Intro to Coding with Python– Working with Files

Dr. Ab Mosca (they/them)

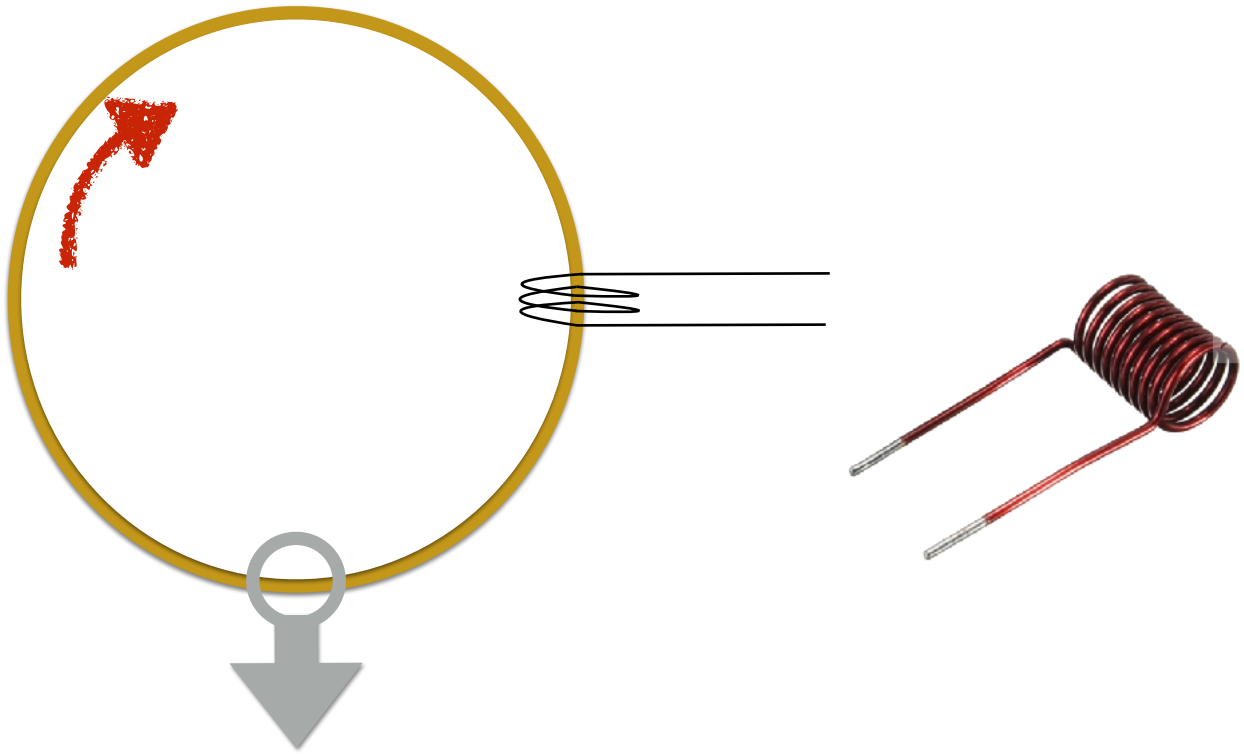
Slides based off slides courtesy of Jordan Crouser (https://jcrouser.github.io/)

Plan for Today

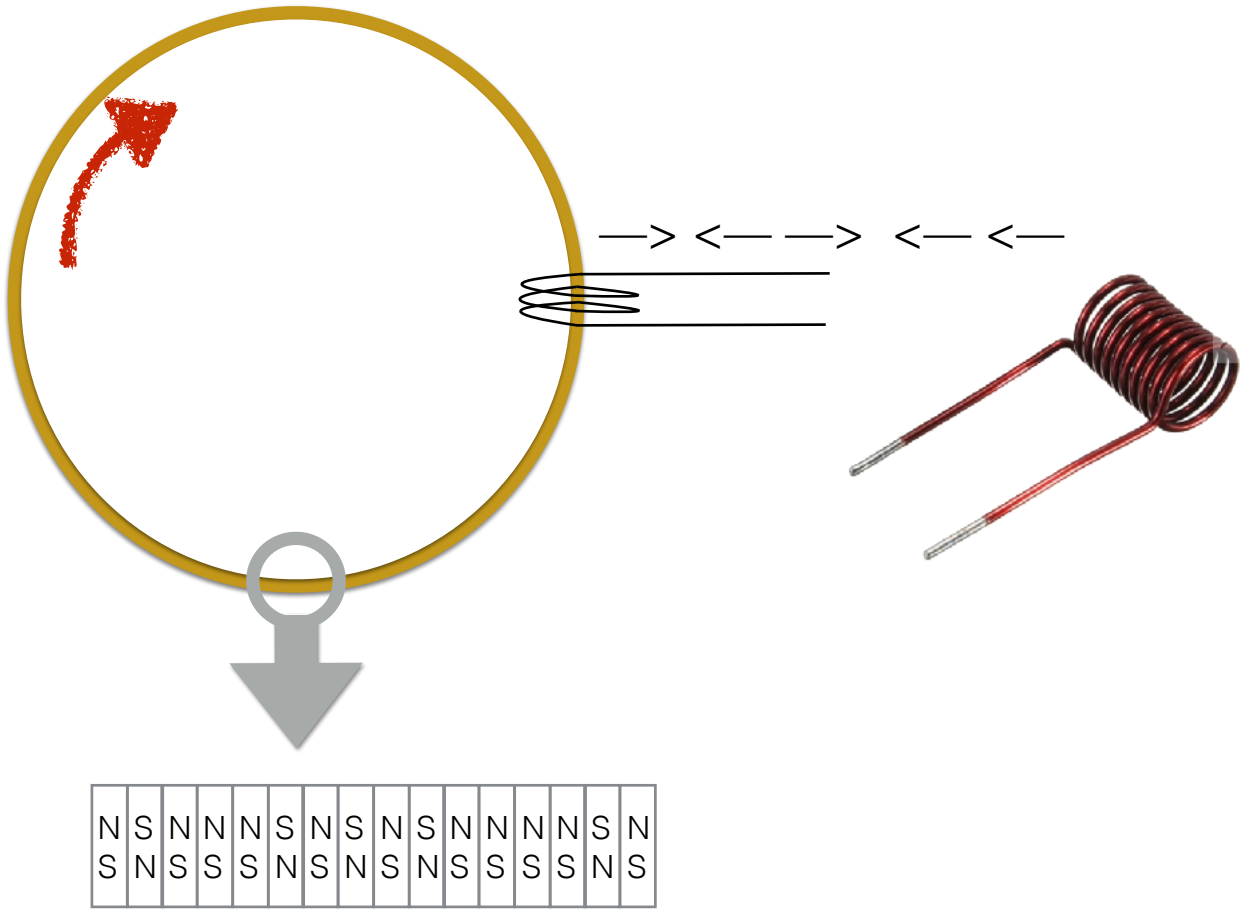
- What is a file
- Read data in
- Writing data out

