Web Programming with Python and JavaScript

Lecture 0: Git

June 26, 2018

Staff

Staff

- Instructors
  - Doug Lloyd (lloyd@cs50.harvard.edu)
  - David J. Malan (malan@harvard.edu)

- Teaching Fellows
  - Rodrigo Daboin Sanchez (daboinsanchez@college.harvard.edu)
  - Alex Mariona (agmariona@college.harvard.edu)
  - David Nunez (david.q.nunez@gmail.com)

- web@cs50.harvard.edu

Lectures

Overview

- Git
- HTML and CSS
- Flask
- SQL
- APIs and ORMs
- JavaScript
- Front ends
- Django
- DevOps, CI, and CD
- Scalability
- Security
- App Party

Projects

Projects

- Project 0
- Project 1
- Project 2
- Project 3
- Final Project

Sections

forms.cs50.io/web/2018/summer/preferences

Office Hours

forms.cs50.io/web/2018/summer/preferences

CS50 IDE

http://cs50.io

CS50 IDE File Edit Find View Go v135 MEMORY CPU DISK Share

~/workspace/

Collaborate Outline Debugger

workspace/

```
~/workspace/ $
```

Workspace name

your-project-name

Description

Make a short description of your workspace

Hosted workspace | Clone workspace | Remote SSH workspace | Salesforce

Private
This is a workspace for your eyes only

Public
This will create a workspace for everybody to see

Clone from Git or Mercurial URL (optional)

e.g. ajaxorg/ace or git@github.com:ajaxorg/ace.git

Choose a template

HTML5 | Node.js | PHP, Apache & ... | Python | Django | Ruby

C++ | Wordpress | Rails Tutorial | Blank | Harvard's CS50

Create workspace

Workspace name

your-project-name

Description

Make a short description of your workspace

Hosted workspace | Clone workspace | Remote SSH workspace | Salesforce

Private
This is a workspace for your eyes only

Public
This will create a workspace for everybody to see

Clone from Git or Mercurial URL (optional)

e.g. ajaxorg/ace or git@github.com:ajaxorg/ace.git

Choose a template

HTML5 | Node.js | PHP, Apache & ... | Python | Django | Ruby

C++ | Wordpress | Rails Tutorial | Blank | Harvard's CS50

Create workspace

Git

Git

- We use Git to keep track of changes to our code.

Git

- We use Git to keep track of changes to our code.

```
a = 1
b = 2
c = 3
```

Create file

Git

- We use Git to keep track of changes to our code.

```
a = 1
b = 2
c = 3
```

Create file

```
a = 1
b = 2
c = 3
d = 4
```

Add a line

Git

- We use Git to keep track of changes to our code.

```
a = 1
b = 2
c = 3
```

Create file

```
a = 1
b = 2
c = 3
d = 4
```

Add a line

```
a = 1
b = 2
d = 4
```

Delete a line

Git

- We also use Git to synchronize code across multiple devices.

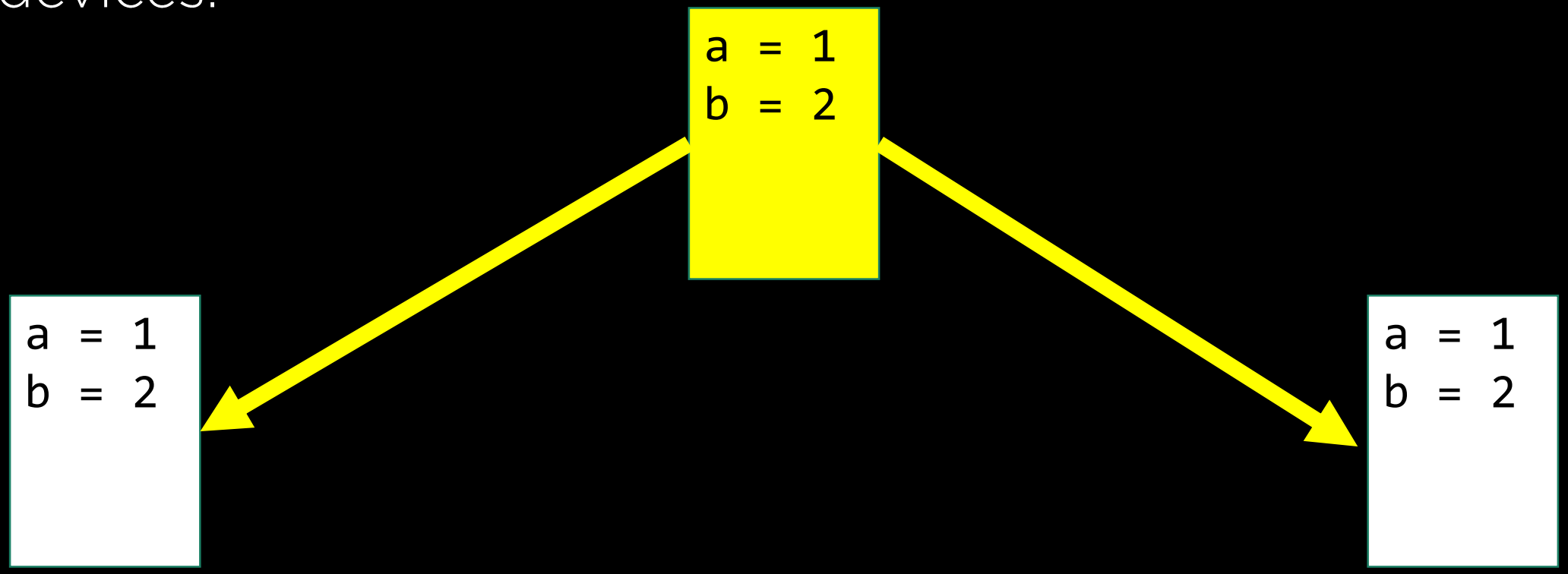
Git

- We also use Git to synchronize code across multiple devices.

```
a = 1
b = 2
```

Git

- We also use Git to synchronize code across multiple devices.



Git

- We also use Git to synchronize code across multiple devices.

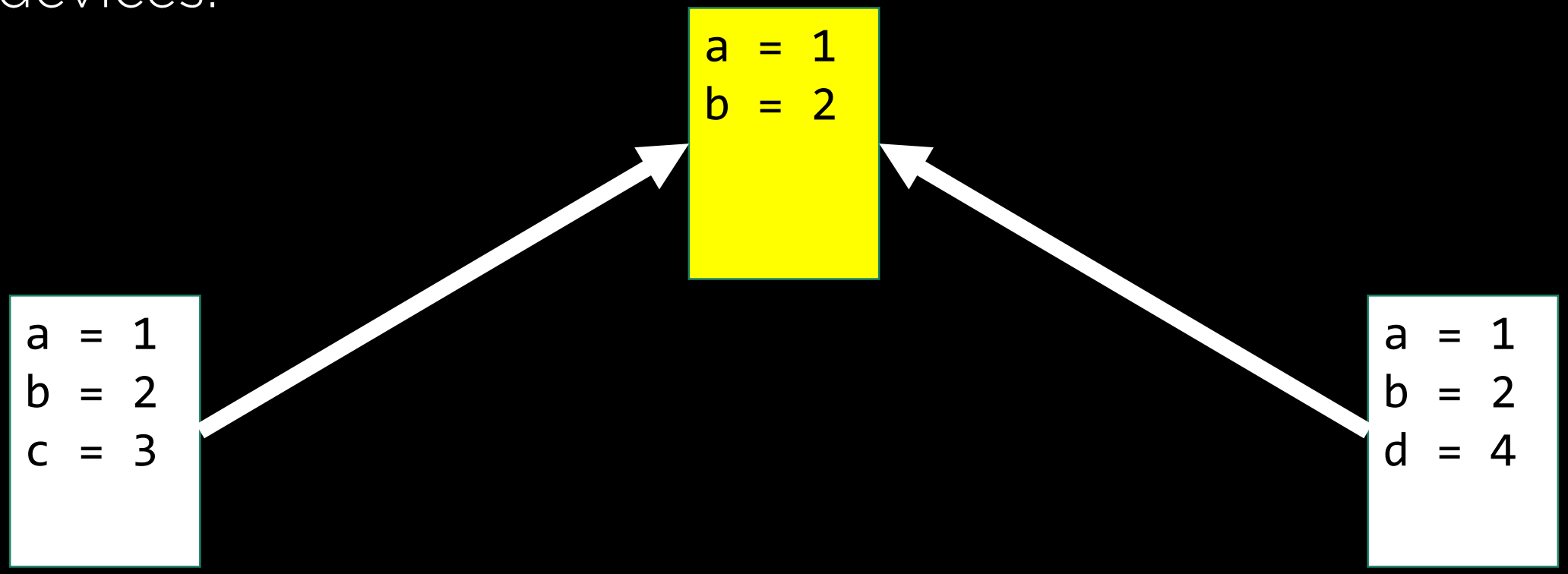
```
a = 1
b = 2
```

```
a = 1
b = 2
c = 3
```

```
a = 1
b = 2
d = 4
```

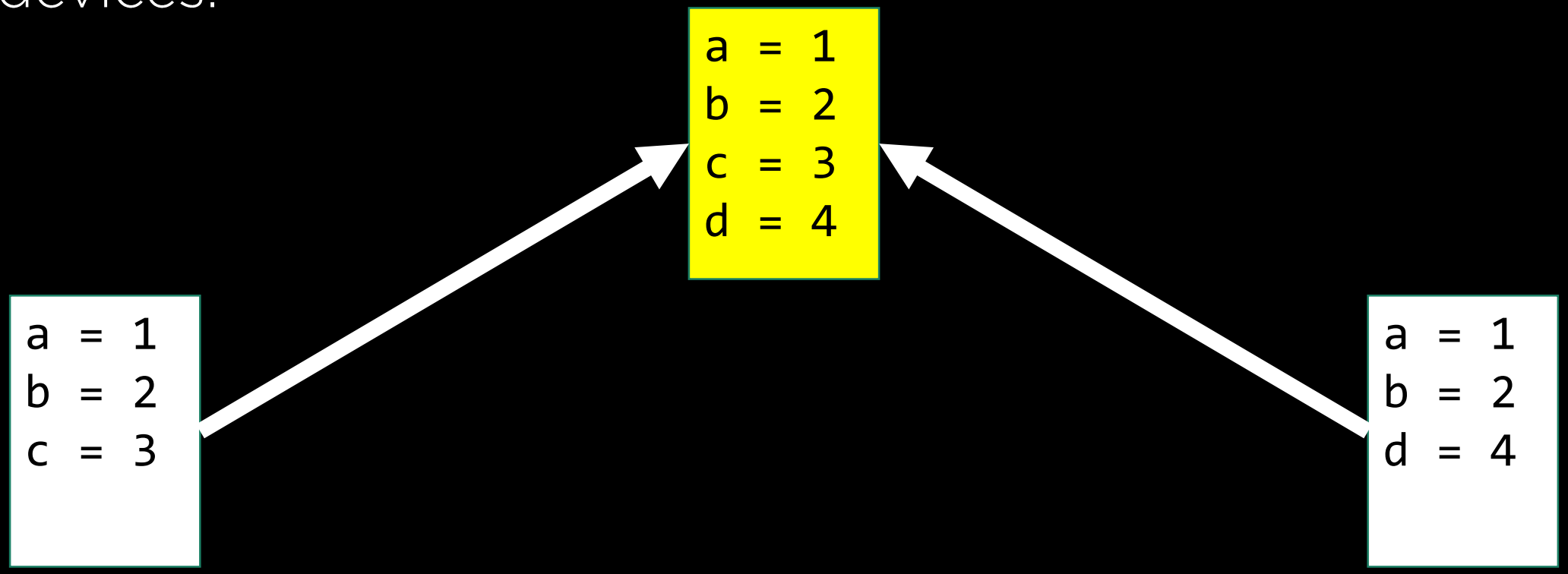
Git

- We also use Git to synchronize code across multiple devices.



