Parent + Child

```
<!doctype html>
      <head>
          <title> Week 1 </title>
      </head>
      <body>
          <div>
          Here's a Great Site.
          </div>
      </body>
</html>
```

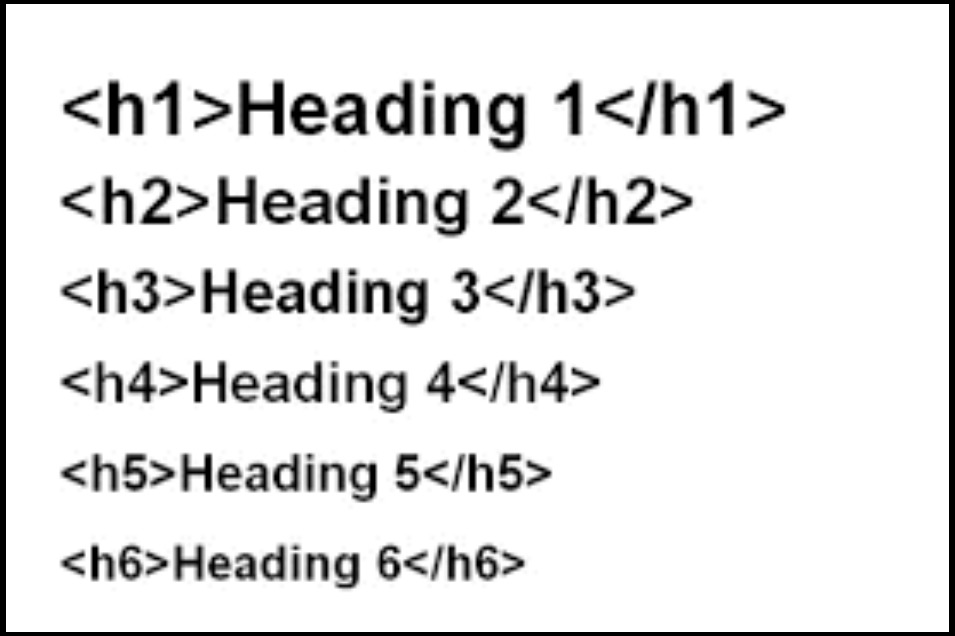
head is the parent of title

div is the child of body

body is the child of html

Text tags

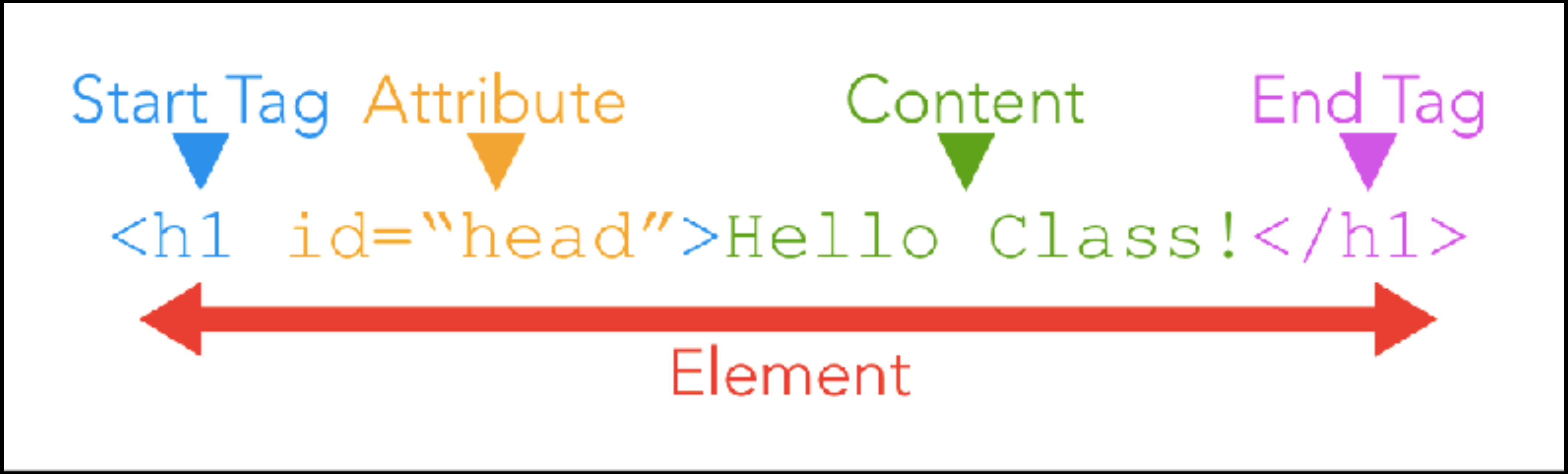
- h1, h2, h3, h4, h5, h6 are text tags for headings
- p is a tag for paragraphs
- b is for bold, i is for italics
- <strong> is for **bold** <em> is for *italics*
- ul, ol, li are used for making lists
  - ul: unordered lists
  - ol: ordered lists
  - li: an individual list tag
- <br/> will break to a new line



```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

HTML Elements / Tags, Attributes, Content

- Elements and tags used interchangeably



tag attribute value

```
<video src= “filepath/file.mov” alt= “this is the video” height=“300”></video>

<html attribute= “value” attribute= “value” attribute= “value”> </html>
```

<a> links </a>

OPENING LINK TAG

WE ARE DIRECTED TO

TEXT WE CLICK ON

```
<a href="http://storm.cis.fordham.edu/~rleopold/" > Professor Leopold </a>
```

< a href — stands for *hyperlink reference*

<a> relative urls </a>

Linking to pages on the same site

Same Directory	`<a href=“week00.html” > Week 1 Page </a>`
Child Directory	`<a href=“myBlog/week00.html” > Week 1 Page </a>`
id attribute	`<a href=“#potatoGallery” > Click here for this week in potatoes! </a`
Parent Directory	`<a href=“../index.html” > Home Page </a>`

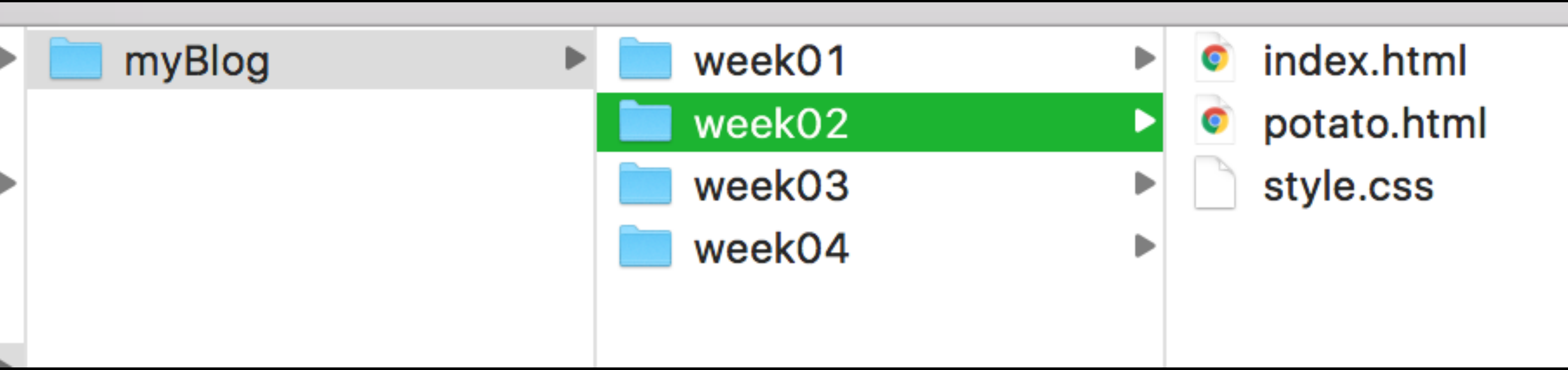
< a href — **stands for *hyperlink reference***

Why index.html?

The main homepage of a site written in HTML (and the homepage of each section in a child folder) is called index.html.

Web servers are usually set up to return the index.html file if no file name is specified. Therefore, it’s always a good idea to name your directories’ root webpages index.html

Why index.html?



The <img> tag has a required attribute called src. The src attribute must be set to the image's source, or the location of the image. In some cases, the value of src must be the *uniform resource locator* (URL) of the image. A URL is the web address or local address where a file is stored.

Images: relative vs. url