RECAP: how does a hard drive work?

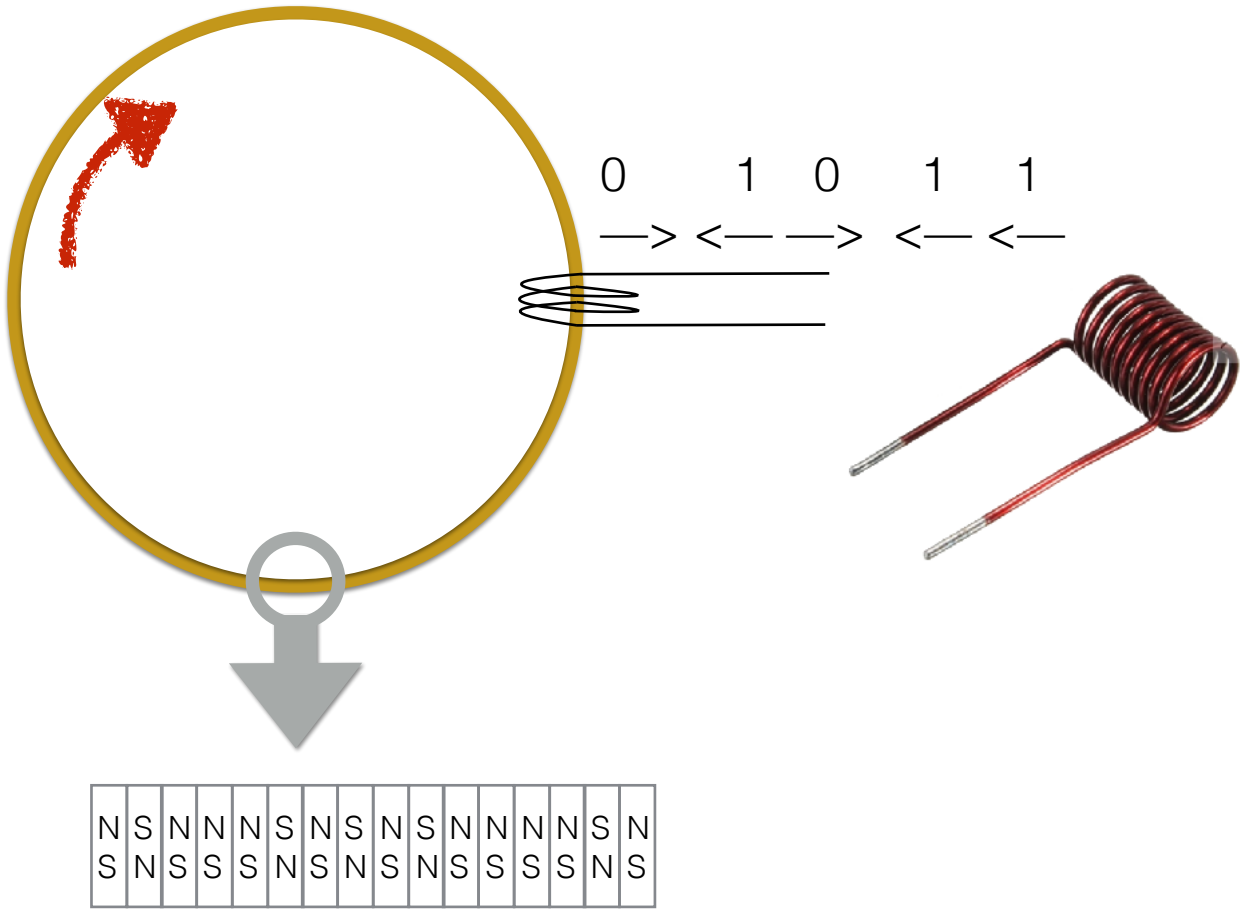
RECAP: how does a hard drive work?



