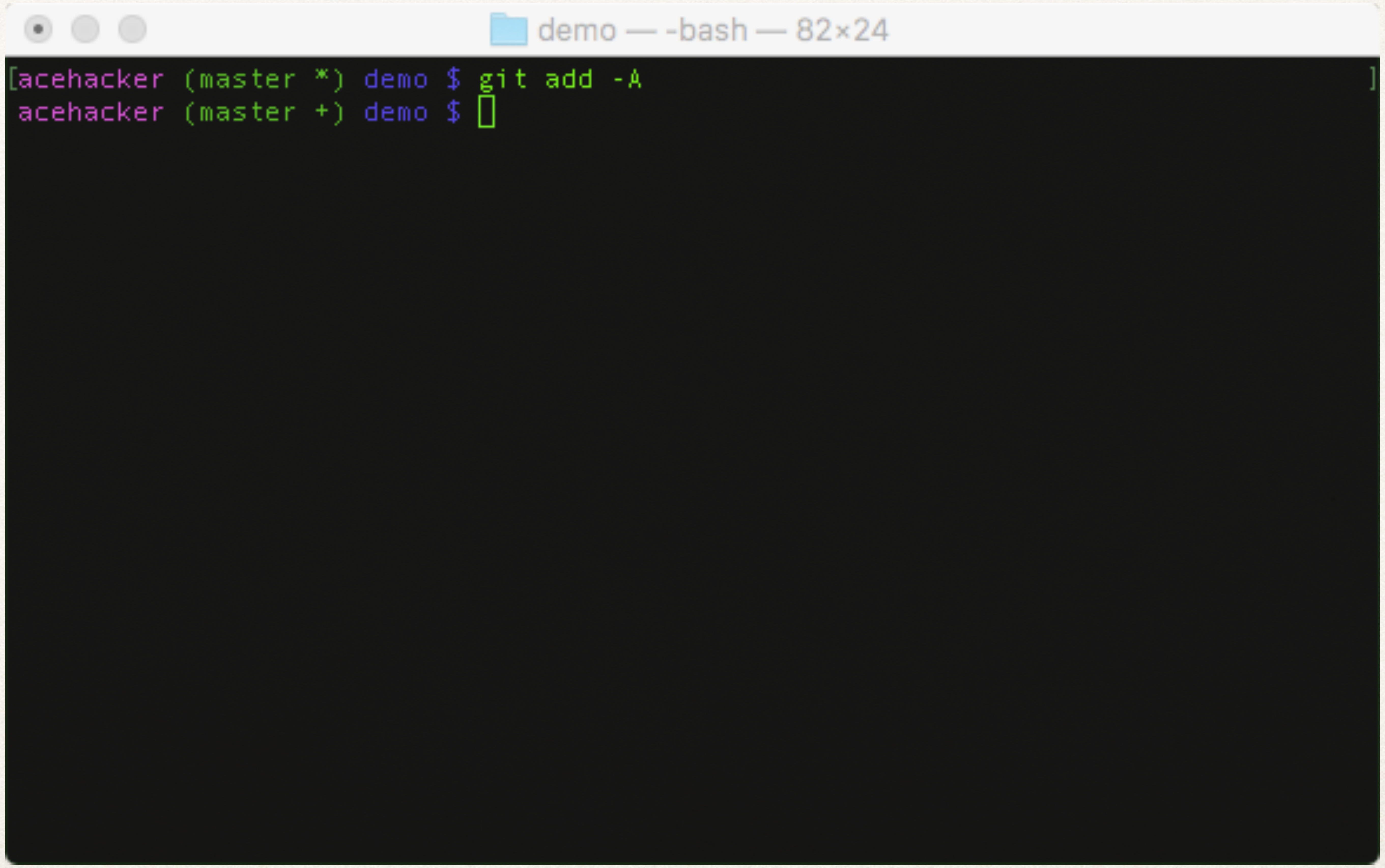
demo — -bash — 82×24

```
acehacker (master *) demo $ 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:   file1.html

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

        file2.html
        file3.html

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

Git

Add all these files to Staging Area all at one

```
git add -A
```

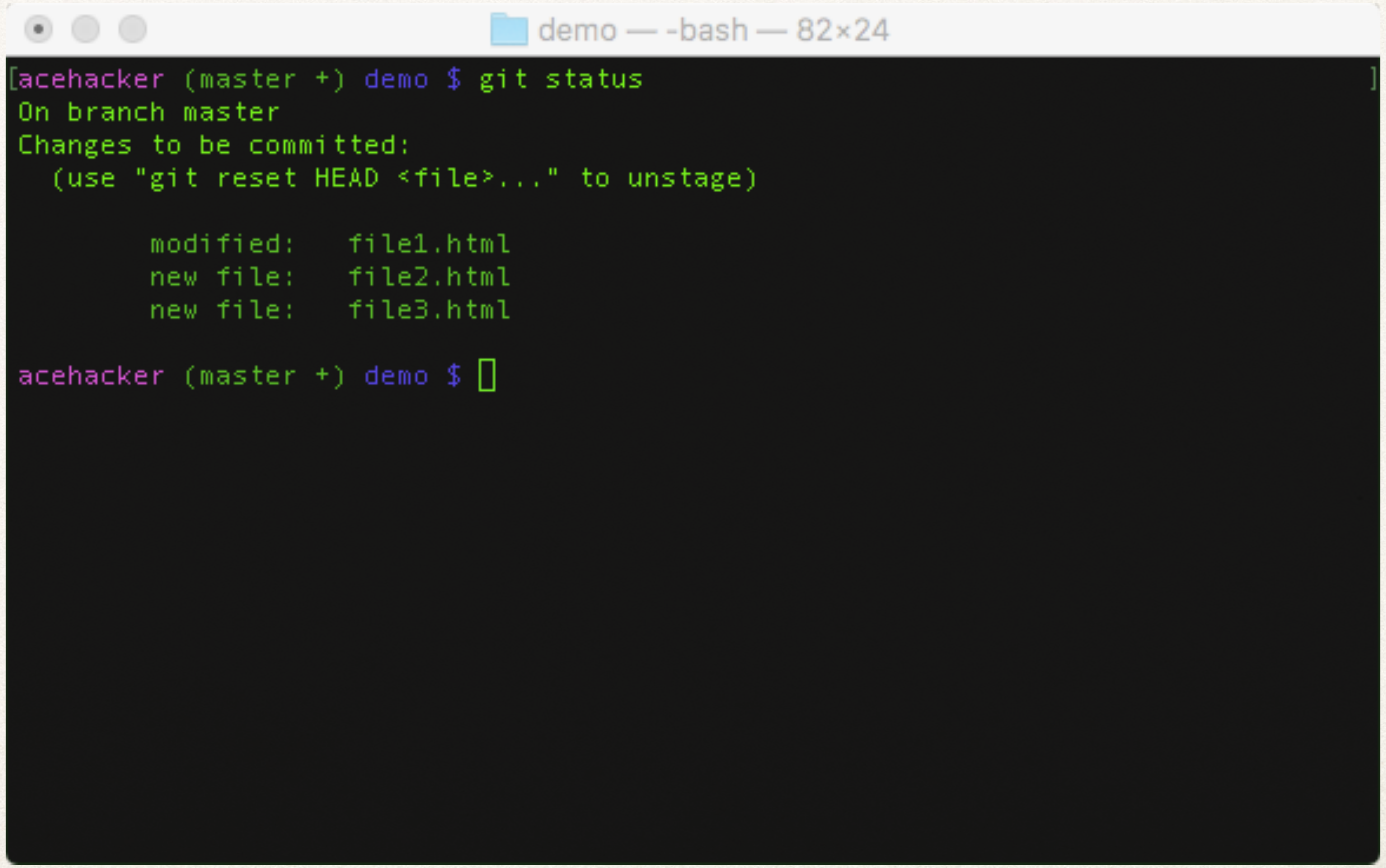


```
[acehacker (master *) demo $ git add -A                                          ]
 acehacker (master +) demo $ 
```

Git

Check the Git Status again

`git status`

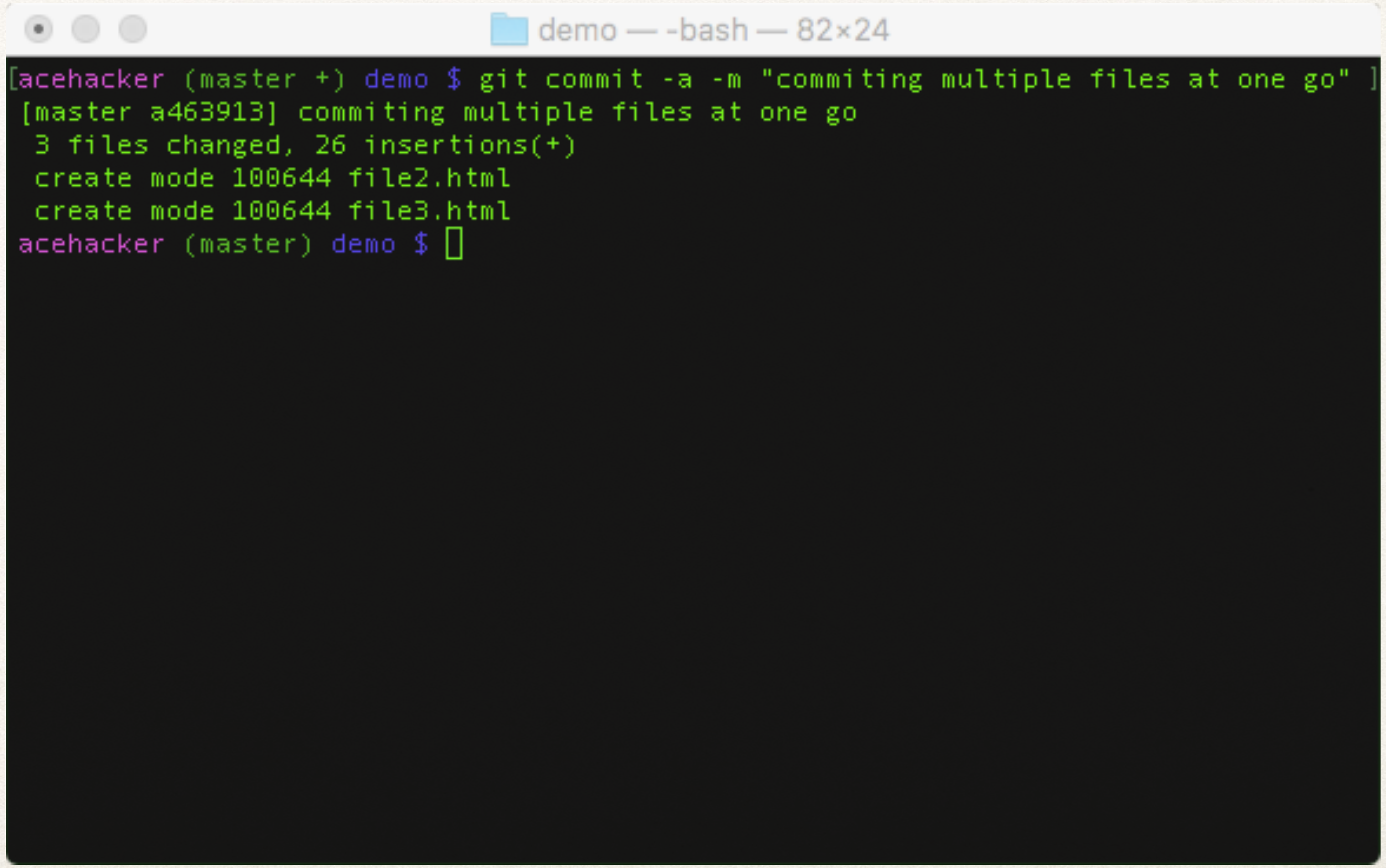

demo — -bash — 82×24