```
<img src= “images/potato07.png” alt= “spud” >
```

```
<img src= “https://pngriver.com/wp-content/uploads/2018/04/Download-Potato-PNG-Pic.png” alt= “spud” >
```

The alt attribute, which means alternative text, brings meaning to the images on our sites. The alt attribute can be added to the image tag just like the src attribute. The value of alt should be a description of the image.

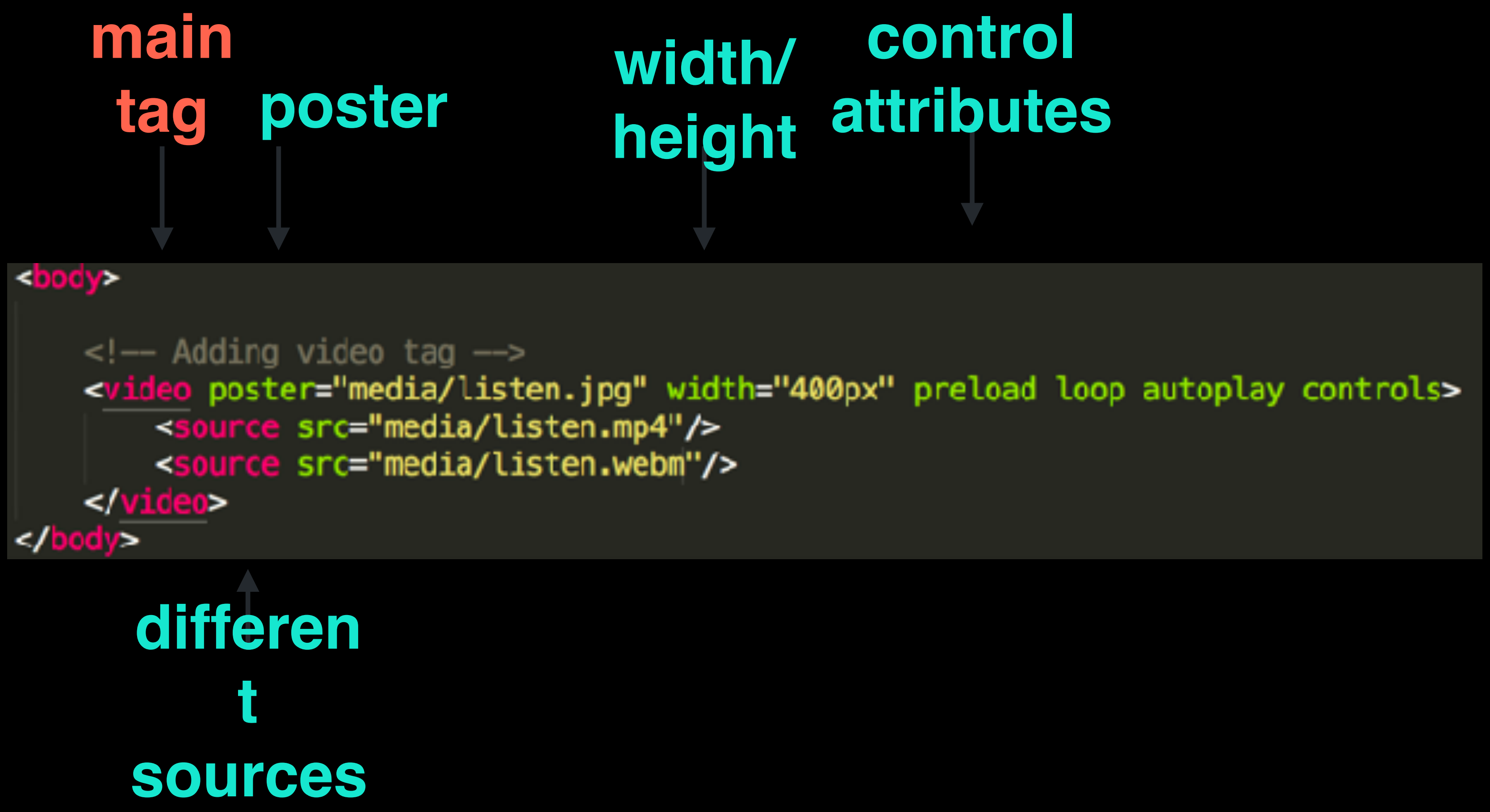
```
<img src="https://www.fordham.edu/
images/fordham_102907_0455.jpg"
alt="Computer Sciences" height="300">
```

1. If an image fails to load on a web page, a user can mouse over the area originally intended for the image and read a brief description of the image. This is made possible by the description you provide in the alt attribute.
2. Visually impaired users often browse the web with the aid of screen reading software. When you include
   the alt attribute, the screen reading software can read the image's description out loud to the visually impaired user.
3. The alt attribute also plays a role in Search EngineOptimization (SEO), because search engines cannot "see" the images on websites as they crawl the internet. Having descriptive alt attributes can improve the ranking of your site.

Like the <img> tag, the <video> tag requires a src attribute with a link to the video source.

Unlike the <img> tag however, the <video> element requires an opening and a closing tag.

<video /> structure


main tag
poster
width/ height
control attributes
differen t sources

```
<body>

    <!-- Adding video tag -->
    <video poster="media/listen.jpg" width="400px" preload loop autoplay controls>
        <source src="media/listen.mp4"/>
        <source src="media/listen.webm"/>
    </video>
</body>
```

After the src attribute, the width
and height attributes are used to set the size
of the video displayed in the browser.

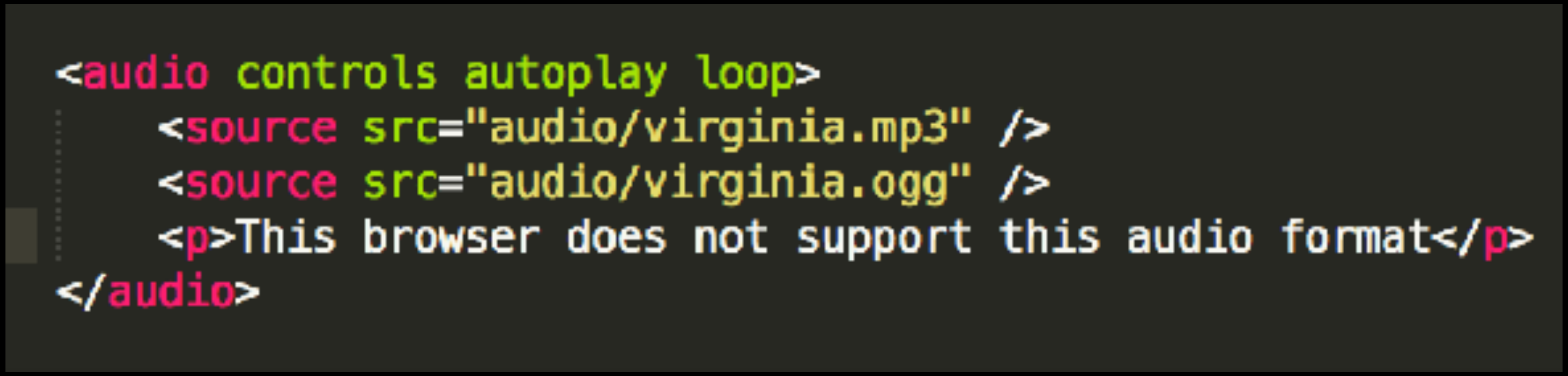
The controls attribute instructs the browser
to include basic video controls: pause, play
and skip.Unlike the <img> tag however,
the <video> element requires an opening
and a closing tag.

The text, "Video not supported", between the opening and closing video
tags will only be displayed if the browser is unable to load the video.

<audio /> structure

main tag

control attributes