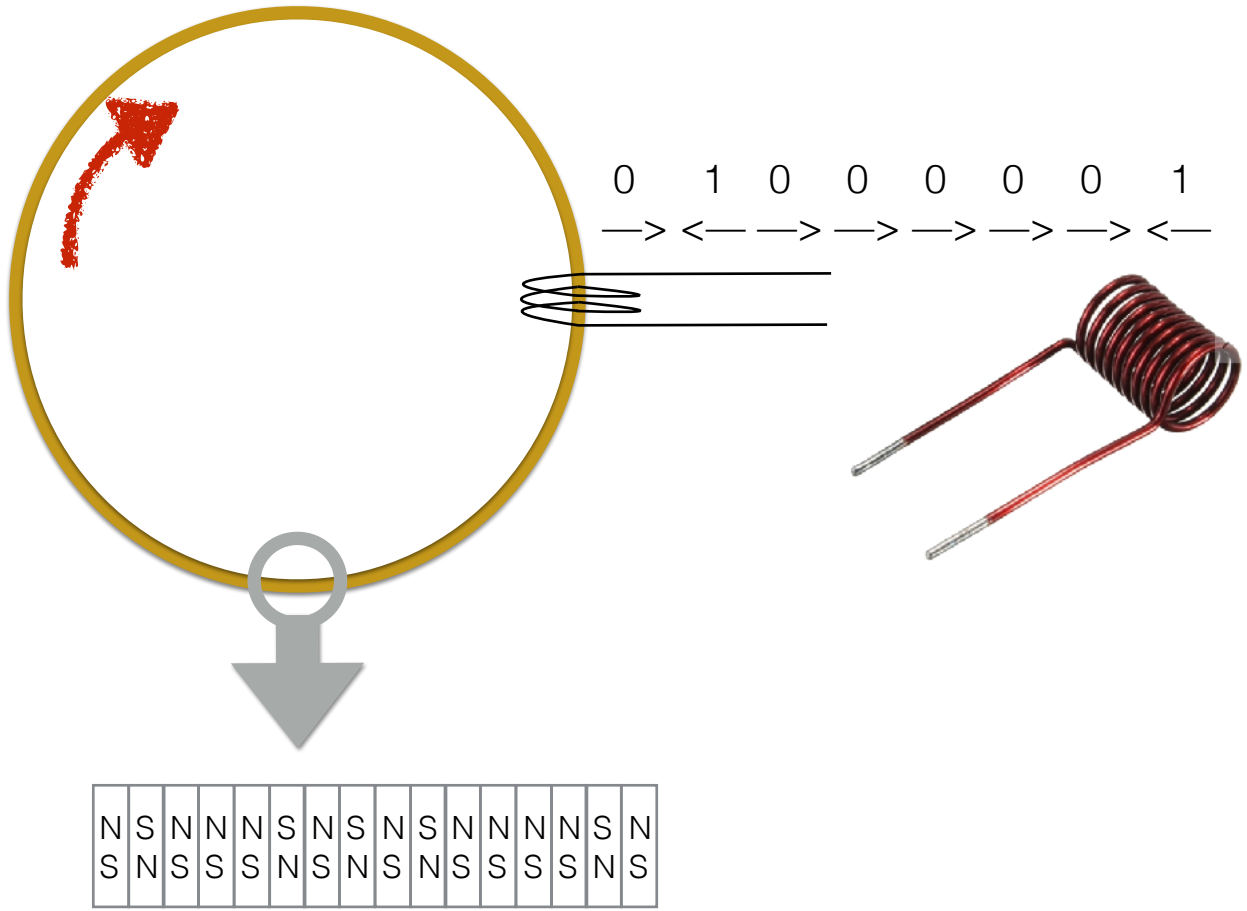
RECAP: how does a hard drive work?