```
acehacker (master +) demo $ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   file1.html
        new file:   file2.html
        new file:   file3.html

acehacker (master +) demo $ 
```

Git

How to commit all at once

```
git commit -a -m "<message>"
```

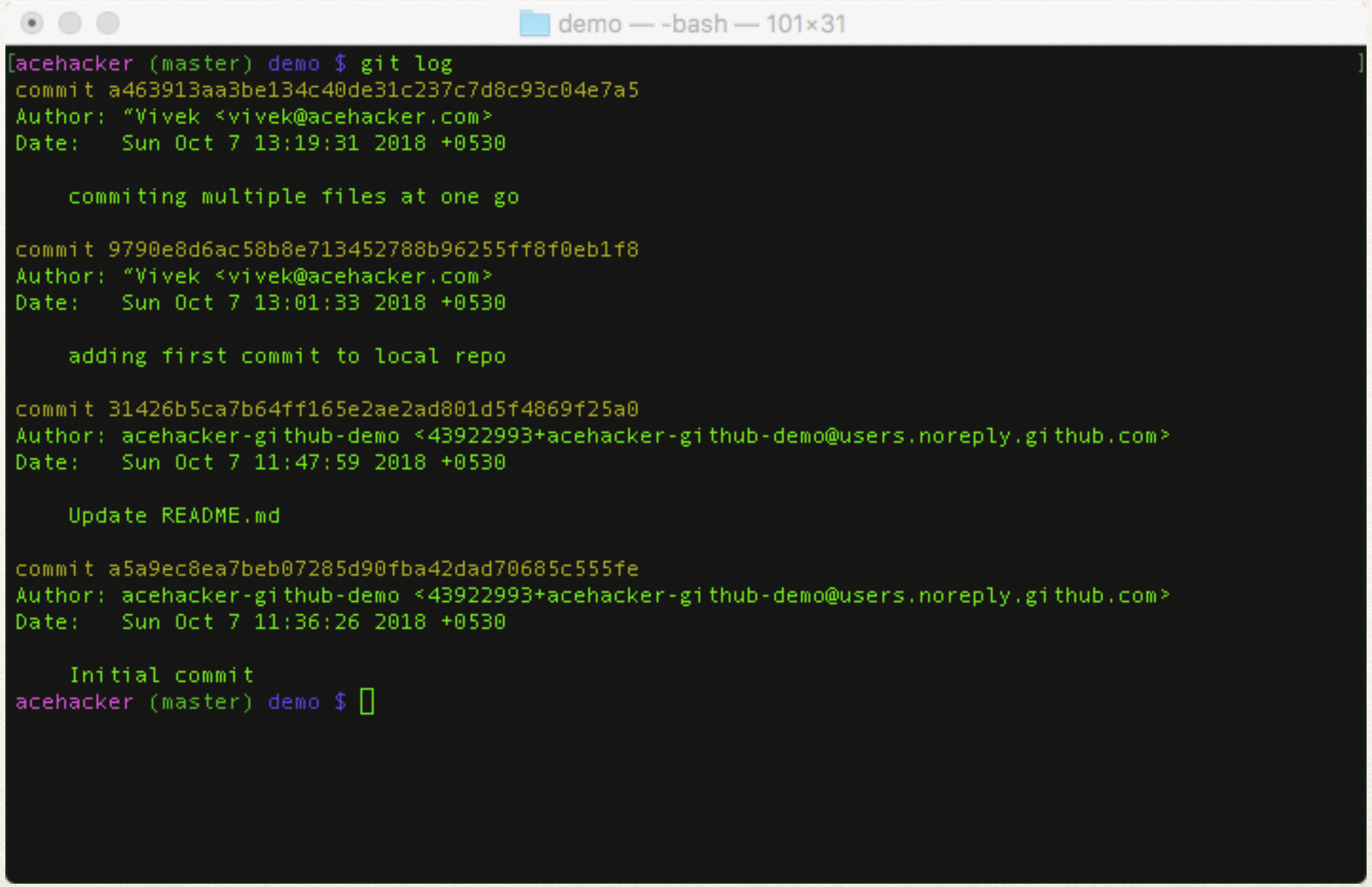


```
[acehacker (master +) demo $ git commit -a -m "commiting multiple files at one go" ]
 [master a463913] commiting multiple files at one go
  3 files changed, 26 insertions(+)
  create mode 100644 file2.html
  create mode 100644 file3.html
 acehacker (master) demo $ 
```

Git

How does Git logs all these changes?

`git log`

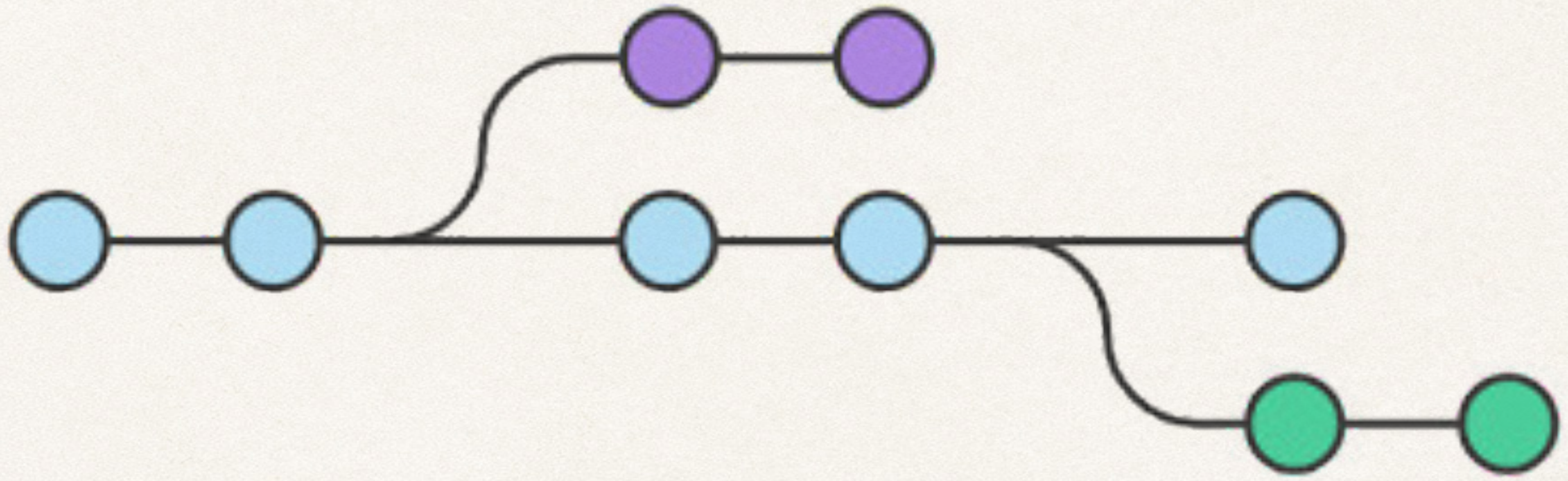


```
acehacker (master) demo $ git log
commit a463913aa3be134c40de31c237c7d8c93c04e7a5
Author: "Vivek <vivek@acehacker.com>
Date:   Sun Oct 7 13:19:31 2018 +0530

    commiting multiple files at one go

commit 9790e8d6ac58b8e713452788b96255ff8f0eb1f8
Author: "Vivek <vivek@acehacker.com>
Date:   Sun Oct 7 13:01:33 2018 +0530

    adding first commit to local repo

commit 31426b5ca7b64ff165e2ae2ad801d5f4869f25a0
Author: acehacker-github-demo <43922993+acehacker-github-demo@users.noreply.github.com>
Date:   Sun Oct 7 11:47:59 2018 +0530

    Update README.md

commit a5a9ec8ea7beb07285d90fba42dad70685c555fe
Author: acehacker-github-demo <43922993+acehacker-github-demo@users.noreply.github.com>
Date:   Sun Oct 7 11:36:26 2018 +0530

    Initial commit
acehacker (master) demo $
```

Git

Non-Linear Development with

Branching



Git

Non-Linear Development
Branching

Two types of Branching:

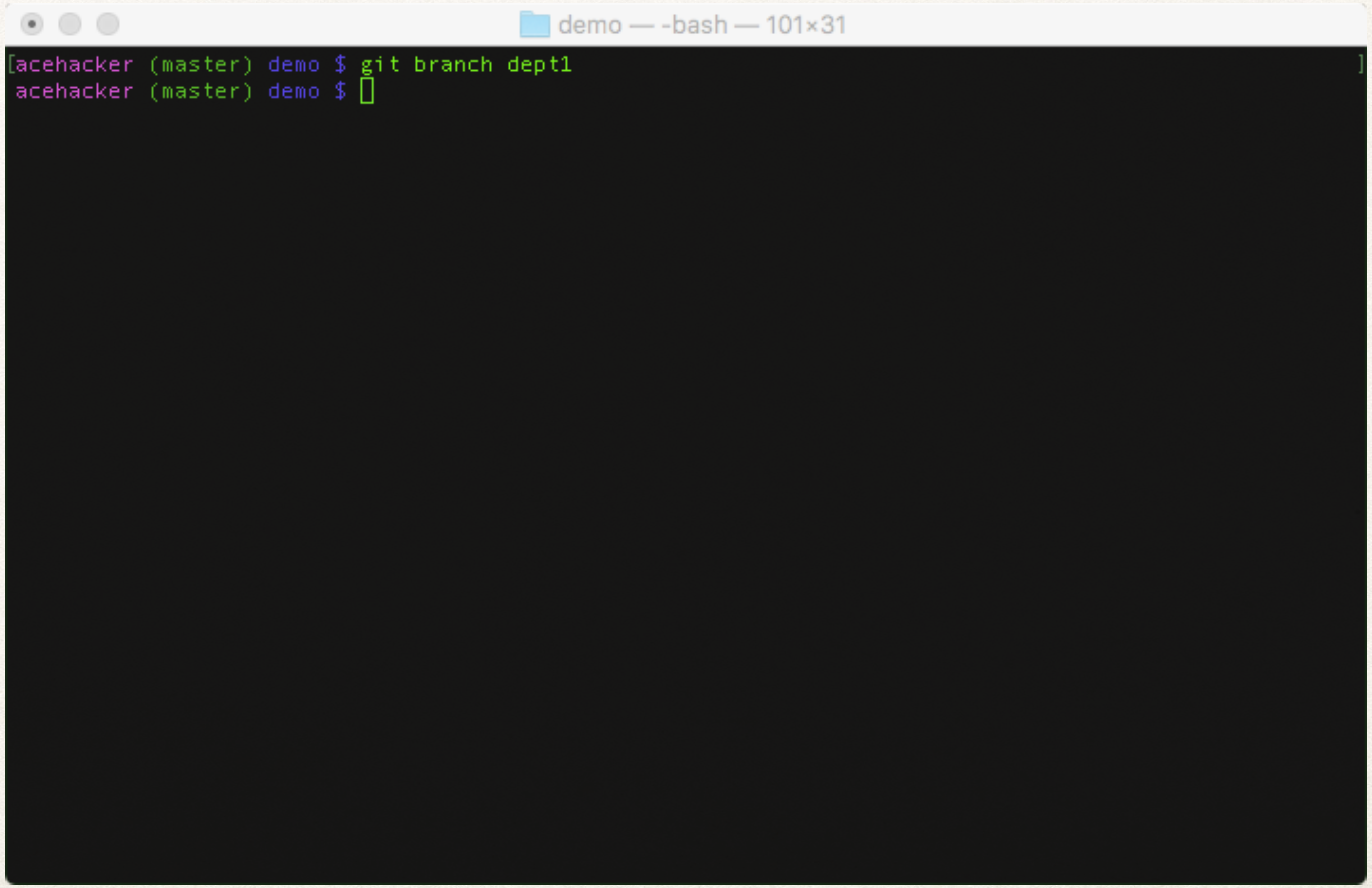
`Local Branches`

`Remote Tracking Branches`

Git

How to create branches

`git branch <branch name>`

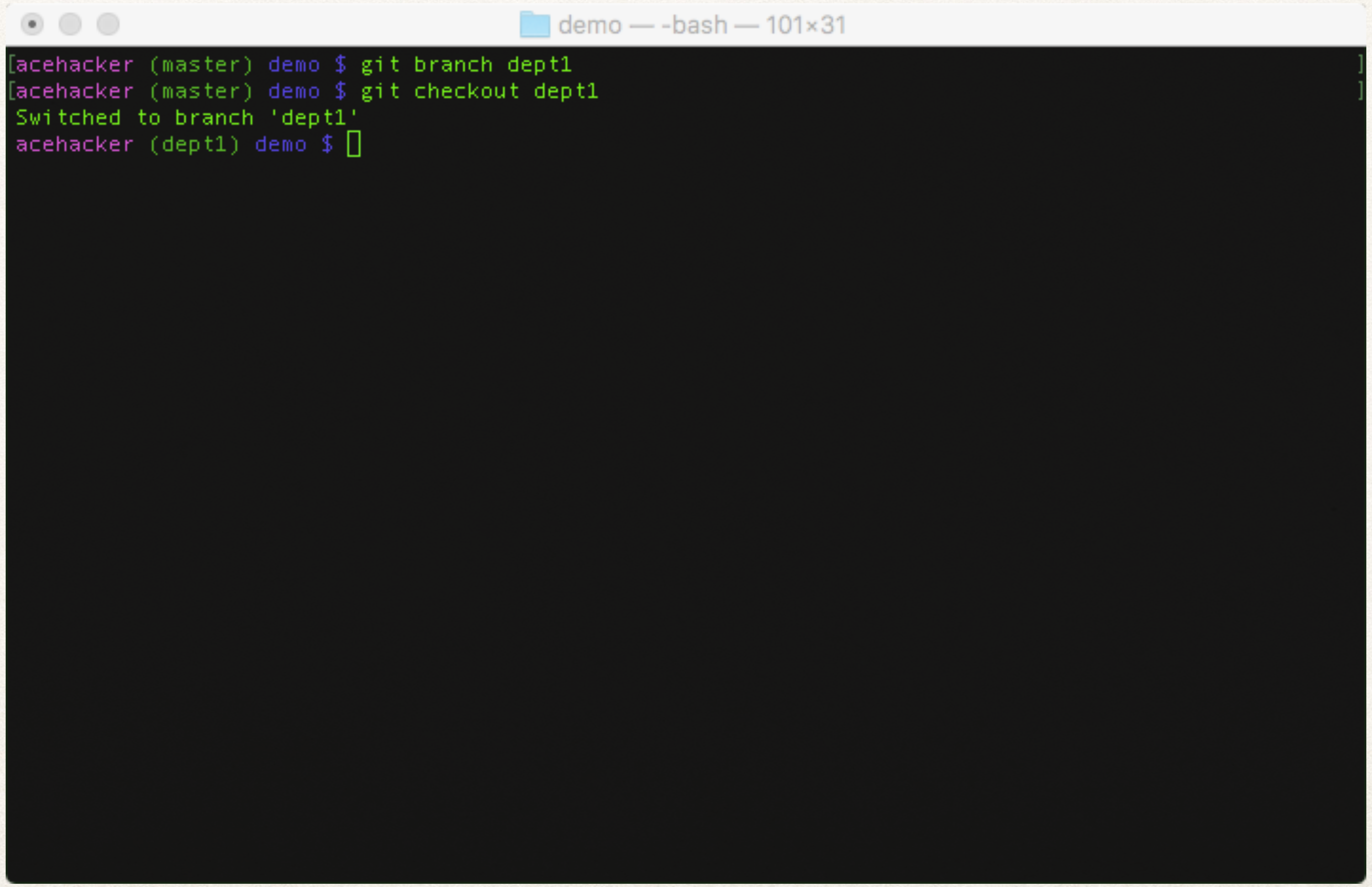