Git

- We also use Git to synchronize code across multiple devices.



Git

- We also use Git to synchronize code across multiple devices.

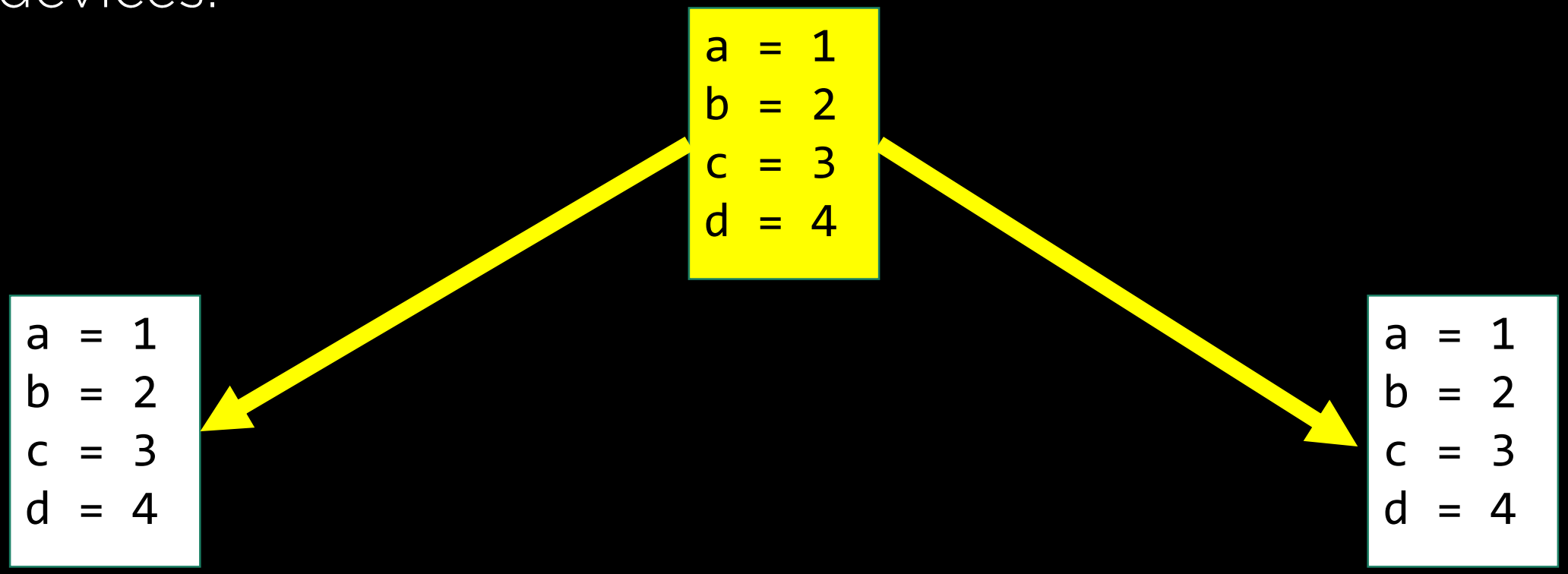
```
a = 1
b = 2
c = 3
d = 4
```

```
a = 1
b = 2
c = 3
```

```
a = 1
b = 2
d = 4
```

Git

- We also use Git to synchronize code across multiple devices.



Git

- Git allows us to test changes to our code without losing the original.

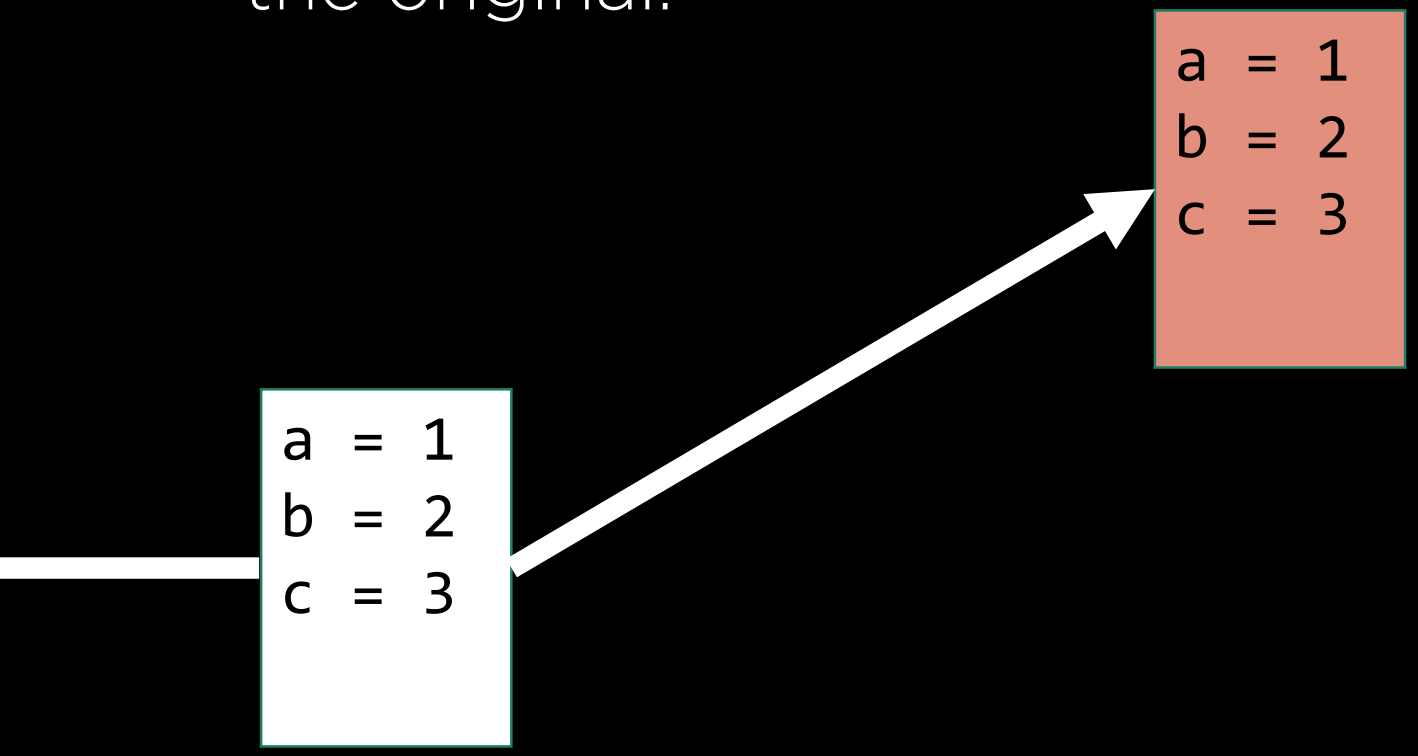
Git

- Git allows us to test changes to our code without losing the original.

```
a = 1
b = 2
c = 3
```

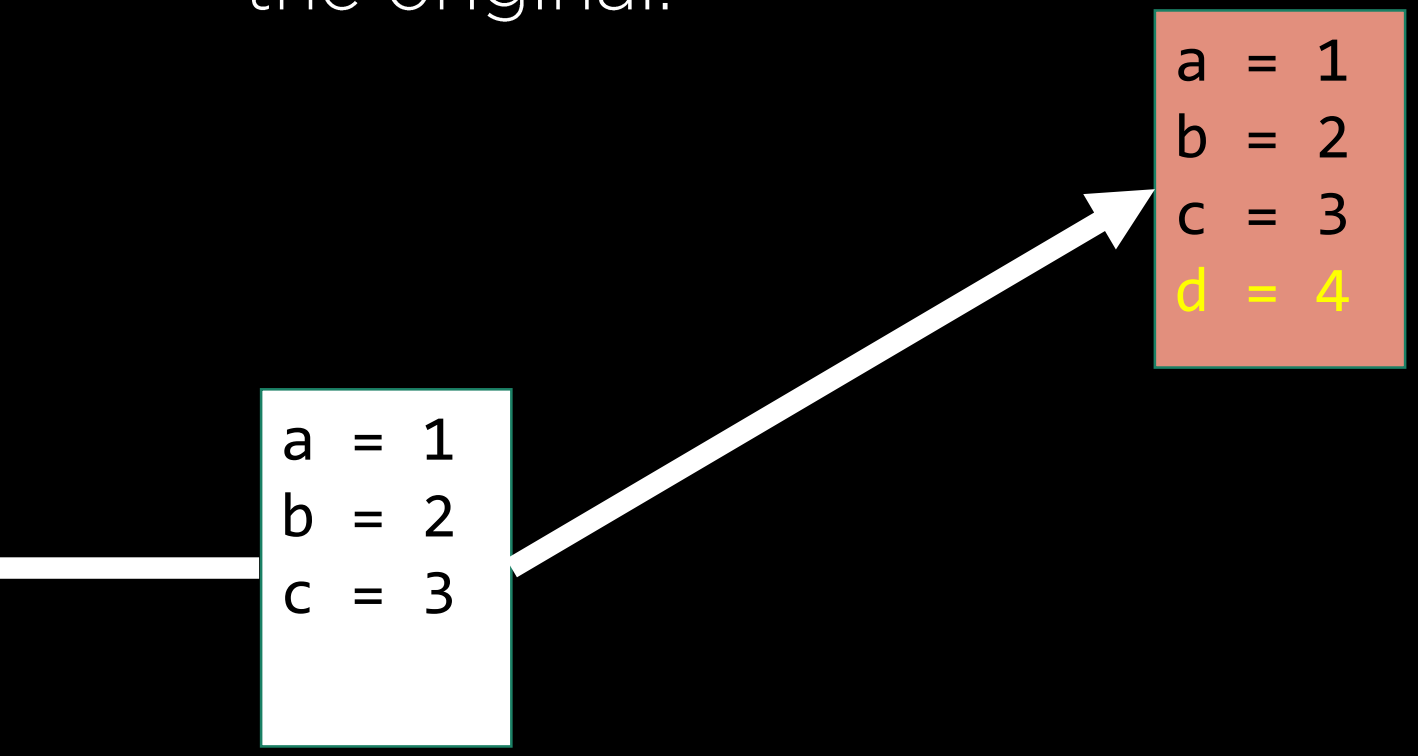
Git

- Git allows us to test changes to our code without losing the original.