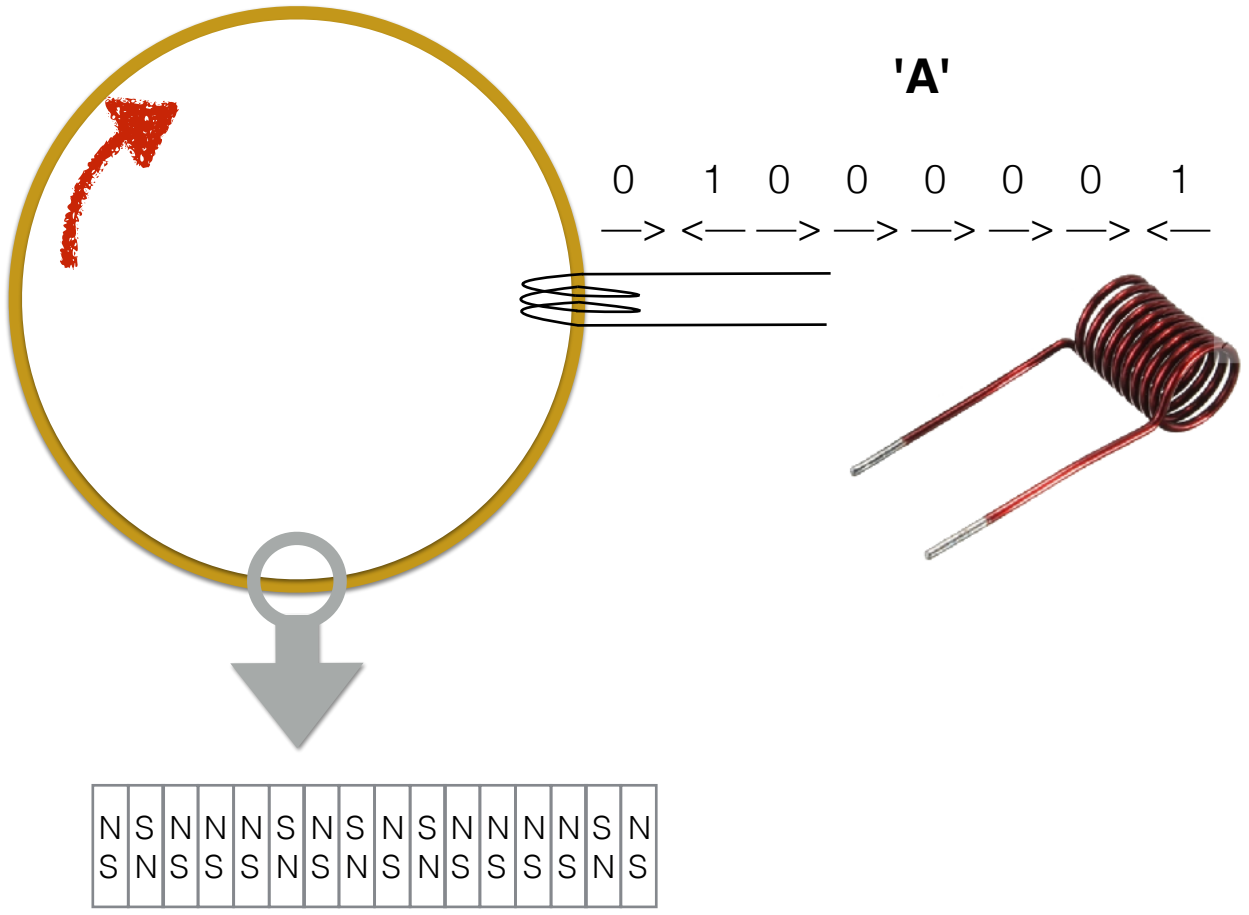
RECAP: how does a hard drive work?



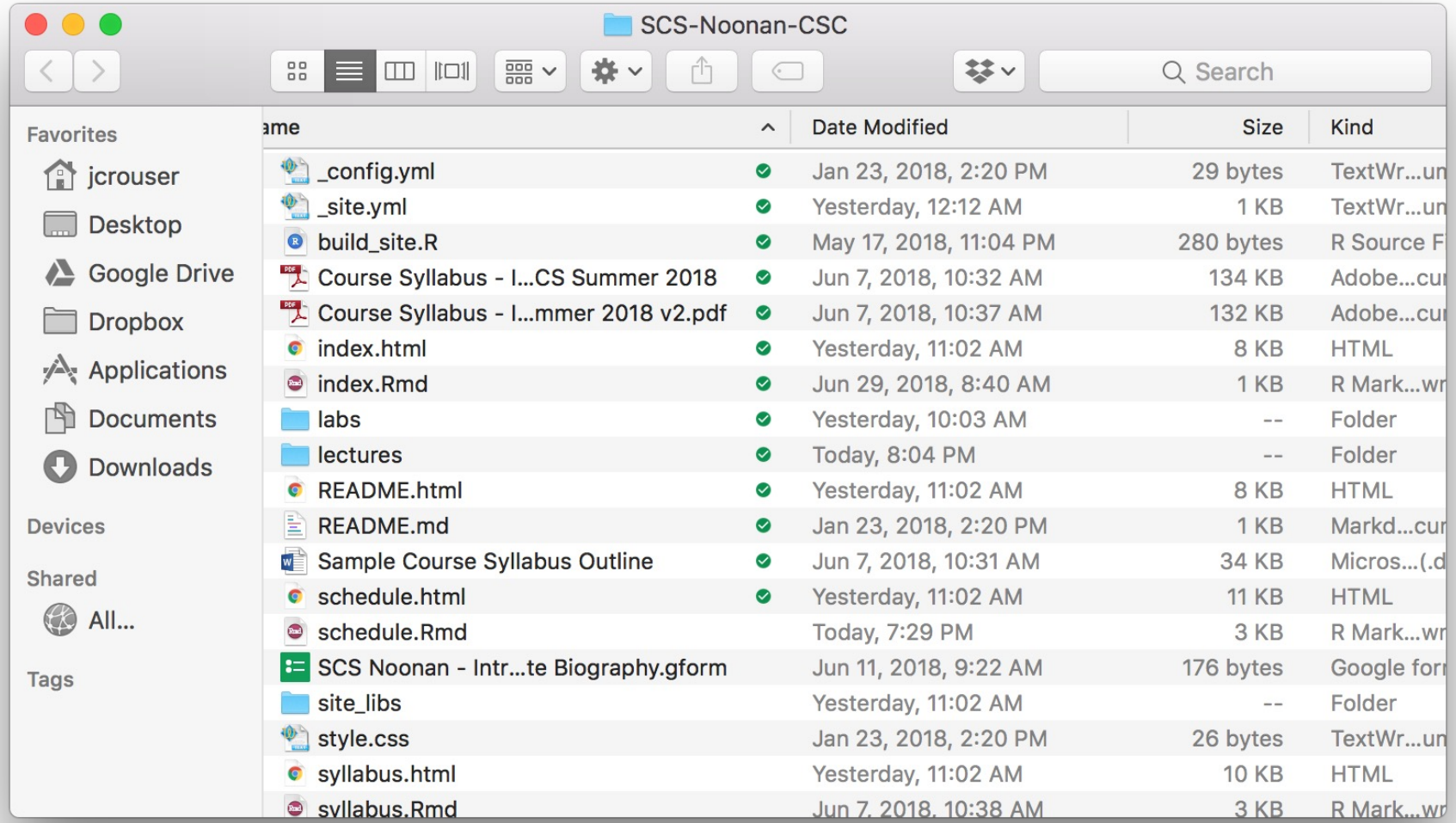
RECAP: how does a hard drive work?