```
<audio controls autoplay loop>
    <source src="audio/virginia.mp3" />
    <source src="audio/virginia.ogg" />
    <p>This browser does not support this audio format</p>
</audio>
```

differen
t
sources

text is the
file cannot
be found

Some Media Attributes

- **Preload** - what preloads when the page loads
- **Controls** - if the play/stop buttons are visible
- **Autoplay** - if the video should start playing automatically
- **Loop** - if the video should loop on completion

Attributes

If we want to expand an element's tag, we can do so using an attribute. Attributes are content added to the opening tag of an element and can be used in several different ways, from providing information to changing styling. Attributes are made up of the following two parts:

1) The name of the attribute
2) The value of the attribute

One commonly used attribute is the id.

We can use the id attribute to specify different content (such as <div>s) and is really helpful when you use an element more than once.

```
<div id="intro">
  <h1>Technology</h1>
</div>
```

<span> contains short pieces of text or other HTML. They are used to separate small pieces of content that are on the same line as other content.

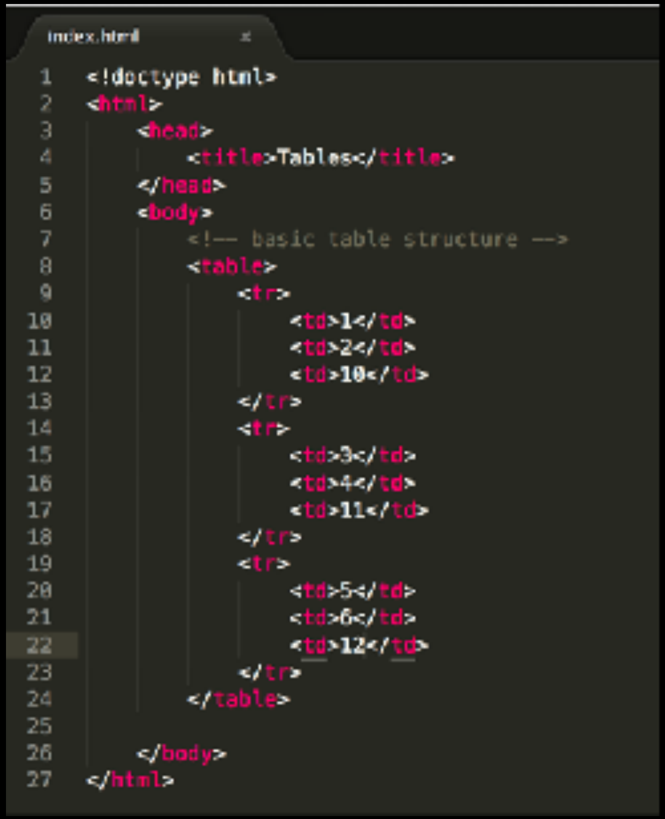
```
<div>
    <h1>Technology</h1>
</div>
<div>
    <p> Wherever there’s a
    <span>computer</span>, there’s a skilled
    person developing, maintaining, hacking,
    advancing or simply using it.</p>
</div>
```

<br>

The line break element is unique because it is only composed of a starting tag. You can use it anywhere within your HTML code and a line break will be shown in the browser.

Table structure

- <table> element is used to create a table (written out row by row)
- <tr> indicates each row
- <td> indicates each cell of a table

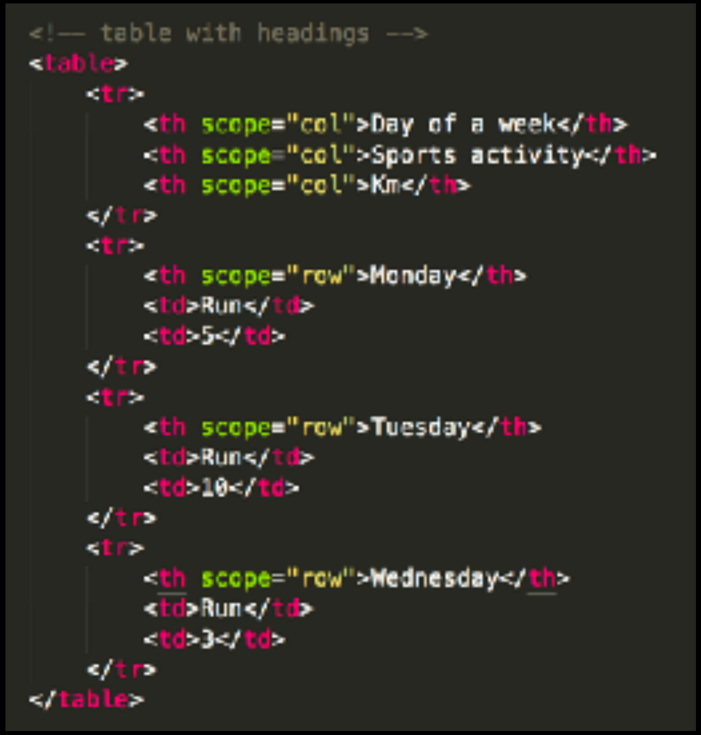


```html
<!doctype html>
<html>
    <head>
        <title>Tables</title>
    </head>
    <body>
        <!-- basic table structure -->
        <table>
            <tr>
                <td>1</td>
                <td>2</td>
                <td>10</td>
            </tr>
            <tr>
                <td>3</td>
                <td>4</td>
                <td>11</td>
            </tr>
            <tr>
                <td>5</td>
                <td>6</td>
                <td>12</td>
            </tr>
        </table>

    </body>
</html>
```

Table headings

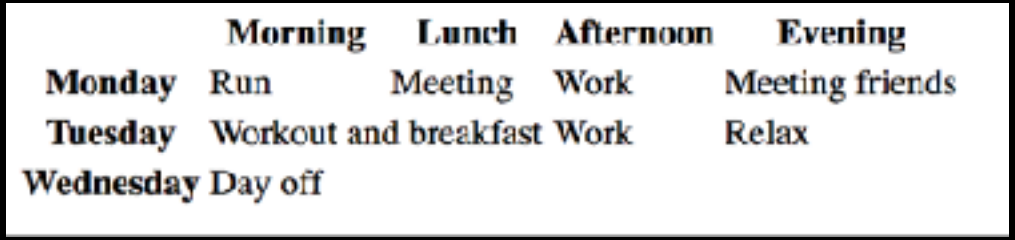
- **<th>** is used to represent the heading for either a column or a row
- Even though there is no content, you should still use it to represent an empty cell
- Add **<scope>** to indicate if it's a heading for row or column