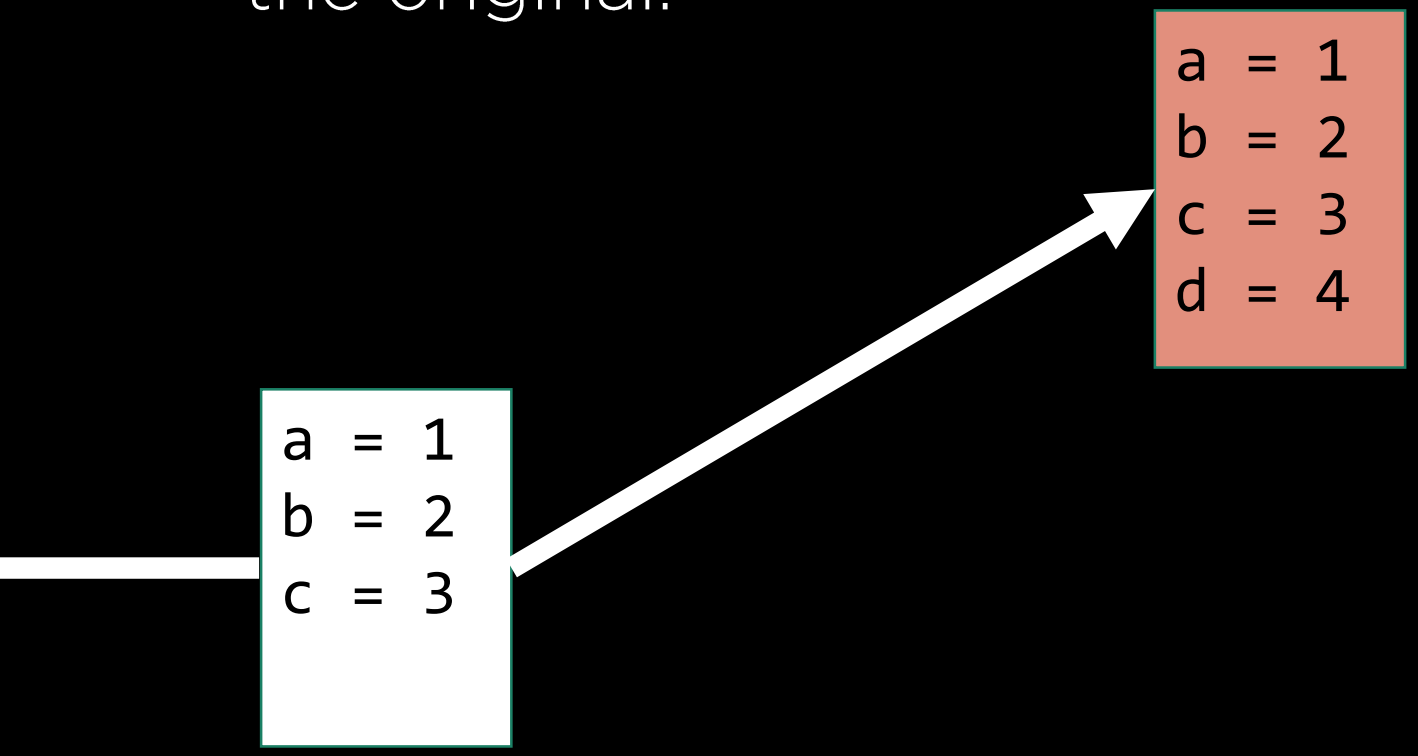
Git

- Git allows us to test changes to our code without losing the original.



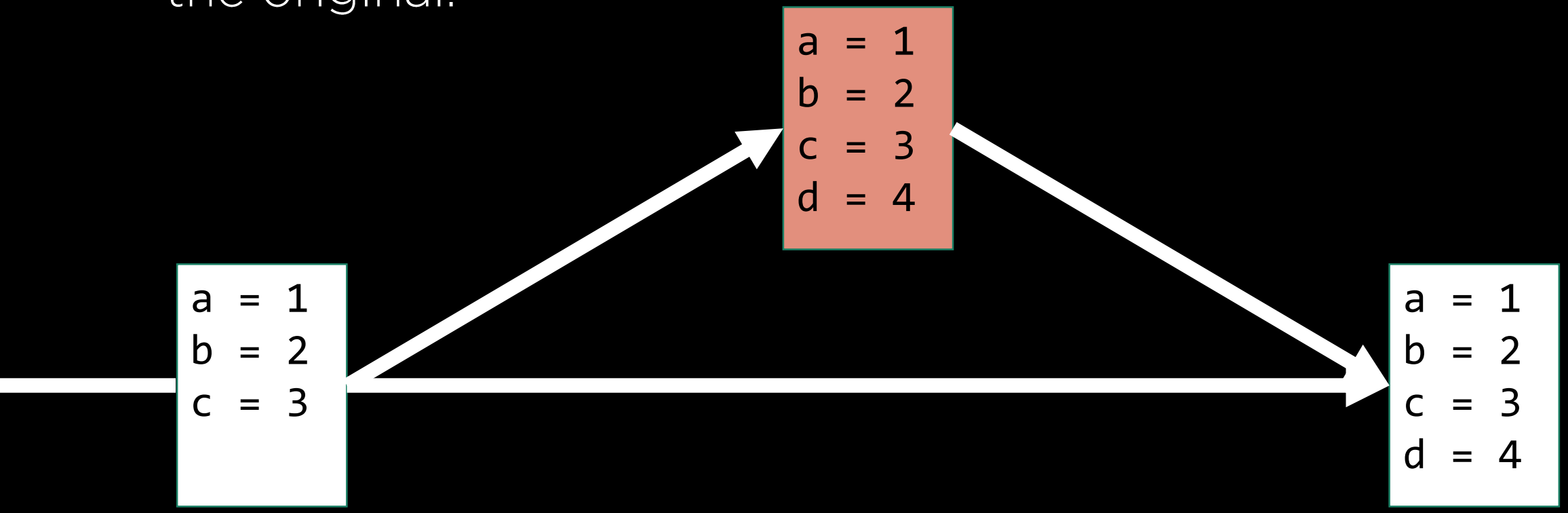
Git

- Git allows us to test changes to our code without losing the original.



Git

- Git allows us to test changes to our code without losing the original.



Git

- We can go back in time to prior versions of our code.

Git

- We can go back in time to prior versions of our code.

```
a = 1
b = 2
c = 3
```

Create file

```
a = 1
b = 2
c = 3
d = 4
```

Add a line

```
a = 1
b = 2
d = 4
```

Delete a line

Git

- We can go back in time to prior versions of our code.

```
a = 1
b = 2
c = 3
```

Create file

```
a = 1
b = 2
c = 3
d = 4
```

Add a line

GitHub

GitHub

- A central hub on the internet for you to store, view, and contribute to your own and others' repositories.

GitHub

- A central hub on the internet for you to store, view, and contribute to your own and others' repositories.

- A "repository" is Git/GitHub terminology for a collection of related code.

GitHub

- A central hub on the internet for you to store, view, and contribute to your own and others' repositories.
- A "repository" is Git/GitHub terminology for a collection of related code.
- You will likely find it easiest to first create a repository on GitHub's web interface, before working on it at the command line.

git clone

`git clone`

- Used to initially get a local copy of a repository stored elsewhere (e.g. GitHub).

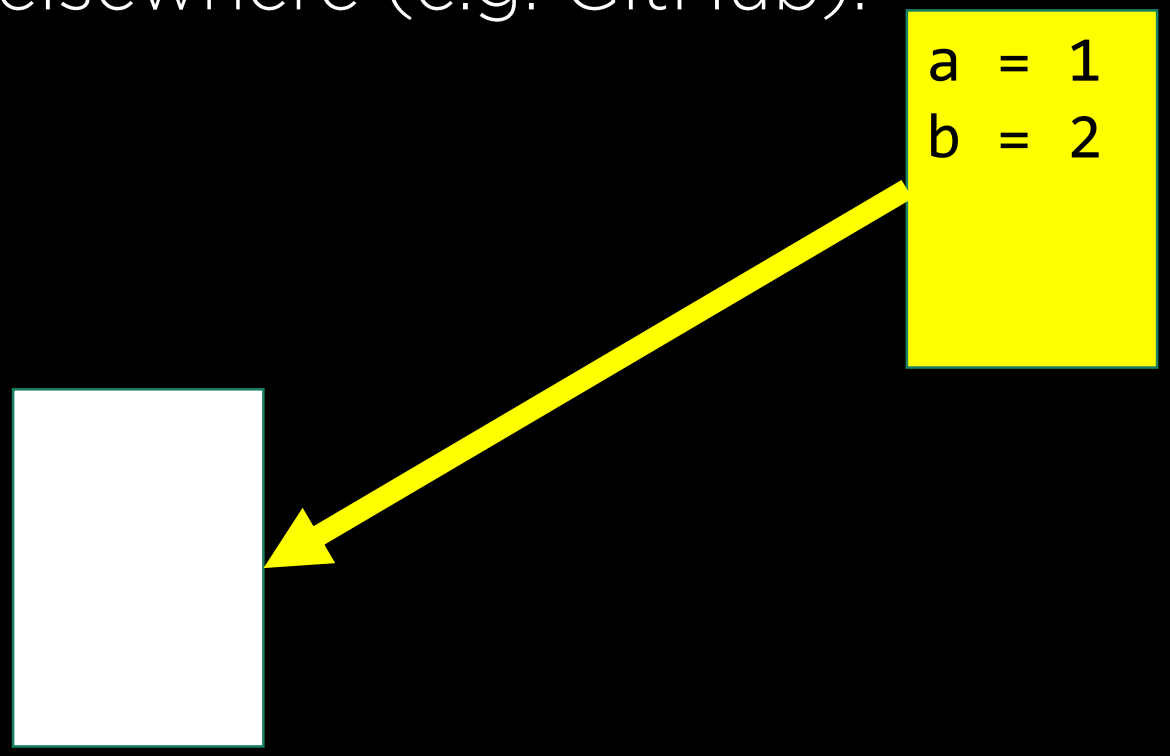
git clone

- Used to initially get a local copy of a repository stored elsewhere (e.g. GitHub).

```
a = 1
b = 2
```