```
[acehacker (master) demo $ git branch dept1
 acehacker (master) demo $ 
```

Git

Switch branches

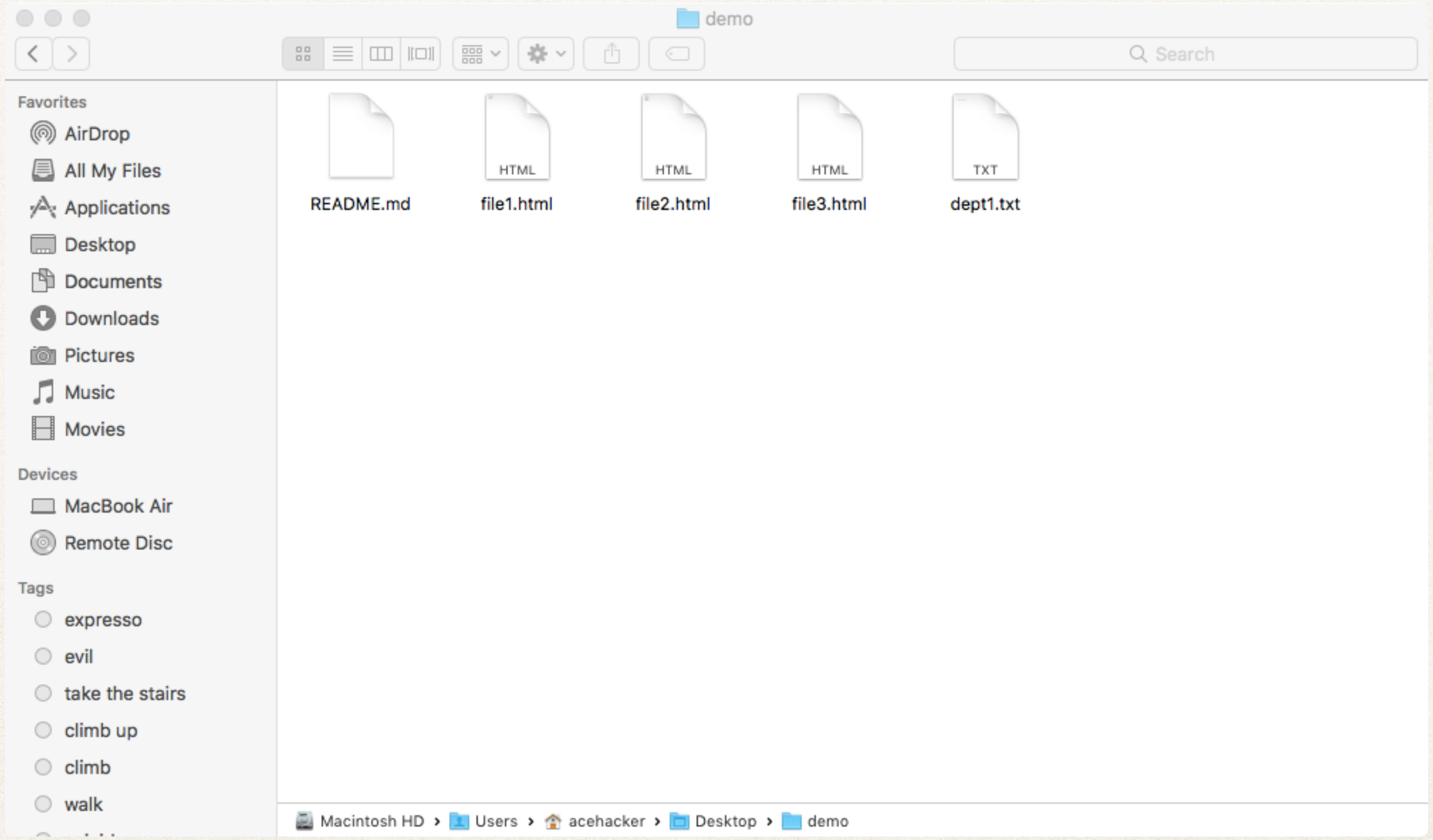
`git checkout <branch name>`



```
acehacker (master) demo $ git branch dept1
acehacker (master) demo $ git checkout dept1
Switched to branch 'dept1'
acehacker (dept1) demo $
```

Git

Create some new files - may be text files this time



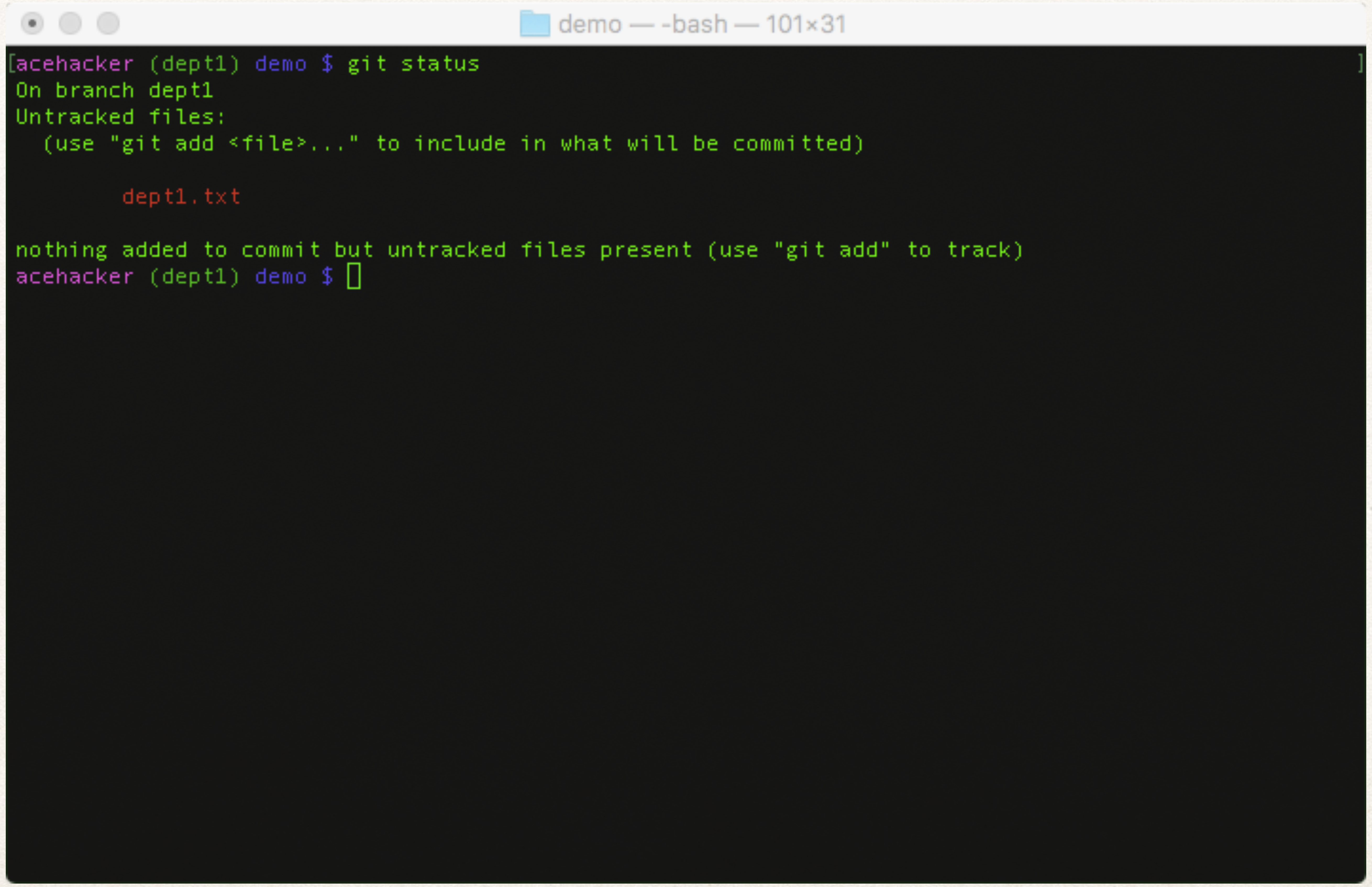
README.md
file1.html
file2.html
file3.html
dept1.txt

Macintosh HD › Users › acehacker › Desktop › demo

Git

How does your Git Status looks now?

`git status`

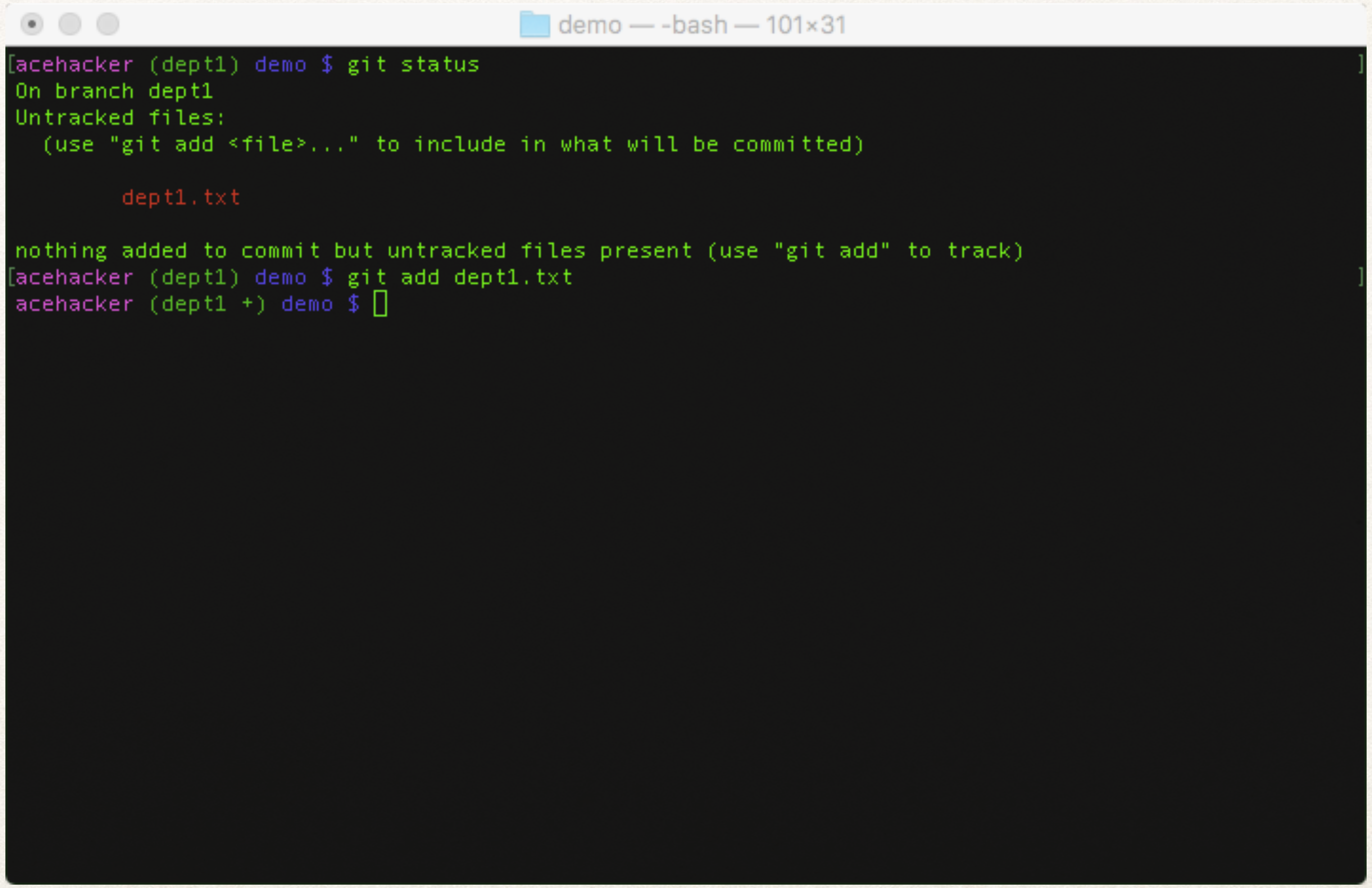