```html
<!-- table with headings -->
<table>
    <tr>
        <th scope="col">Day of a week</th>
        <th scope="col">Sports activity</th>
        <th scope="col">Km</th>
    </tr>
    <tr>
        <th scope="row">Monday</th>
        <td>Run</td>
        <td>5</td>
    </tr>
    <tr>
        <th scope="row">Tuesday</th>
        <td>Run</td>
        <td>10</td>
    </tr>
    <tr>
        <th scope="row">Wednesday</th>
        <td>Run</td>
        <td>3</td>
    </tr>
</table>
```

Spanning columns

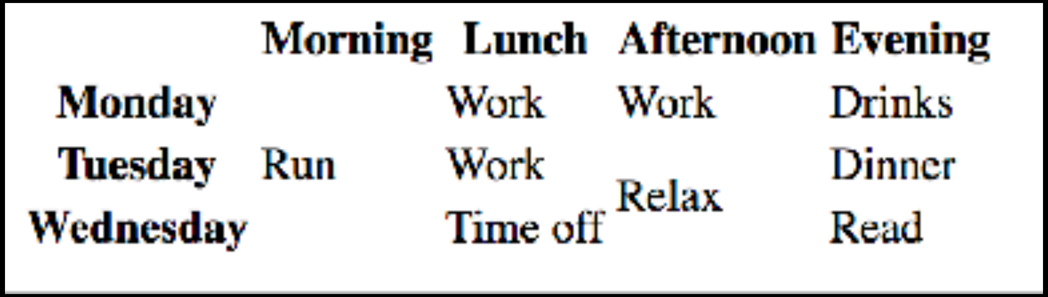
- Sometimes you may need the entries in a table to stretch across more than one column
- You can add ***colspan*** attribute on **<th>** or **<td>** to indicate how many columns that cell should run across



	Morning	Lunch	Afternoon	Evening
Monday	Run	Meeting	Work	Meeting friends
Tuesday	Workout and breakfast		Work	Relax
Wednesday	Day off			

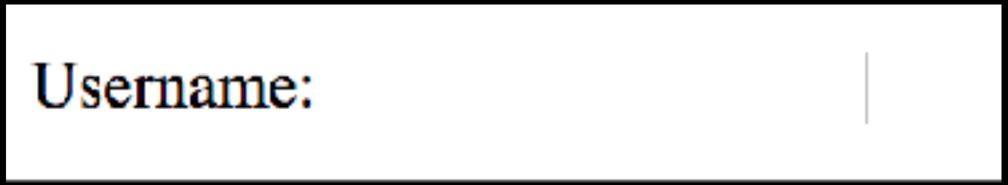
Spanning rows

- Add ***rowspan*** attribute on **<th>** or **<td>** to indicate how many columns that cell should run across

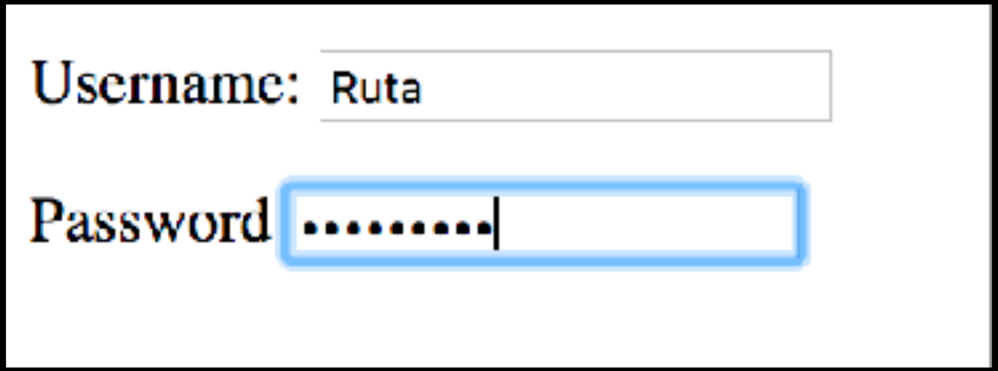


	Morning	Lunch	Afternoon	Evening
Monday		Work	Work	Drinks
Tuesday	Run	Work	Relax	Dinner
Wednesday		Time off		Read

Text input

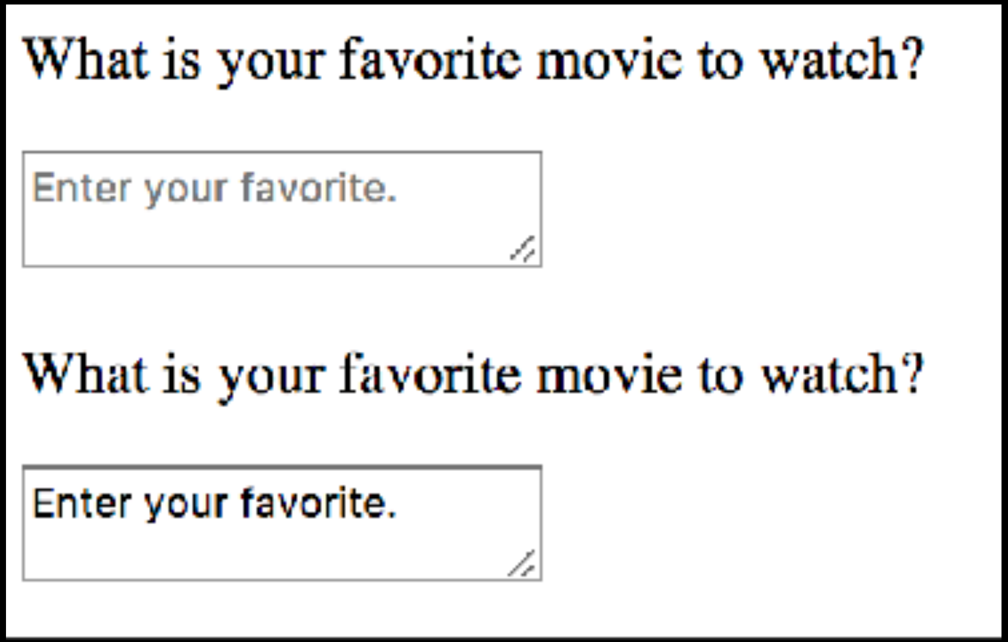

Username:

Password input


Username: Ruta

Password ••••••••••

Text area

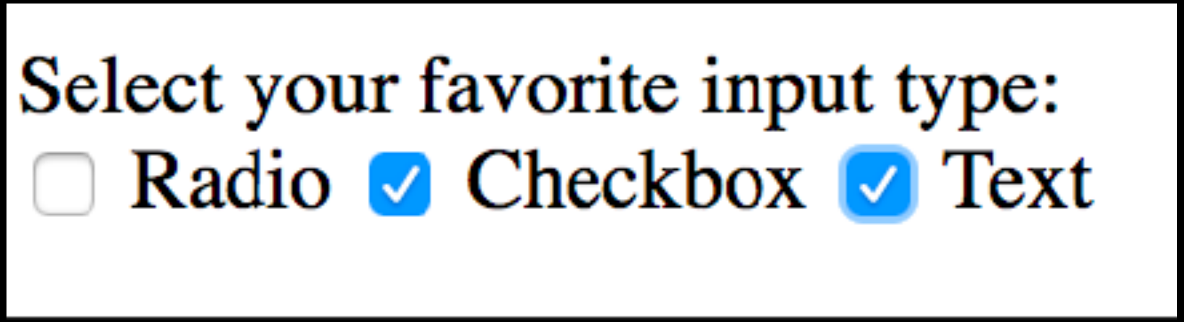

What is your favorite movie to watch?

Enter your favorite.

What is your favorite movie to watch?

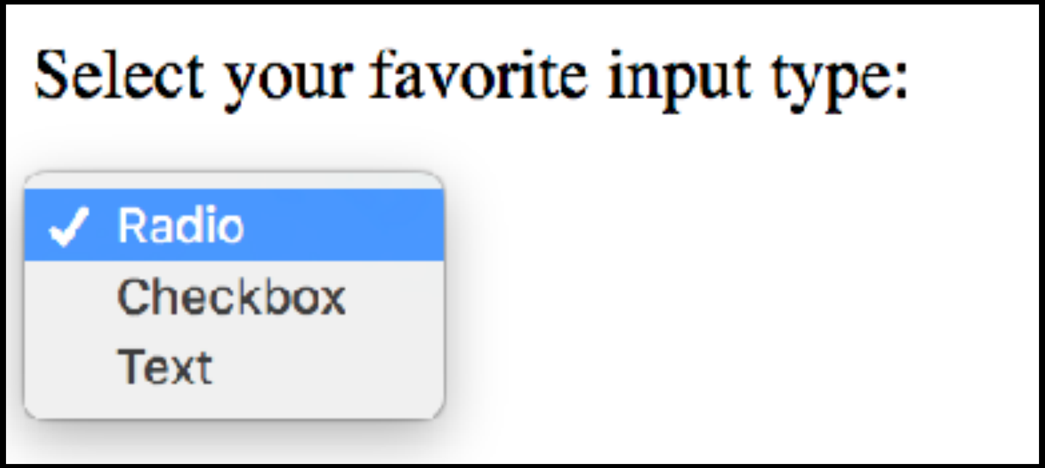
Enter your favorite.

Checkbox

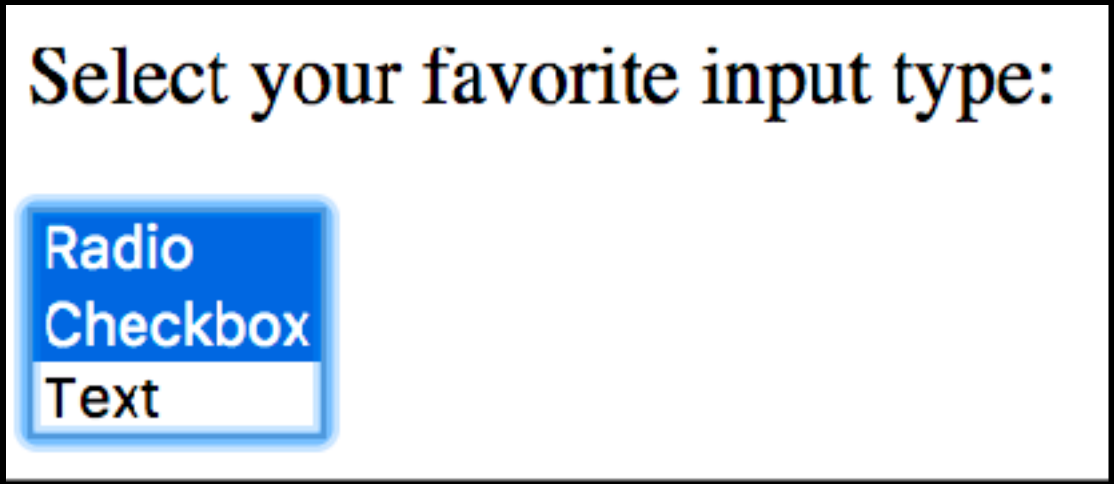

Select your favorite input type:

☐ Radio ✓ Checkbox ✓ Text

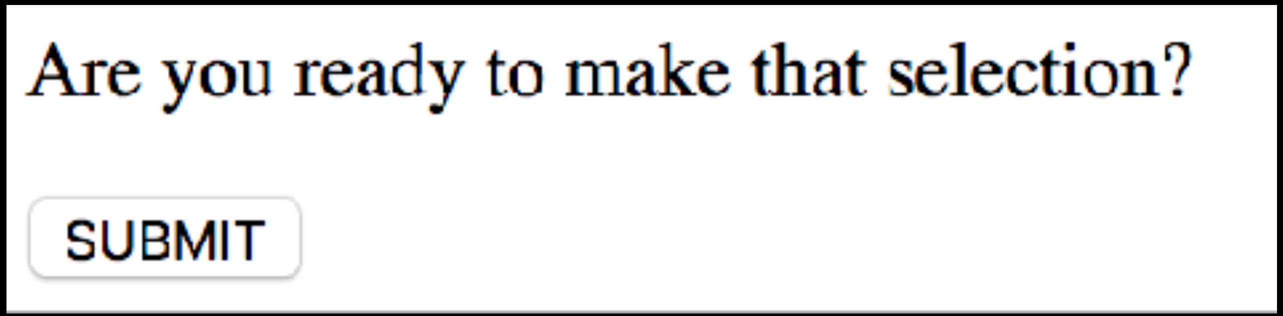
Drop down list


Select your favorite input type:

Multiple select box


Select your favorite input type:

Submit button


Are you ready to make that selection?

Git

What is Git?

A version control system meant to make it easier to have multiple versions of code, sometimes across multiple developers or teams

At its most simple, git helps with the ‘**indexv1.html**, **indexv2.html**, **indexv3FINAL.html**' problem

At its most complex, git allows developers to work together worldwide on code without stepping on each other's toes

Github

Github is a service to host yr projects on the web.

Code is pushed (uploaded) from a local directory (folder) called a repository or rep.

example - our class site:
http://www.github.com/rebleo/webDevSpring2020

Git vs GitHub.com

git is a version control system that takes snapshots of your code at certain points in development

These snapshots are stored in a '**repo**', or '**repository**' on your local machine

GitHub.com is a website that hosts git repositories on a remote server + is available for all the web to see, copy + implement.

Git Terminology

repository - where data is managed. the directory containing your files.

local - the copy that exists on your machine, no one else can access this

remote - the copy in your github account, anyone with access to your github repo can access the remote instance (we won't be doing this!)

push - once you make changes to the local copy you *upload the changes to the remote copy

pull - if someone else makes changes to the remote copy (we won't be doing this this semester)

clone a repository - download the entire codebase of the repo you can pull in changes + and push your own changes if you are given access

Github pages

github.io

easily allows you to host web pages using github servers + workflow

url (uniform resource locator)

http://
yrUsername.github.io

yrUsername.github.io

HTTPS

Hypertext Transfer Protocol Secure - is an extension of the Hypertext Transfer Protocol (HTTP). It is used for secure communication over a computer network, and is widely used on the Internet.

SSH

Secure Shell or Secure Socket - a network protocol that gives users, particularly system administrators, a secure way to access a computer over an unsecured network. **SSH** also refers to the suite of utilities that implement the **SSH** protocol.

HTTPS vs SSH

1. There are several ways to clone and set up repositories in Github
2. HTTPS - easier but you always have to enter your usrnm + psswrd
3. SSH - more complicated, but once set up - easy to go from terminal to github.com

Git commands

Git Terminal Commands

clone - download a copy for repo to your local machine

status - view the change status of the repo

add - add changed files to be committed

commit - **-m** “My saved message here” (this message will be public)

push - send your committed updates to github

Github pages

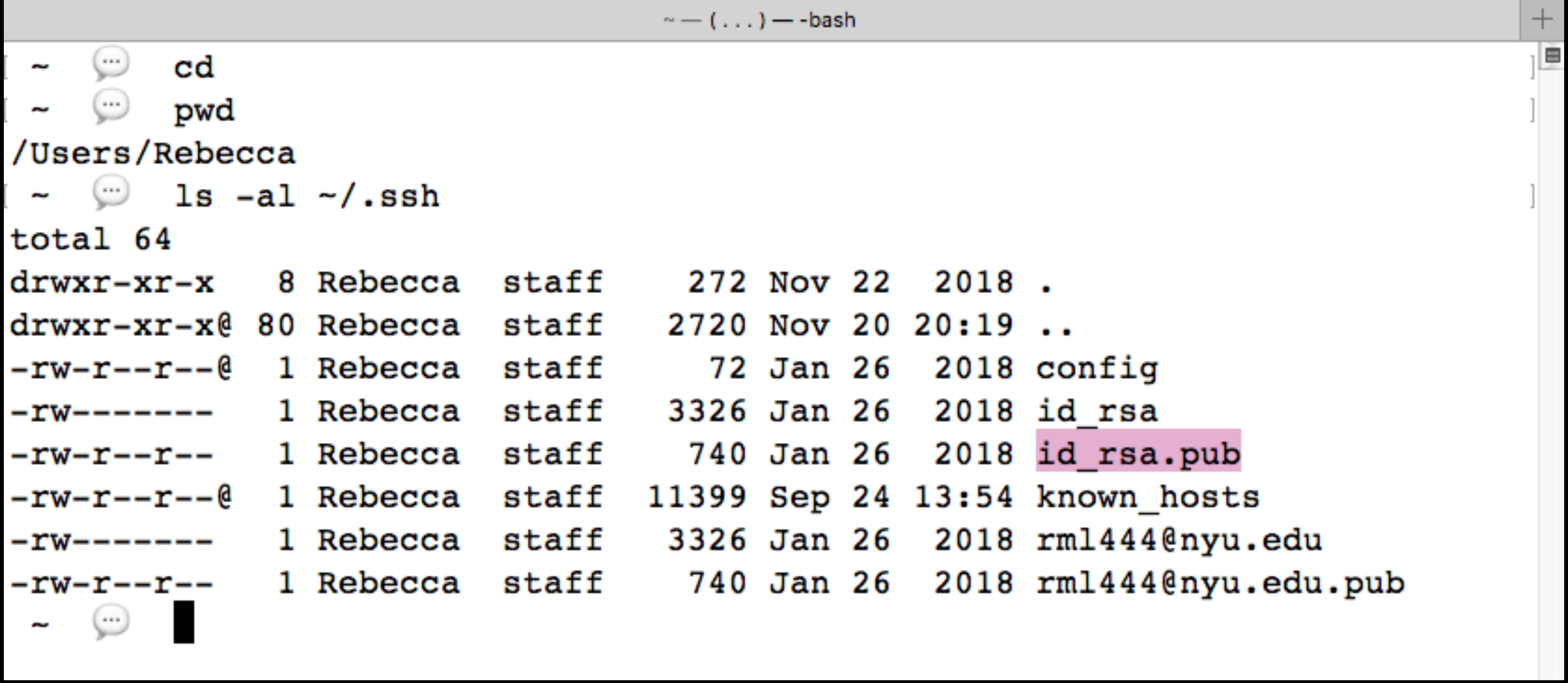
https://pages.github.com

Git Steps

1. Create a new repo on yr Github (git init)

2. Clone yr new repo somewhere on yr local machine

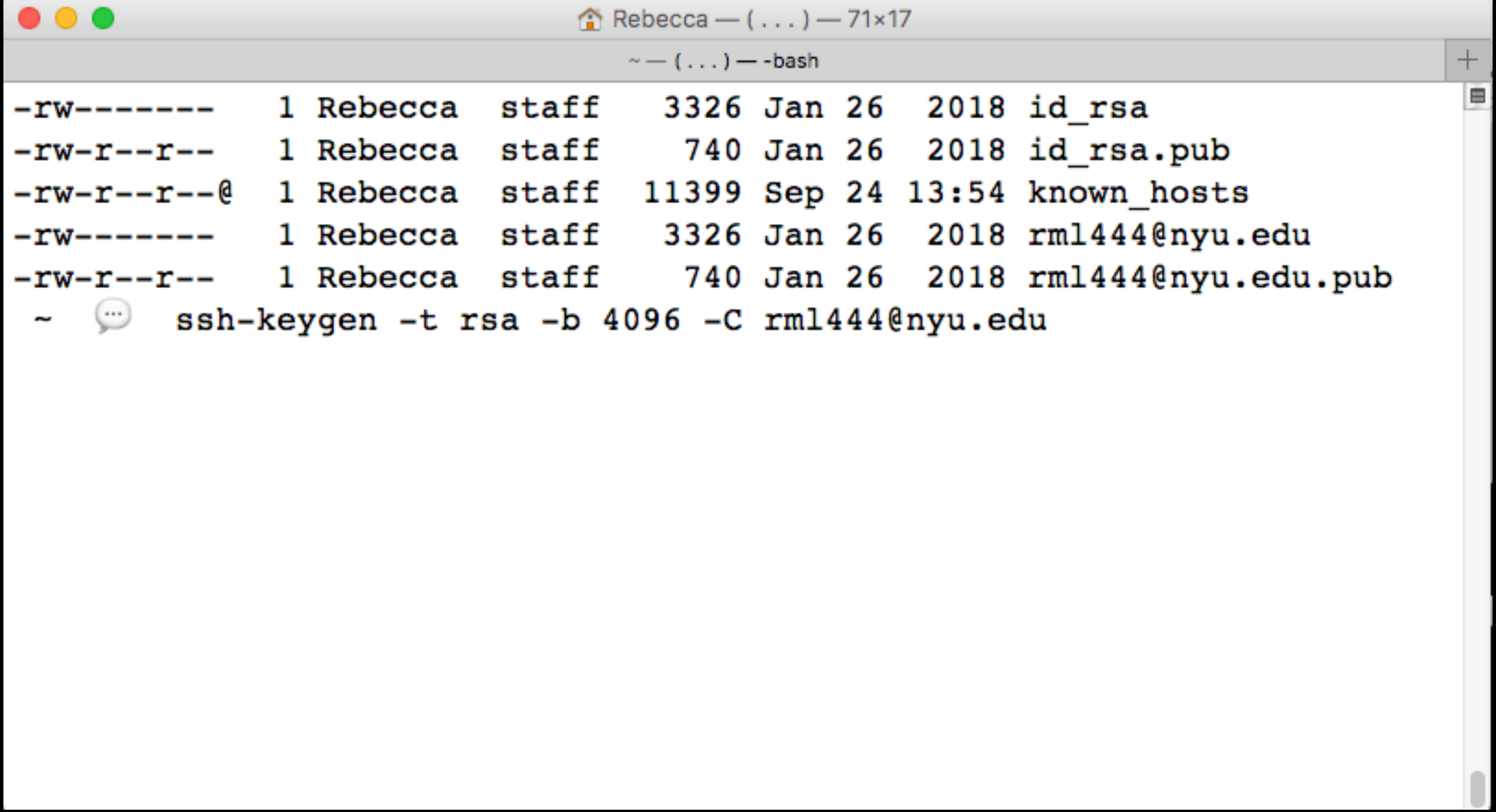
3. Make yr first commit


~ — (. . .) — -bash