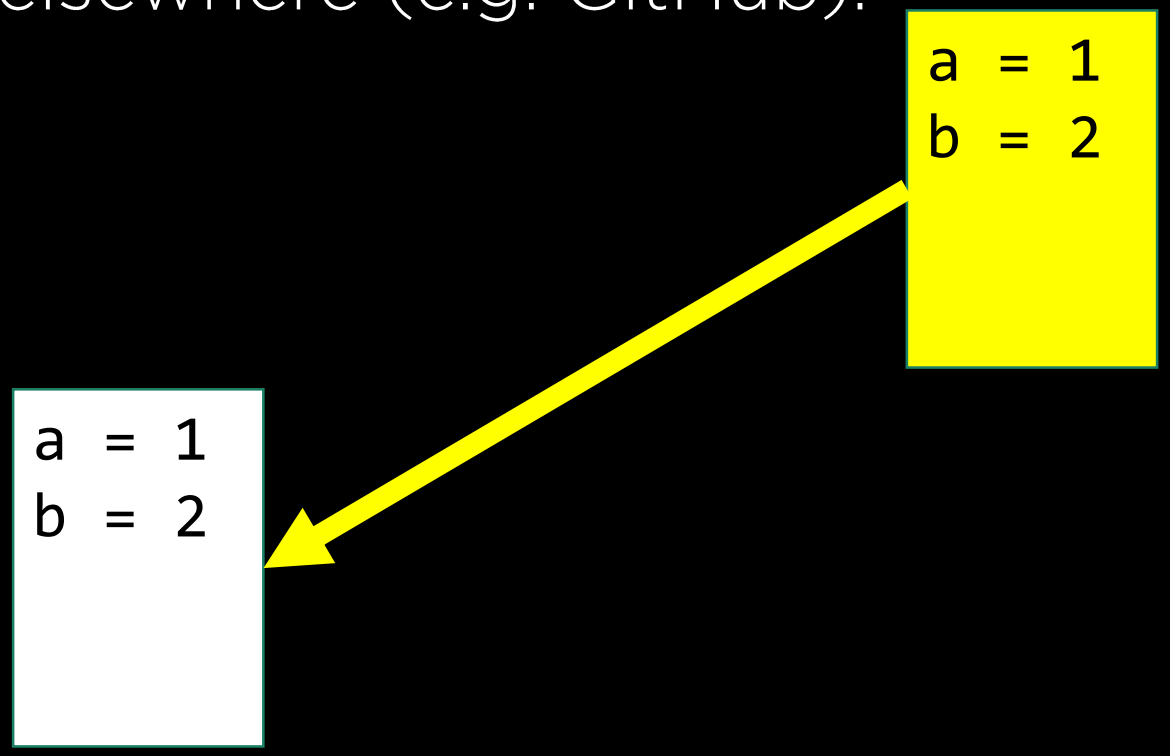
git clone

- Used to initially get a local copy of a repository stored elsewhere (e.g. GitHub).



git clone

- Used to initially get a local copy of a repository stored elsewhere (e.g. GitHub).



`git clone`

- Used to initially get a local copy of a repository stored elsewhere (e.g. GitHub).

```
git clone <url>
```

git add

`git add`

- Used to let Git know which files we want it to look at when we prepare our next “snapshot” of the repository.

`git add`

- Used to let Git know which files we want it to look at when we prepare our next "snapshot" of the repository.

```
a = 1
b = 2
```

```
a = 1
b = 2
```

`git add`

- Used to let Git know which files we want it to look at when we prepare our next "snapshot" of the repository.

```
a = 1
b = 2
```

```
a = 1
b = 2
c = 3
```

`git add`

- Used to let Git know which files we want it to look at when we prepare our next "snapshot" of the repository.

```
a = 1
b = 2
```

```
a = 1
b = 2
c = 3
```

`git add`

- Used to let Git know which files we want it to look at when we prepare our next "snapshot" of the repository.

```
a = 1
b = 2
```

```
a = 1
b = 2
c = 3
```

git add

- Used to let Git know which files we want it to look at when we prepare our next "snapshot" of the repository.

```
a = 1
b = 2
```

```
a = 1
b = 2
c = 3
```

git add

- Used to let Git know which files we want it to look at when we prepare our next “snapshot” of the repository.

```
git add <filename>
git add -A
```

git status

`git status`

- Used to let you know the current state of your directory (i.e., what's changed)

git commit

git commit

- Used to tell Git that it's time to take that next "snapshot" of the repository.

```
a = 1
b = 2
```

```
a = 1
b = 2
c = 3
```

git commit

- Used to tell Git that it's time to take that next "snapshot" of the repository.

```
a = 1
b = 2
```

```
a = 1
b = 2
```

```
a = 1
b = 2
c = 3
```

git commit

- Used to tell Git that it's time to take that next "snapshot" of the repository.

```
a = 1
b = 2
```

```
a = 1
b = 2
```

```
a = 1
b = 2
c = 3
```

Add a line

git commit

- Used to tell Git that it’s time to take that next “snapshot” of the repository.

```
git commit –m "message"
git commit –m "Add a line"
```

git push

`git push`

- Used to take all of the snapshots and changes made locally and send them to your “remote” (i.e., GitHub).

git push

- Used to take all of the snapshots and changes made locally and send them to your “remote” (i.e., GitHub).

```
a = 1
b = 2
```

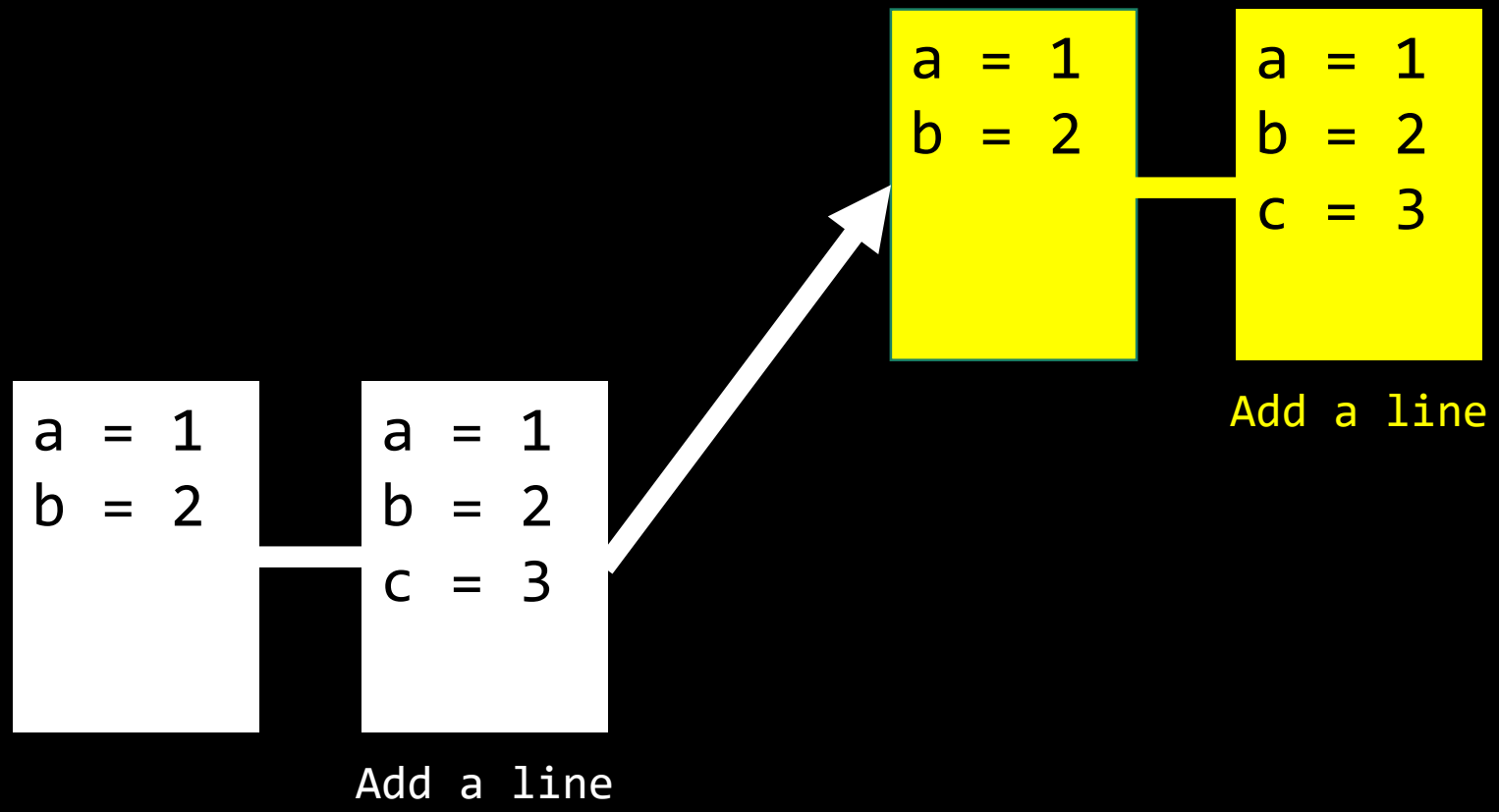
```
a = 1
b = 2
```

```
a = 1
b = 2
c = 3
```

Add a line

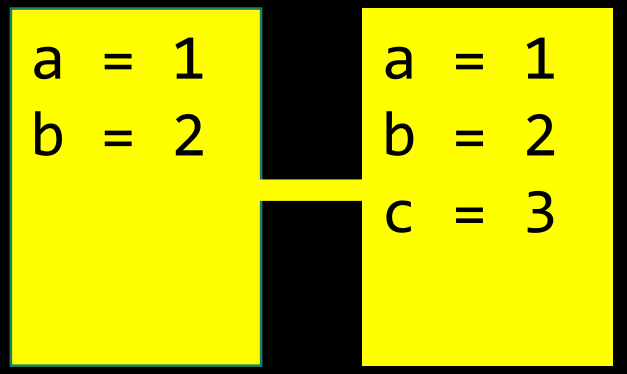
git push

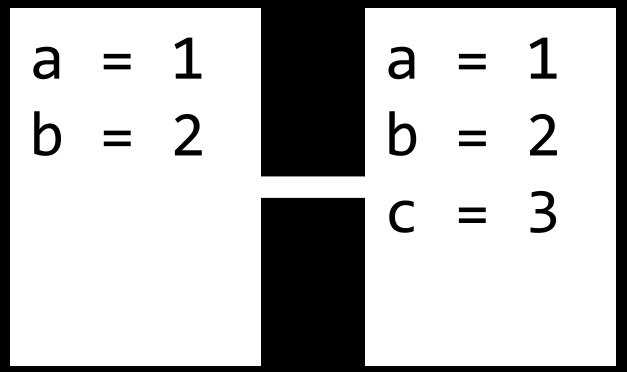
- Used to take all of the snapshots and changes made locally and send them to your “remote” (i.e., GitHub).



git push

- Used to take all of the snapshots and changes made locally and send them to your “remote” (i.e., GitHub).





