

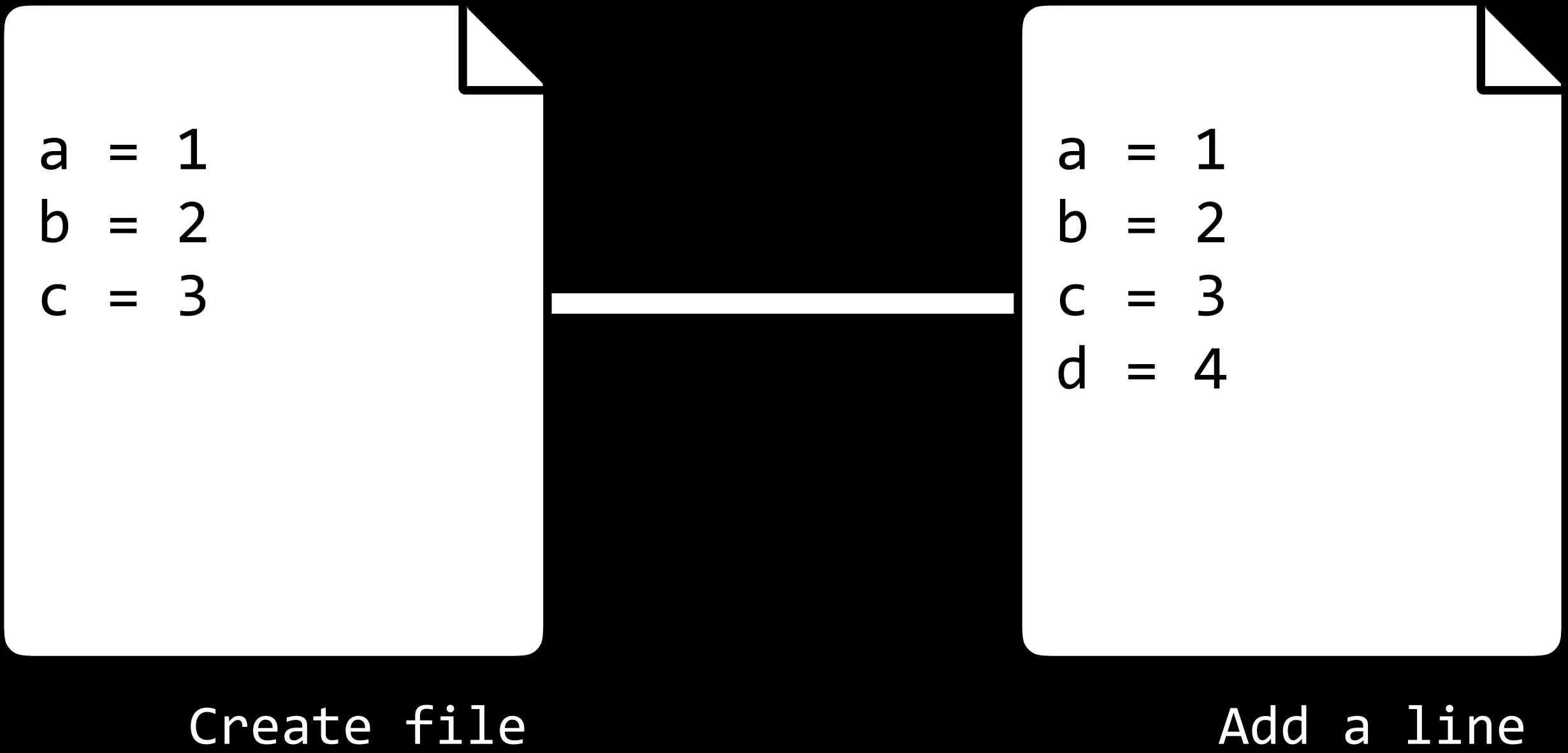
# CS50 Beyond

# Agenda

- Git
  - Repositories
  - Commits
  - Branches and Merges
  - GitHub
- Sass

Git

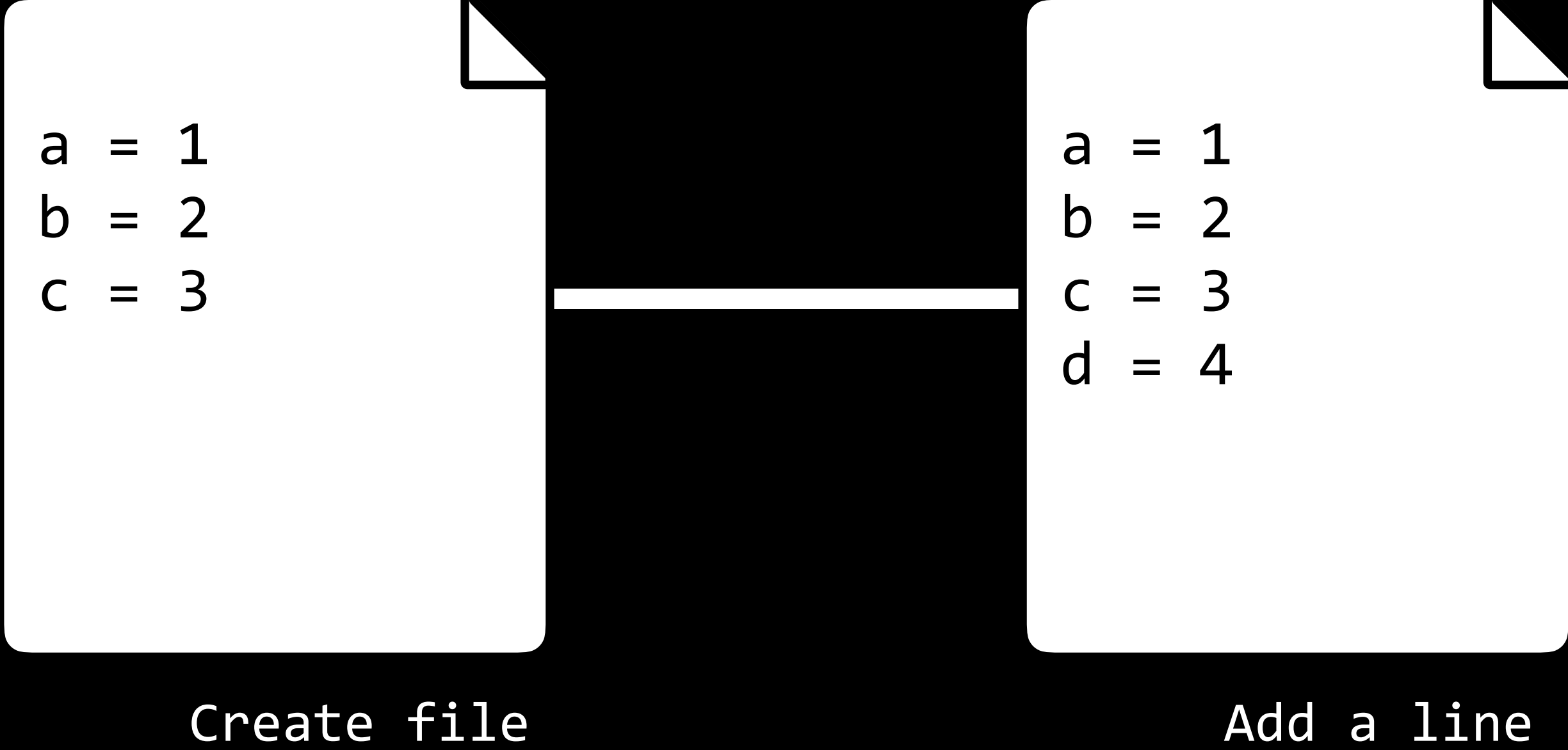
# Keep track of changes to code.



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

Create file

# Keep track of changes to code.



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

Create file

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

Add a line

# Keep track of changes to code.

The diagram illustrates a sequence of three code states connected by horizontal lines. Each state is represented by a white document icon with a folded top-right corner. The first document contains three lines of code: 'a = 1', 'b = 2', and 'c = 3'. The second document contains four lines: 'a = 1', 'b = 2', 'c = 3', and 'd = 4'. The third document contains three lines: 'a = 1', 'c = 3', and 'd = 4'. The label 'Remove a line' is positioned below the third document, indicating that the line 'b = 2' was removed from the second state to reach the third state.

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

Create file

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

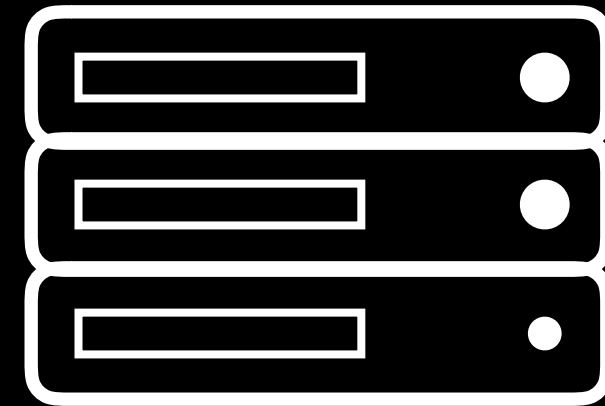
Add a line

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

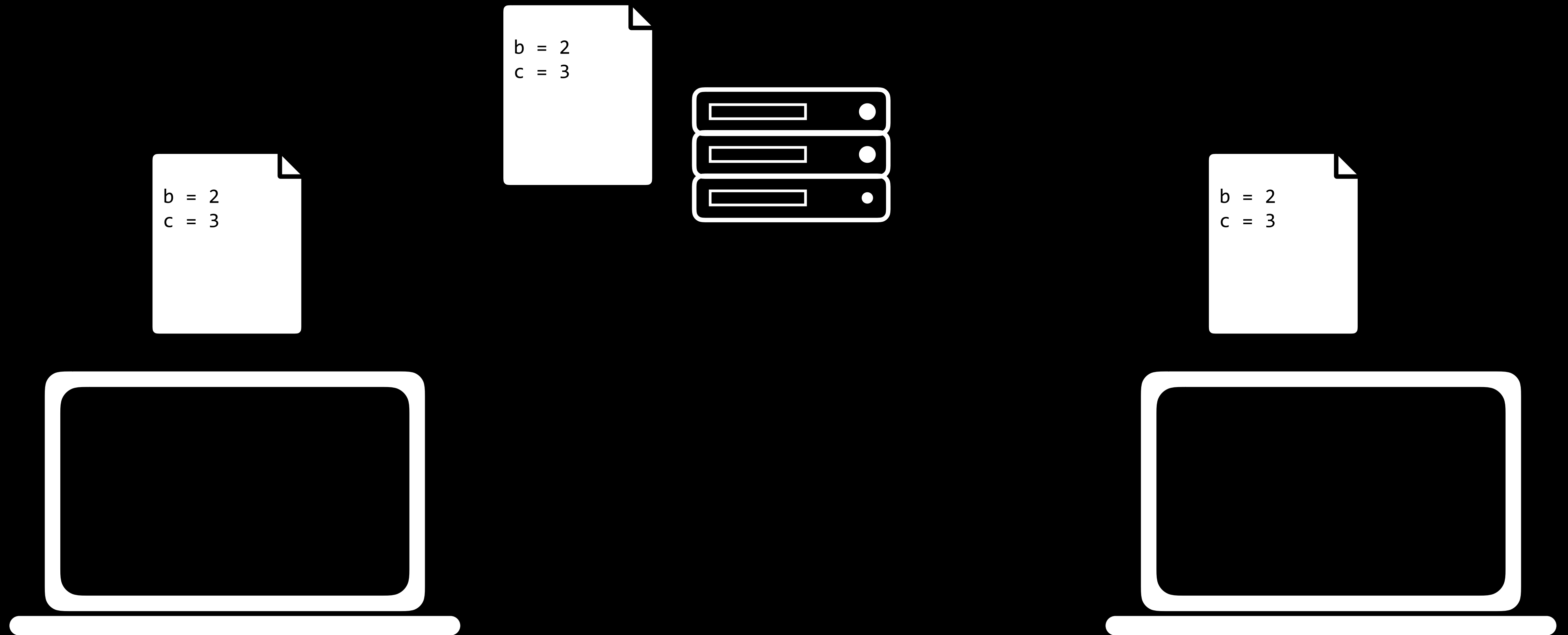
Remove a line

Synchronizes code between different people.

```
b = 2  
c = 3
```

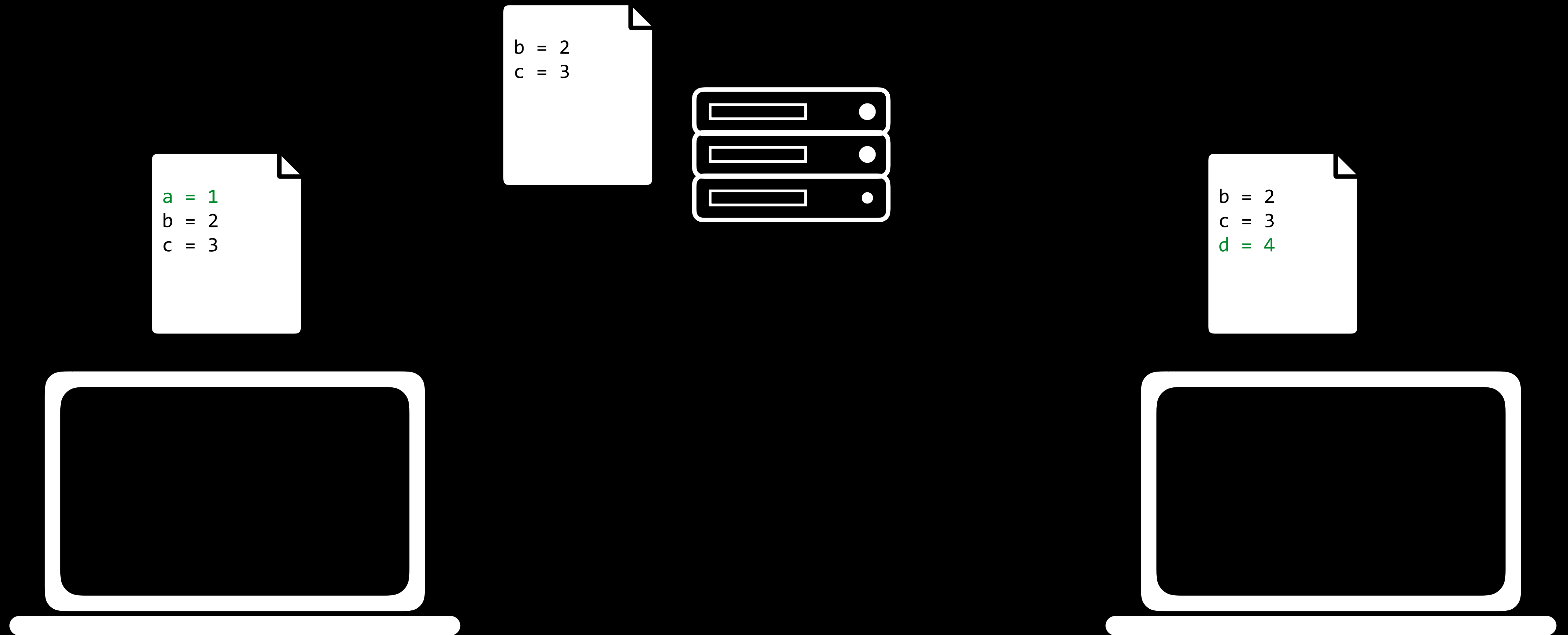


# Synchronizes code between different people.

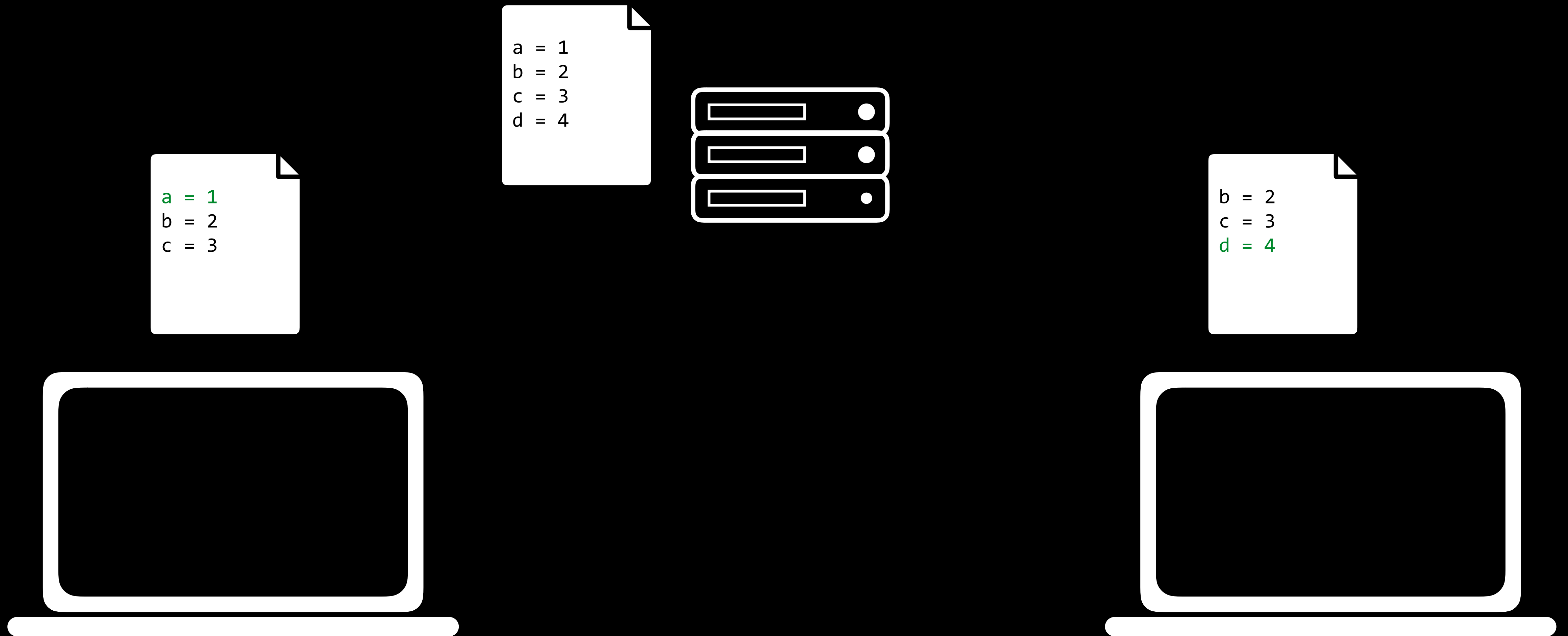




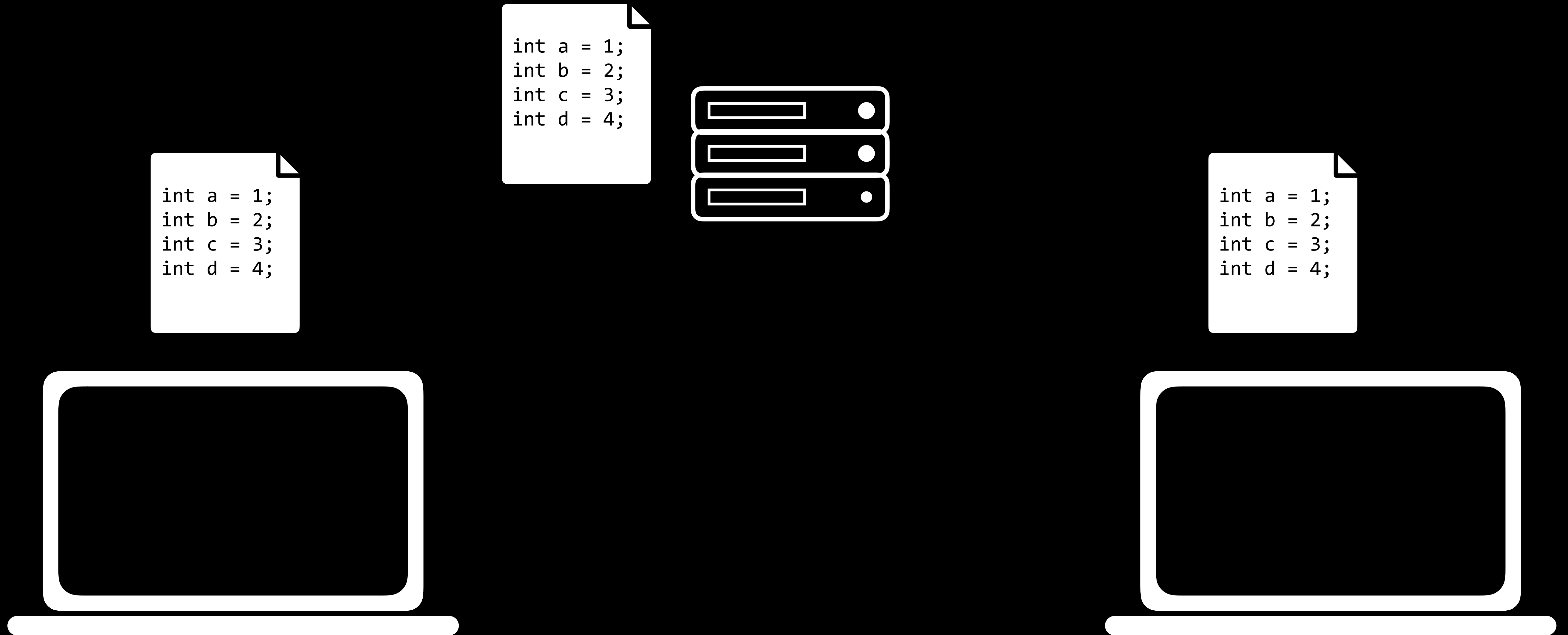
# Synchronizes code between different people.