RECAP: how does a hard drive work?

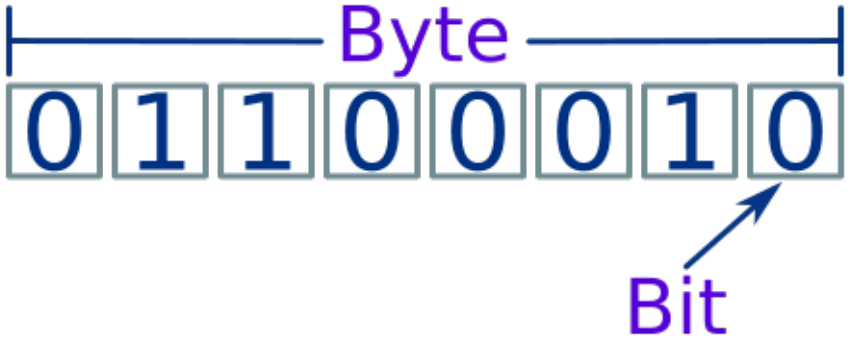


What is a **file**?



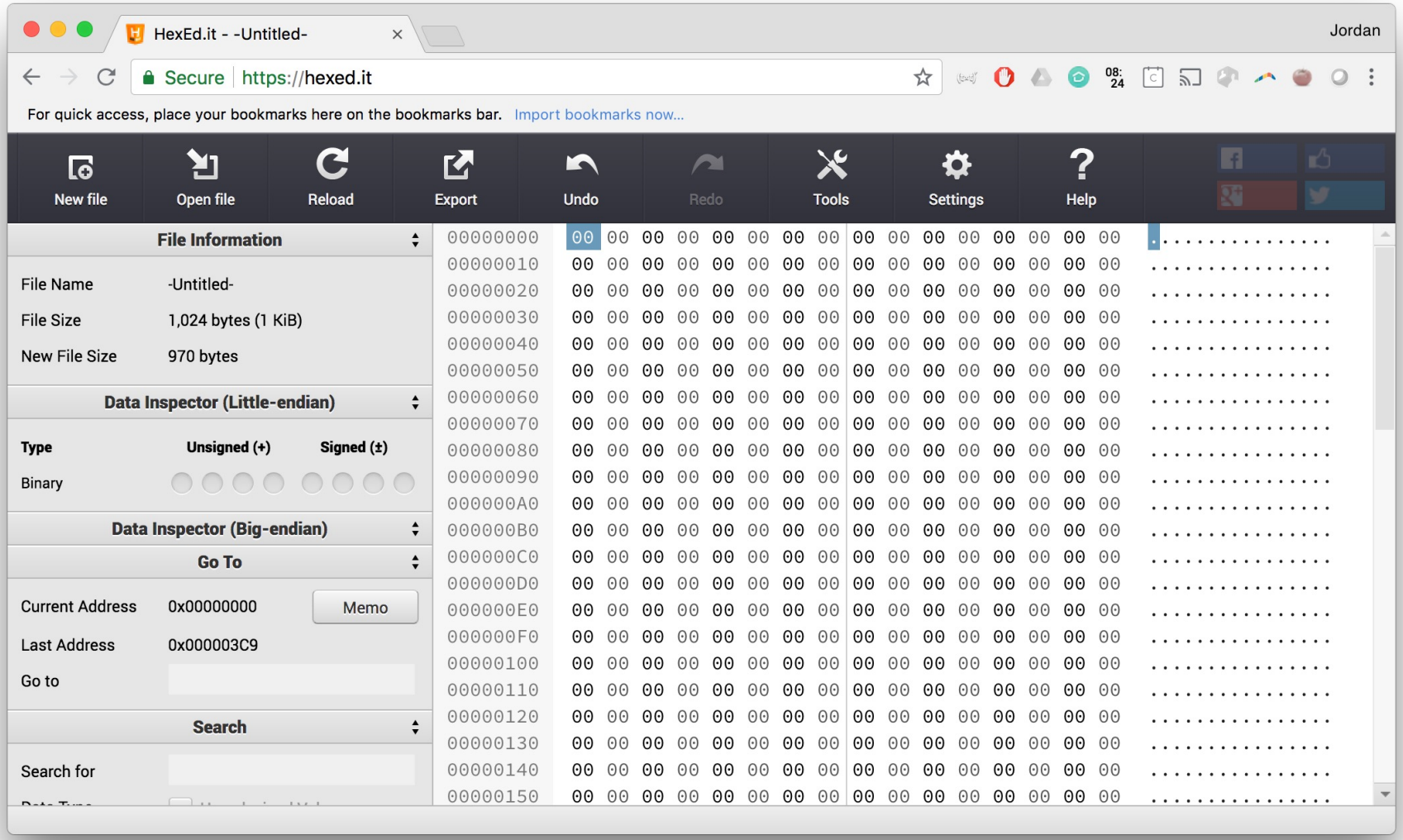
Working with **files**

- Containers of **bits**, organized into **bytes**



- Could represent text, images, music, movies, programs, applications, list of files (folders)… but underneath, they're all the same: 0s and 1s
- We'll start by playing with text files