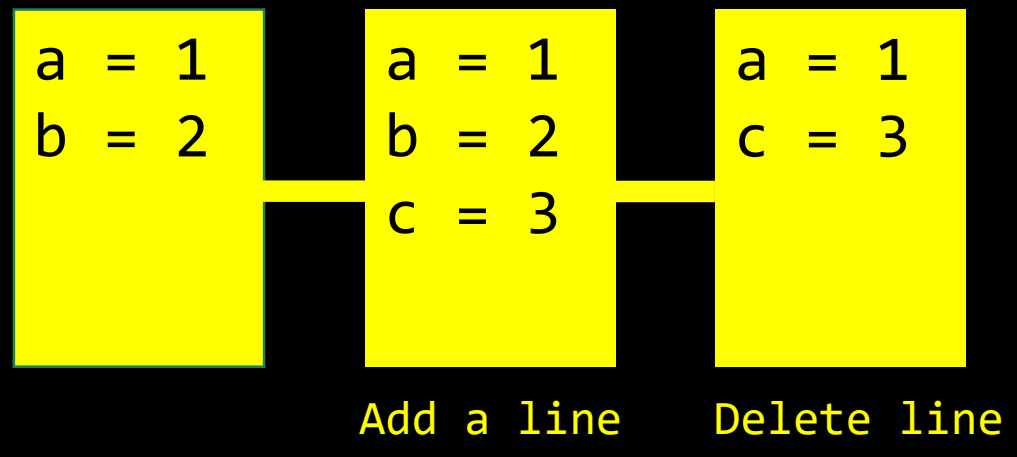
git pull

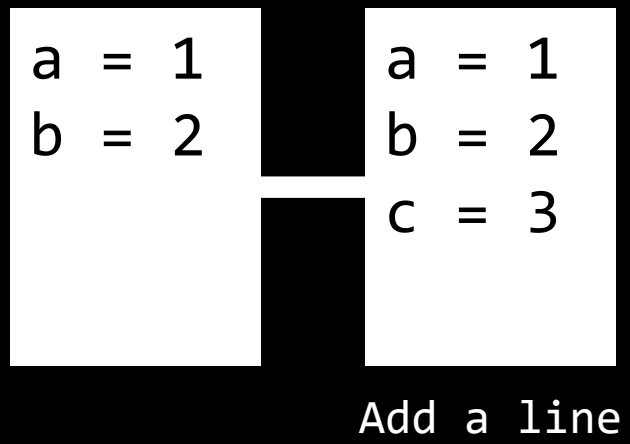
`git pull`

- Used to get all of the snapshots and changes that are stored on your “remote” (i.e., GitHub) onto your local machine.

git pull

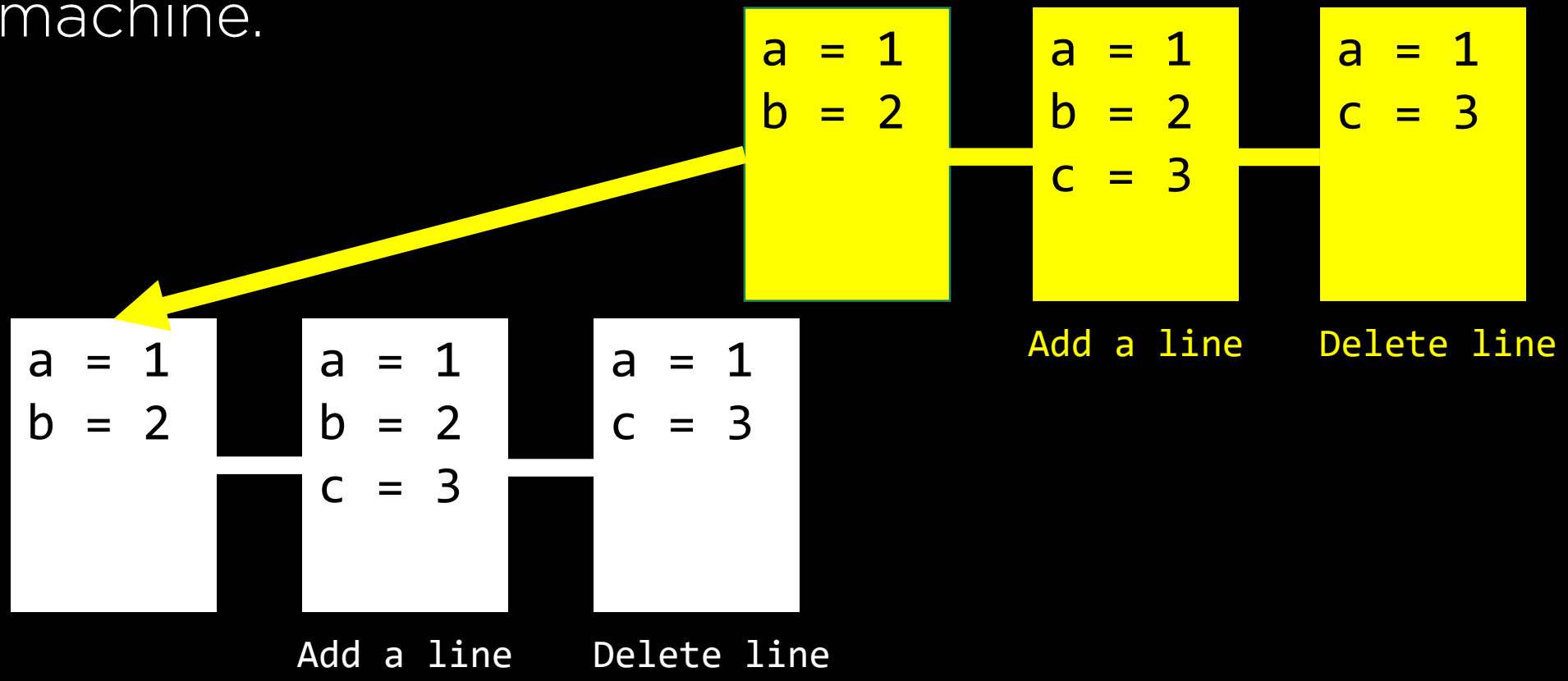
- Used to get all of the snapshots and changes that are stored on your “remote” (i.e., GitHub) onto your local machine.





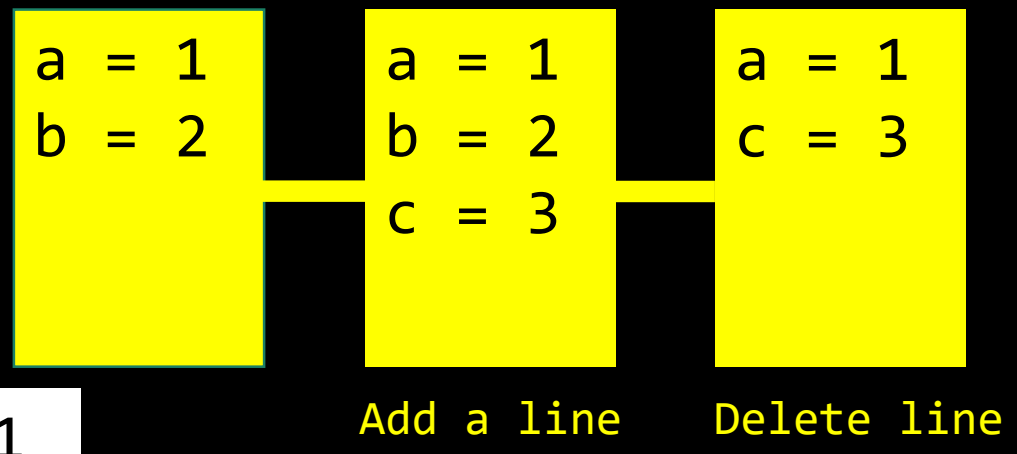
git pull

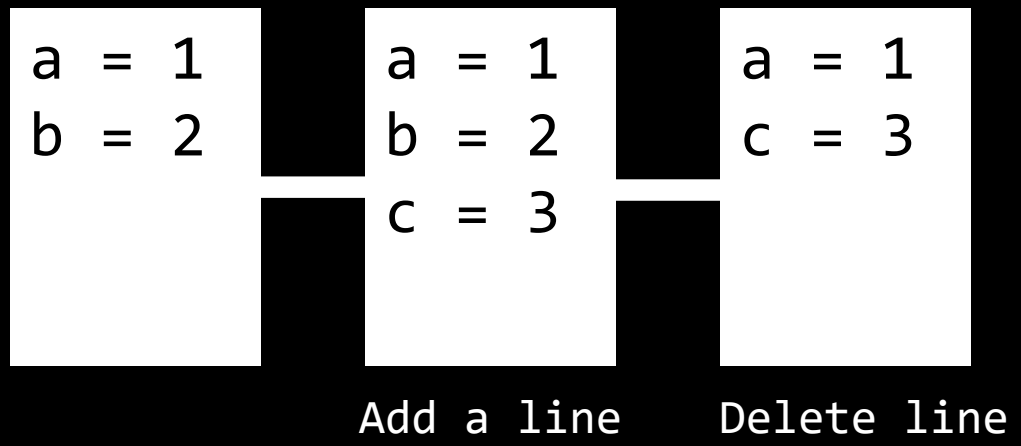
- Used to get all of the snapshots and changes that are stored on your “remote” (i.e., GitHub) onto your local machine.



git pull

- Used to get all of the snapshots and changes that are stored on your “remote” (i.e., GitHub) onto your local machine.