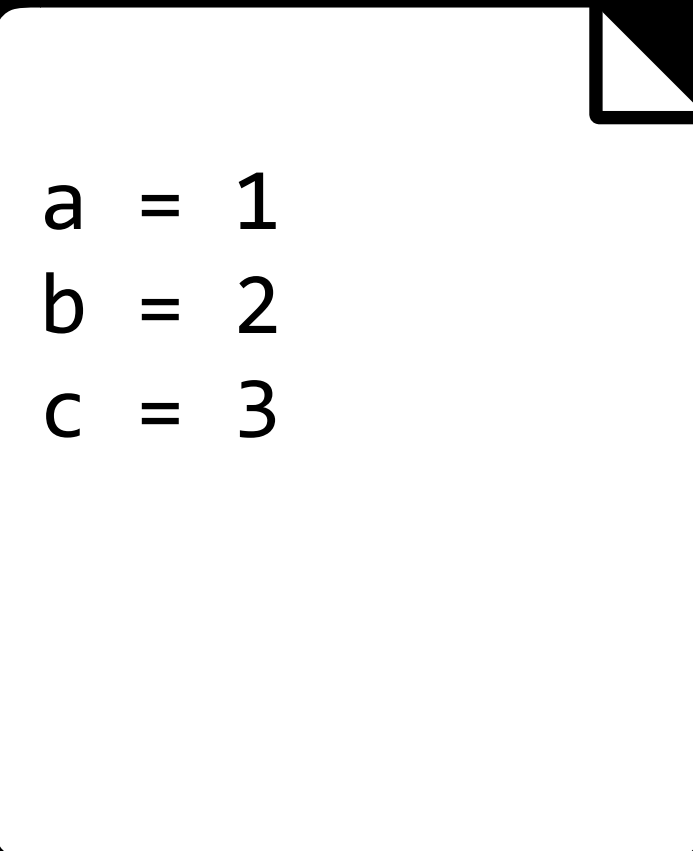
# Synchronizes code between different people.



# Synchronizes code between different people.

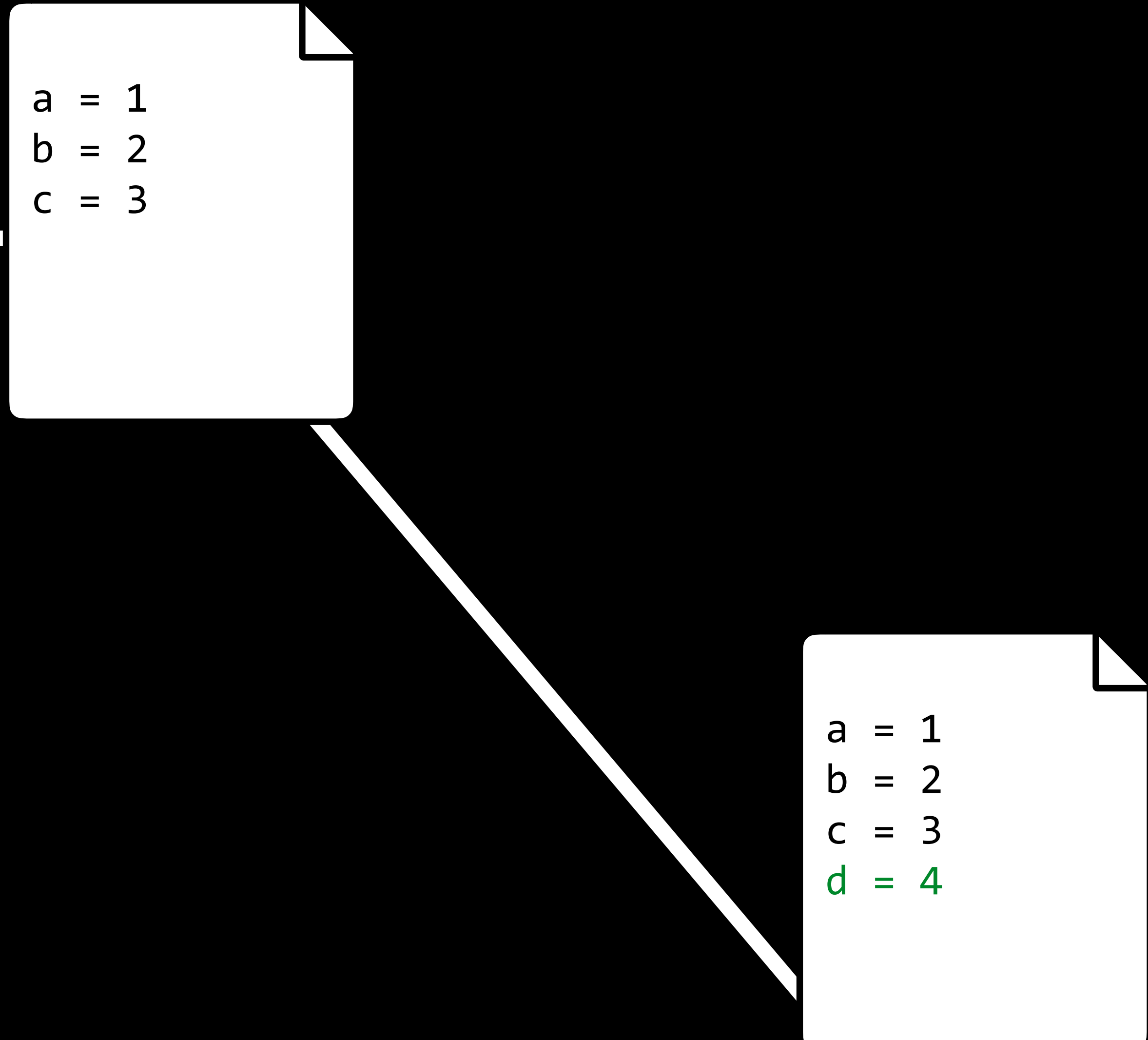


Test changes to code without losing the original.



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

# Test changes to code without losing the original.

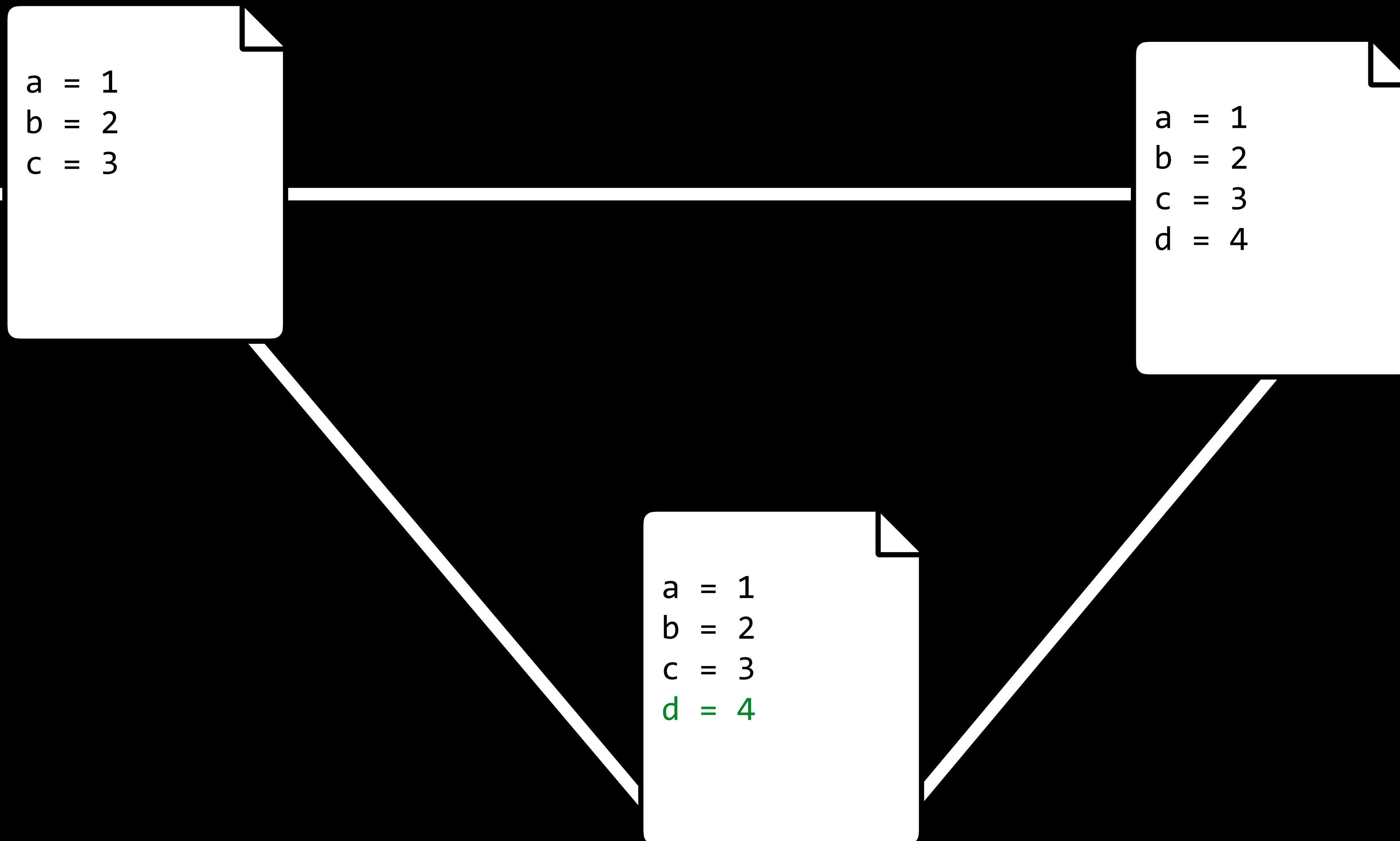


The diagram illustrates a branching strategy for testing code. It starts with a main code file on the left containing three lines of code: `a = 1`, `b = 2`, and `c = 3`. A horizontal line extends from the left edge of this file. From the bottom-right corner of the main file, a diagonal line branches down to a second file on the right. This second file contains the same three lines of code as the first, but with a fourth line added: `d = 4`. The original code in the second file is black, while the new line is green, indicating a change. The main file is connected to the left edge of the slide by a horizontal line.

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

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

# Test changes to code without losing the original.



# Revert back to old versions of code.

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

Create file

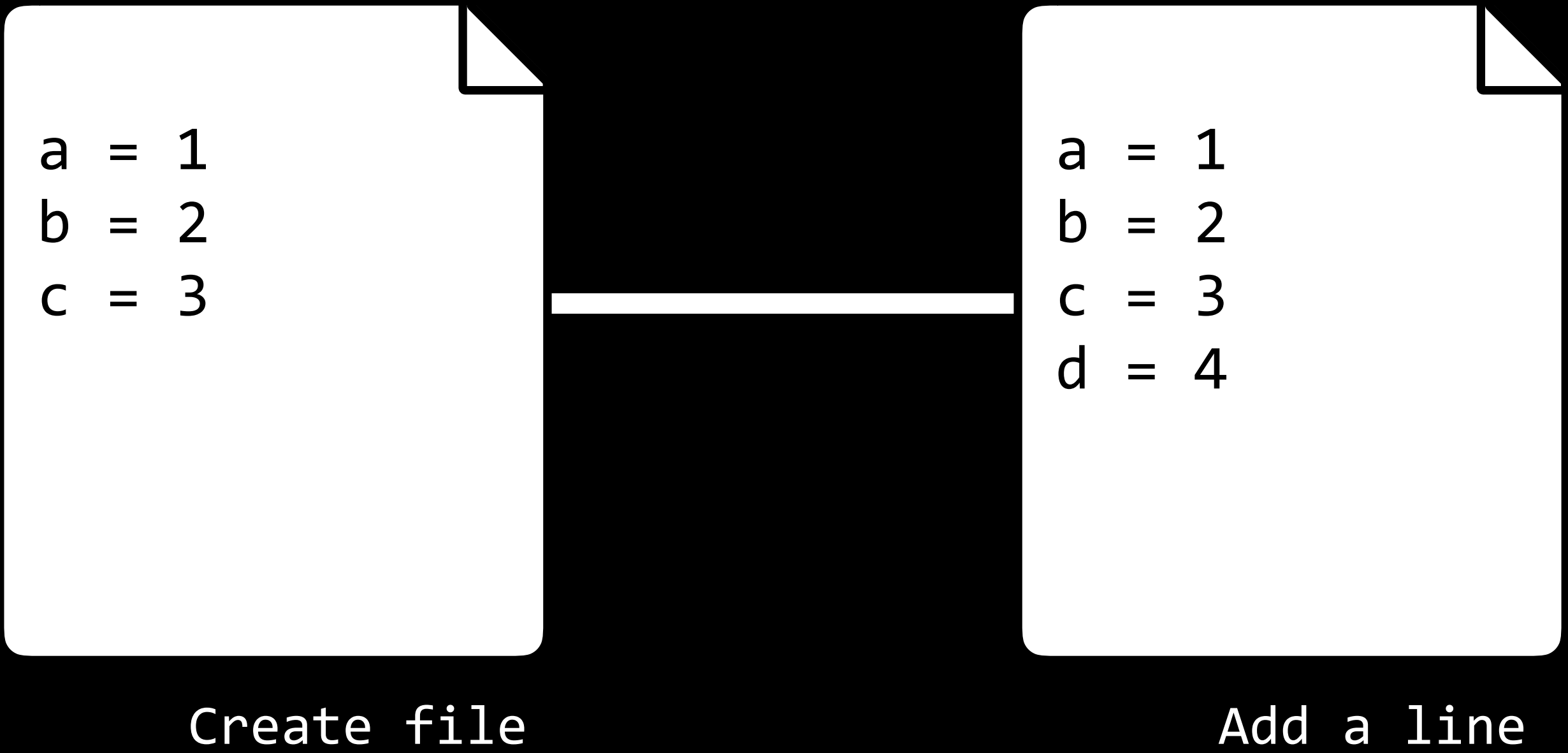
```
a = 1  
b = 2  
c = 3  
d = 4
```

Add a line

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

Remove a line

# Revert back to old versions of code.



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

Create file

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

Add a line

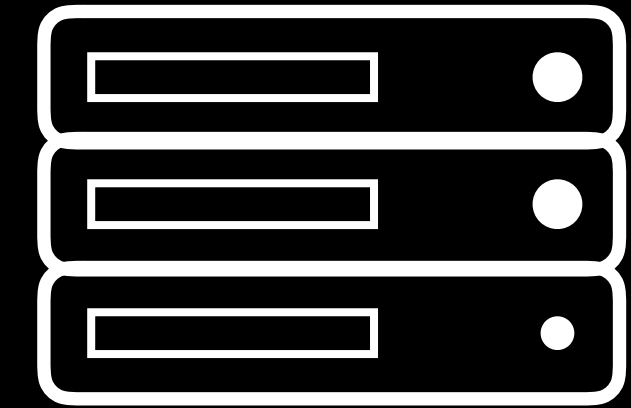


GitHub

```
git clone
```

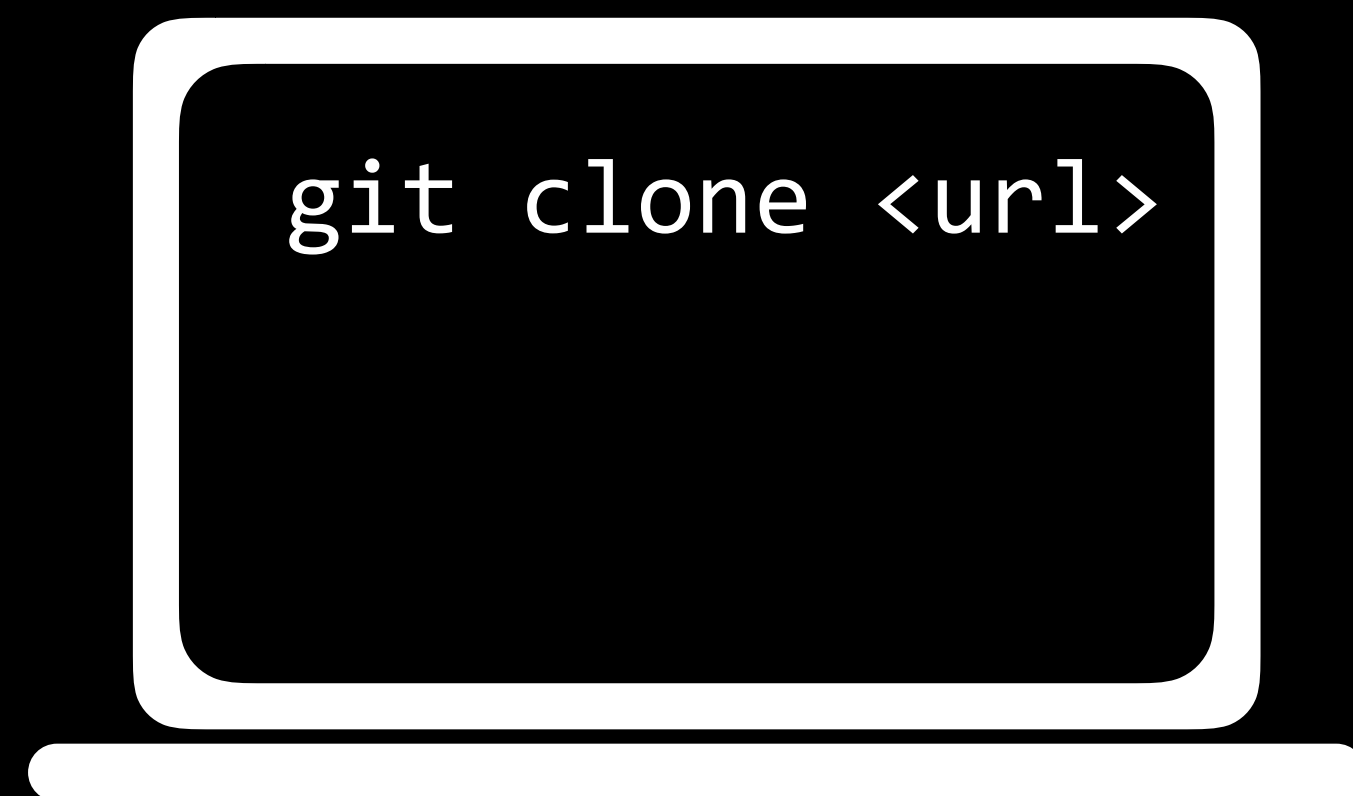
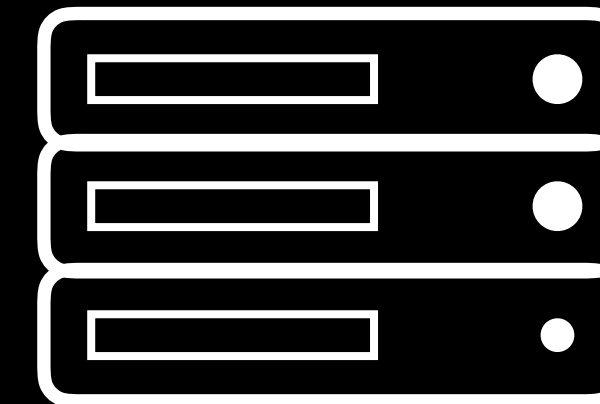
```
git clone <url>
```