Demo: looking inside a text file



Reading a text file

- In order to bring data stored in a text file into a Python program, we need to **read** it:

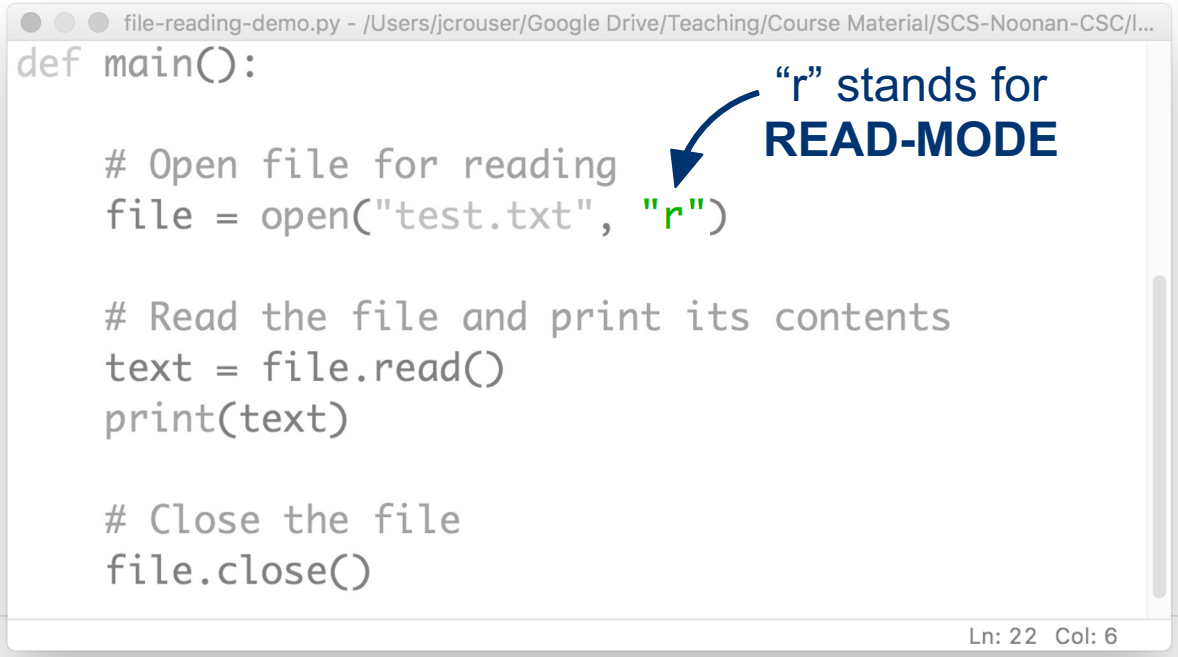
file-reading-demo.py - /Users/jcrouser/Google Drive/Teaching/Course Material/SCS-Noonan-CSC/I...

```
def main():

    # Open file for reading
    file = open("test.txt", "r")

    # Read the file and print its contents
    text = file.read()
    print(text)

    # Close the file
    file.close()
```

Ln: 22 Col: 6

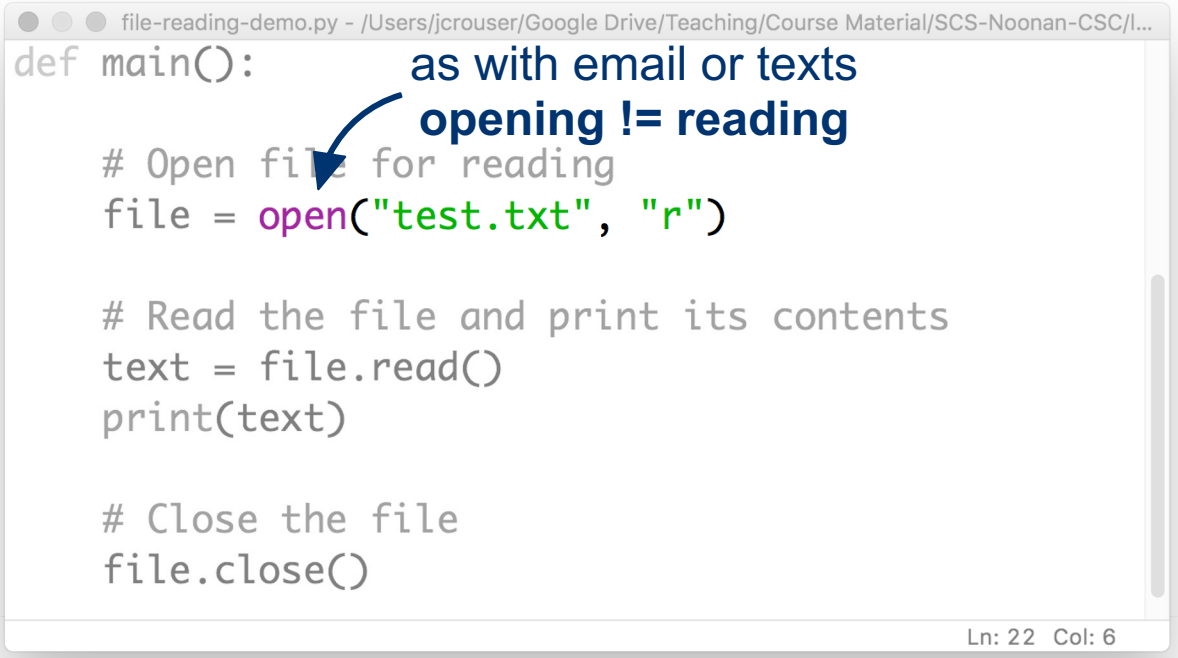
Reading a text file

- In order to bring data stored in a text file into a Python program, we need to **read** it:



Reading a text file

- In order to bring data stored in a text file into a Python program, we need to **read** it:



Reading a text file

- In order to bring data stored in a text file into a Python program, we need to **read** it:

file-reading-demo.py - /Users/jcrouser/Google Drive/Teaching/Course Material/SCS-Noonan-CSC/I...

```
def main():

    # Open file for reading
    file = open("test.txt", "r")

    # Read the file and print its contents
    text = file.read()
    print(text)

    # Close the file
    file.close()
```

`file` is now an **object** with a `.read()` method that returns a string

Ln: 22 Col: 6

Reading a text file

- In order to bring data stored in a text file into a Python program, we need to **read** it:

file-reading-demo.py - /Users/jcrouser/Google Drive/Teaching/Course Material/SCS-Noonan-CSC/I...

```
def main():

    # Open file for reading
    file = open("test.txt", "r")

    # Read the file and print its contents
    text = file.readlines()
    print(text)

    # Close the file
    file.close()
```

it also has a `.readlines()` method that returns a `list` of strings

Ln: 22 Col: 6

Reading a text file

- In order to bring data stored in a text file into a Python program, we need to **read** it:

file-reading-demo.py - /Users/jcrouser/Google Drive/Teaching/Course Material/SCS-Noonan-CSC/I...

```
def main():

    # Open file for reading
    file = open("test.txt", "r")

    # Read the file and print its contents
    text = file.read()
    print(text)

    # Close the file
    file.close()
```

Ln: 22 Col: 6

after you **`.read()`** or **`.readlines()`** a file, remember to **`.close()`** it

Key points for **reading** files

- Three-step process:
  1. `.open()`
  2. `.read()` or `.readlines()`
  3. `.close()`
- All three steps, always in that order
- If you want to `.read()` a file multiple times, you have to repeat the whole process

```
file = open("test.txt", "r")
text1 = file.read()
text2 = file.read() # Empty!
file.close()
```

Writing data to a text file

- The process looks very similar when we want to **write** data to a file:

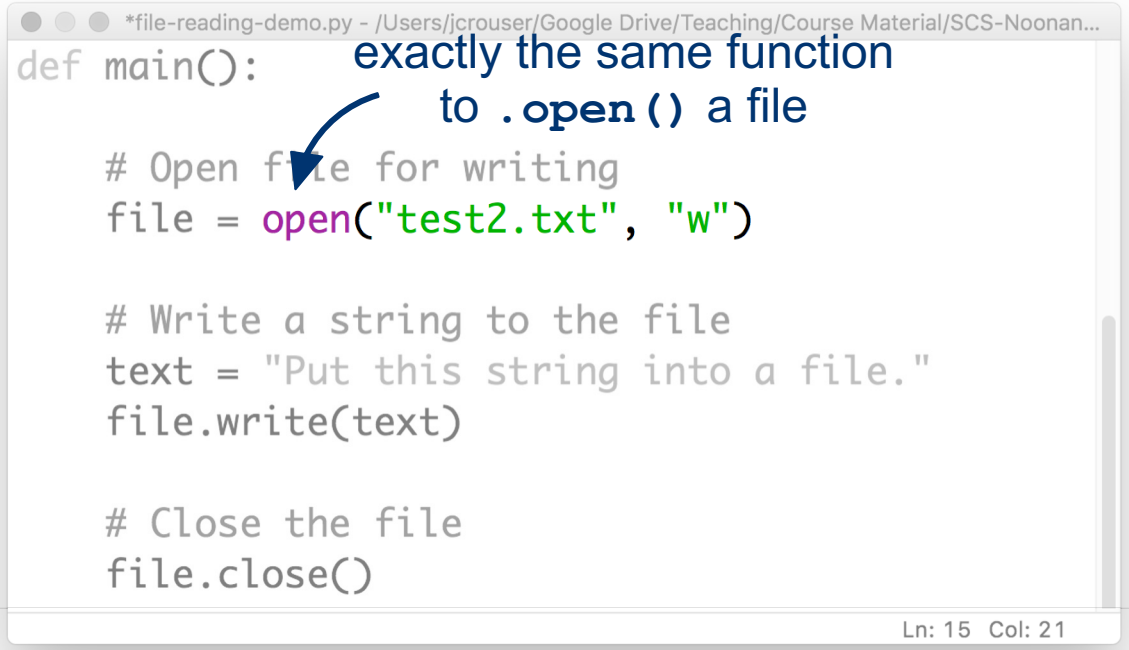
*file-reading-demo.py - /Users/jcrouser/Google Drive/Teaching/Course Material/SCS-Noonan...

```
def main():

    # Open file for writing
    file = open("test2.txt", "w")

    # Write a string to the file
    text = "Put this string into a file."
    file.write(text)

    # Close the file
    file.close()
```