Conflicts

Conflicts

- A conflict can occur when locally saved changes are incompatible with changes on the remote.

```
a = 1
b = 2
c = 5
d = 4
```

```
a = 1
b = 2
c = 3
d = 4
```

Conflicts

- A conflict can occur when locally saved changes are incompatible with changes on the remote.

```
a = 1
b = 2
c = 5
d = 4
```

```
a = 1
b = 2
c = 3
d = 4
```

```
CONFLICT (content): Merge conflict in foo.py
Automatic merge failed; fix conflicts and
then commit the result.
```

Conflicts

- A conflict can occur when locally saved changes are incompatible with changes on the remote.

```
a = 1
b = 2
c = 5
d = 4
```

```
a = 1
b = 2
c = 3
d = 4
```

```
CONFLICT (content): Merge conflict in foo.py
Automatic merge failed; fix conflicts and
then commit the result.
```

Conflicts

- A conflict can occur when locally saved changes are incompatible with changes on the remote.

```
a = 1
b = 2
<<<<< HEAD
c = 3
=====
c = 5
>>>>> 2828abcdef0123456789
d = 4
```

Conflicts

- A conflict can occur when locally saved changes are incompatible with changes on the remote.

```
a = 1
b = 2
<<<<< HEAD
c = 3
=====
c = 5
>>>>> 2828abcdef0123456789
d = 4
```

Conflicts

- A conflict can occur when locally saved changes are incompatible with changes on the remote.

```
a = 1
b = 2
<<<<< HEAD
c = 3
=====
c = 5
>>>>> 2828abcdef0123456789
d = 4
```

`git log`

`git log`

- Used to get a history of all of the commits you've made locally.

git reset

`git reset`

- Used to jump back in time to a prior version of your code (perhaps because the current version is broken).

git reset

- Used to jump back in time to a prior version of your code (perhaps because the current version is broken).

```
git reset --hard <commit>
git reset --hard origin/master
```

Break

HTML

HTML Tags

- <h1>, <h2>, ..., <h6>
- , <ol>,
- <img>
- <a>
- <table>
- <form>
- <b>, <i>
- <p>

DOM

DOM

- The *Document Object Model*, a way of considering more visually the natural nesting structure that HTML seems to take on.

DOM

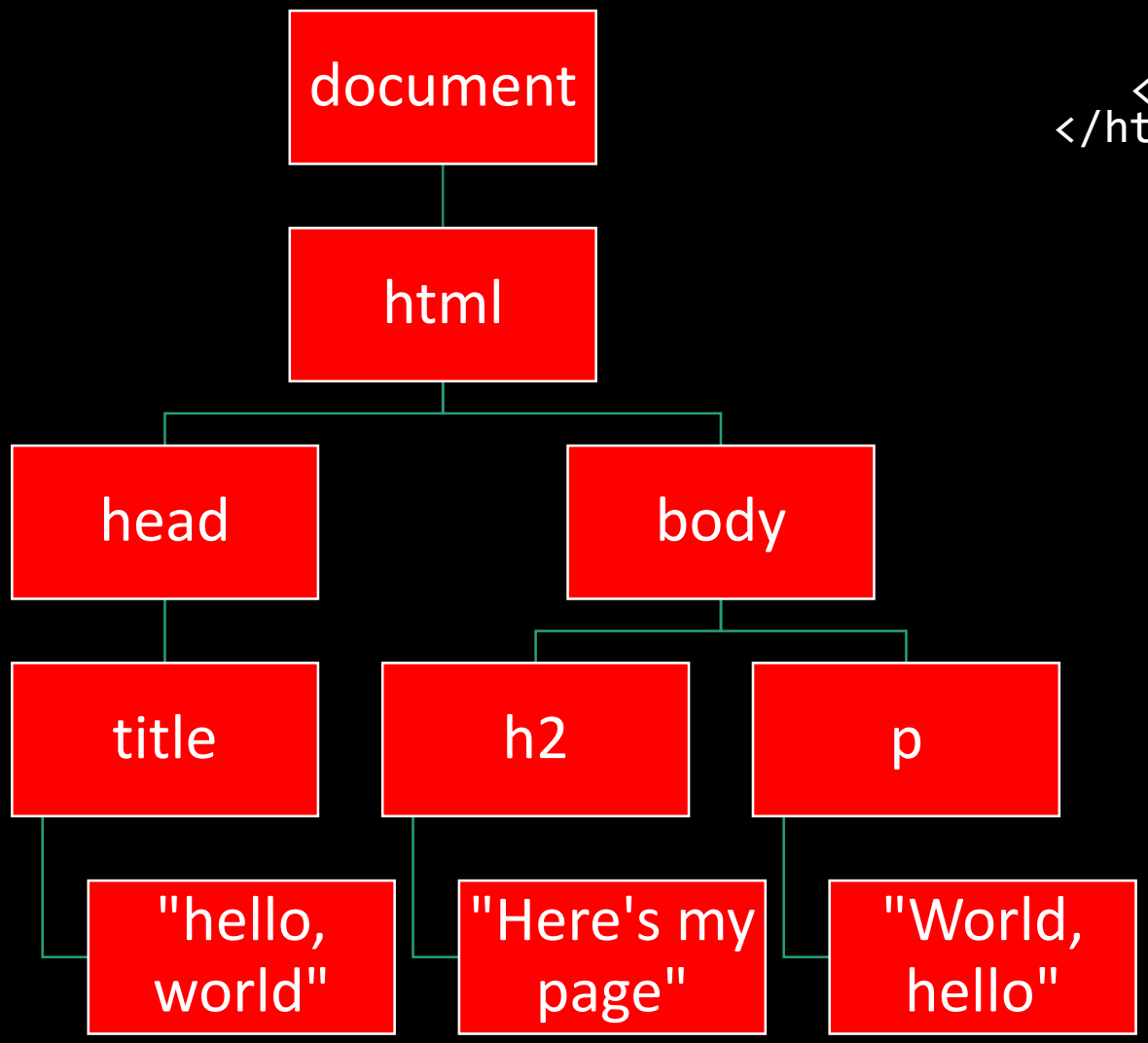
- The *Document Object Model*, a way of considering more visually the natural nesting structure that HTML seems to take on.

- Later, we'll be able to leverage the power of this document object to style and manipulate our sites programmatically.

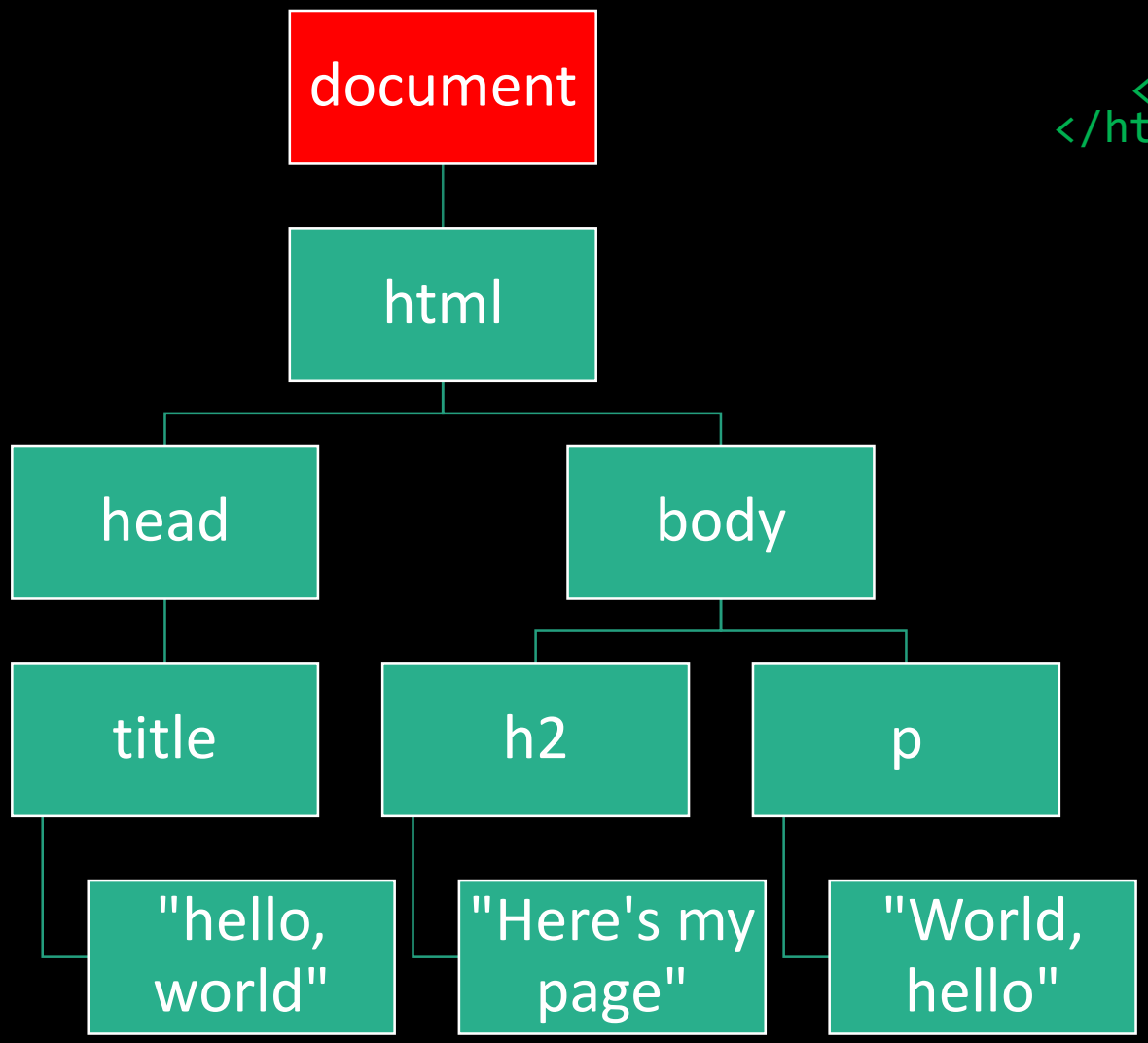
```html
<!DOCTYPE html>
<html>
   <head>
      <title>Hello, world</title>
   </head>
   <body>
      <h2>Here's my page</h2>
      <p>World, hello</p>
   </body>
</html>
```

```html
<!DOCTYPE html>
<html>
   <head>
      <title>Hello, world</title>
   </head>
   <body>
      <h2>Here's my page</h2>
      <p>World, hello</p>
   </body>
</html>
```