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



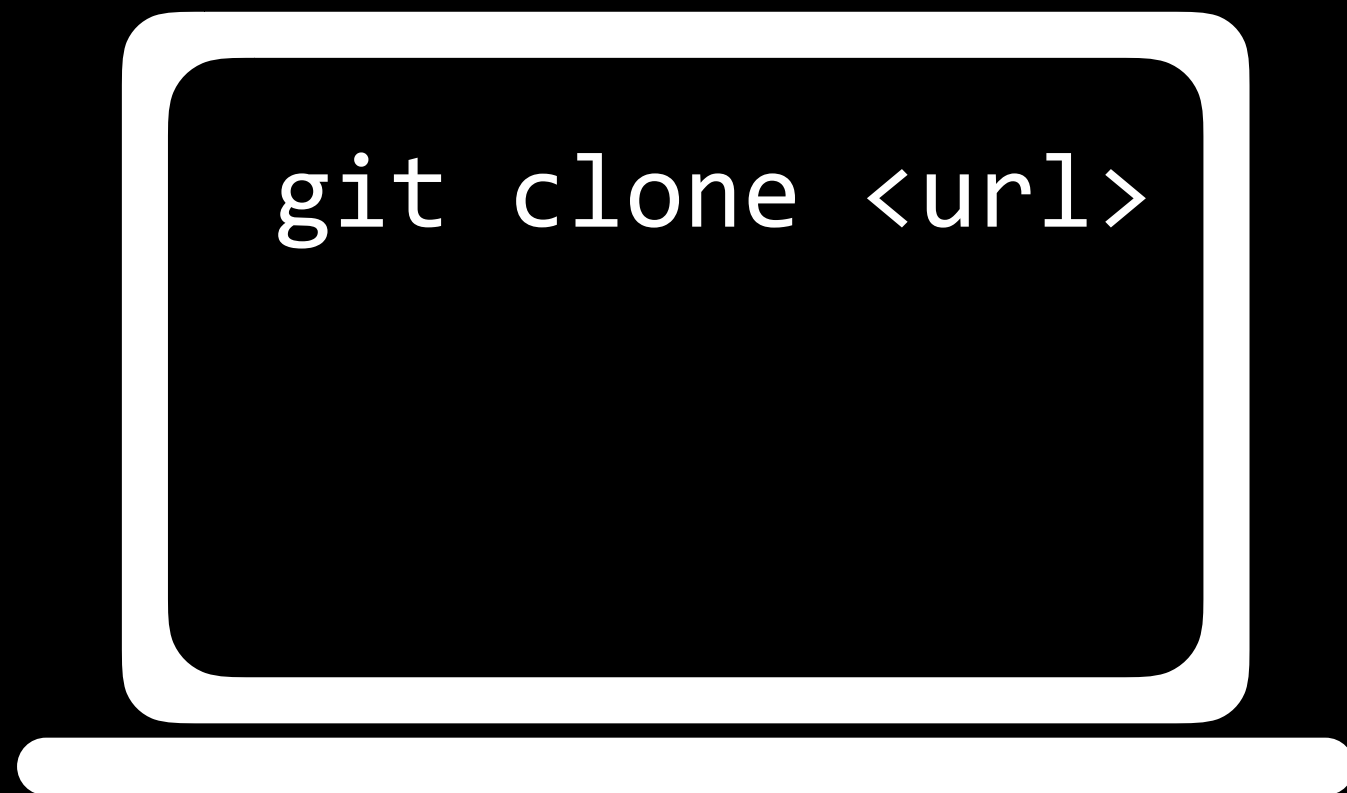
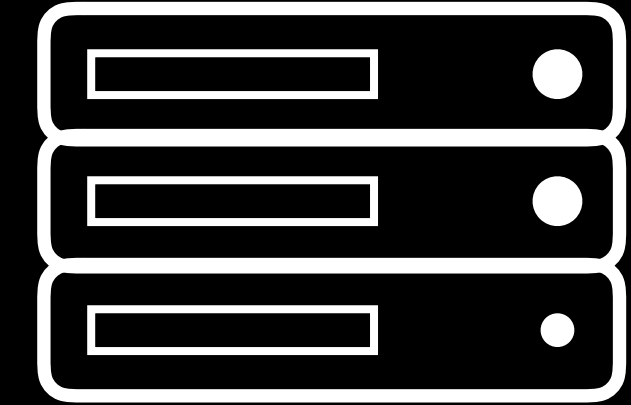
```
git clone <url>
```

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



```
git clone <url>
```

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

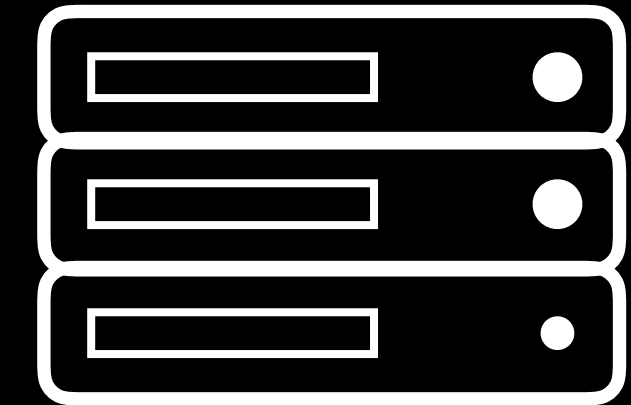


```
a = 1  
b = 2  
c = 3  
d = 4
```

```
git add
```

```
git add <filename>
```

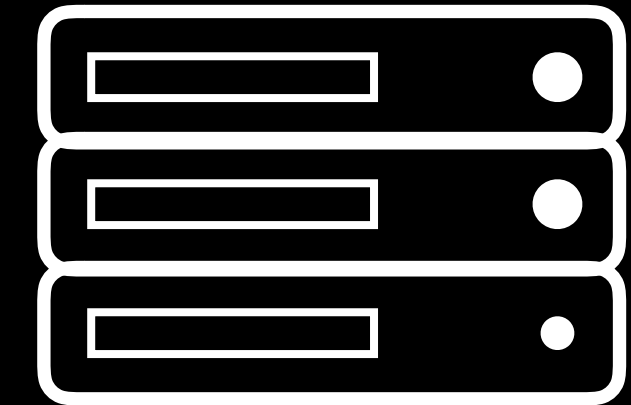
```
a = 1  
b = 2  
c = 3  
d = 4
```



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

```
git add <filename>
```

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

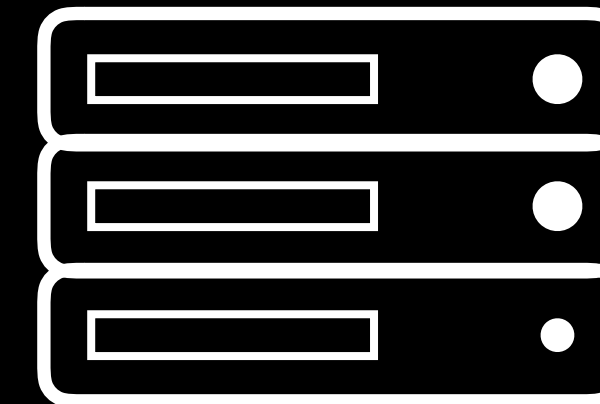


```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```



```
git add <filename>
```

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

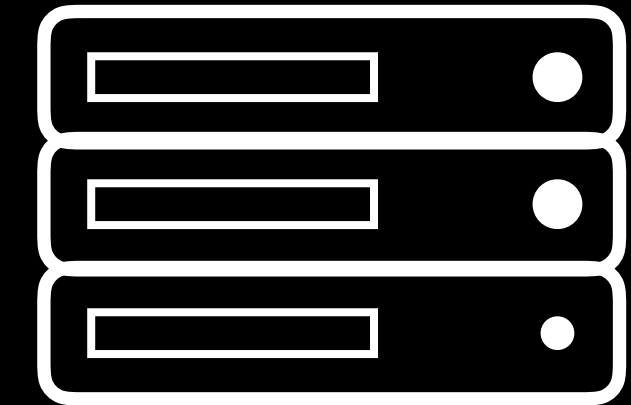


```
git add foo.py
```

```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

```
git add <filename>
```

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



```
git add foo.py
```

```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

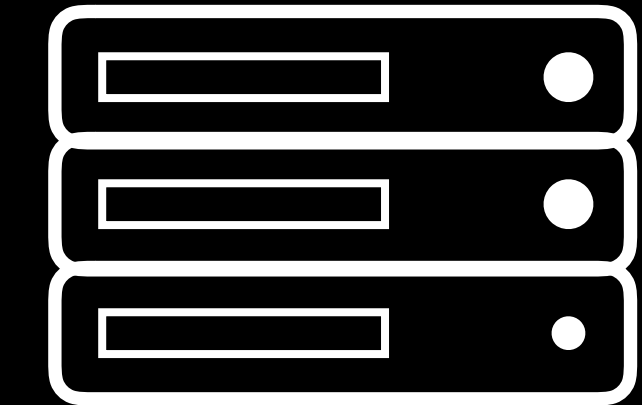
Changes to be committed:

```
modified: foo.py
```

```
git commit
```

```
git commit -m "message"
```

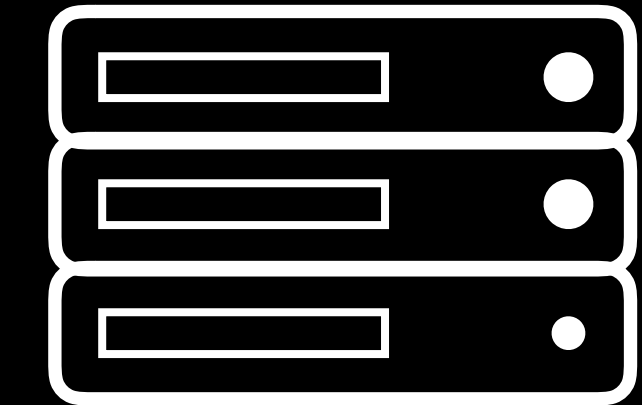
```
a = 1  
b = 2  
c = 3  
d = 4
```



```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

```
git commit -m "message"
```

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

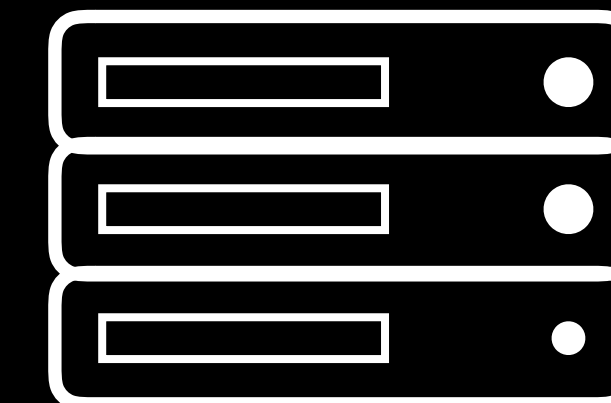


```
git commit -m  
"Add line"
```

```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

```
git commit -m "message"
```

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



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



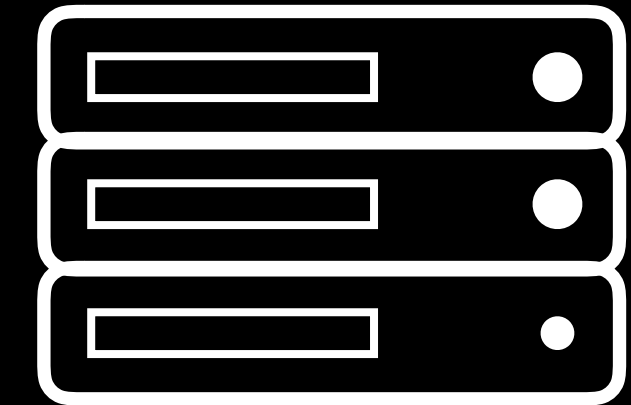
```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

Add line

```
git status
```

git status

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



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



```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

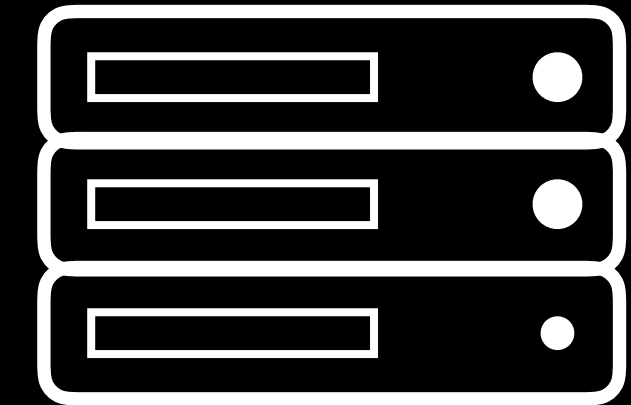
Add line





git status

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



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

```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

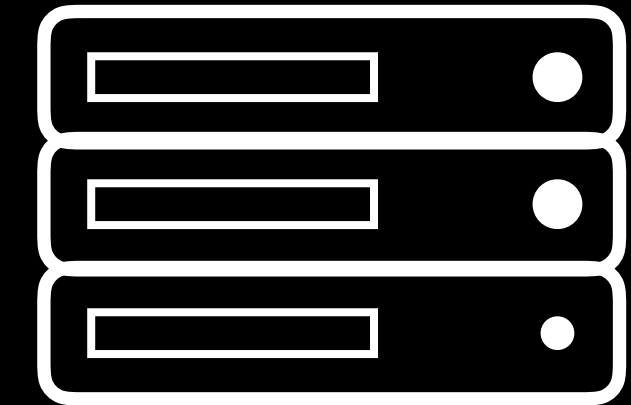
Add line



```
git status
```

git status

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



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

```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

Add line

git status

On branch master

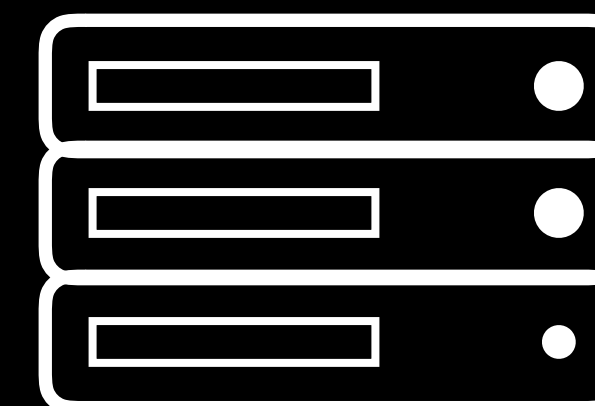
Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

```
git push
```

git push

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



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

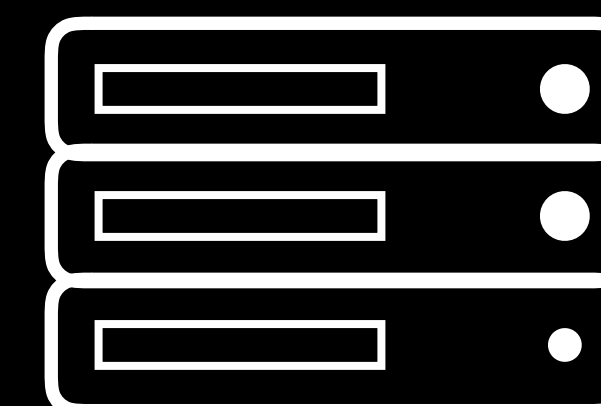