```
acehacker (dept1) demo $ git status
On branch dept1
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        dept1.txt

nothing added to commit but untracked files present (use "git add" to track)
acehacker (dept1) demo $ 
```

Git

Add this file to the Staging Area

`git add`

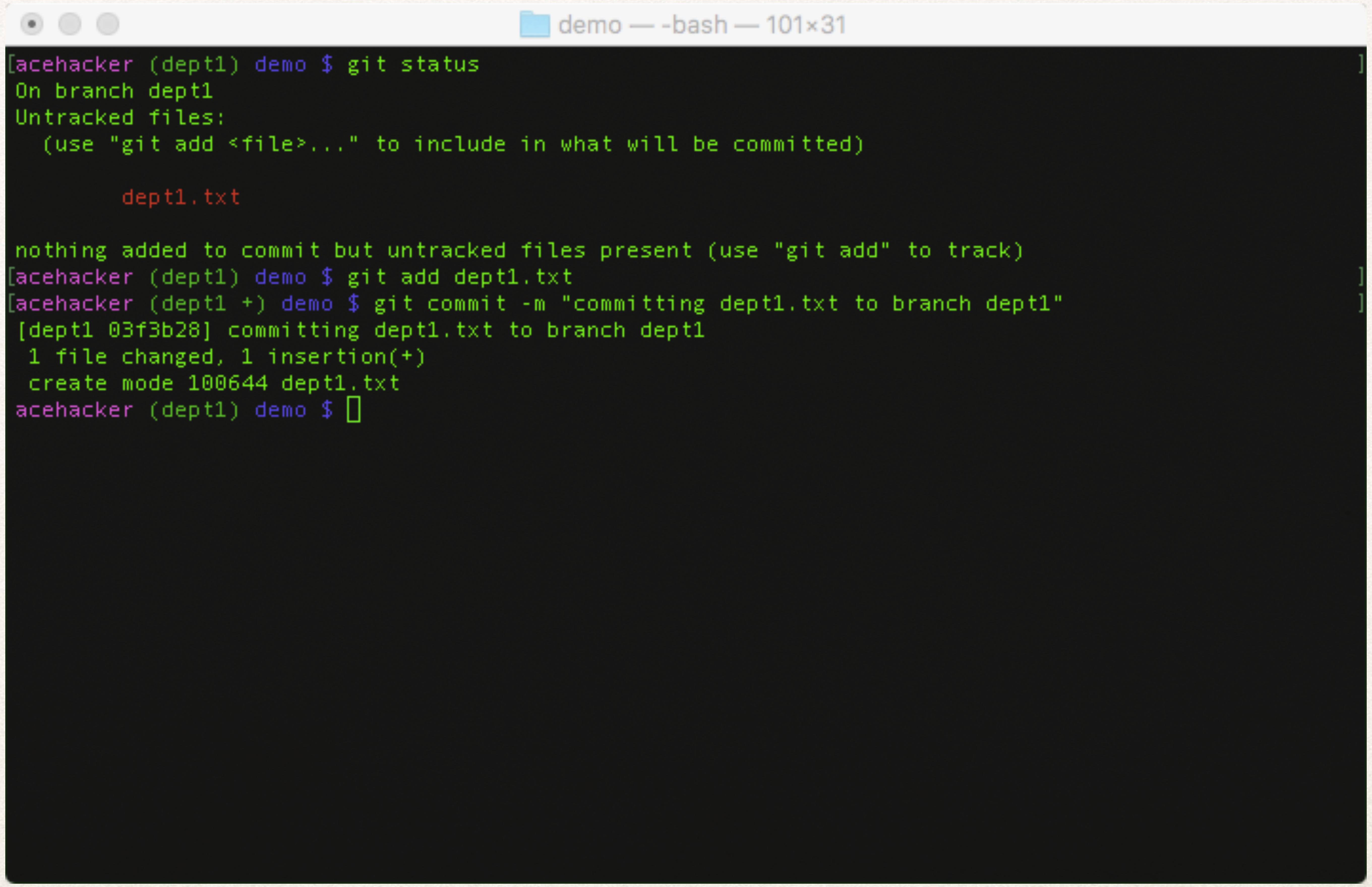


```
[acehacker (dept1) demo $ git status                                                                  ]
On branch dept1
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        dept1.txt

nothing added to commit but untracked files present (use "git add" to track)
[acehacker (dept1) demo $ git add dept1.txt                                                           ]
acehacker (dept1 +) demo $ 
```

Git

You now need to commit this file

```
git commit -m “<message>”
```

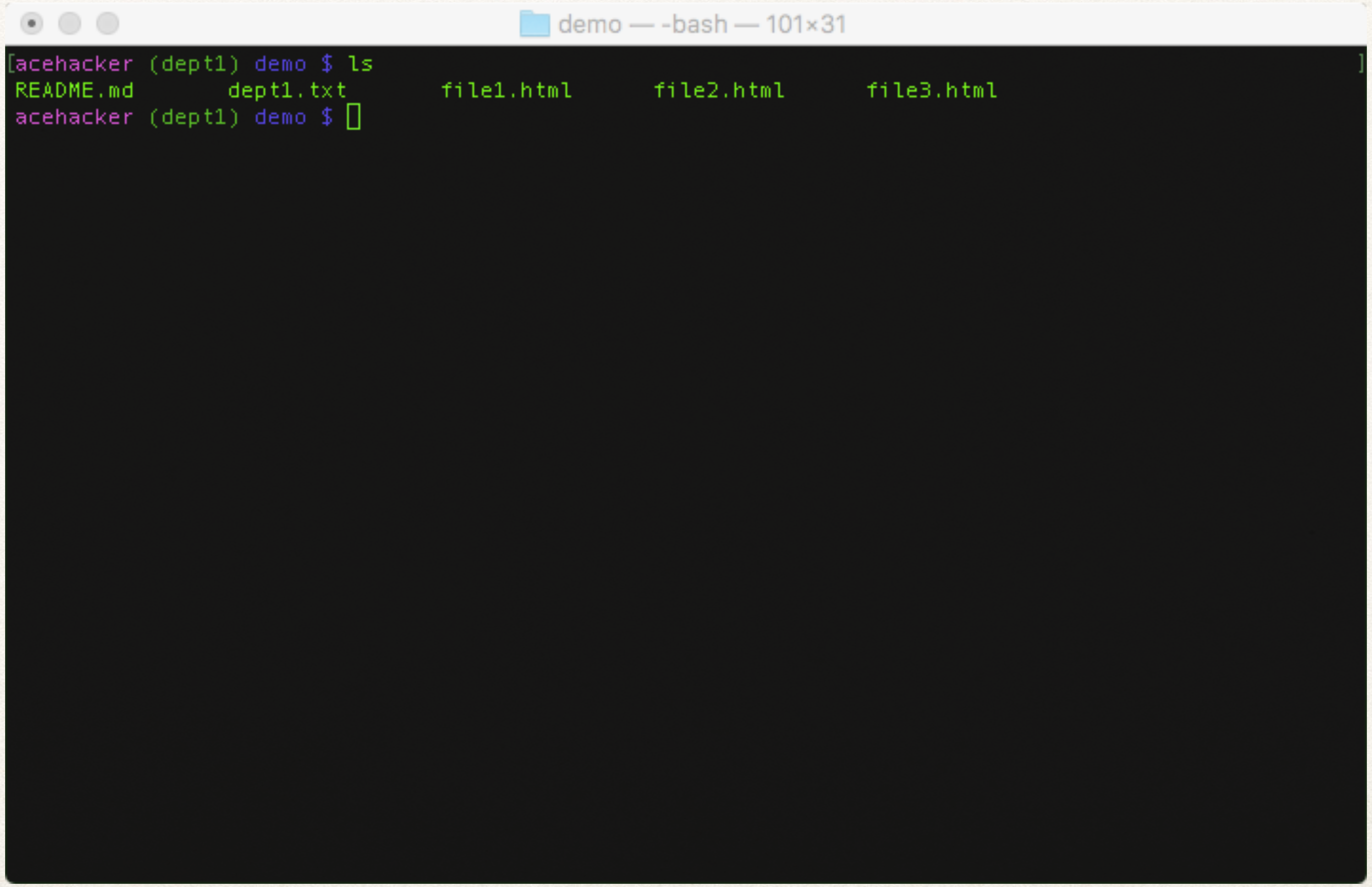


demo — -bash — 101×31

```
acehacker (dept1) demo $ git status
On branch dept1
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        dept1.txt

nothing added to commit but untracked files present (use "git add" to track)
acehacker (dept1) demo $ git add dept1.txt
acehacker (dept1 +) demo $ git commit -m "committing dept1.txt to branch dept1"
[dept1 03f3b28] committing dept1.txt to branch dept1
 1 file changed, 1 insertion(+)
 create mode 100644 dept1.txt
acehacker (dept1) demo $
```

Git

Remember that you are making changes in a branch.
See all files in the branch with `ls`

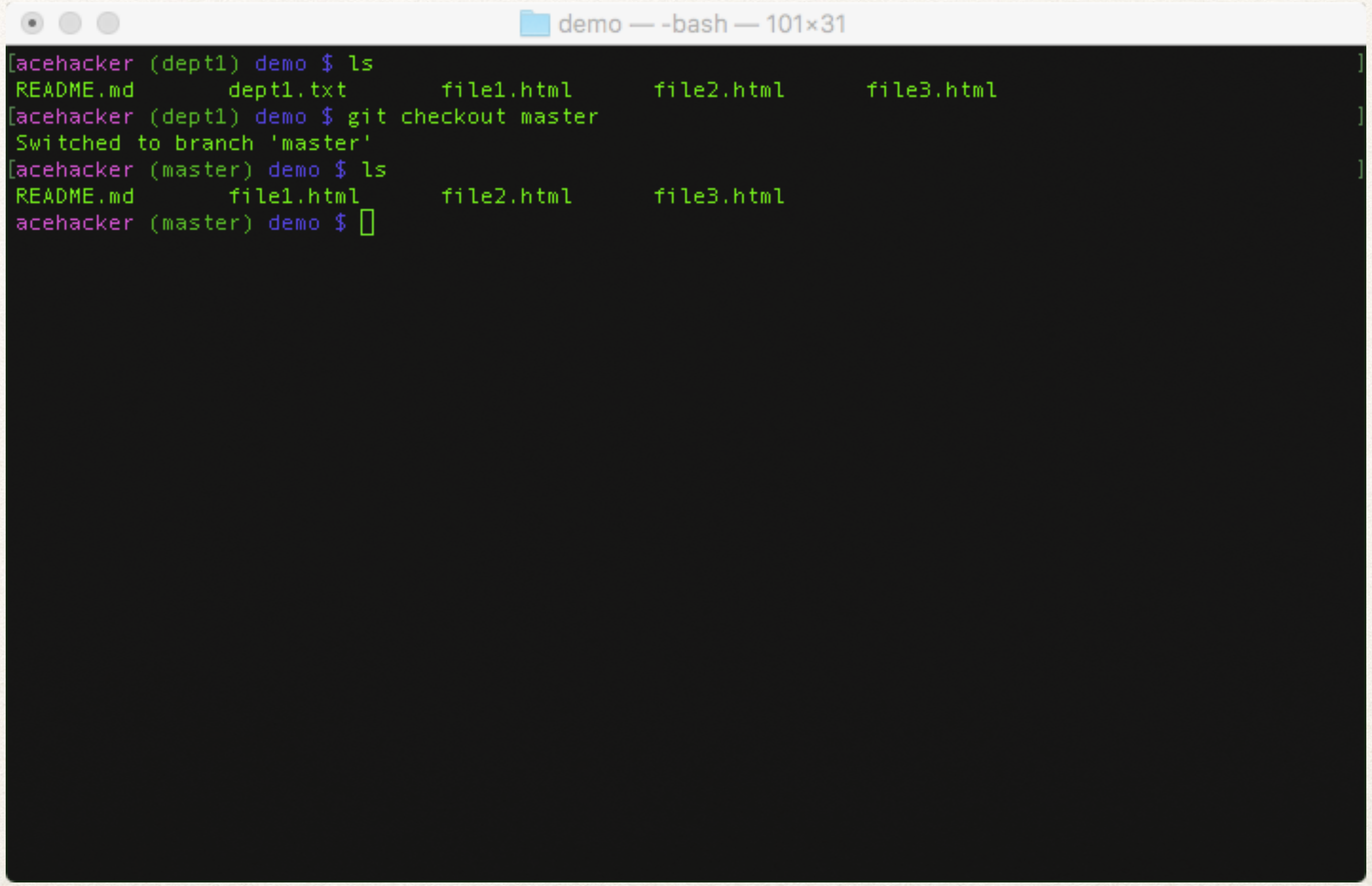


```
[acehacker (dept1) demo $ ls
 README.md        dept1.txt        file1.html       file2.html       file3.html
 acehacker (dept1) demo $ 
```

Git

Now move back to the master branch.
`git checkout master`

See all files in master with `ls`

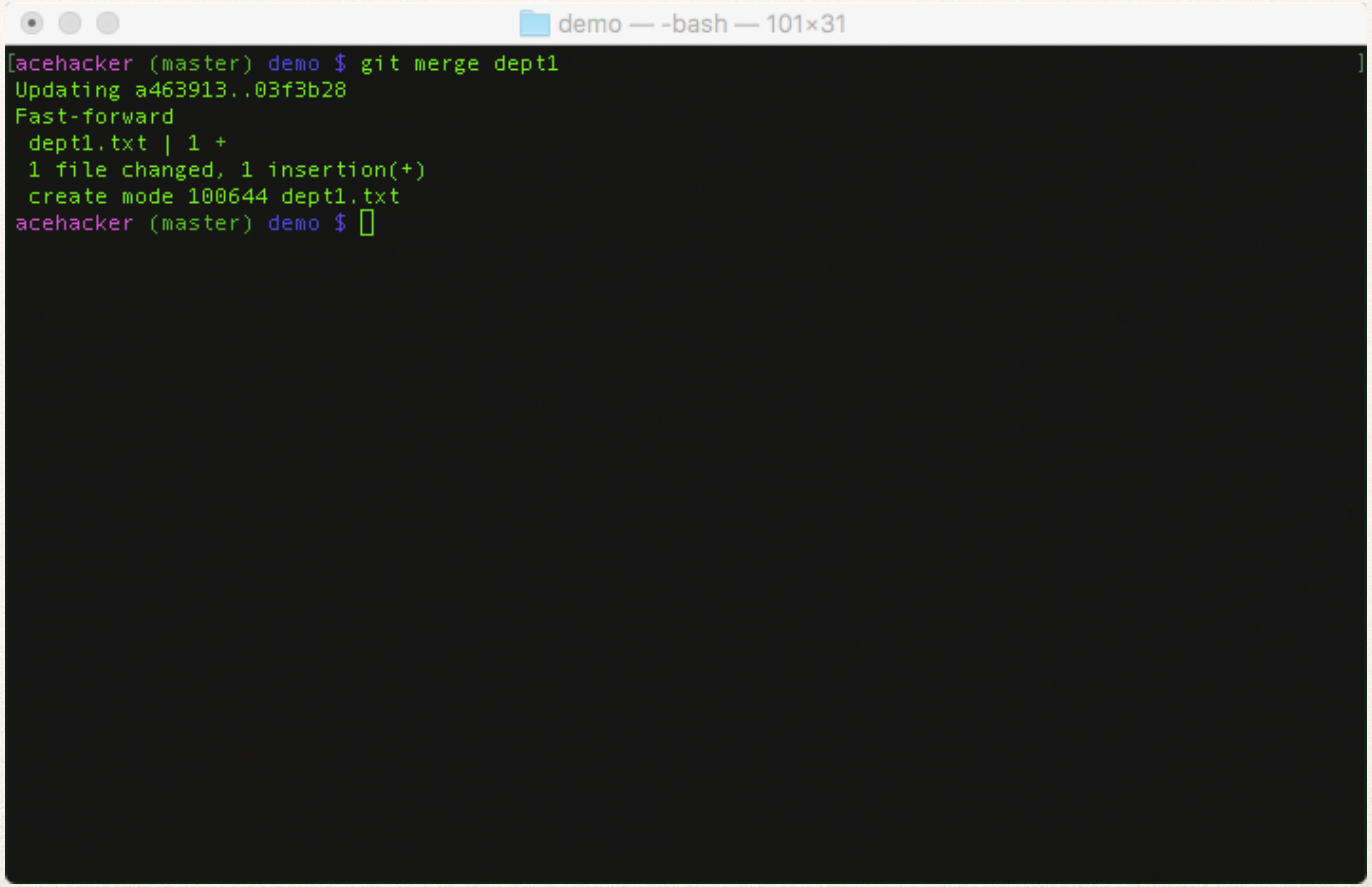