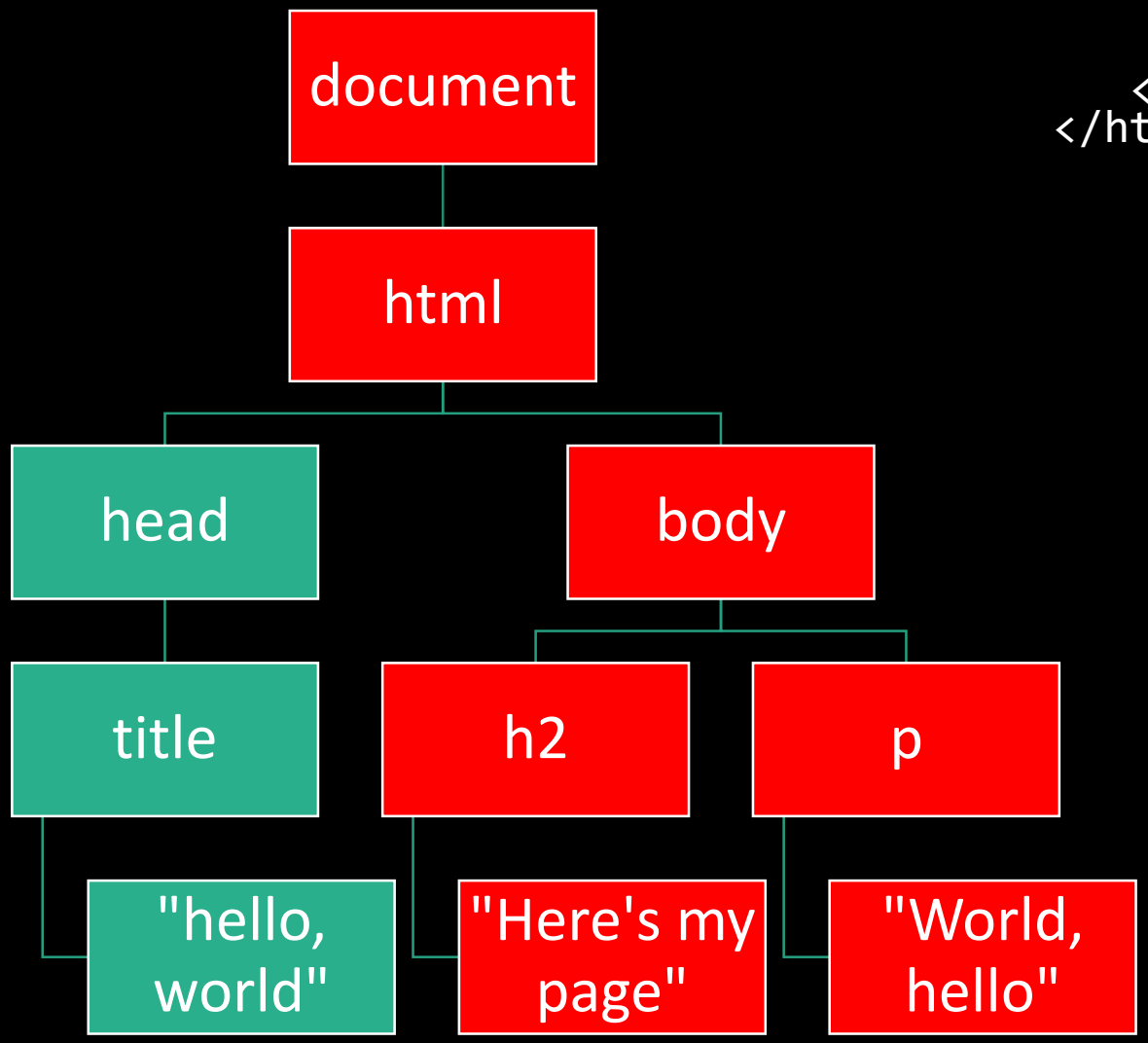
```html
<!DOCTYPE html>
<html>
    <head>
        <title>Hello, world</title>
    </head>
    <body>
        <h2>Here's my page</h2>
        <p>World, hello</p>
    </body>
</html>
```



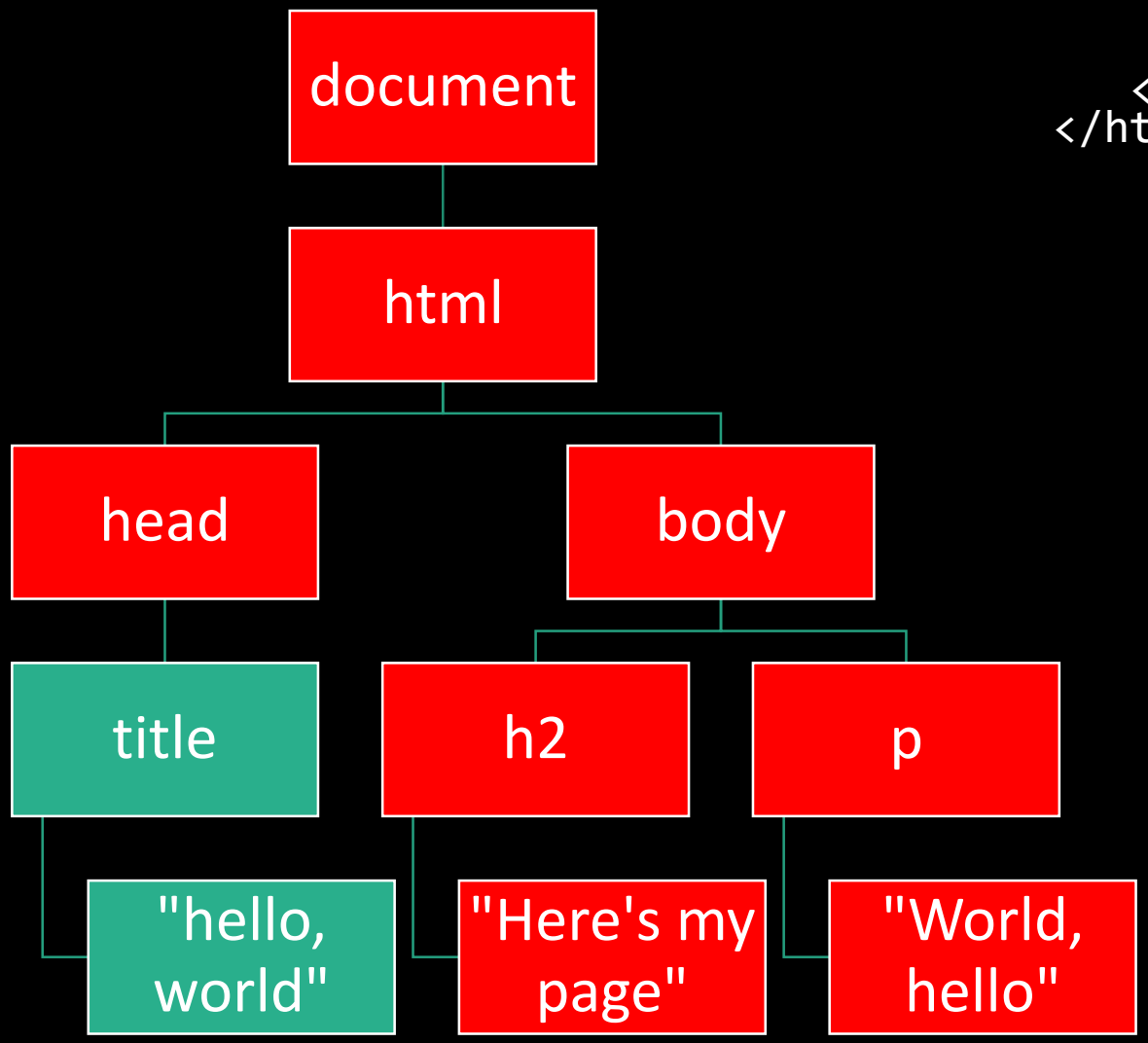
```html
<!DOCTYPE html>
<html>
    <head>
        <title>Hello, world</title>
    </head>
    <body>
        <h2>Here's my page</h2>
        <p>World, hello</p>
    </body>
</html>
```



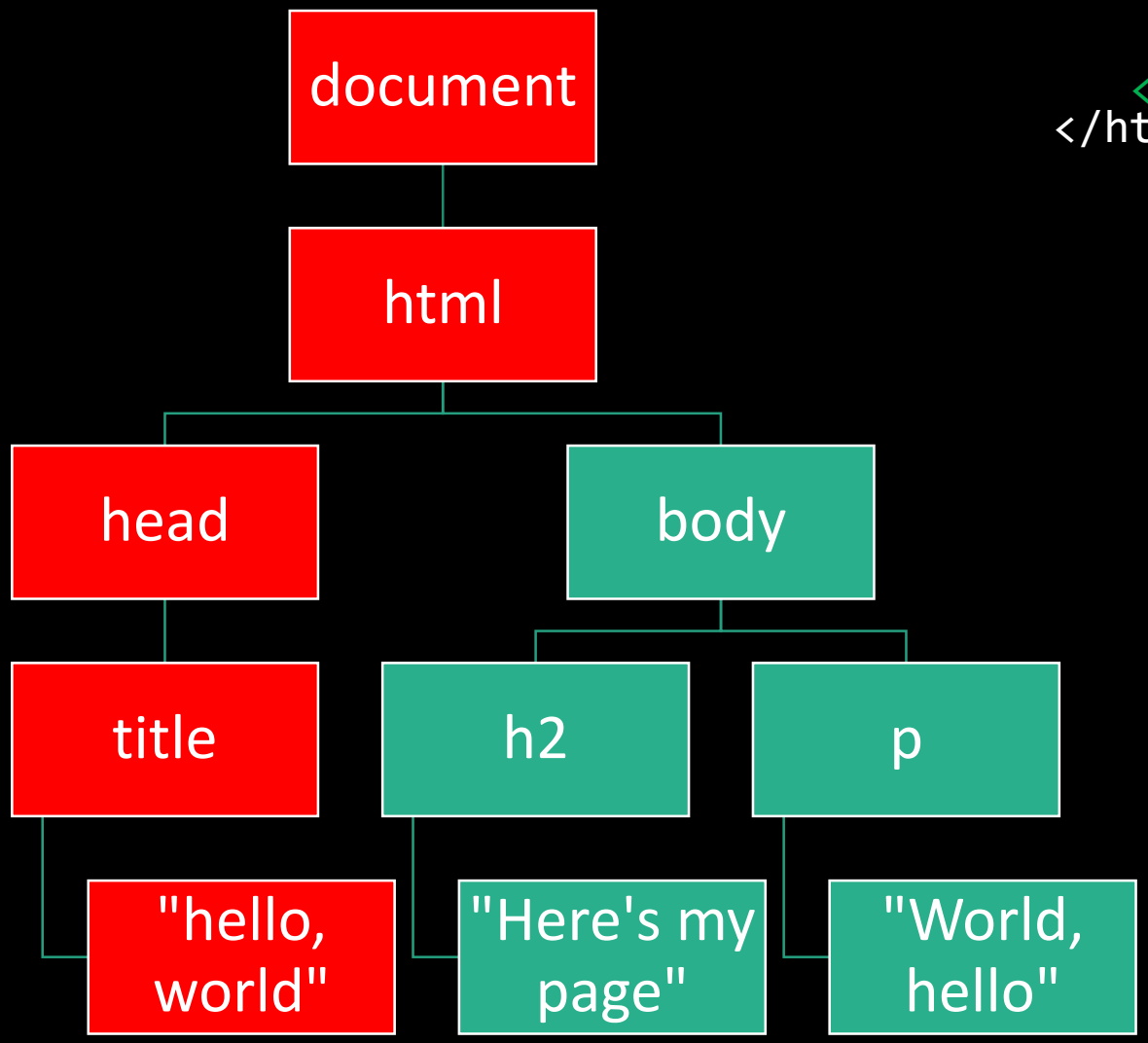
```html
<!DOCTYPE html>
<html>
    <head>
        <title>Hello, world</title>
    </head>
    <body>
        <h2>Here's my page</h2>
        <p>World, hello</p>
    </body>
</html>
```