```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

Add line

git push

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



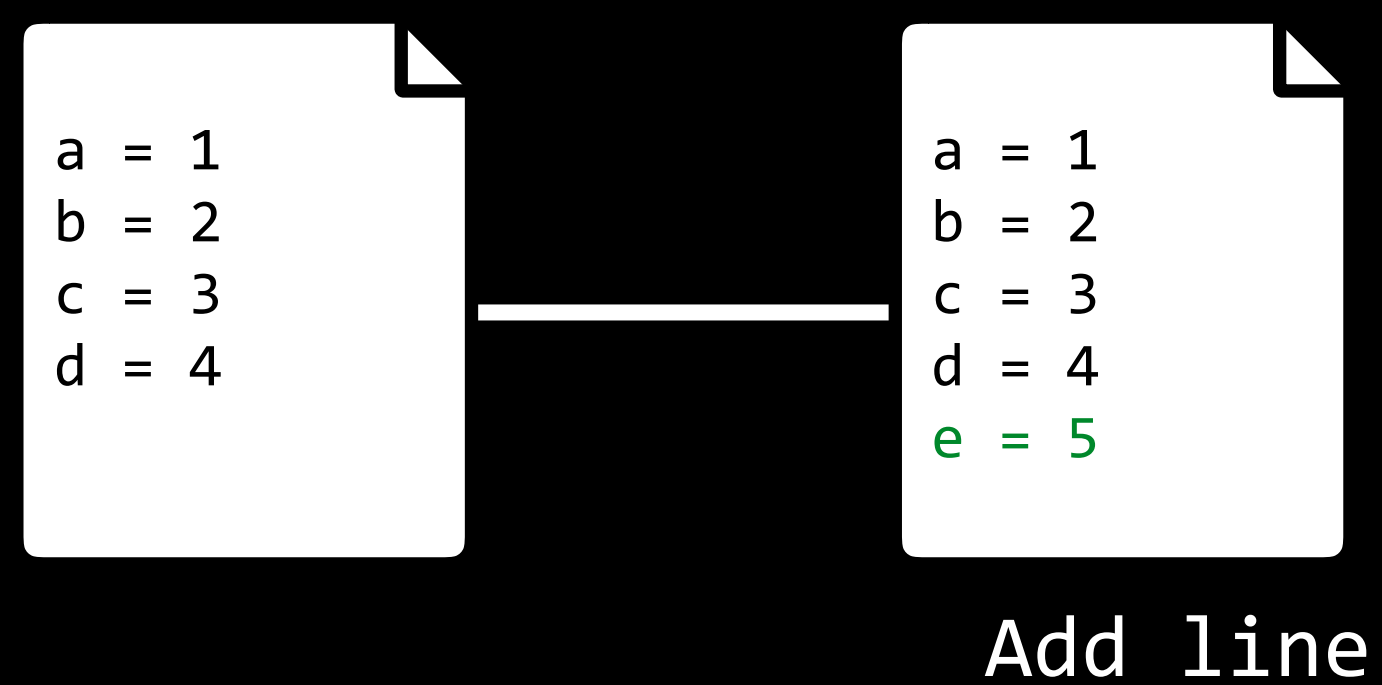
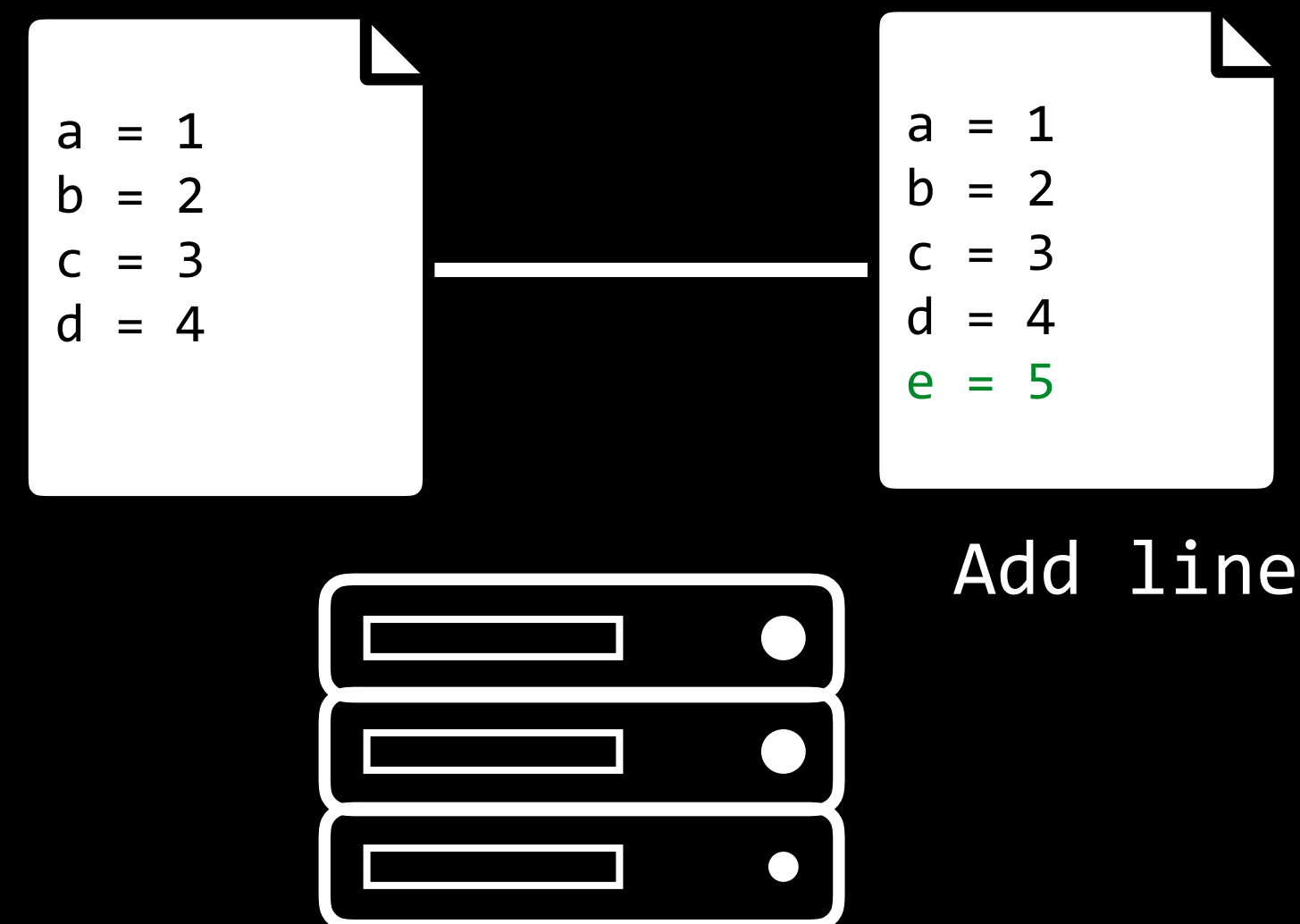
```
a = 1  
b = 2  
c = 3  
d = 4
```



```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

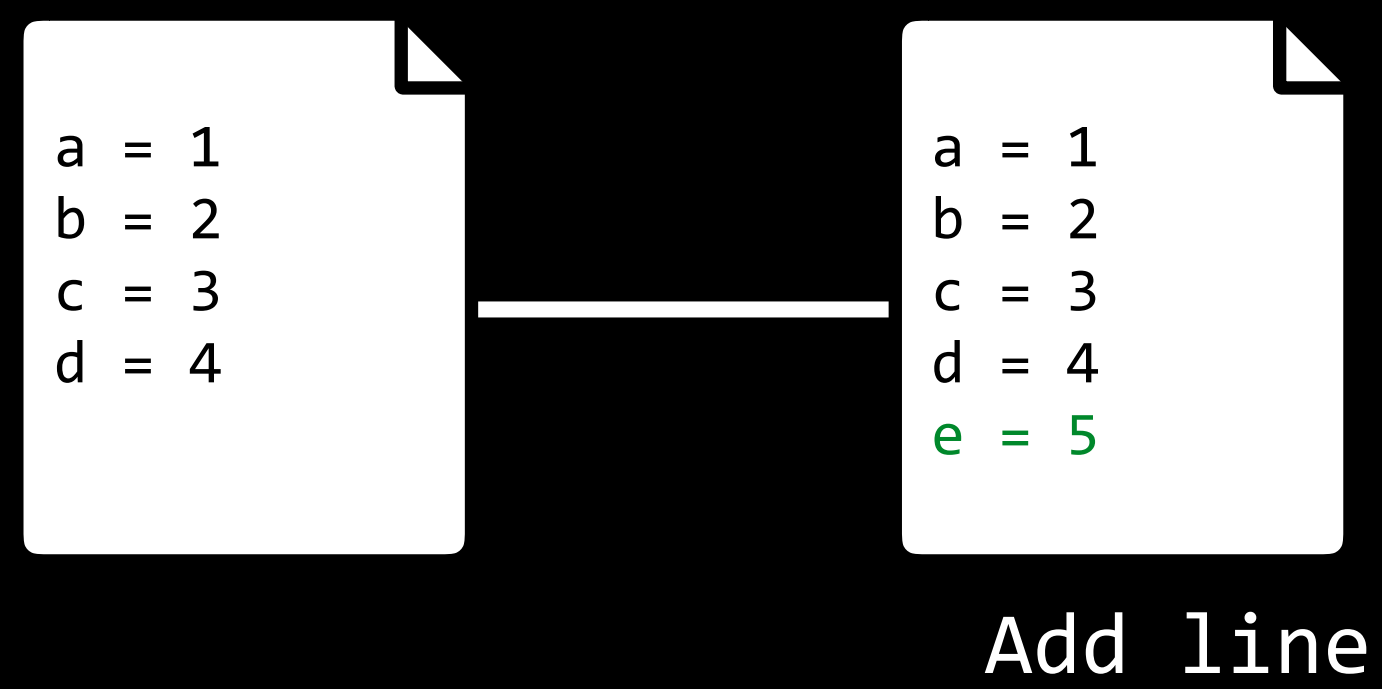
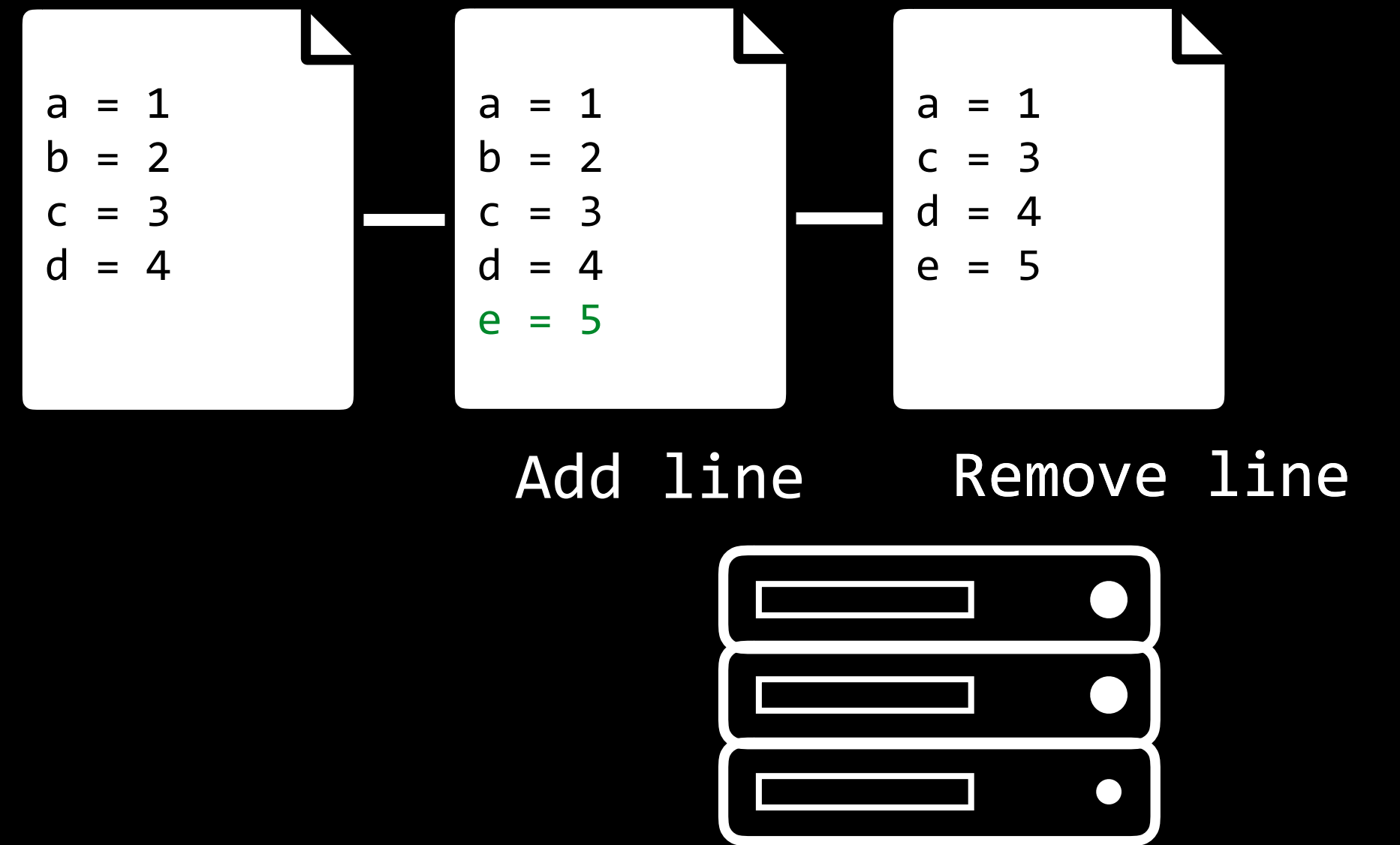
Add line

git push



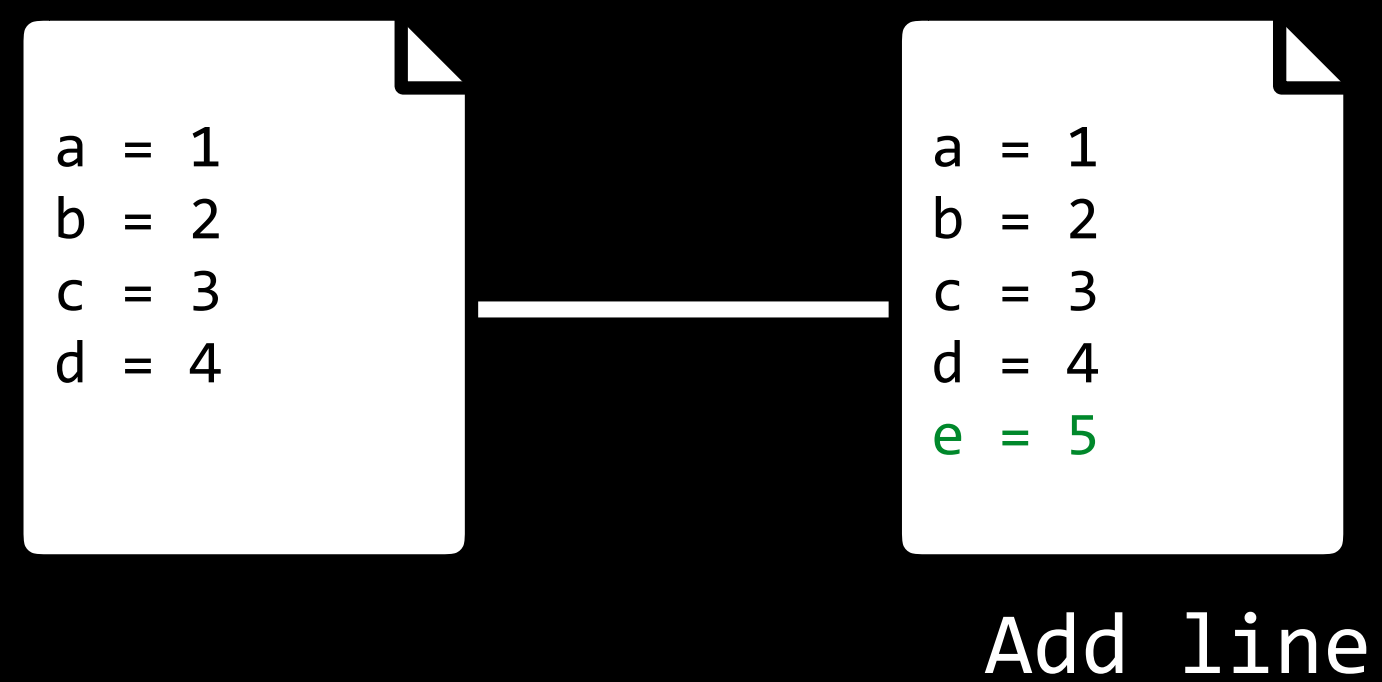
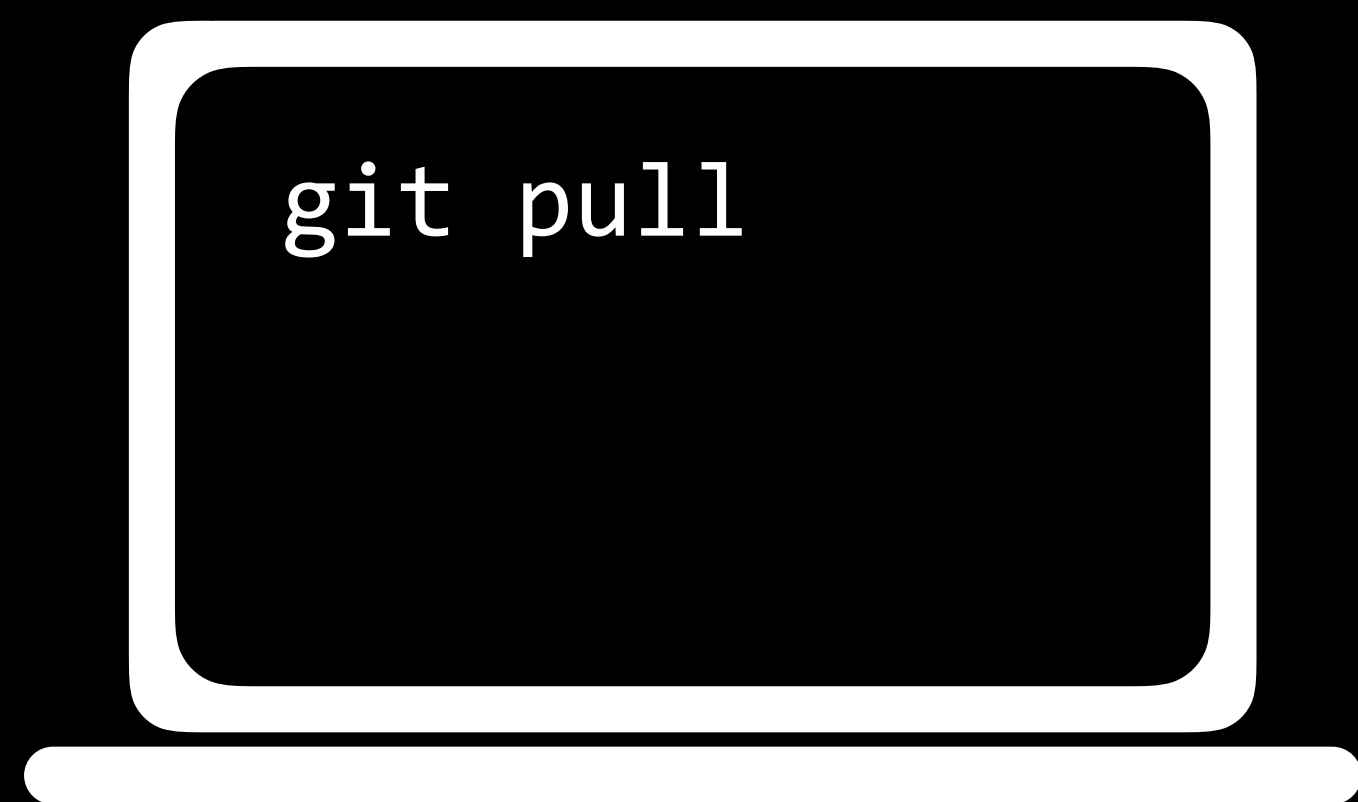
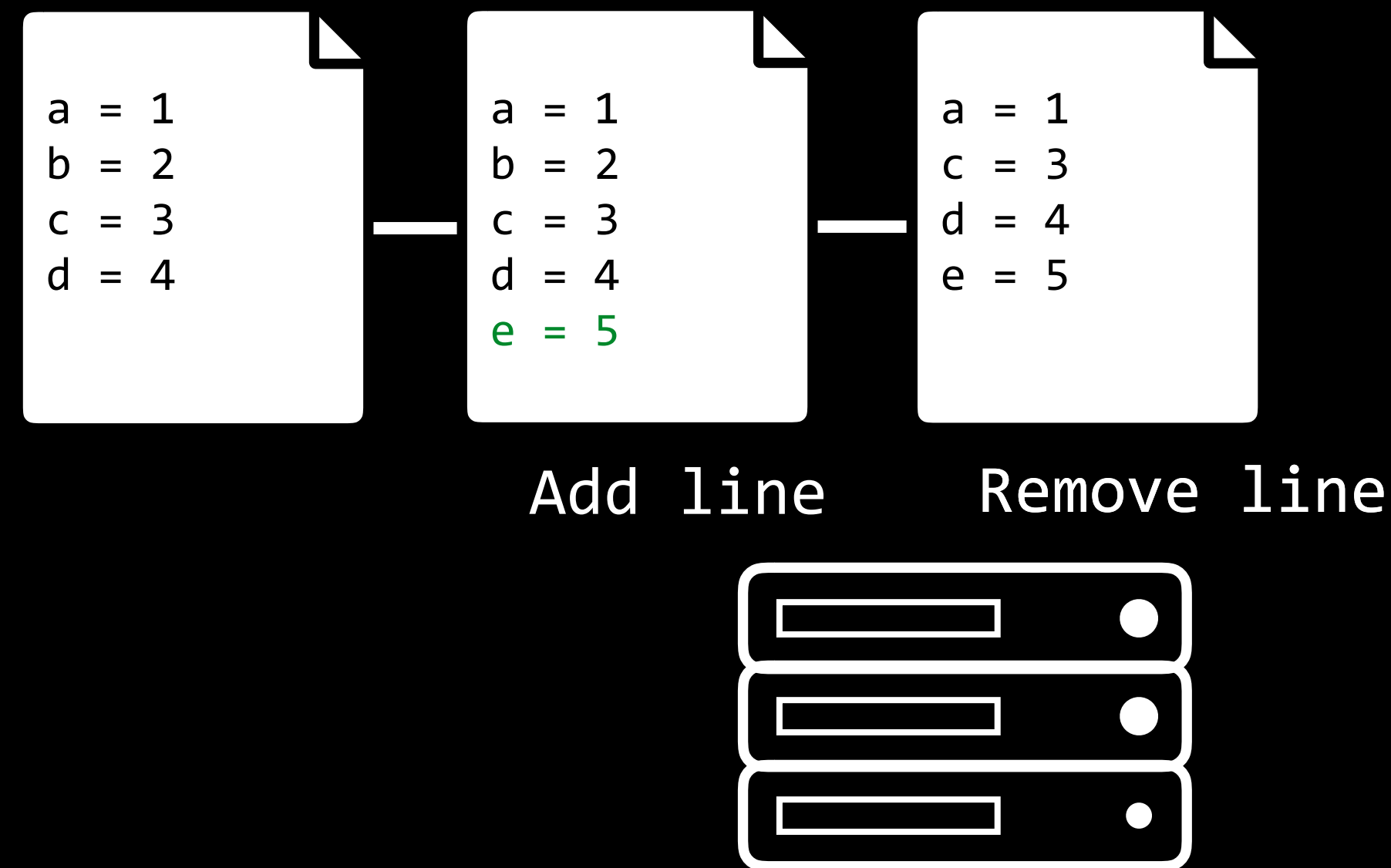
```
git pull
```

git pull

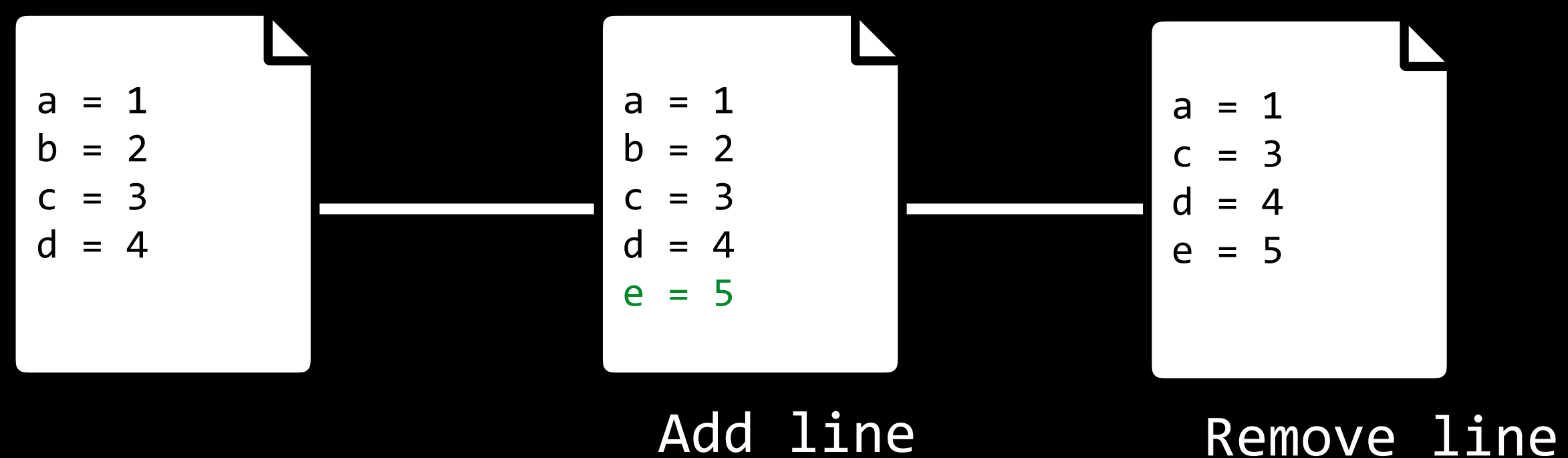
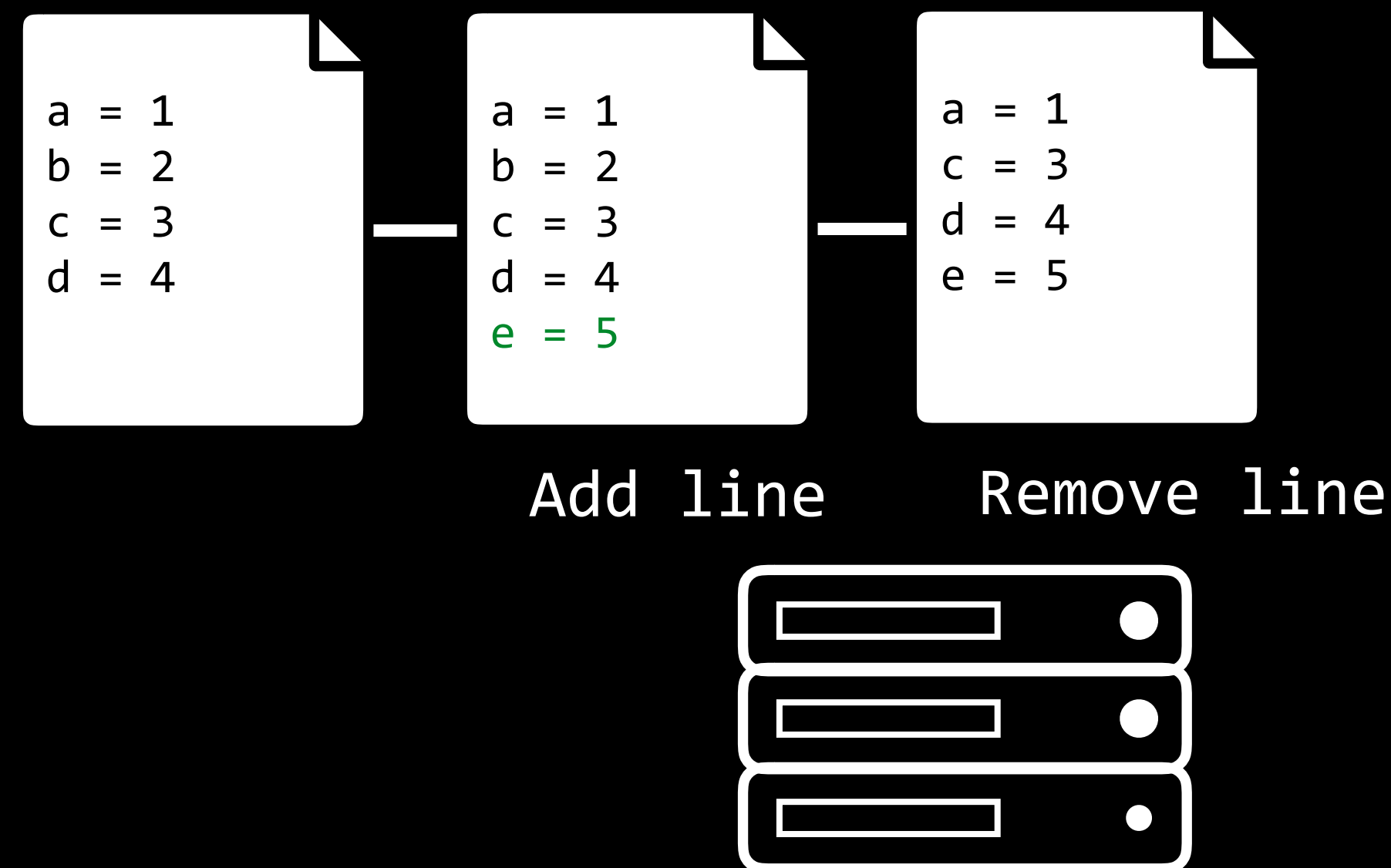




git pull

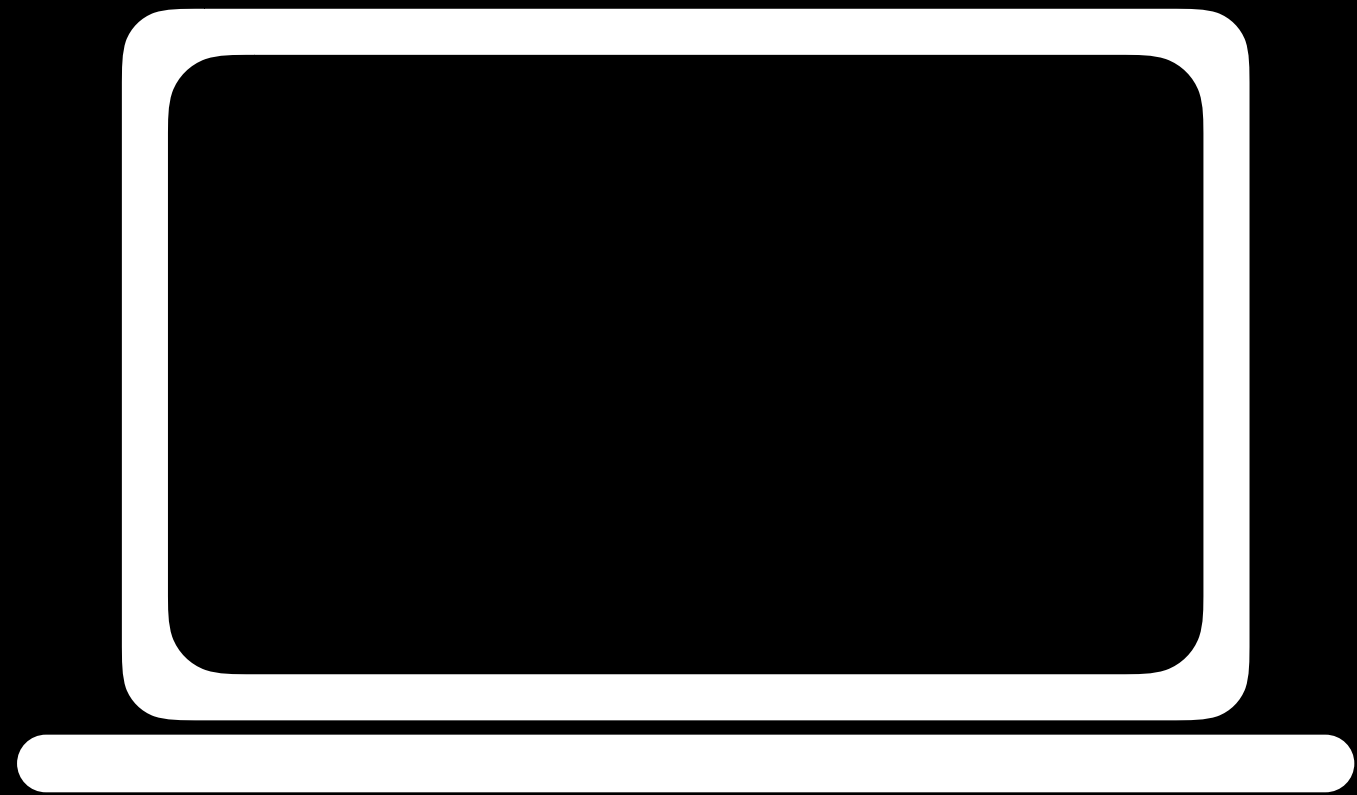


git pull



# Merge Conflicts

# Merge Conflicts



# Merge Conflicts

A white outline of a terminal window with a rounded rectangle for the screen and a horizontal bar for the base. The text "git pull" is centered on the screen.