```
[acehacker (dept1) demo $ ls                                                                        ]
 README.md        dept1.txt        file1.html       file2.html       file3.html
[acehacker (dept1) demo $ git checkout master                                                       ]
 Switched to branch 'master'
[acehacker (master) demo $ ls                                                                       ]
 README.md        file1.html       file2.html       file3.html
 acehacker (master) demo $ 
```

Git

Merging

`git merge` *`<branch name>`*

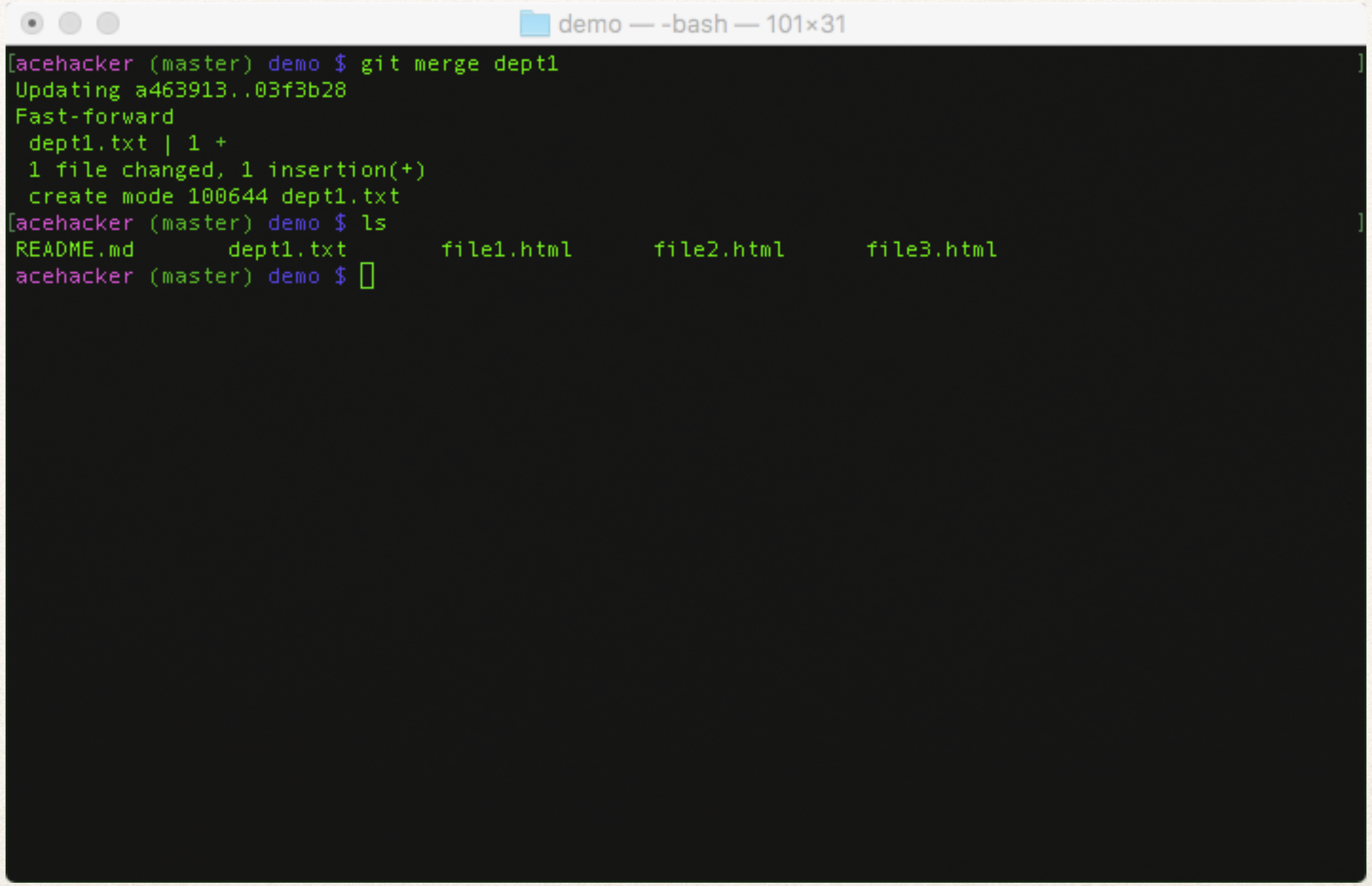


```
acehacker (master) demo $ git merge dept1
Updating a463913..03f3b28
Fast-forward
 dept1.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 dept1.txt
acehacker (master) demo $ 
```

Git

Merging

List out the files in master with `ls` to see file in branch show up here

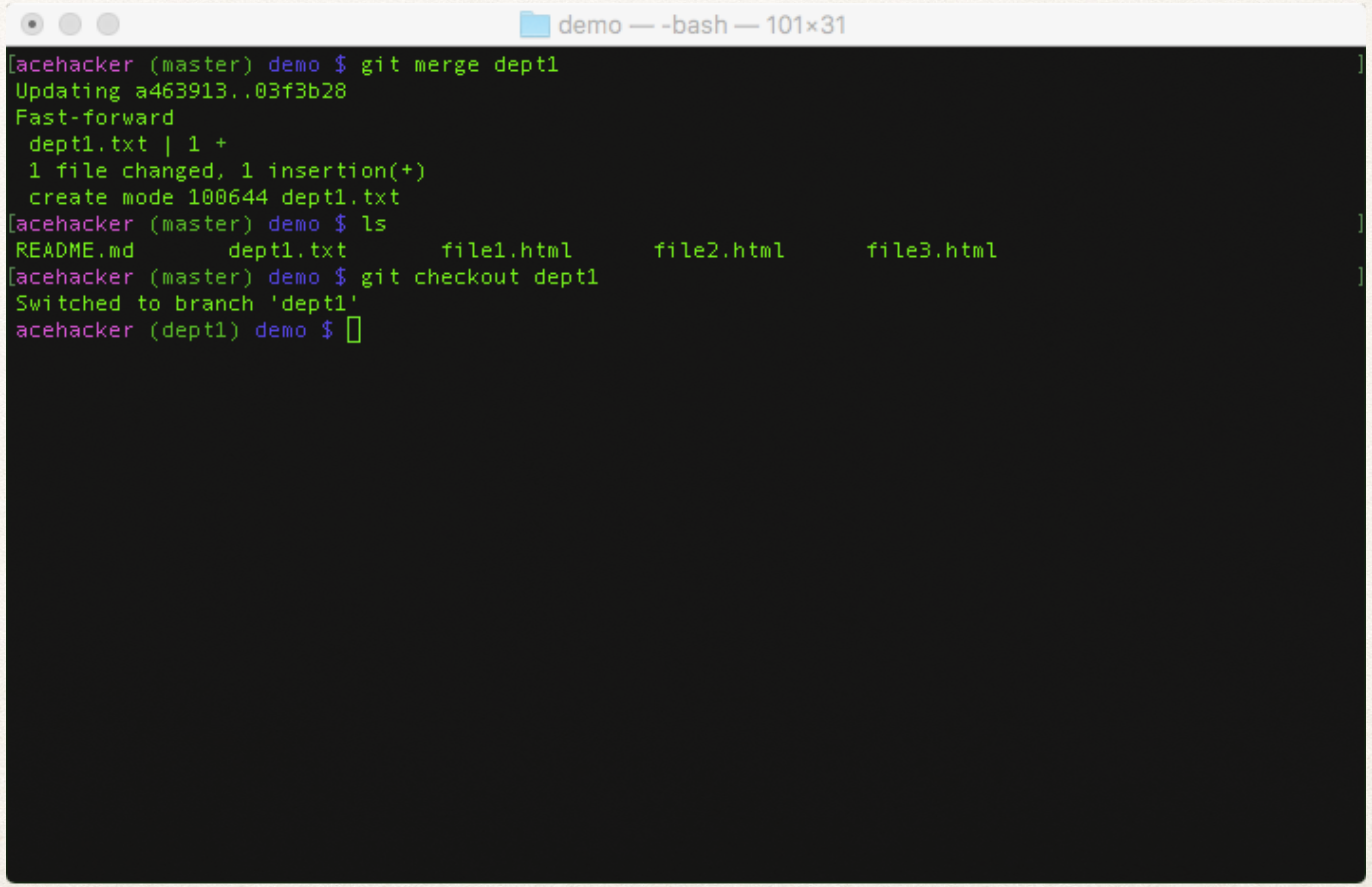


```
demo — -bash — 101×31
[acehacker (master) demo $ git merge dept1
Updating a463913..03f3b28
Fast-forward
 dept1.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 dept1.txt
[acehacker (master) demo $ ls
README.md       dept1.txt       file1.html      file2.html      file3.html
acehacker (master) demo $
```

Git

Remember that whatever you do in your branch will not reflect in master until you merge it. Try making some changes in the branch now.

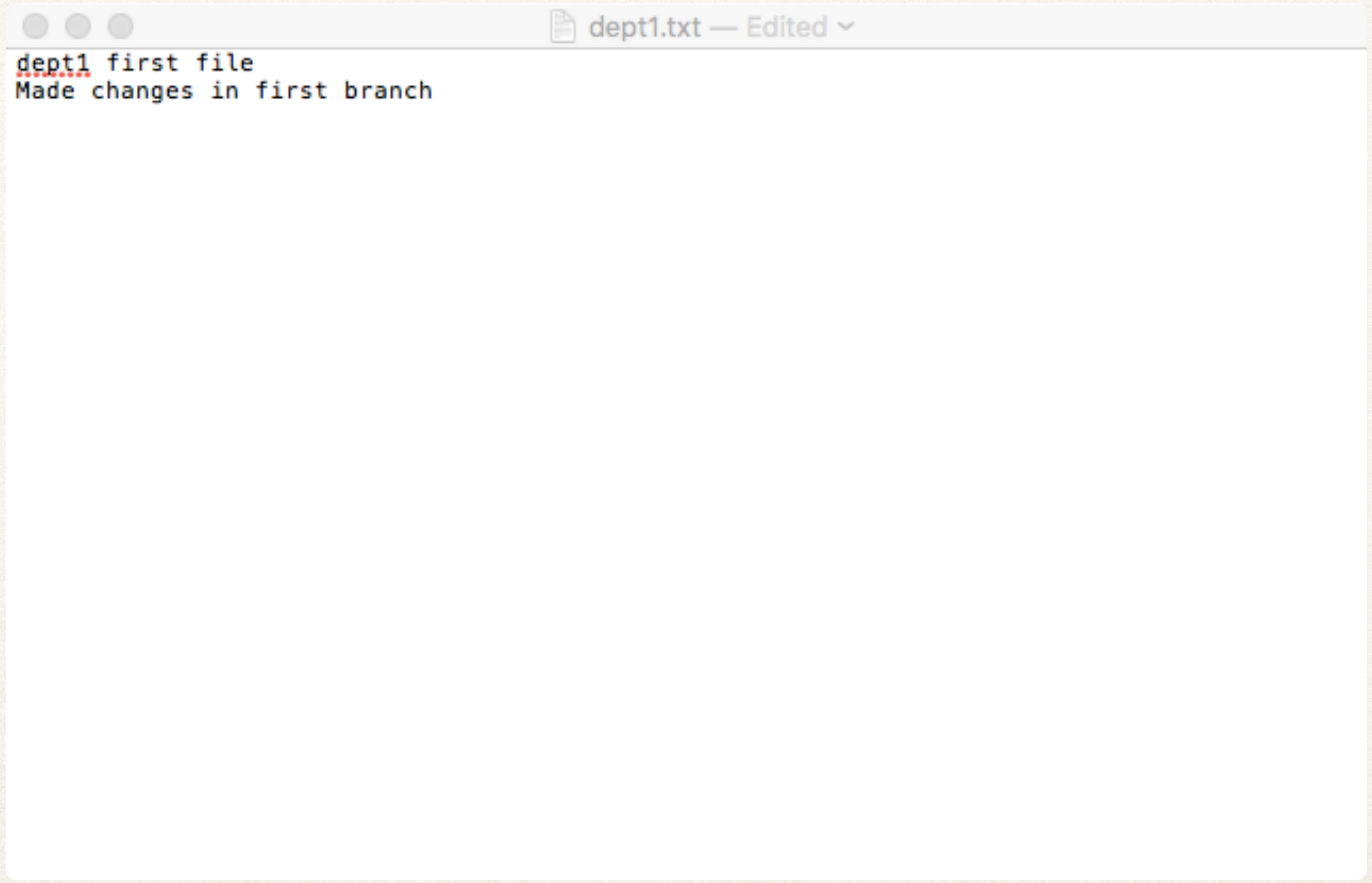
Go back to the branch using `git checkout <branch name>`