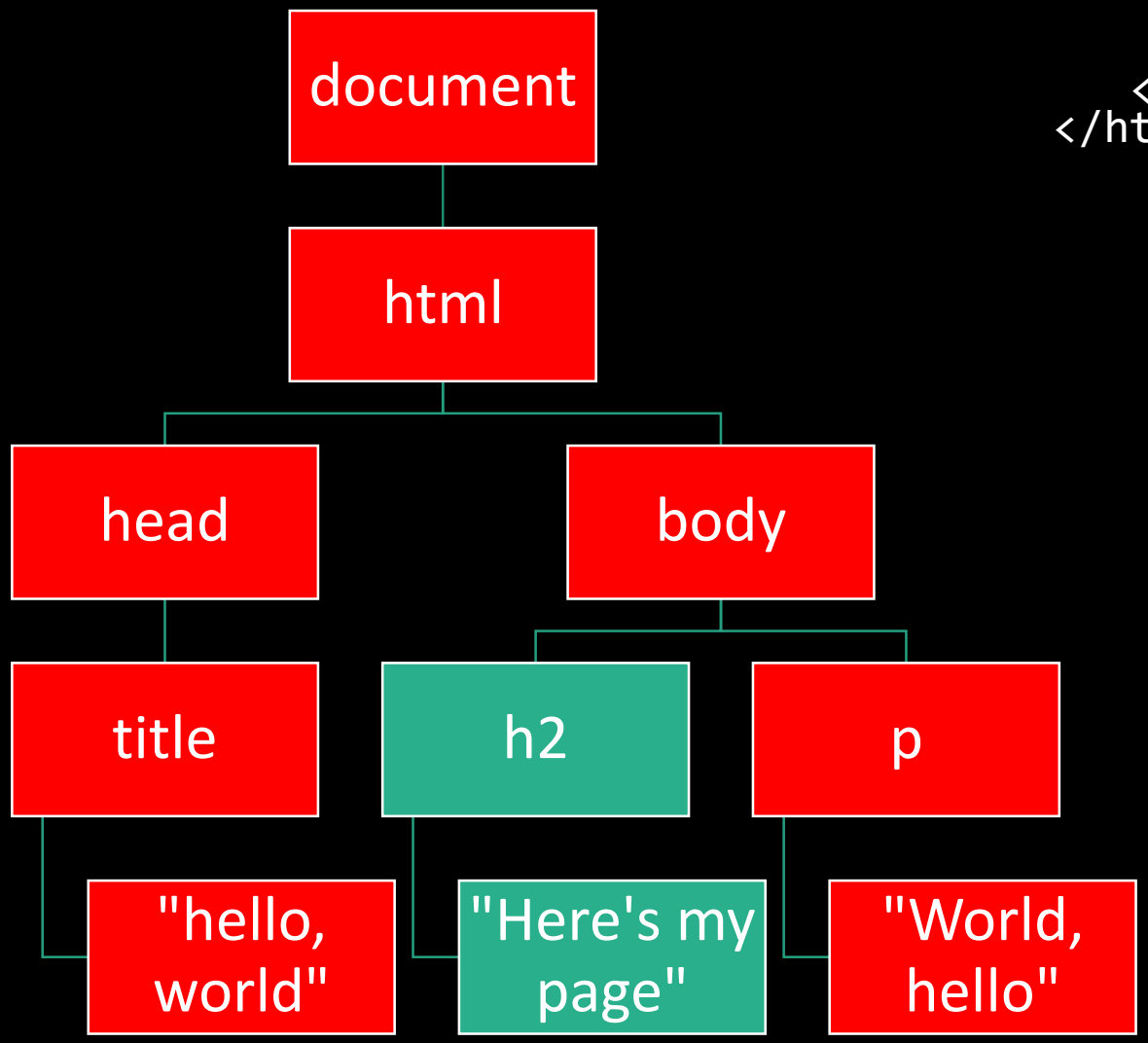
```html
<!DOCTYPE html>
<html>
    <head>
        <title>Hello, world</title>
    </head>
    <body>
        <h2>Here's my page</h2>
        <p>World, hello</p>
    </body>
</html>
```



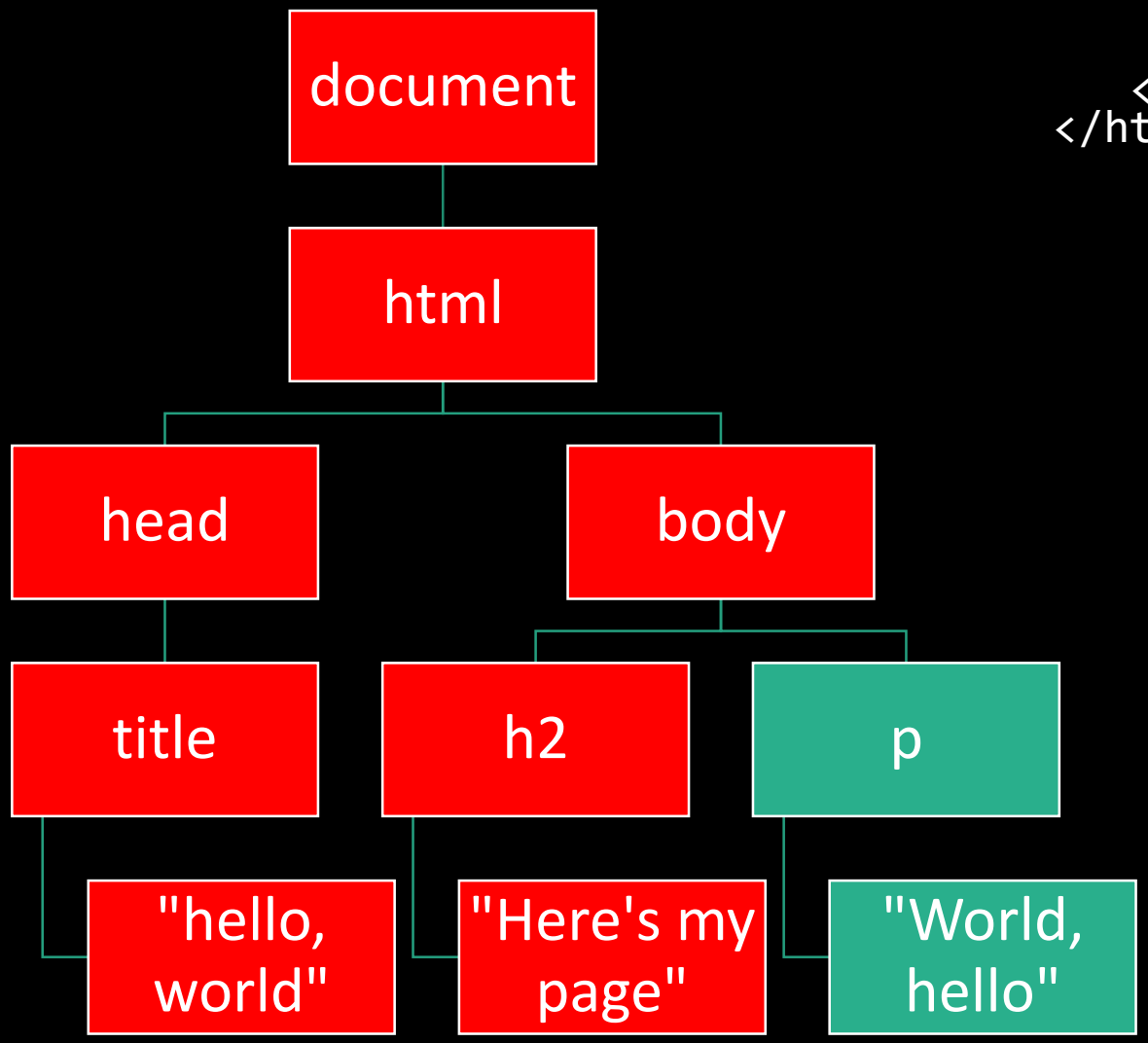
```html
<!DOCTYPE html>
<html>
    <head>
        <title>Hello, world</title>
    </head>
    <body>
        <h2>Here's my page</h2>
        <p>World, hello</p>
    </body>
</html>
```



```html
<!DOCTYPE html>
<html>
    <head>
        <title>Hello, world</title>
    </head>
    <body>
        <h2>Here's my page</h2>
        <p>World, hello</p>
    </body>
</html>
```



```html
<!DOCTYPE html>
<html>
    <head>
        <title>Hello, world</title>
    </head>
    <body>
        <h2>Here's my page</h2>
        <p>World, hello</p>
    </body>
</html>
```



CSS

CSS

- If HTML is the content of our web pages, then CSS (*Cascading Style Sheets*) are how we will style our sites to make them more aesthetically pleasing.

CSS Properties

- color
- text-align
- width
- height
- margin
- padding
- font-family, font-size, font-weight
- border

Abstract Elements and Attributes

- `div`
- `span`

- `id`
- `class`

GitHub Pages

GitHub Pages

- So far, all of the code we’ve written has only been “run” locally, on my own machine or IDE.

GitHub Pages

- So far, all of the code we've written has only been "run" locally, on my own machine or IDE.

- GitHub Pages is a feature of GitHub that effectively allows you to deploy the contents of a repository to live on the internet.