```
~      cd
~      pwd
/Users/Rebecca
~      ls -al ~/.ssh
total 64
drwxr-xr-x    8 Rebecca  staff    272 Nov 22  2018 .
drwxr-xr-x@  80 Rebecca  staff   2720 Nov 20 20:19 ..
-rw-r--r--@   1 Rebecca  staff     72 Jan 26  2018 config
-rw-------    1 Rebecca  staff   3326 Jan 26  2018 id_rsa
-rw-r--r--    1 Rebecca  staff    740 Jan 26  2018 id_rsa.pub
-rw-r--r--@   1 Rebecca  staff  11399 Sep 24 13:54 known_hosts
-rw-------    1 Rebecca  staff   3326 Jan 26  2018 rml444@nyu.edu
-rw-r--r--    1 Rebecca  staff    740 Jan 26  2018 rml444@nyu.edu.pub
~
```

Check for SSH Keys

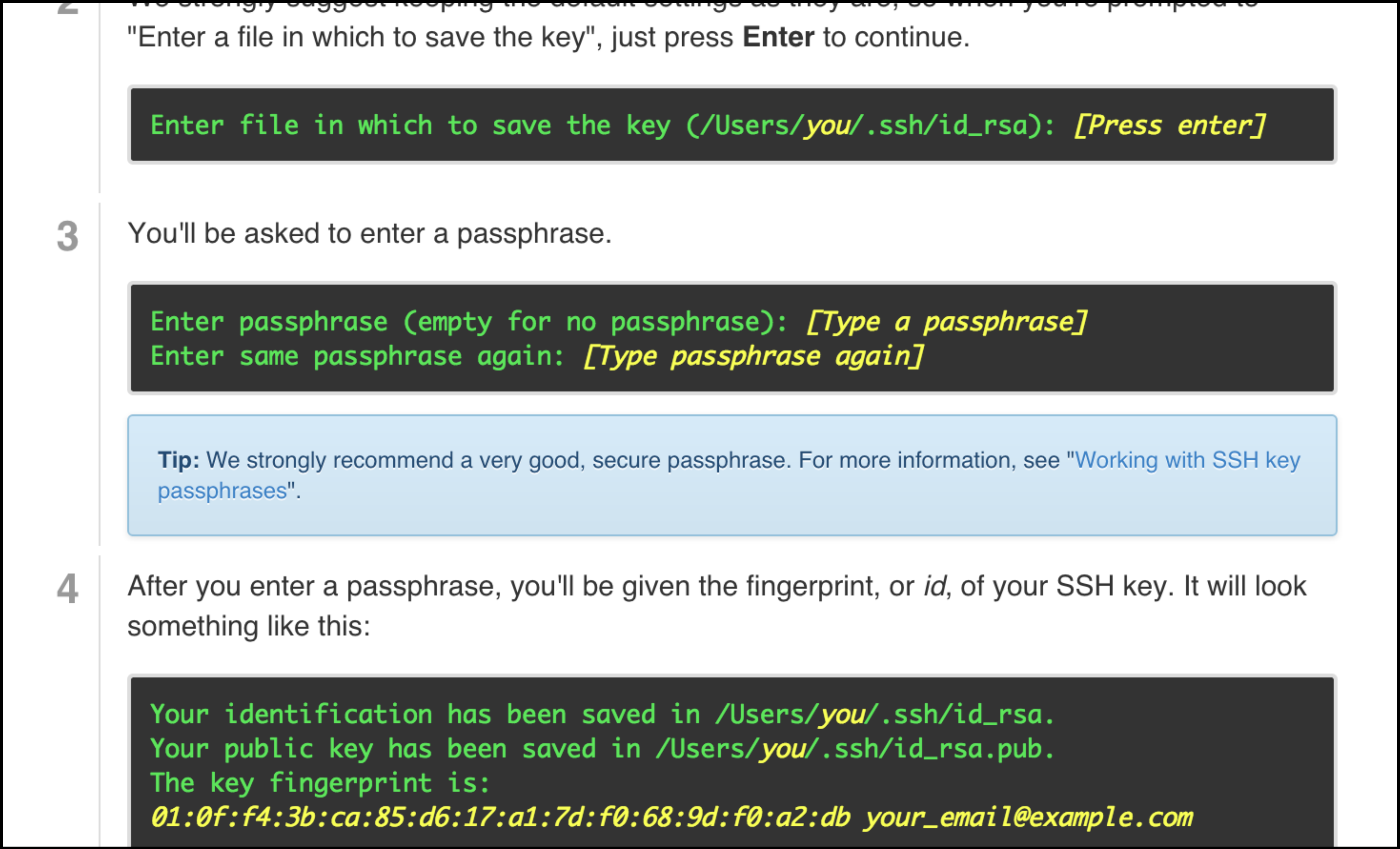
At the root of your machine (your user) - in Bash: type **cd**. Then **pwd** to check…
Using **ls ~/.ssh** (list specific computer readable files w/ exertions .ssh). Look for **id_rsa.pub**, if it's not there we'll fix that!