```
[acehacker (master) demo $ git merge dept1
Updating a463913..03f3b28
Fast-forward
 dept1.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 dept1.txt
[acehacker (master) demo $ ls
README.md       dept1.txt       file1.html      file2.html      file3.html
[acehacker (master) demo $ git checkout dept1
Switched to branch 'dept1'
acehacker (dept1) demo $ 
```

Git

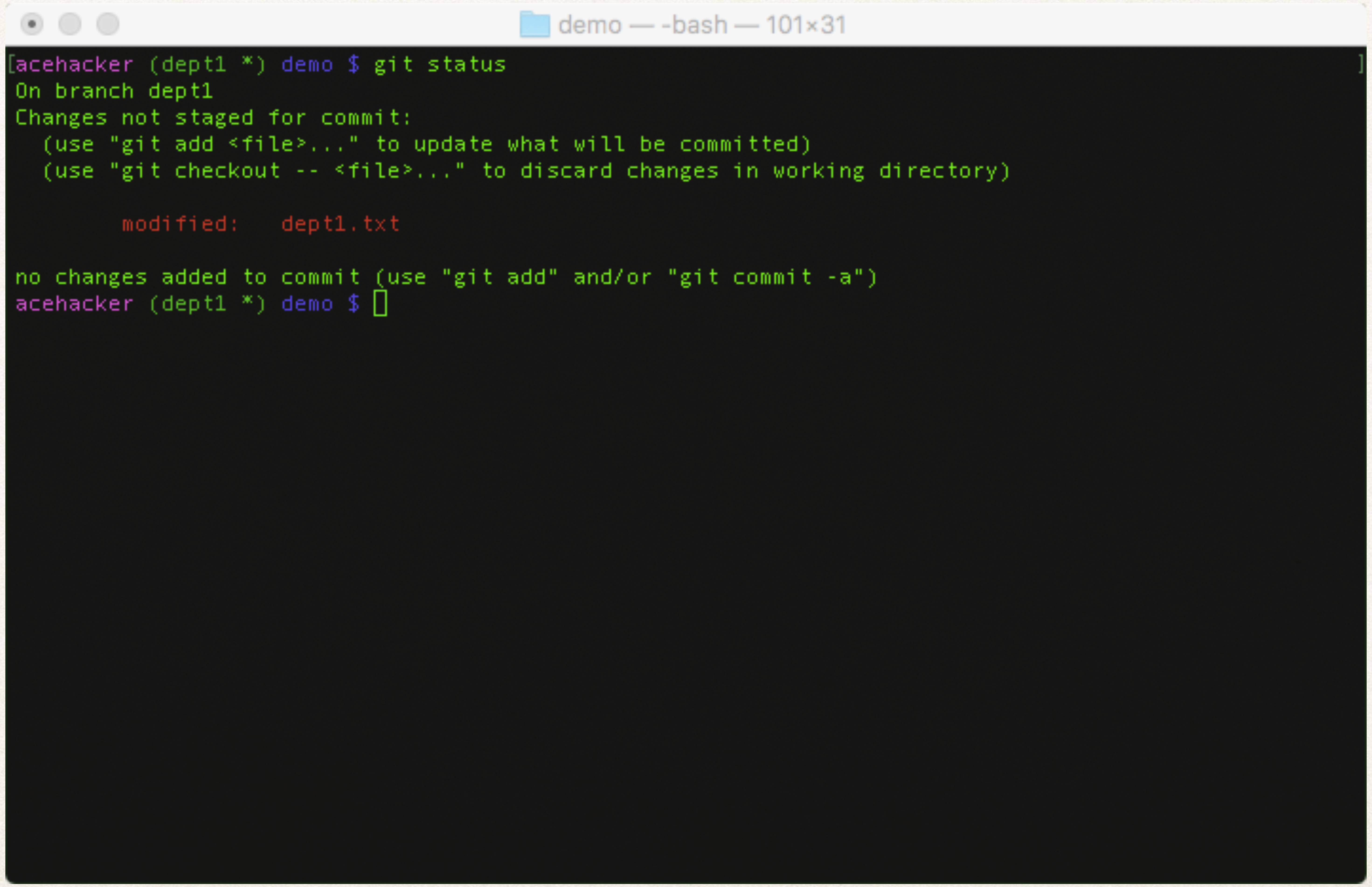
Go to your folder and make changes to the file in branch.



```
dept1 first file
Made changes in first branch
```

Git

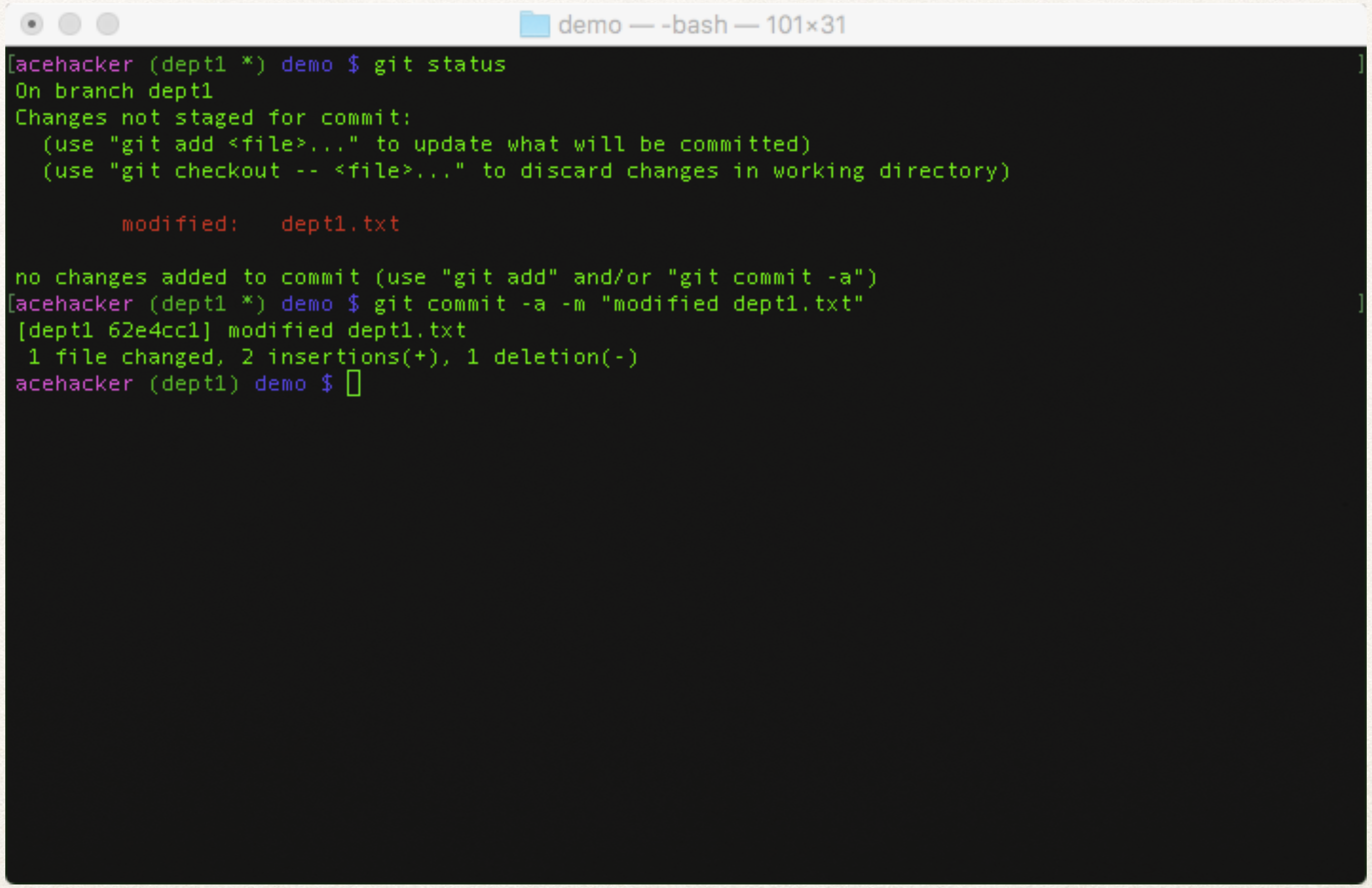
Check Git Status with `git status`


demo — -bash — 101×31

```
[acehacker (dept1 *) demo $ git status
 On branch dept1
 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:   dept1.txt

 no changes added to commit (use "git add" and/or "git commit -a")
 acehacker (dept1 *) demo $
```

Git

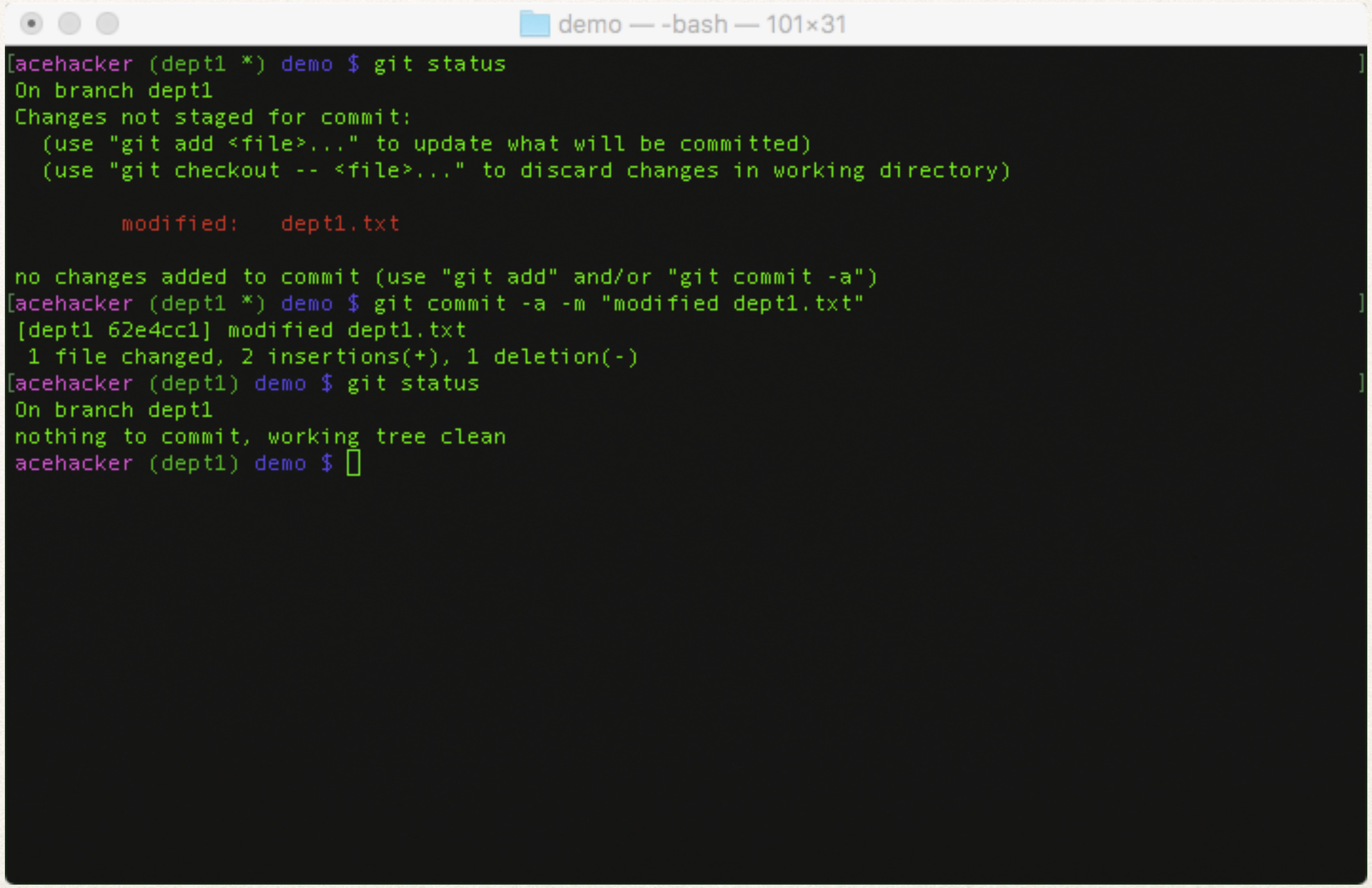
Because this is already a tracked file, you can commit directly without explicitly running the `git add` command. `git commit` at this stage will do it for you.