```
git pull
```

# Merge Conflicts



```
git pull
```

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

# Merge Conflicts

```
git pull
```

```
a = 1  
<<<<< HEAD  
b = 2  
=====  
b = 0  
>>>>> 57656c636f6d6520746f20576562  
c = 3  
d = 4  
e = 5
```

# Merge Conflicts




```
git pull
```

your  
changes

remote  
changes

```
a = 1
<<<<< HEAD
{ b = 2
  =====
  { b = 0
    >>>>> 57656c636f6d6520746f20576562
c = 3
d = 4
e = 5
```

conflicting  
commit





# Merge Conflicts

```
git pull
```

```
a = 1  
<<<<< HEAD  
b = 2  
=====  
b = 0  
>>>>> 57656c636f6d6520746f20576562  
c = 3  
d = 4  
e = 5
```

# Merge Conflicts

```
git pull
```

```
a = 1
```

```
b = 2
```

```
c = 3
```

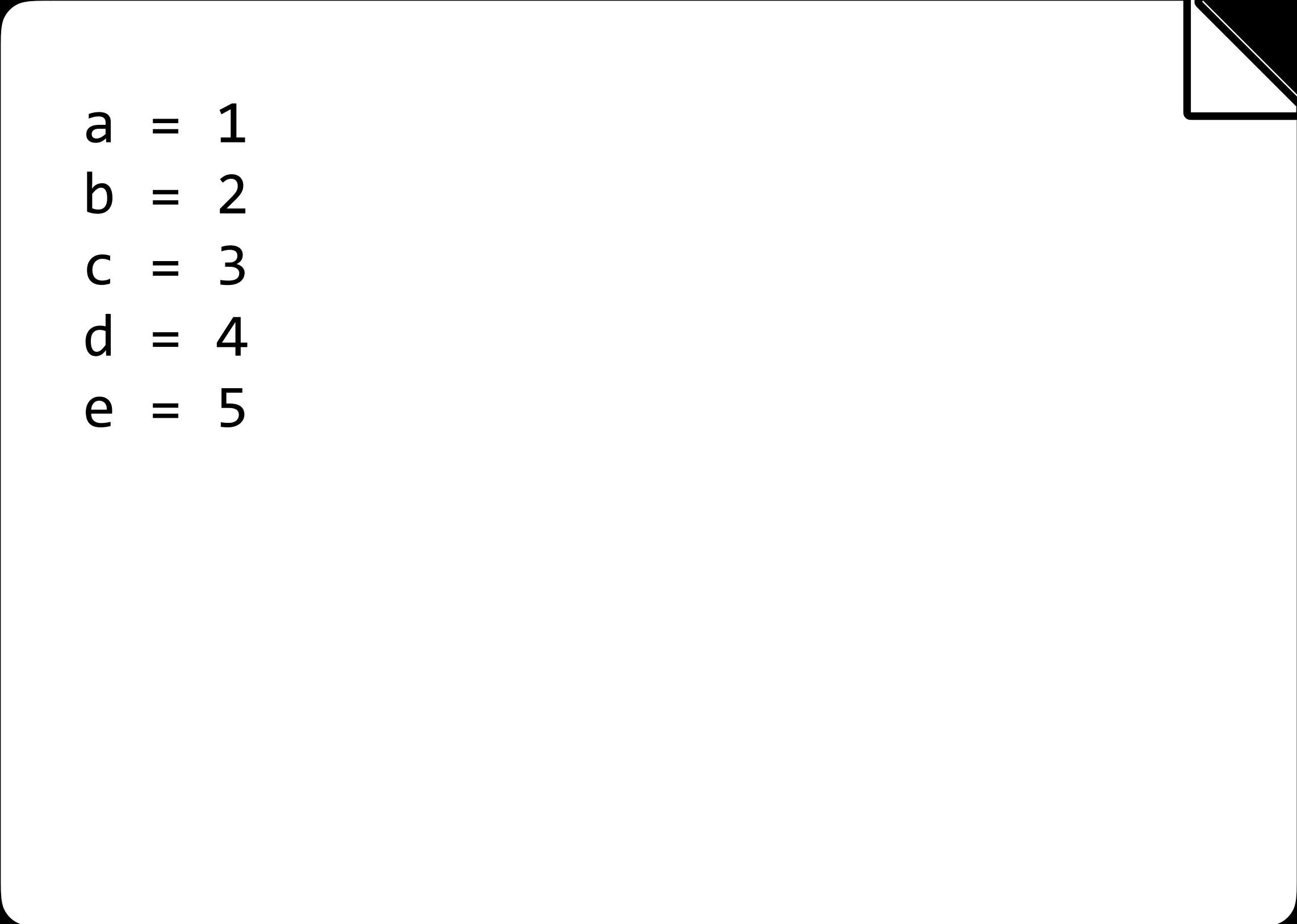
```
d = 4
```

```
e = 5
```

# Merge Conflicts



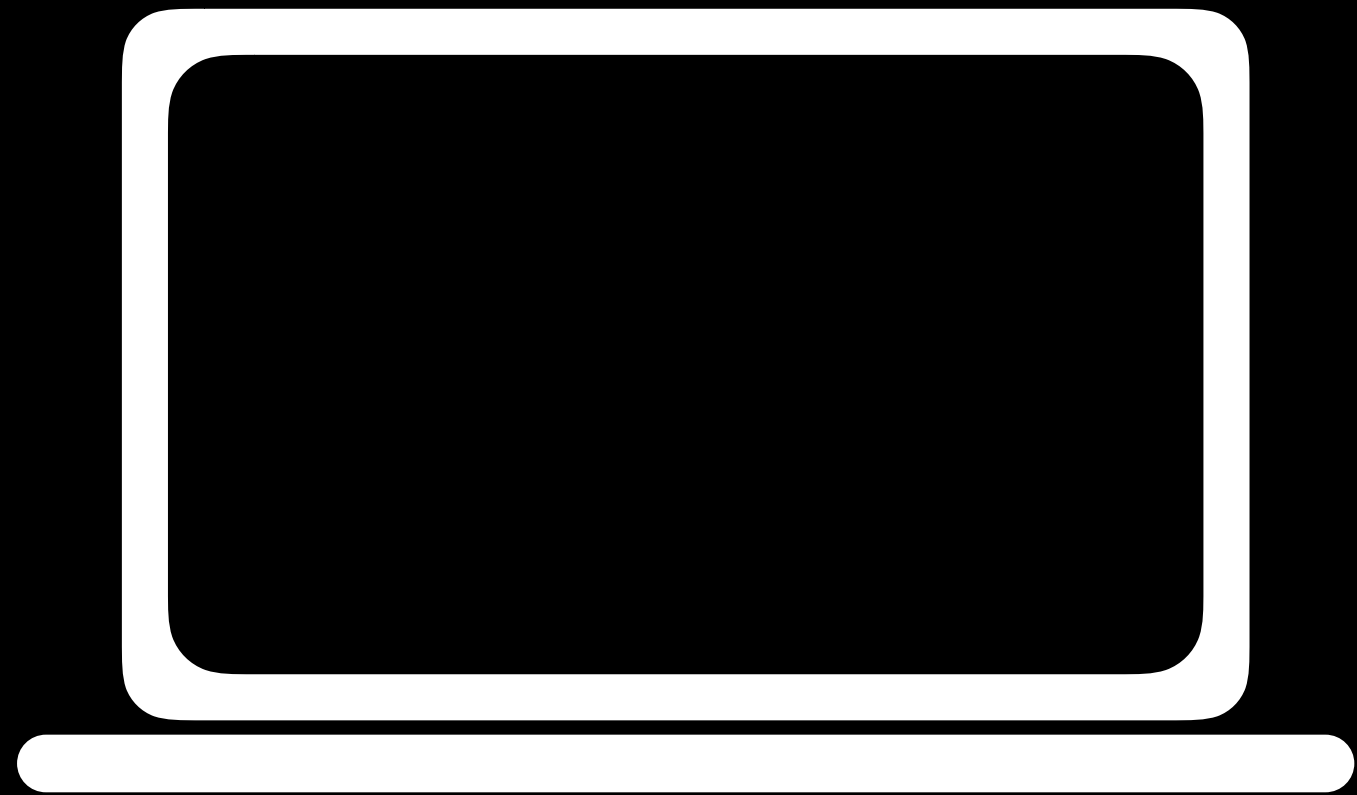
```
git pull
```



```
a = 1  
b = 2  
c = 3  
d = 4  
e = 5
```

```
git log
```

git log



```
git log
```



git log

```
commit 436f6d6d6974204d73672048657265
Author: Brian Yu <brian@cs.harvard.edu>
Date:   Mon Jan 22 14:06:28 2018 -0400
```

Remove a line

```
commit 57656c636f6d6520746f20576562
Author: Brian Yu <brian@cs.harvard.edu>
Date:   Mon Jan 22 14:05:28 2018 -0400
```

Add a line



git log

```
git reset
```



# git reset

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



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

```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
int e = 5;
```

Add line  
57656c6

```
int a = 1;
int c = 3;
int d = 4;
int e = 5;
```

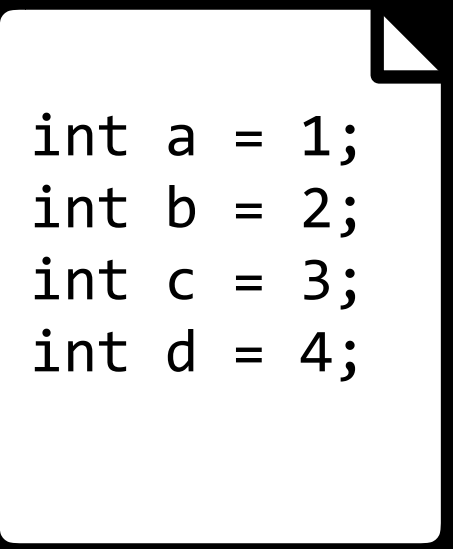
Remove line  
436f6d6

# git reset

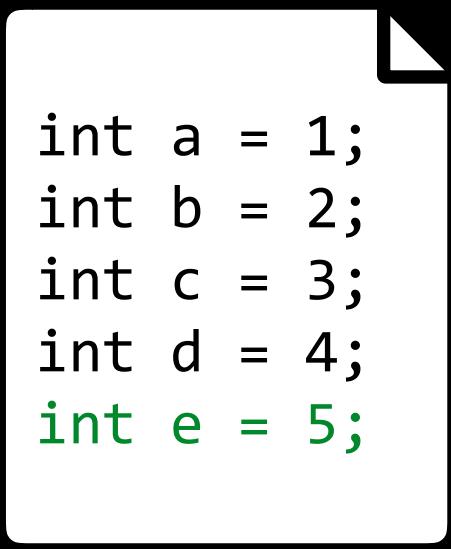
- `git reset --hard <commit>`
- `git reset --hard origin/master`



```
git reset --hard  
57656c6
```

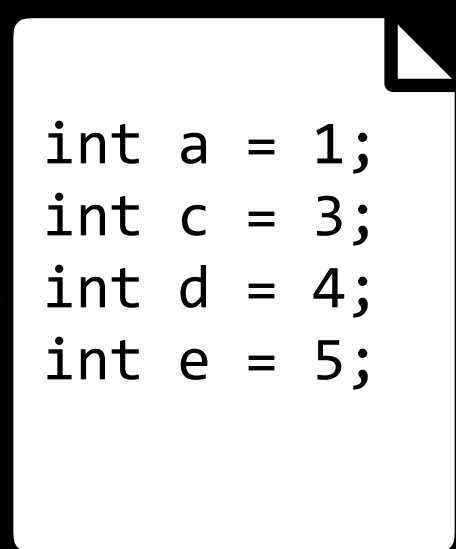


```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```

Add line  
57656c6



```
int a = 1;  
int c = 3;  
int d = 4;  
int e = 5;
```

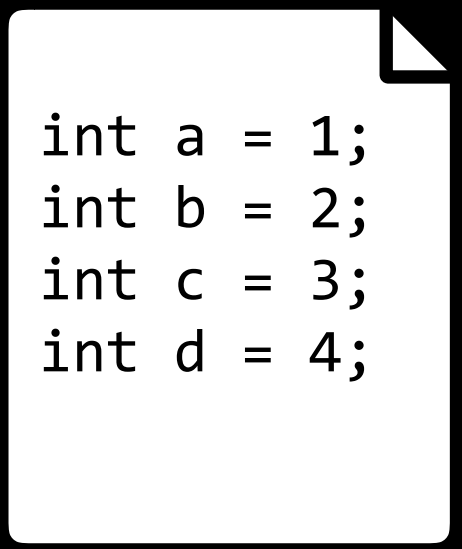
Remove line  
436f6d6

# git reset

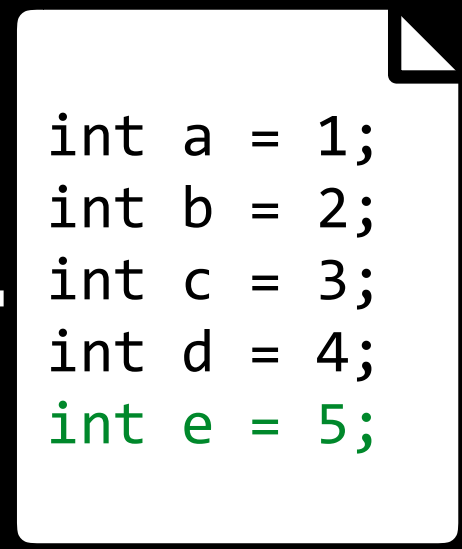
- `git reset --hard <commit>`
- `git reset --hard origin/master`



```
git reset --hard  
57656c6
```