```
-rw-------    1 Rebecca  staff   3326 Jan 26  2018 id_rsa
-rw-r--r--    1 Rebecca  staff    740 Jan 26  2018 id_rsa.pub
-rw-r--r--@   1 Rebecca  staff  11399 Sep 24 13:54 known_hosts
-rw-------    1 Rebecca  staff   3326 Jan 26  2018 rml444@nyu.edu
-rw-r--r--    1 Rebecca  staff    740 Jan 26  2018 rml444@nyu.edu.pub
 ~       ssh-keygen -t rsa -b 4096 -C rml444@nyu.edu
```

Generate a new SSH Key

Make sure to use the same email you used to create your Github account


"Enter a file in which to save the key", just press **Enter** to continue.

```
Enter file in which to save the key (/Users/you/.ssh/id_rsa): [Press enter]
```

3 You'll be asked to enter a passphrase.

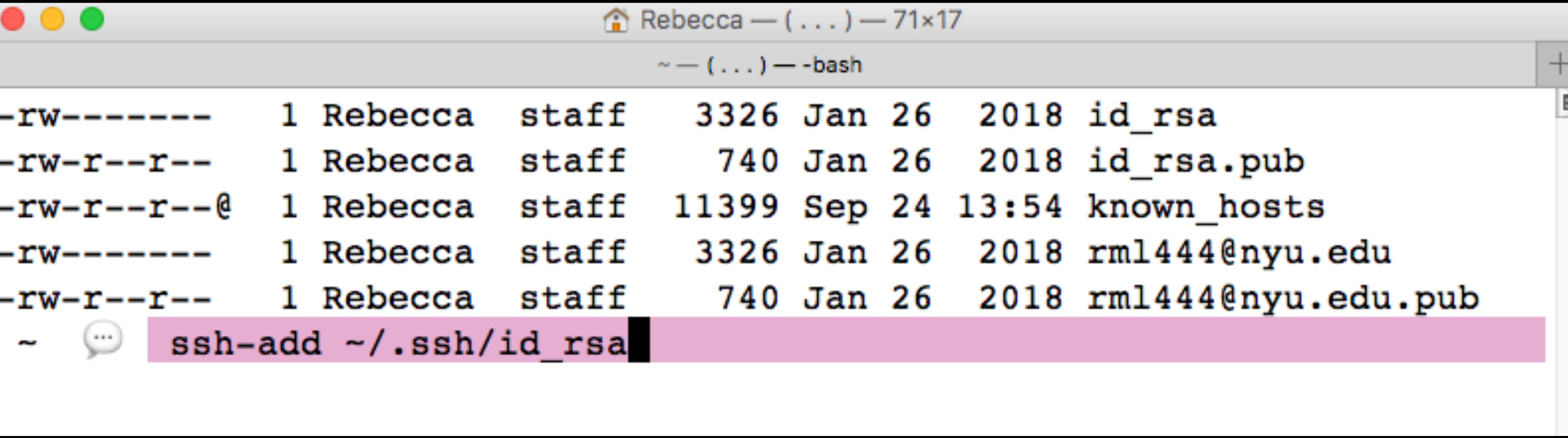
```
Enter passphrase (empty for no passphrase): [Type a passphrase]
Enter same passphrase again: [Type passphrase again]
```

Tip: We strongly recommend a very good, secure passphrase. For more information, see "Working with SSH key passphrases".

4 After you enter a passphrase, you'll be given the fingerprint, or *id*, of your SSH key. It will look something like this:

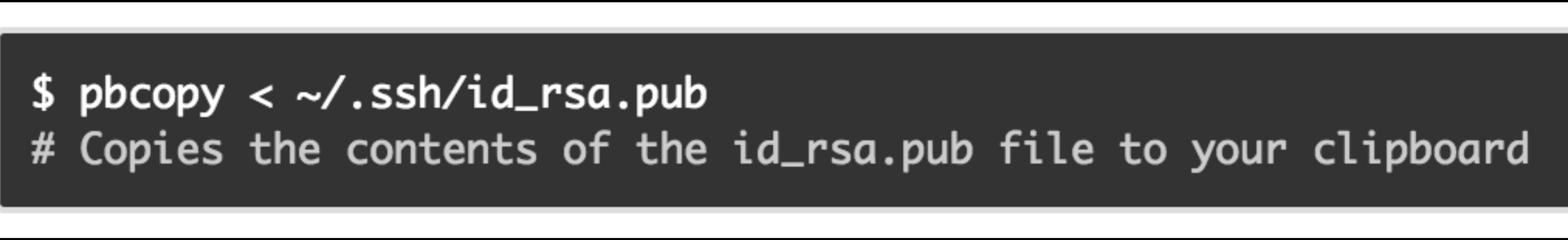
```
Your identification has been saved in /Users/you/.ssh/id_rsa.
Your public key has been saved in /Users/you/.ssh/id_rsa.pub.
The key fingerprint is:
01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db your_email@example.com
```

Finish creating the key