```
acehacker (dept1 *) demo $ git status
On branch dept1
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:   dept1.txt

no changes added to commit (use "git add" and/or "git commit -a")
acehacker (dept1 *) demo $ git commit -a -m "modified dept1.txt"
[dept1 62e4cc1] modified dept1.txt
 1 file changed, 2 insertions(+), 1 deletion(-)
acehacker (dept1) demo $ 
```

Git

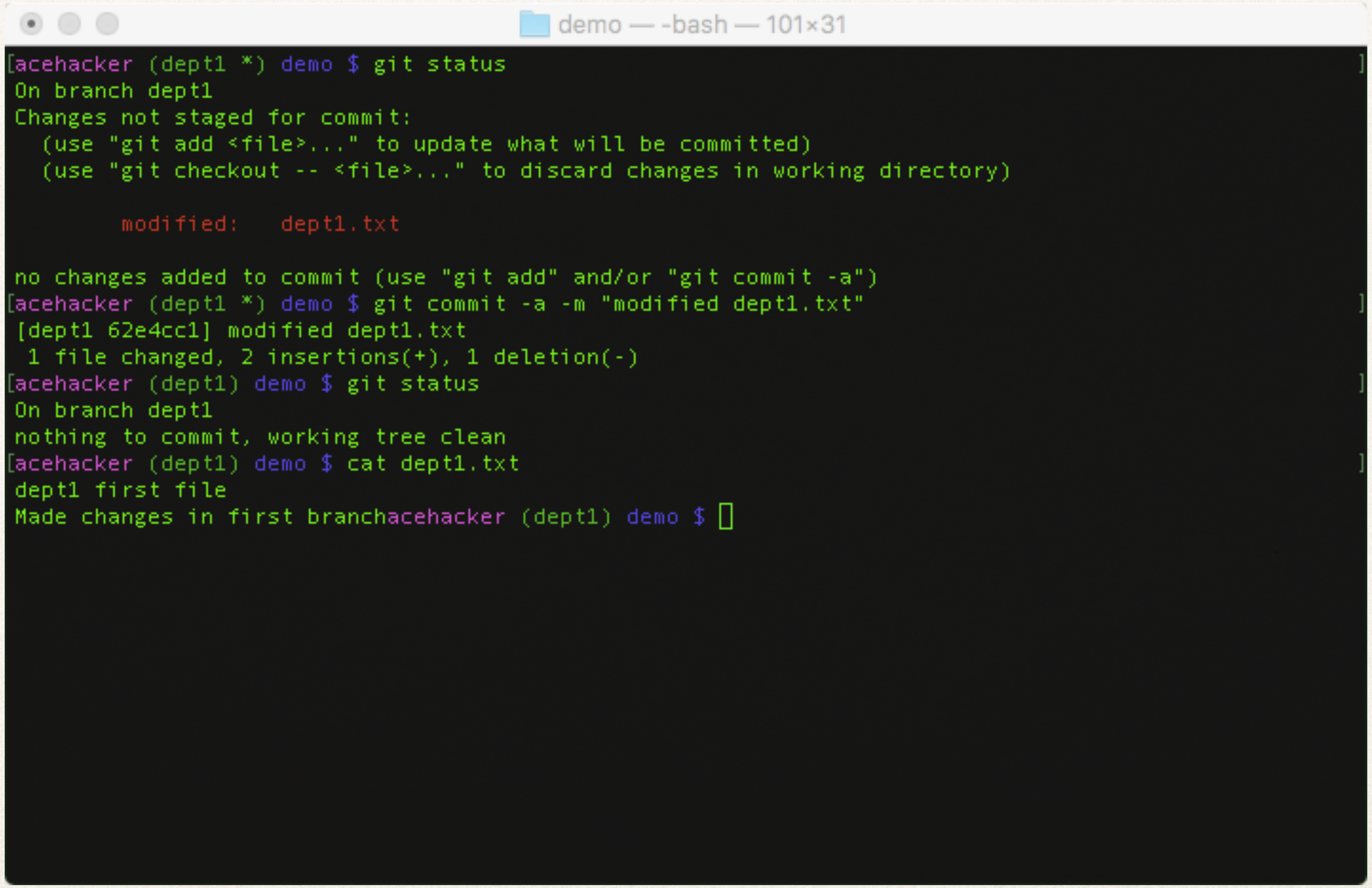
Check Git Status with `git status`



```
[acehacker (dept1 *) demo $ git status
On branch dept1
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:   dept1.txt

no changes added to commit (use "git add" and/or "git commit -a")
[acehacker (dept1 *) demo $ git commit -a -m "modified dept1.txt"
[dept1 62e4cc1] modified dept1.txt
 1 file changed, 2 insertions(+), 1 deletion(-)
[acehacker (dept1) demo $ git status
On branch dept1
nothing to commit, working tree clean
acehacker (dept1) demo $
```

Git

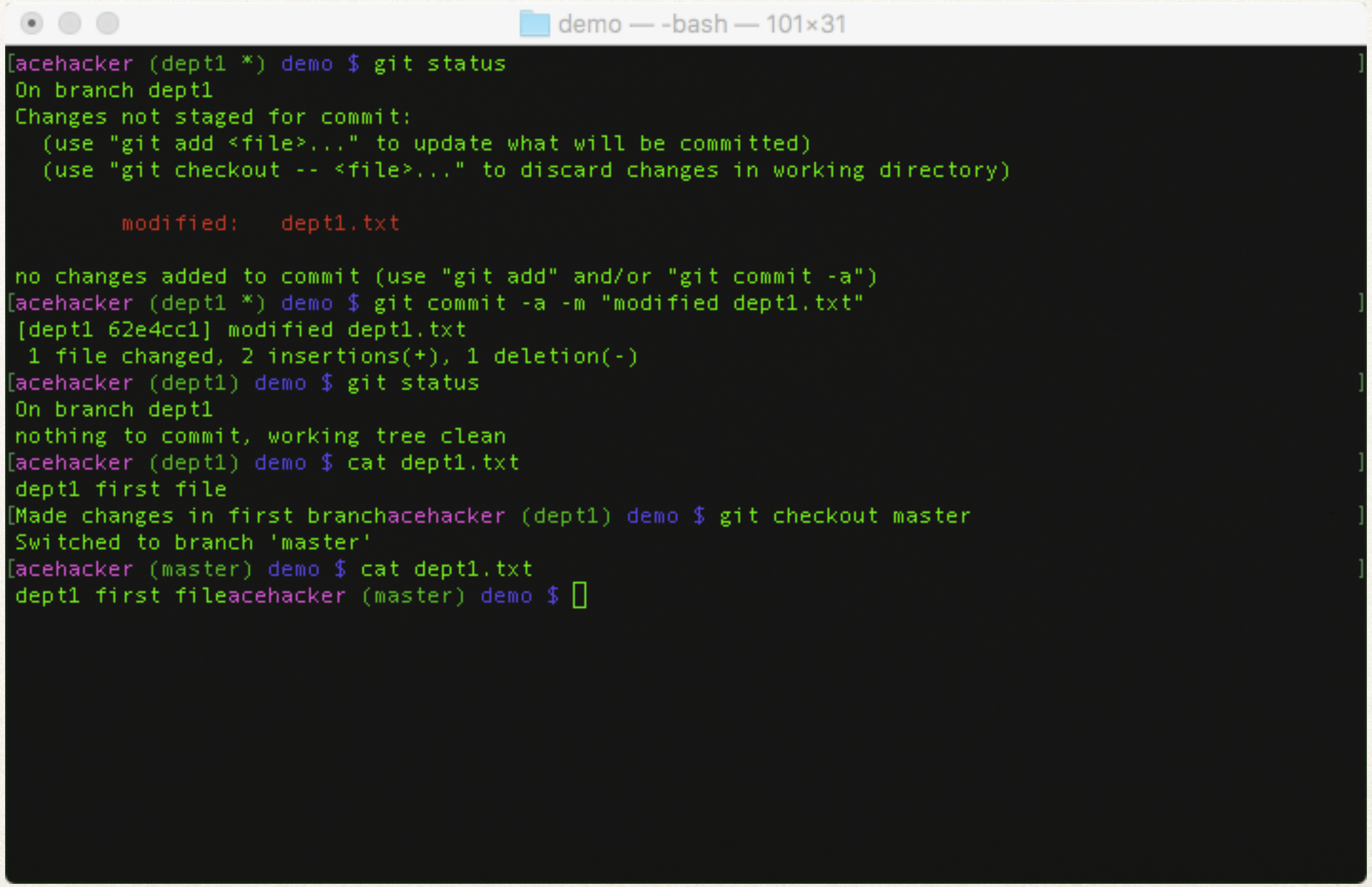
Also check content of the file you just committed using `cat`


demo — -bash — 101×31

```
[acehacker (dept1 *) demo $ git status
 On branch dept1
 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:   dept1.txt

 no changes added to commit (use "git add" and/or "git commit -a")
[acehacker (dept1 *) demo $ git commit -a -m "modified dept1.txt"
 [dept1 62e4cc1] modified dept1.txt
  1 file changed, 2 insertions(+), 1 deletion(-)
[acehacker (dept1) demo $ git status
 On branch dept1
 nothing to commit, working tree clean
[acehacker (dept1) demo $ cat dept1.txt
 dept1 first file
 Made changes in first branchacehacker (dept1) demo $ 
```

Git

Remember that you have not merged this file after making changes. Switch back to master using `git checkout master` and see the difference in the files using `cat`.

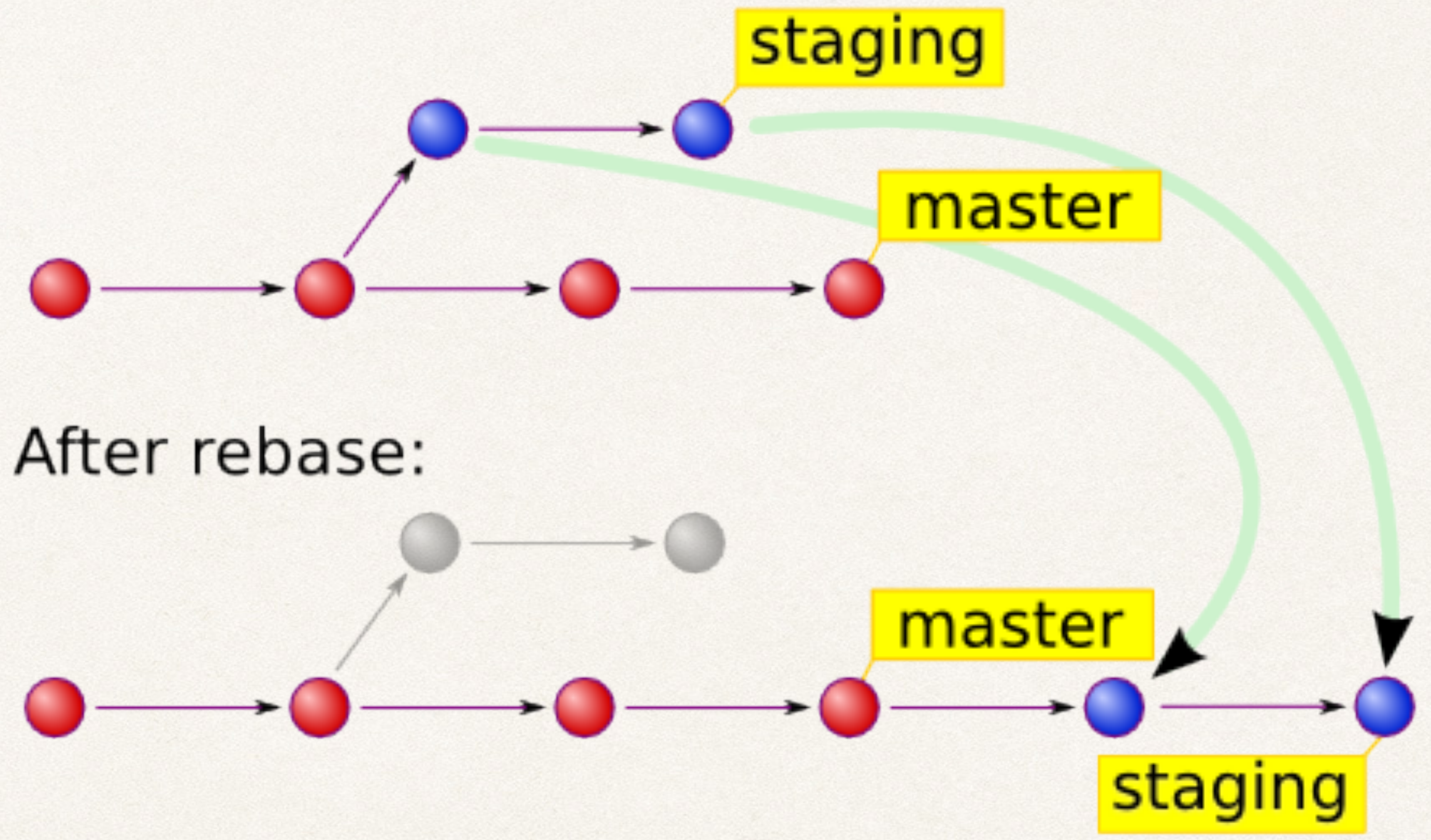


demo — -bash — 101×31