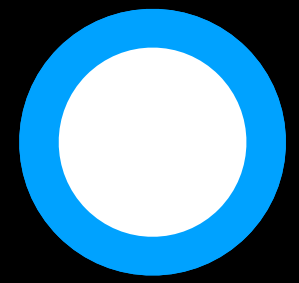
```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```

Add line  
57656c6

# Making Changes

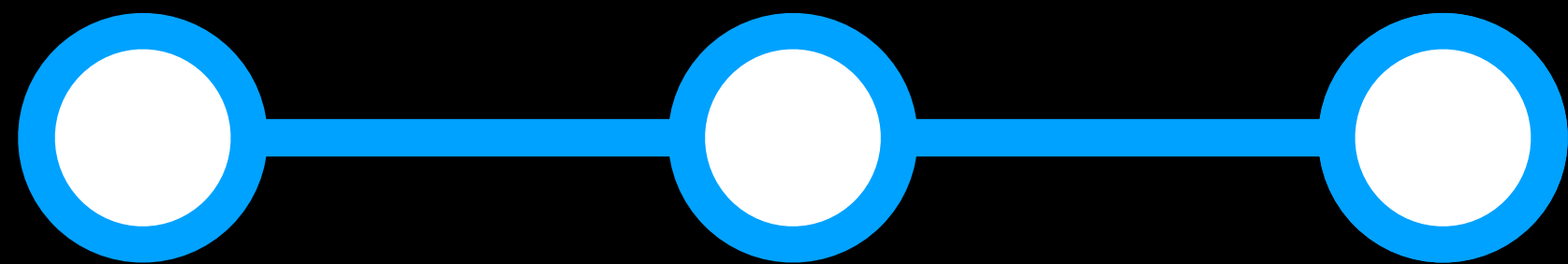


first  
commit



first  
commit

changes



first  
commit

changes

more  
changes



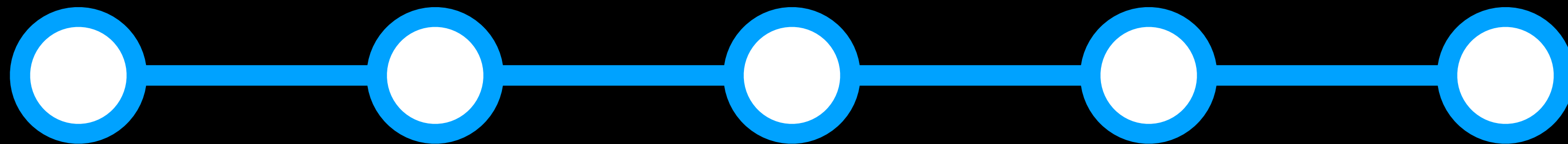
first  
commit

changes

more  
changes

start new  
feature





first  
commit

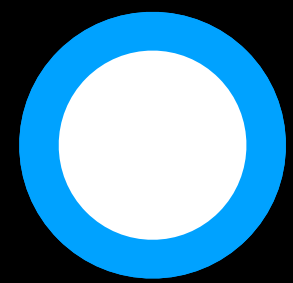
changes

more  
changes

start new  
feature

keep working  
on new feature

# Branching

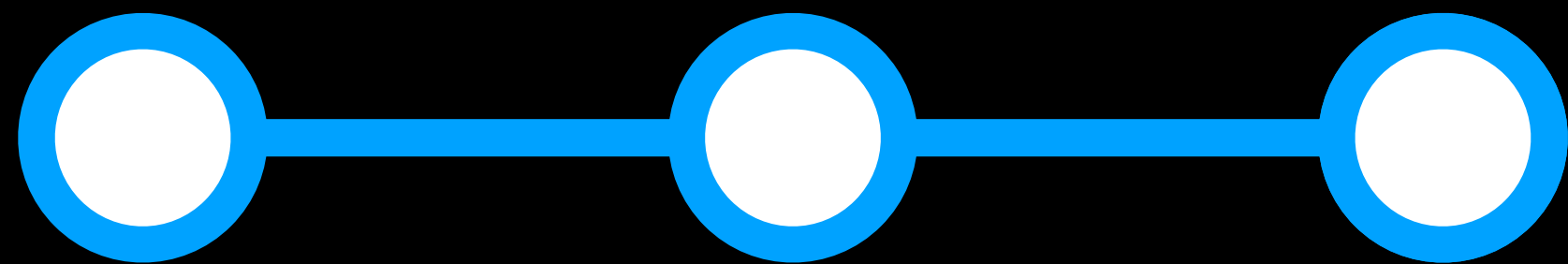


first  
commit



first  
commit

changes

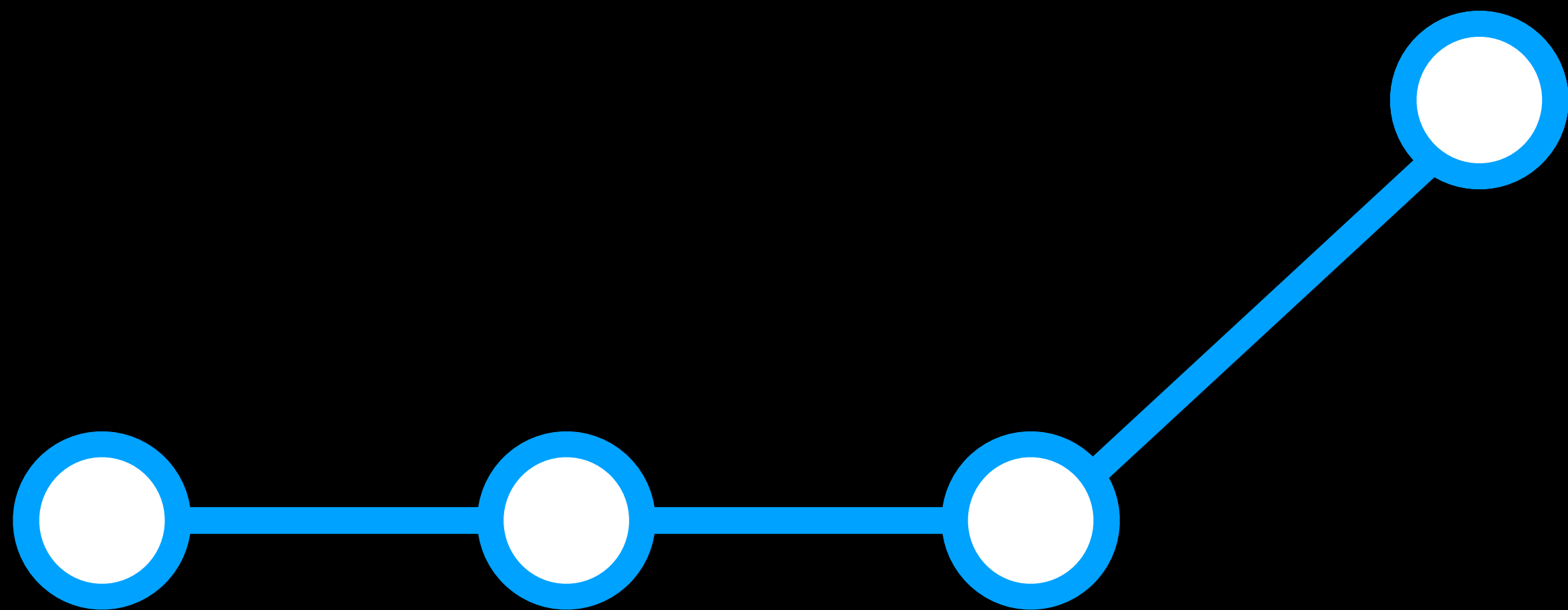


first  
commit

changes

more  
changes

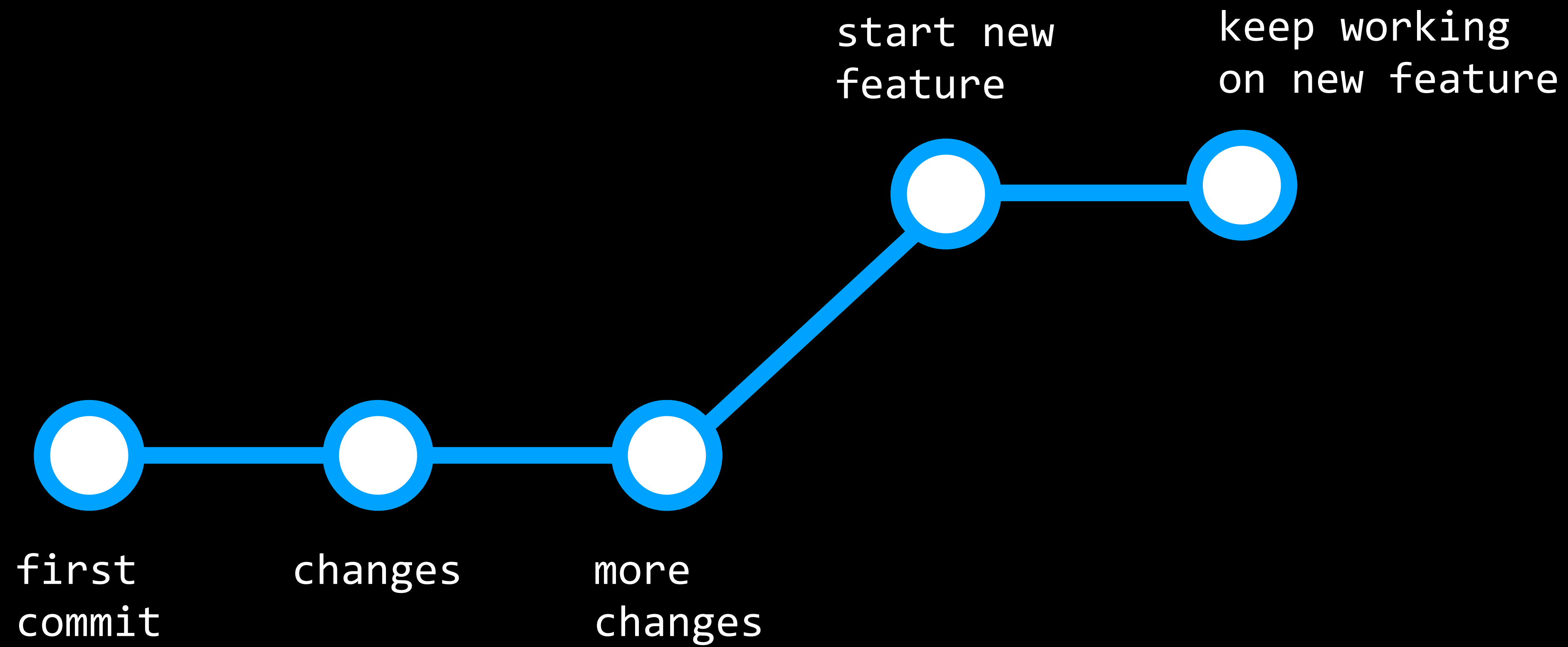
start new  
feature

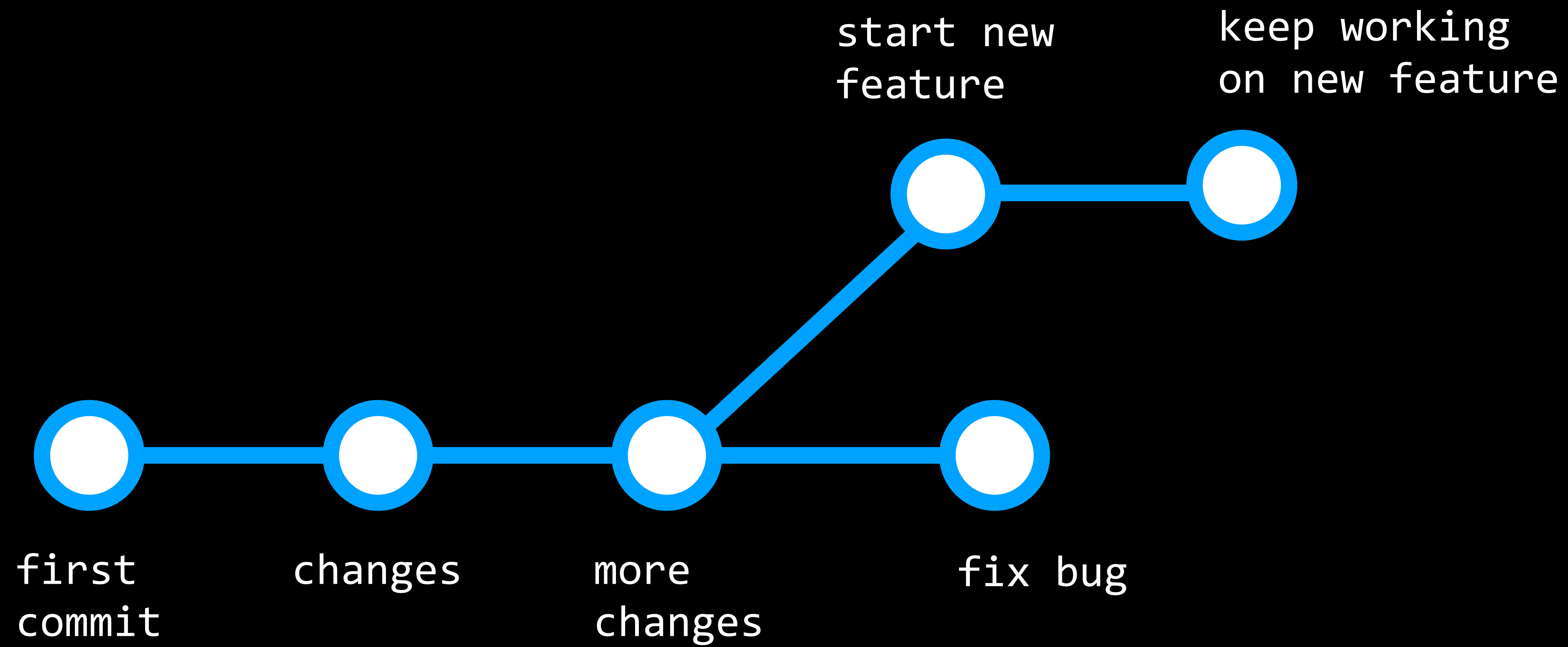


first  
commit