```
-rw-------    1 Rebecca  staff   3326 Jan 26  2018 id_rsa
-rw-r--r--    1 Rebecca  staff    740 Jan 26  2018 id_rsa.pub
-rw-r--r--@   1 Rebecca  staff  11399 Sep 24 13:54 known_hosts
-rw-------    1 Rebecca  staff   3326 Jan 26  2018 rml444@nyu.edu
-rw-r--r--    1 Rebecca  staff    740 Jan 26  2018 rml444@nyu.edu.pub
 ~           ssh-add ~/.ssh/id_rsa
```

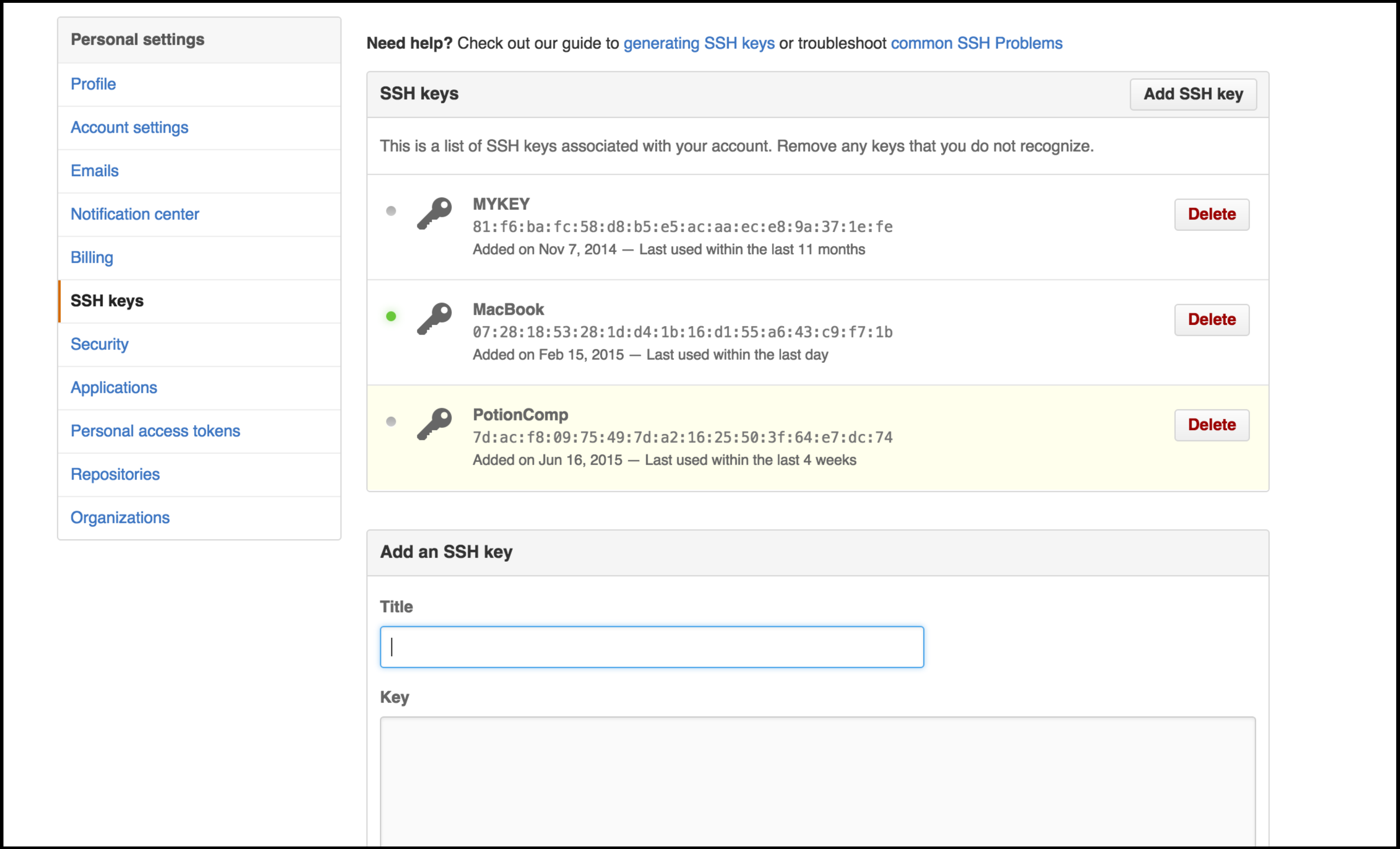
Add your key to the SSH Agent



```
$ pbcopy < ~/.ssh/id_rsa.pub
# Copies the contents of the id_rsa.pub file to your clipboard
```

copy key to clipboard

This command reads your key file (id_rsa.pub) and copies the contents to your clip board.



Personal settings

Profile

Account settings

Emails

Notification center

Billing

SSH keys

Security

Applications

Personal access tokens

Repositories

Organizations

Need help? Check out our guide to generating SSH keys or troubleshoot common SSH Problems

SSH keys

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

MYKEY
`81:f6:ba:fc:58:d8:b5:e5:ac:aa:ec:e8:9a:37:1e:fe`
Added on Nov 7, 2014 — Last used within the last 11 months

MacBook
`07:28:18:53:28:1d:d4:1b:16:d1:55:a6:43:c9:f7:1b`
Added on Feb 15, 2015 — Last used within the last day

PotionComp
`7d:ac:f8:09:75:49:7d:a2:16:25:50:3f:64:e7:dc:74`
Added on Jun 16, 2015 — Last used within the last 4 weeks

Add an SSH key

Title

Key

Add key to Github by going to Settings. Paste your key in the key section (command + v in the key section and it will appear) now you can create new repos + clone them to your local machine!!

Example flow of operations

git status

git add .

git commit -m “i am saving my work to github. I am writing..”

git status

git push origin master

Where is git? Once you've installed git, you can verify it has been installed by

opening up the Terminal and typing git

```
$ git
```

If you see a 'command not recognized' error, you probably haven't installed git

Go through the process of creating a repo for your Github pages site. Clone it inside the webDev directory you made earlier. Ta Da!

http://yourUserName.github.io