```
[acehacker (dept1 *) demo $ git status                                                              ]
 On branch dept1
 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:   dept1.txt

 no changes added to commit (use "git add" and/or "git commit -a")
[acehacker (dept1 *) demo $ git commit -a -m "modified dept1.txt"                                   ]
 [dept1 62e4cc1] modified dept1.txt
  1 file changed, 2 insertions(+), 1 deletion(-)
[acehacker (dept1) demo $ git status                                                                ]
 On branch dept1
 nothing to commit, working tree clean
[acehacker (dept1) demo $ cat dept1.txt                                                             ]
 dept1 first file
[Made changes in first branchacehacker (dept1) demo $ git checkout master                           ]
 Switched to branch 'master'
[acehacker (master) demo $ cat dept1.txt                                                            ]
 dept1 first fileacehacker (master) demo $ 
```

Git

Linear sequence of commits - Rebasing



Git

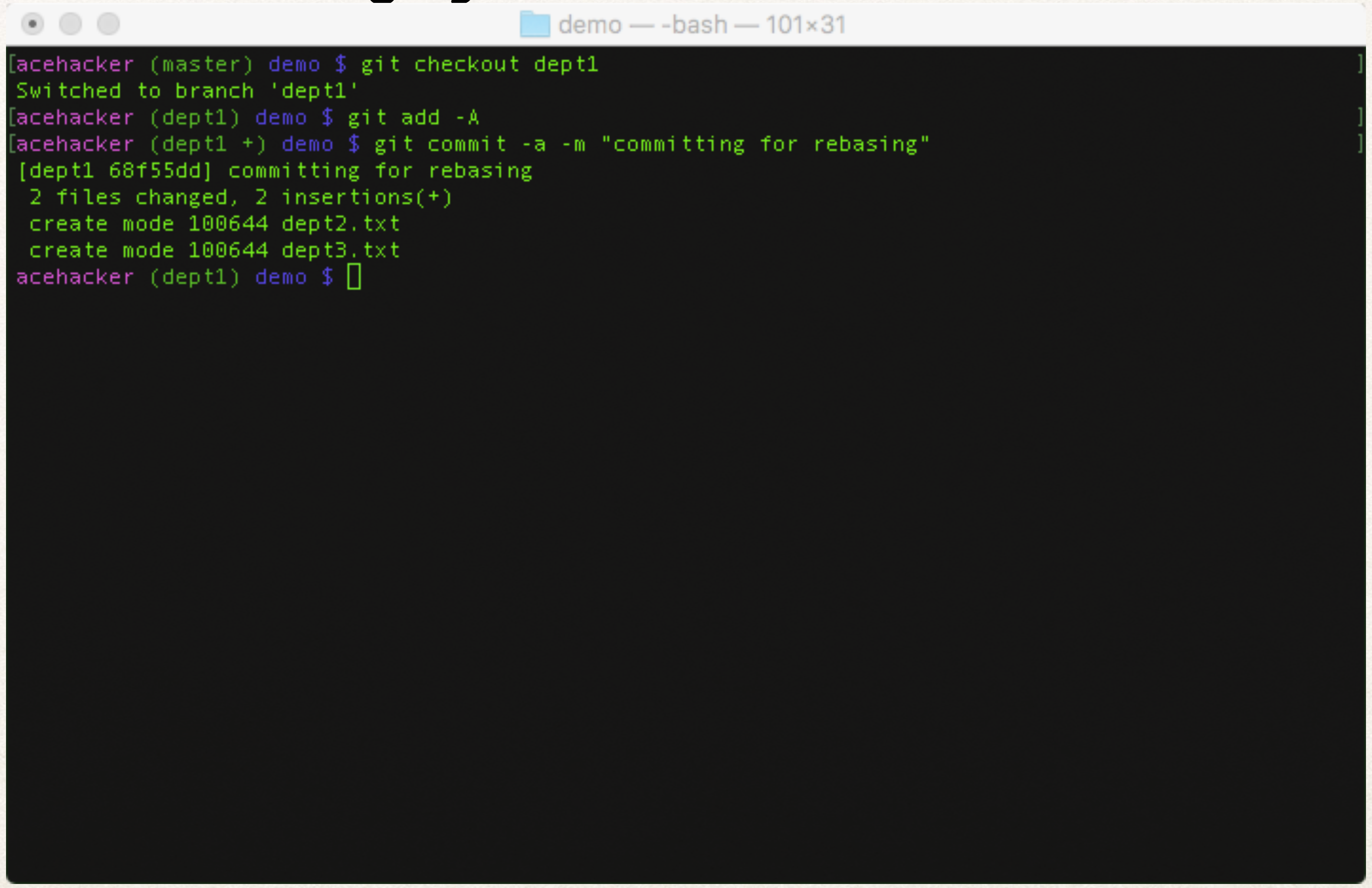
Rebasing

Switch to the branch.

Go to your folder and create some new files.

These files will be untracked, so you will need to use `git add -A`

Then commit using `git commit -a -m “<message>”`



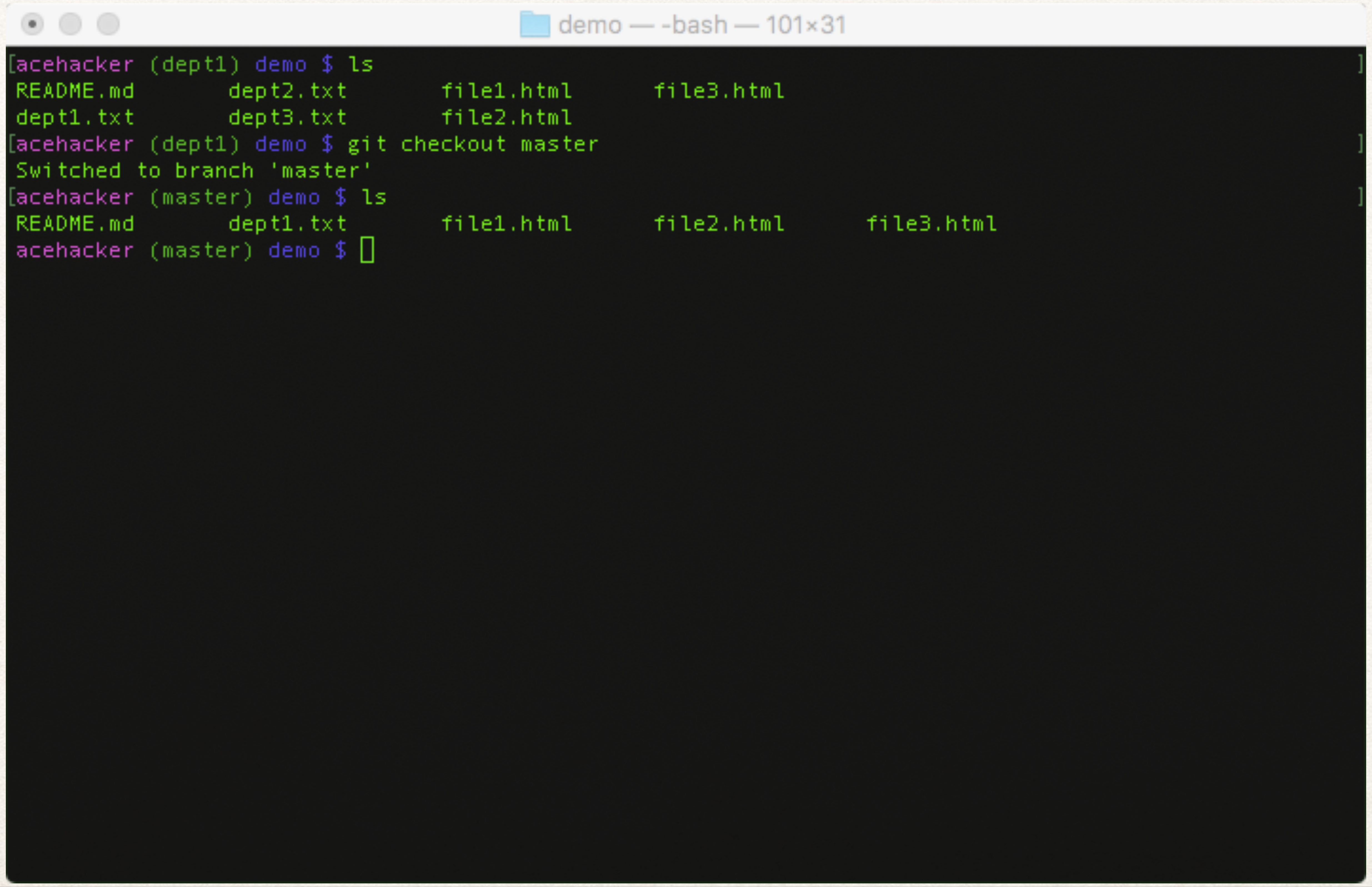
demo — -bash — 101×31

```
[acehacker (master) demo $ git checkout dept1
 Switched to branch 'dept1'
[acehacker (dept1) demo $ git add -A
[acehacker (dept1 +) demo $ git commit -a -m "committing for rebasing"
 [dept1 68f55dd] committing for rebasing
  2 files changed, 2 insertions(+)
  create mode 100644 dept2.txt
  create mode 100644 dept3.txt
 acehacker (dept1) demo $ 
```

Git

Rebasing

See the files in the branch using `ls`.
Switch to the master and see the files there using `ls`.



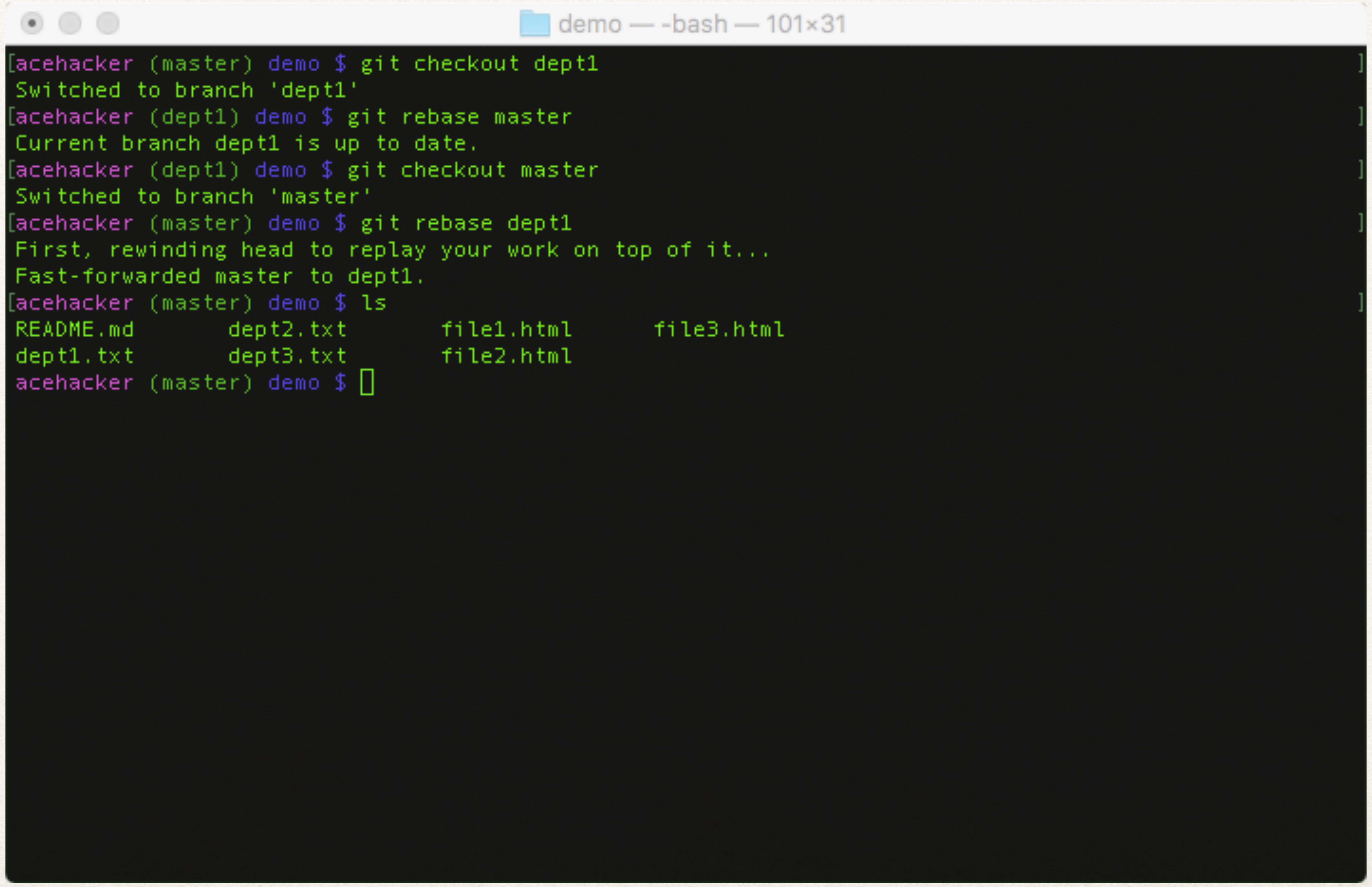
demo — -bash — 101×31

```
[acehacker (dept1) demo $ ls
 README.md       dept2.txt       file1.html      file3.html
 dept1.txt       dept3.txt       file2.html
[acehacker (dept1) demo $ git checkout master
 Switched to branch 'master'
[acehacker (master) demo $ ls
 README.md       dept1.txt       file1.html      file2.html      file3.html
 acehacker (master) demo $
```

Git

Rebasing

Rebase from the master using `git rebase <branch name>`



```
[acehacker (master) demo $ git checkout dept1
 Switched to branch 'dept1'
[acehacker (dept1) demo $ git rebase master
 Current branch dept1 is up to date.
[acehacker (dept1) demo $ git checkout master
 Switched to branch 'master'
[acehacker (master) demo $ git rebase dept1
 First, rewinding head to replay your work on top of it...
 Fast-forwarded master to dept1.
[acehacker (master) demo $ ls
 README.md       dept2.txt       file1.html      file3.html
 dept1.txt       dept3.txt       file2.html
 acehacker (master) demo $
```

Attend the next Webinar on
Oct 14, 2018
at
15:00 IST

More details here: https://codefundo.io