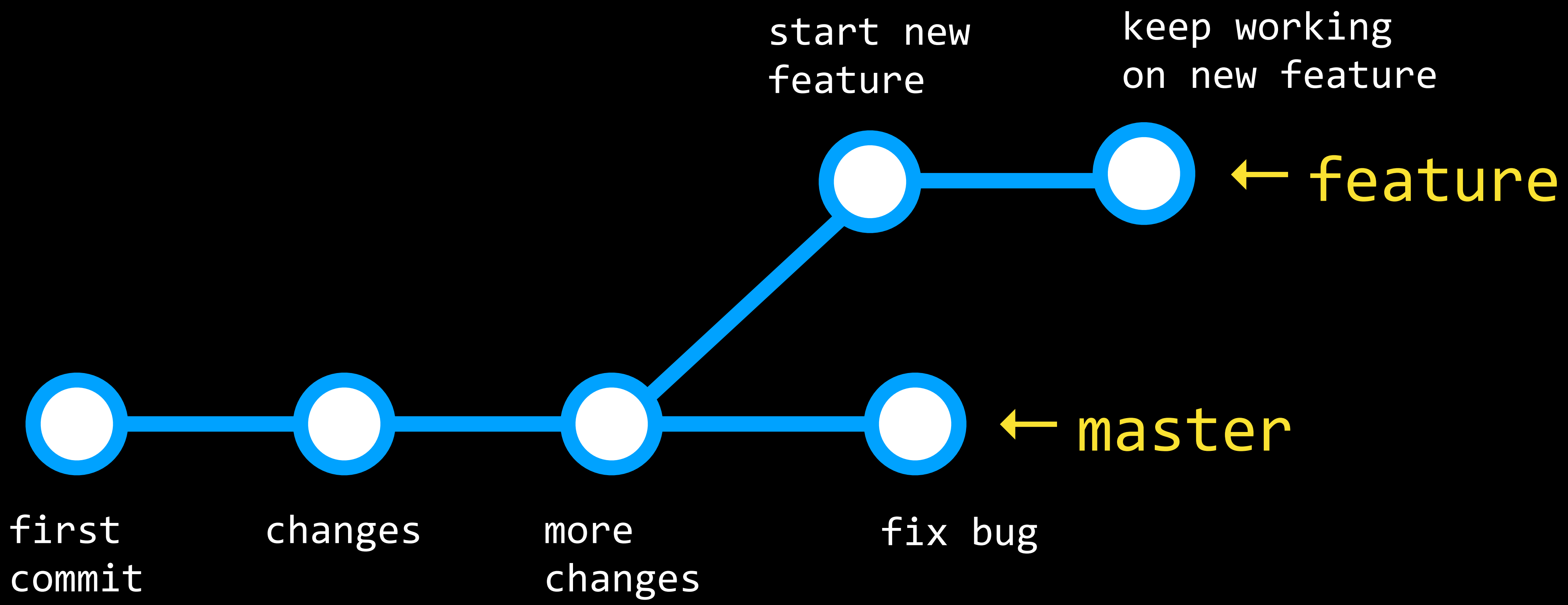
changes

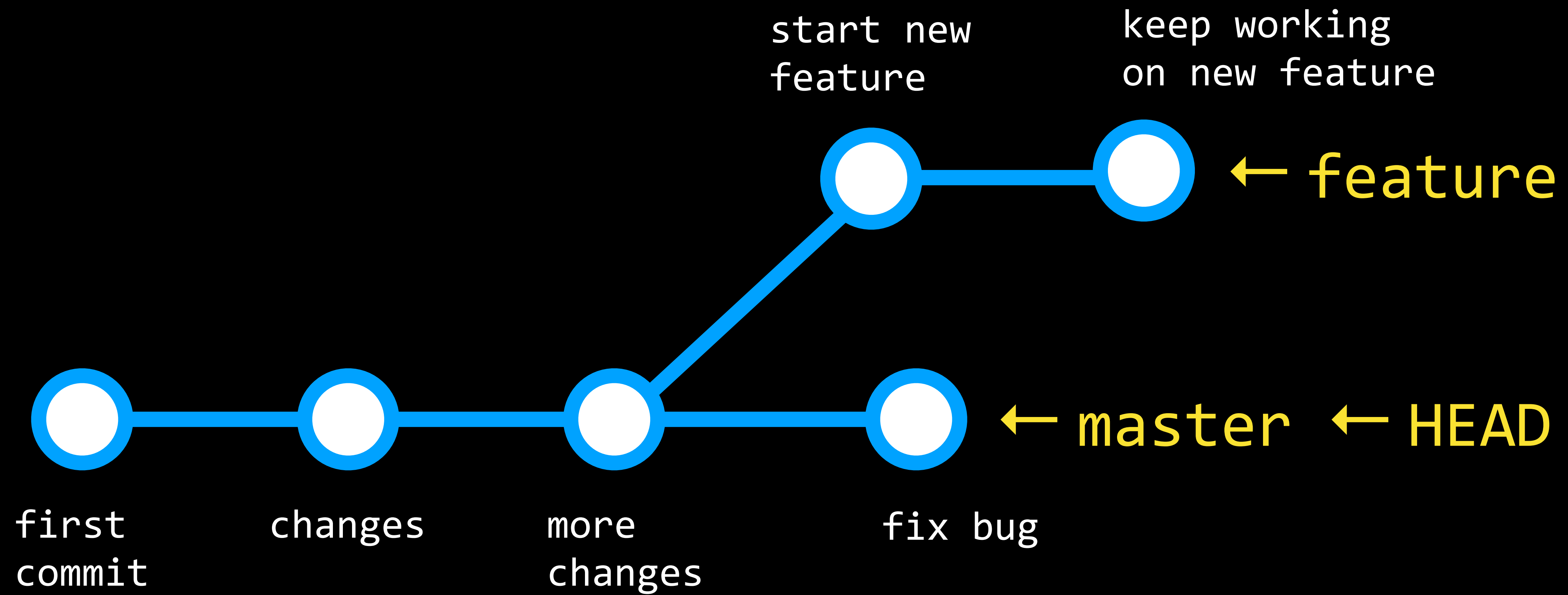
more  
changes

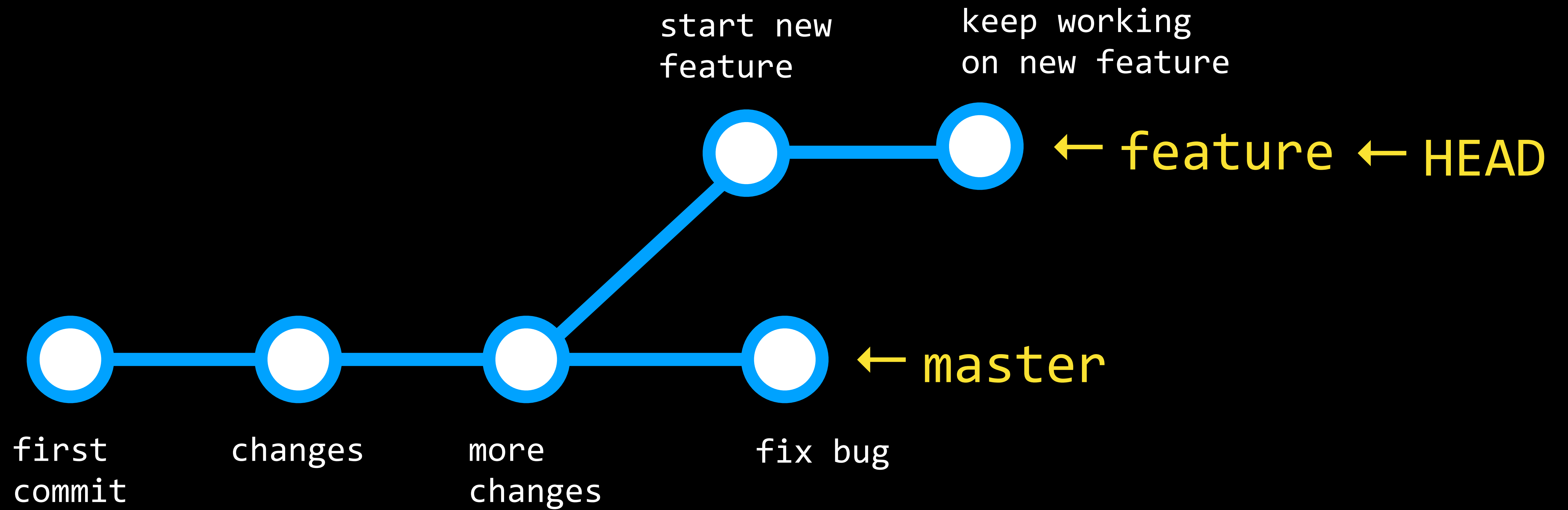


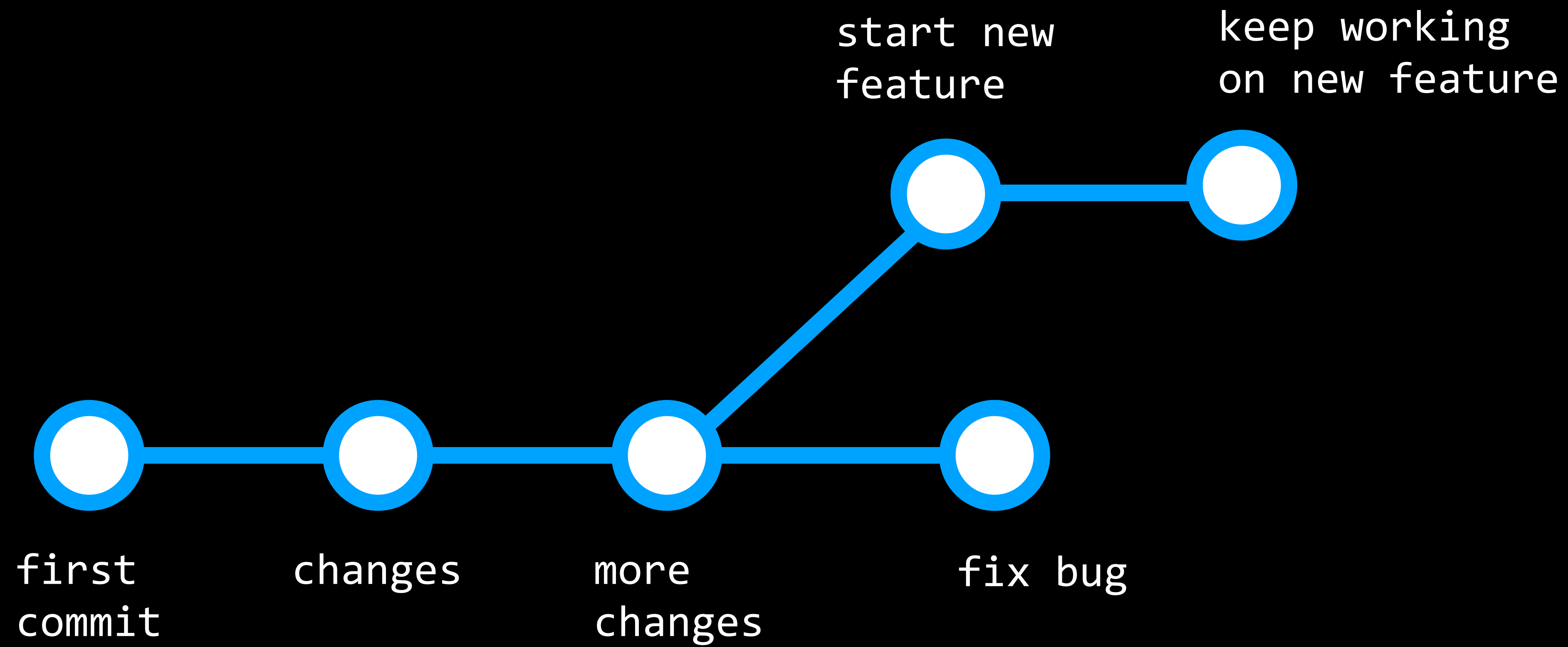


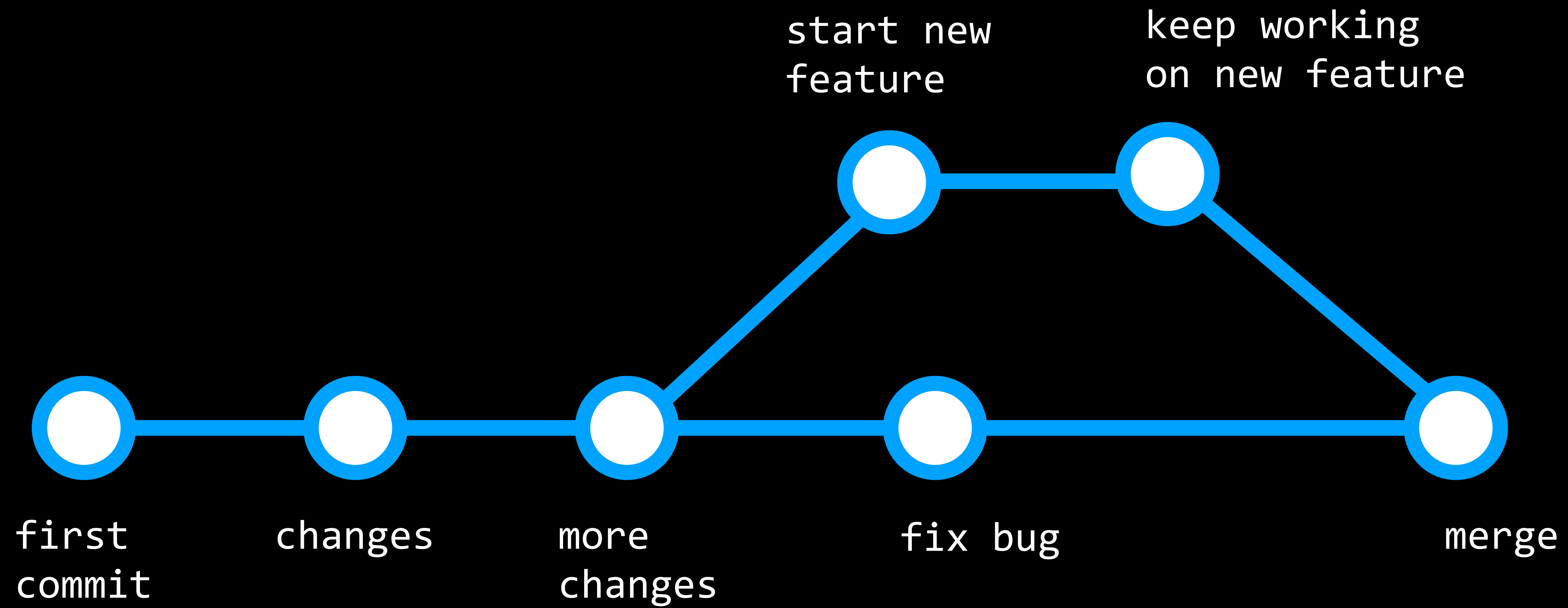










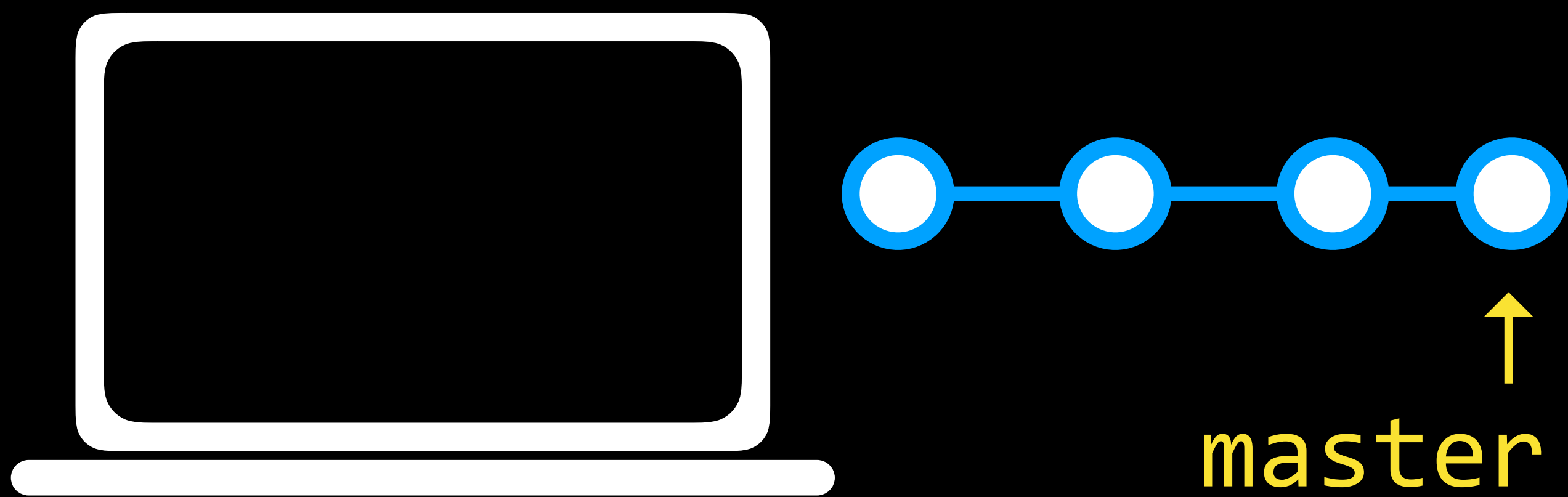
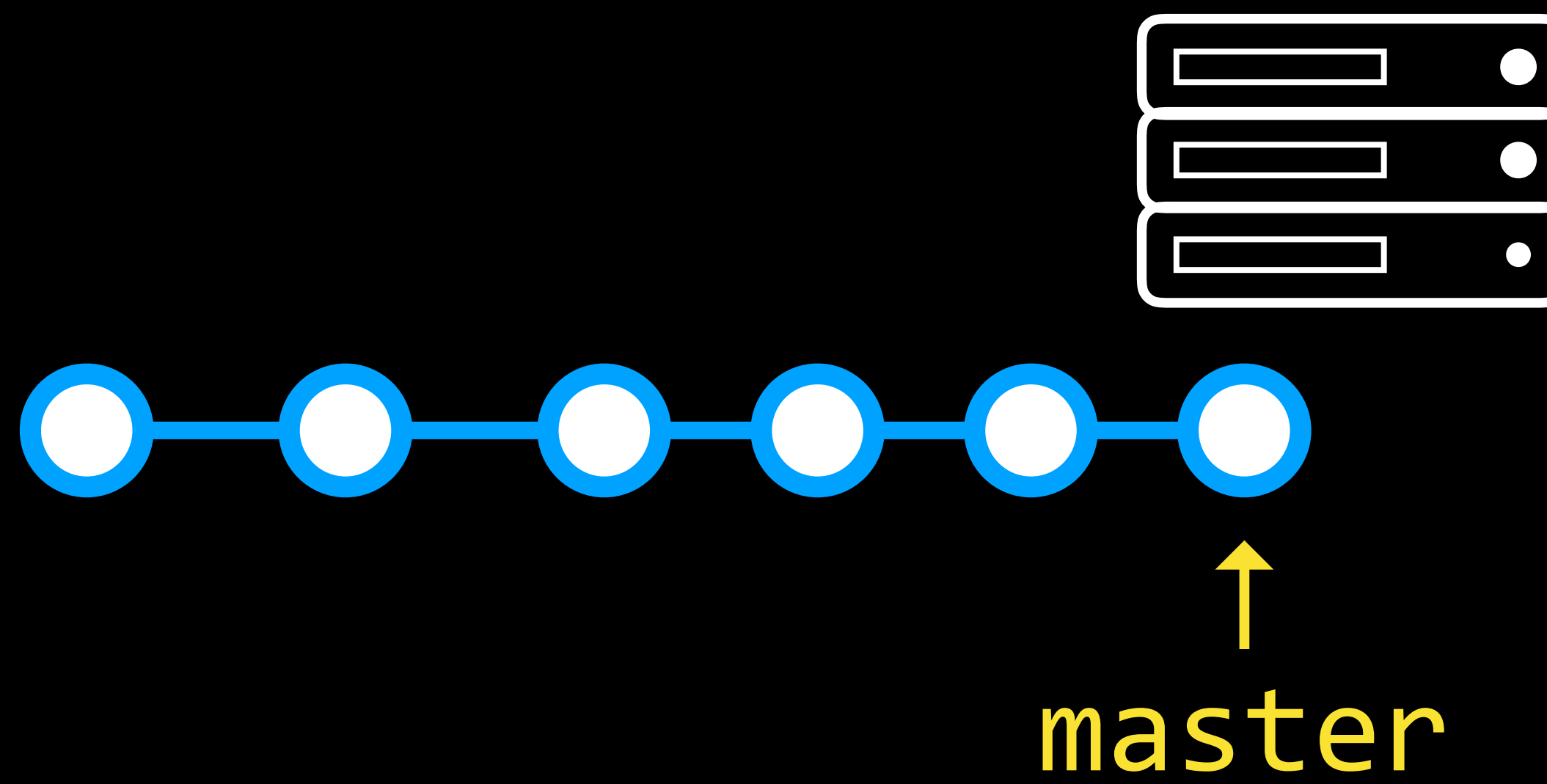


# Branching

- `git branch`
- `git checkout`
- `git merge`

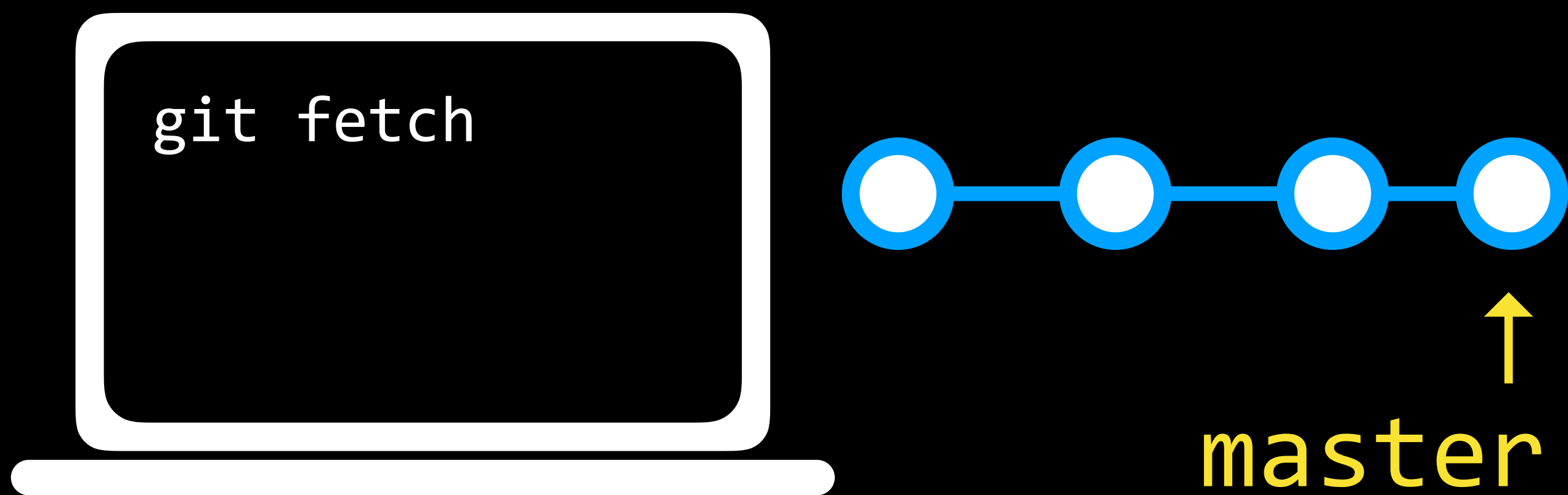
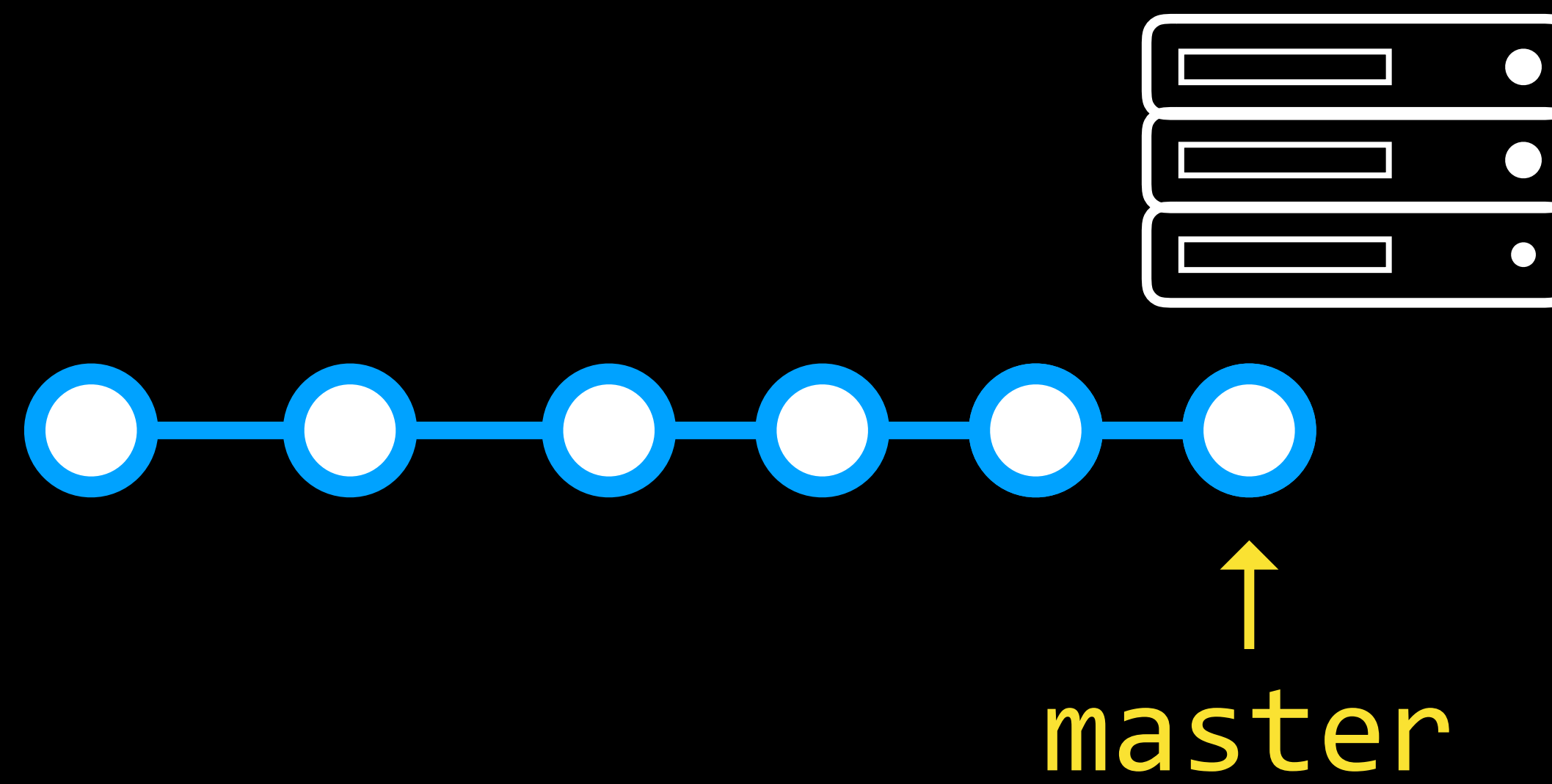
Remotes

# Remotes

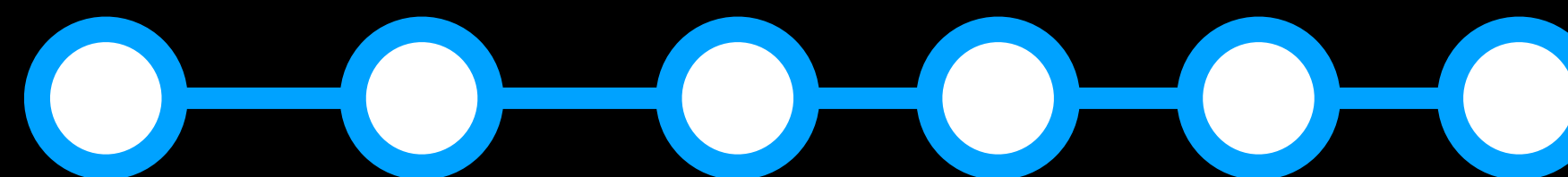
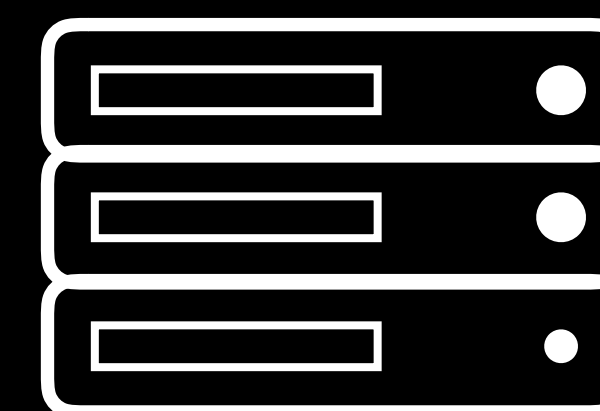




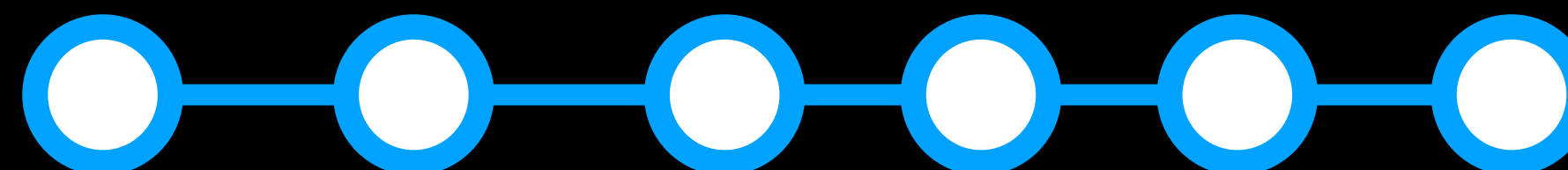
# Remotes



# Remotes



↑  
master

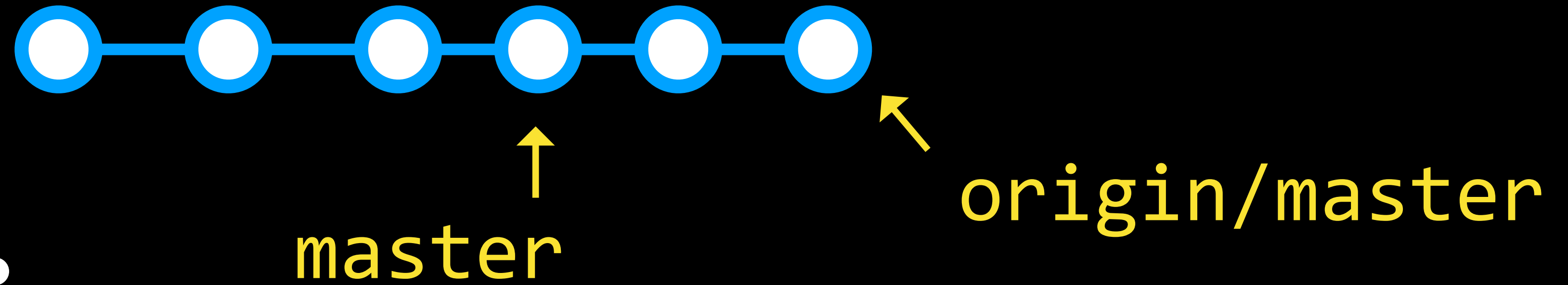
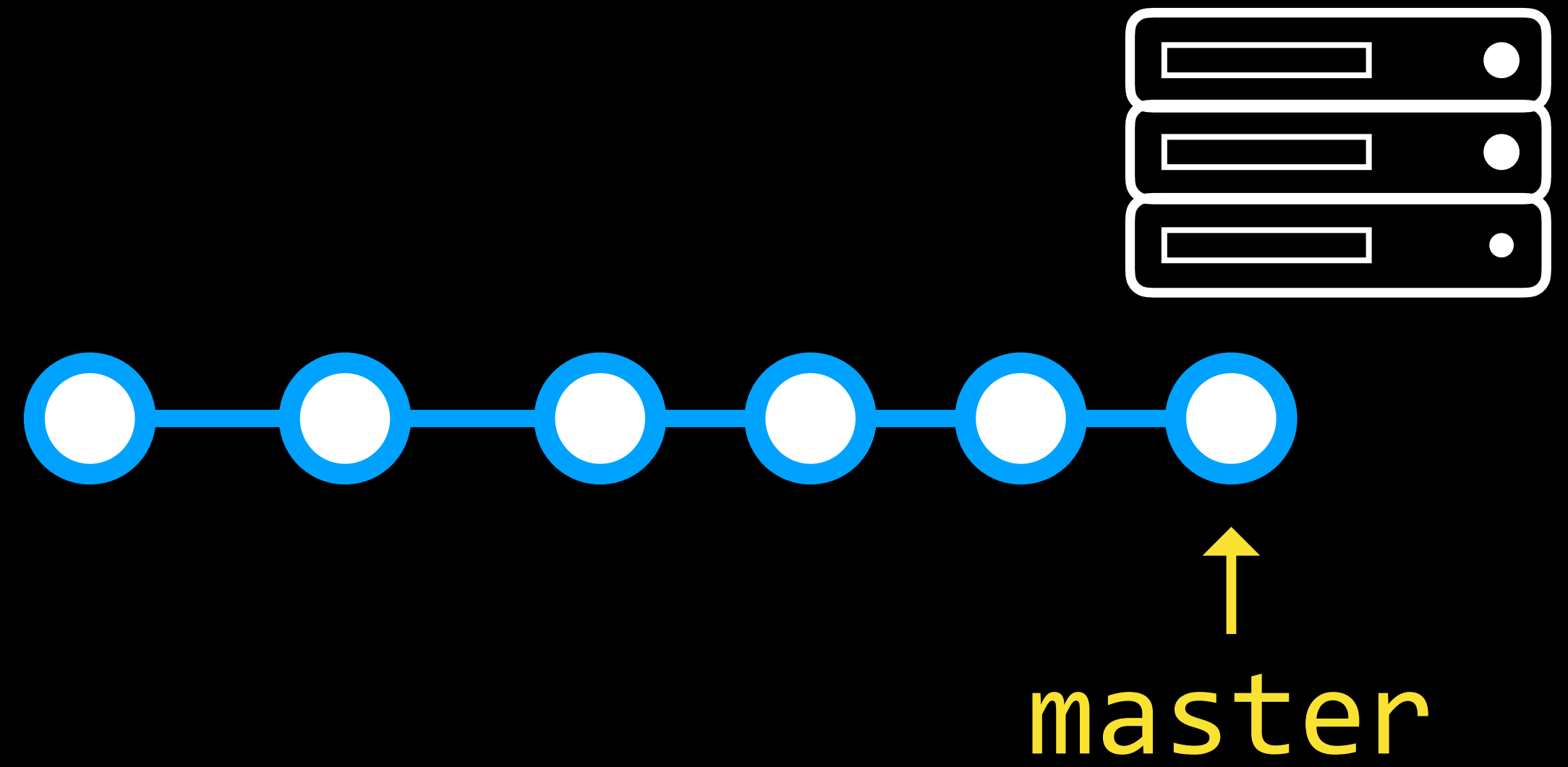


↑  
master

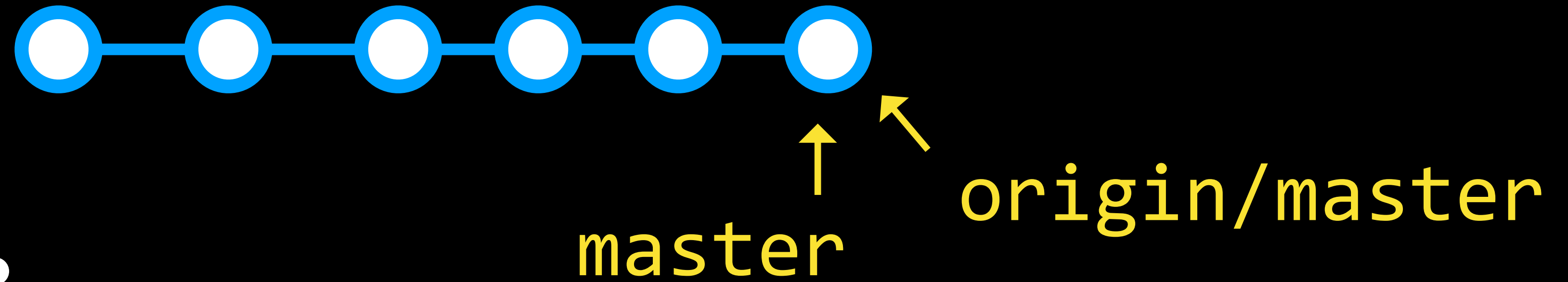
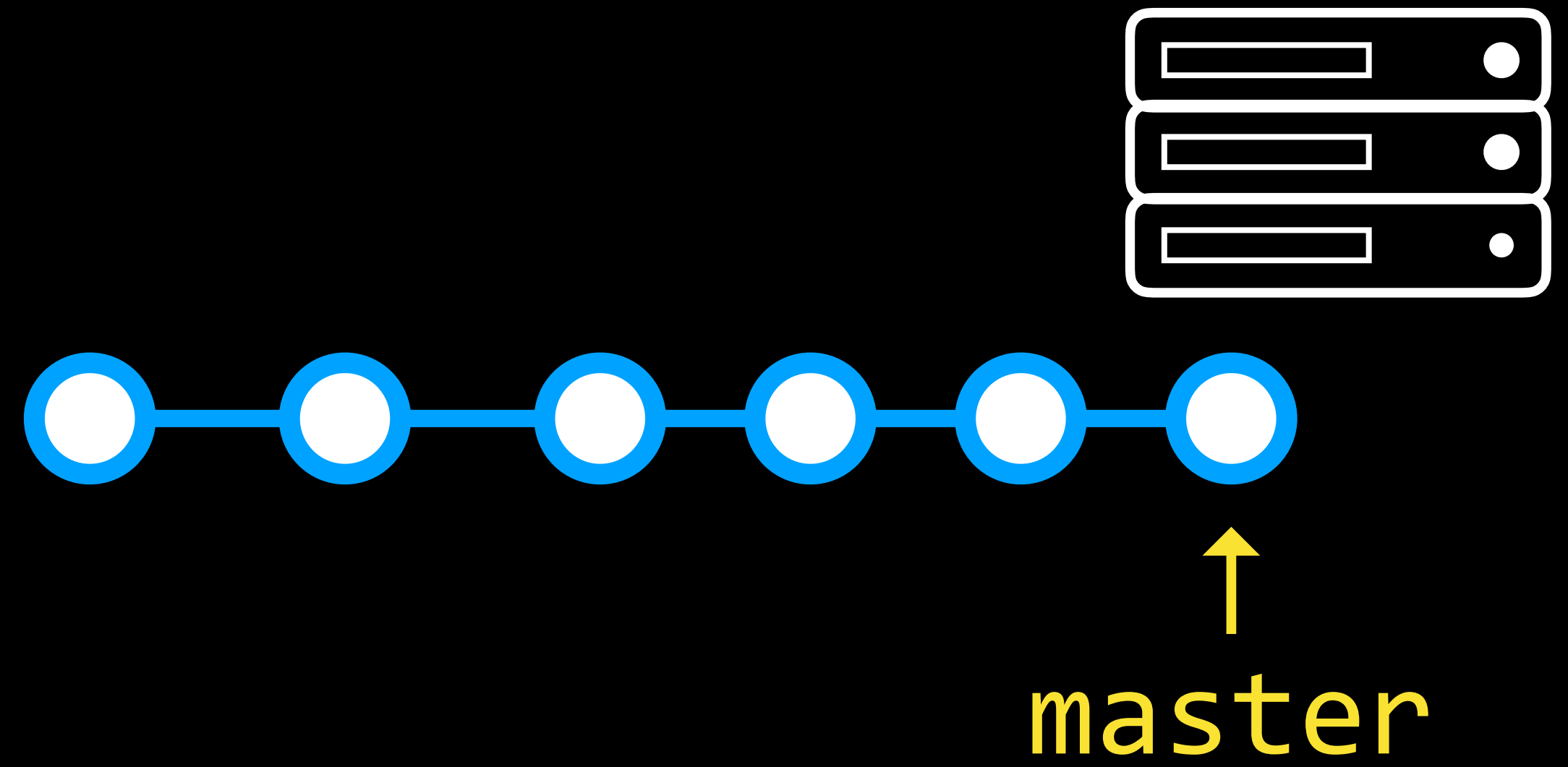
↖  
origin/master



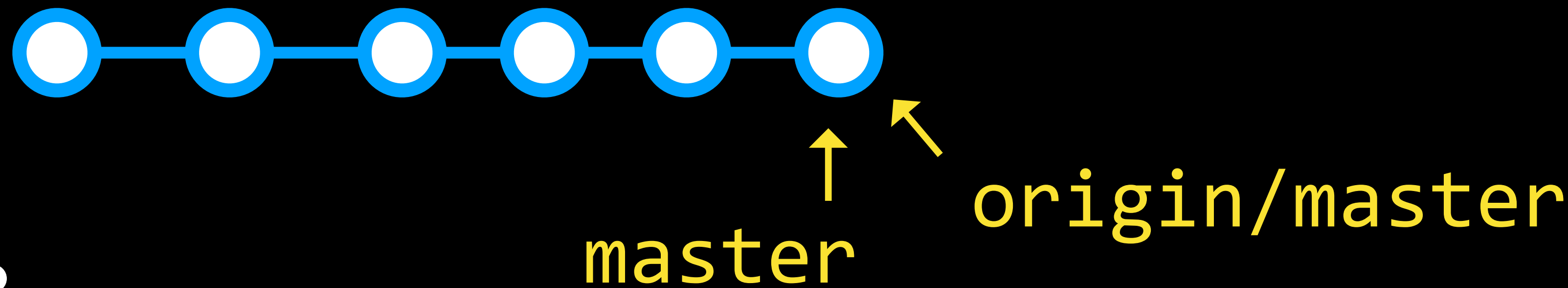
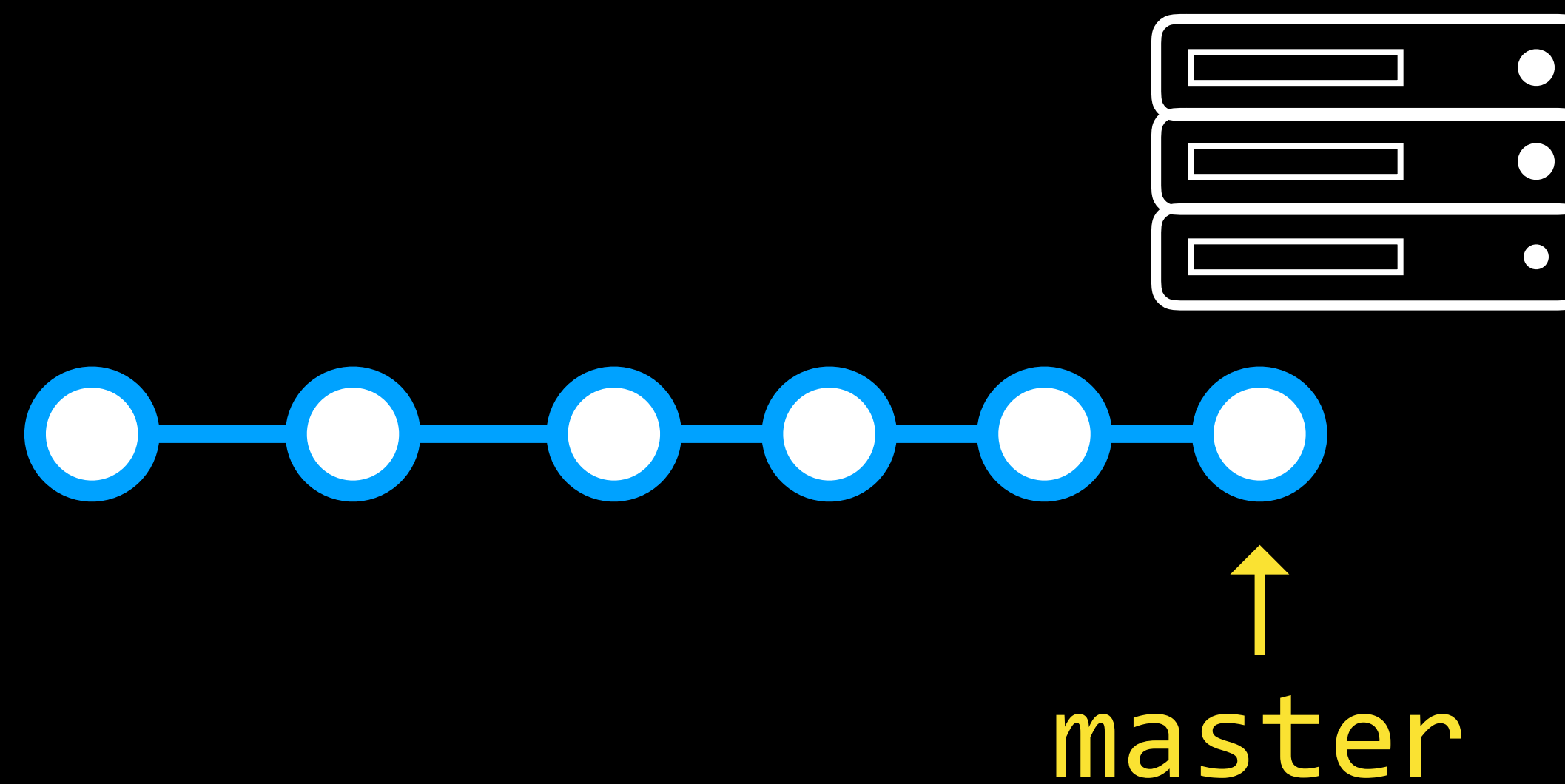
# Remotes



# Remotes



# Remotes



# GitHub

- Issues
- Forks
- Pull Requests

Sass

# Sass Features

- Variables
- Nesting
- Inheritance
- ...



# Variables

CSS

```
color: blue;
```

Sass

```
$foo: blue;
```

```
color: $foo;
```

# Nesting

## CSS

```
div {  
  font-size: 18px;  
}  
div p {  
  color: blue;  
}  
div ul {  
  color: green;  
}
```

## Sass

```
div {  
  font-size: 18px;  
  p {  
    color: blue;  
  }  
  ul {  
    color: green;  
  }  
}
```

# Afternoon Project

- Install Sass and add it to your HTML page(s).
- Push your HTML page(s) to GitHub.
- Find a partner, add each other as collaborators.  
Clone each others' repos. Make a change, and push it to a different branch. Make a pull request!

# CS50 Beyond