Ln: 15 Col: 21

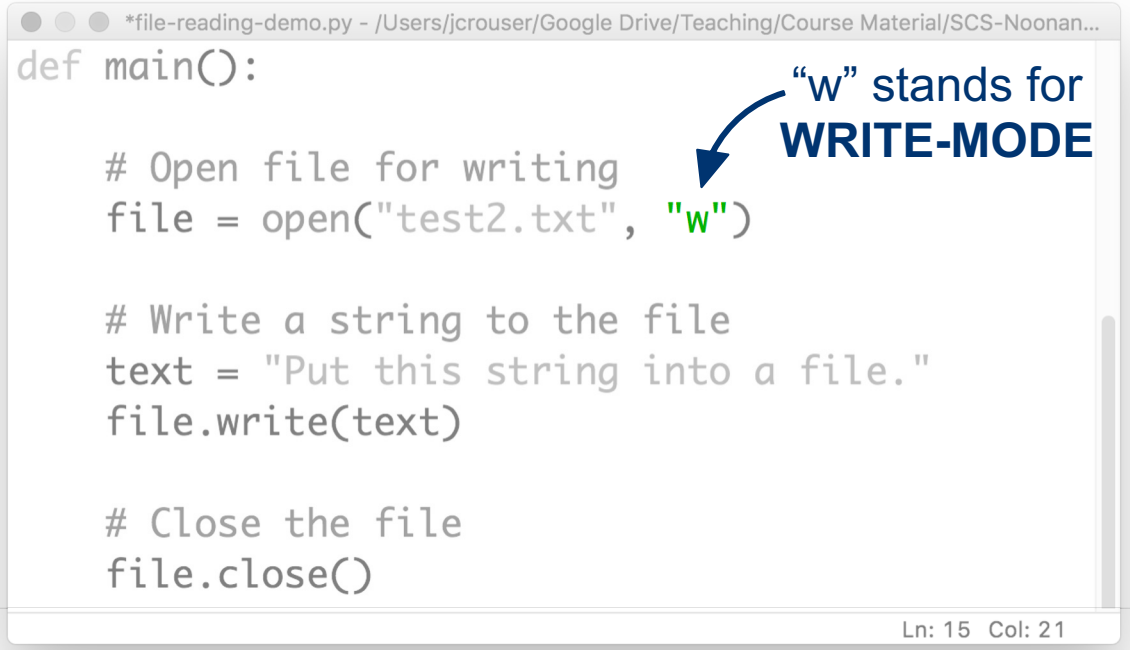
Writing data to a text file

- The process looks very similar when we want to **write** data to a file:



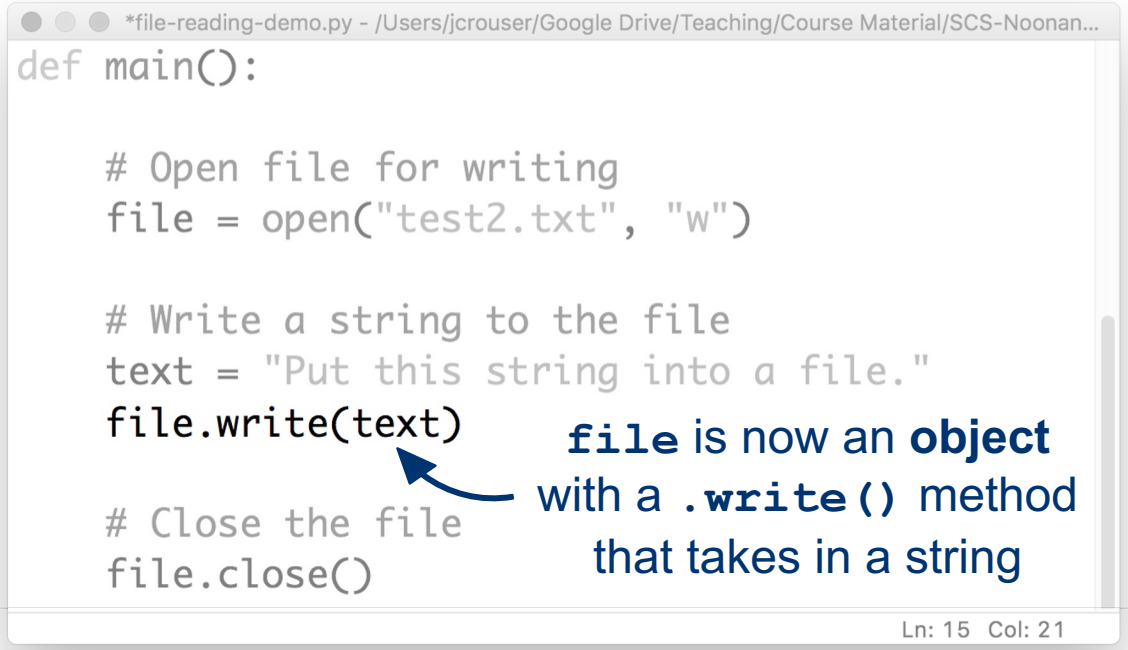
Writing data to a text file

- The process looks very similar when we want to **write** data to a file:



Writing data to a text file

- The process looks very similar when we want to **write** data to a file:



Writing data to a text file

- The process looks very similar when we want to **write** data to a file:

*file-reading-demo.py - /Users/jcrouser/Google Drive/Teaching/Course Material/SCS-Noonan...

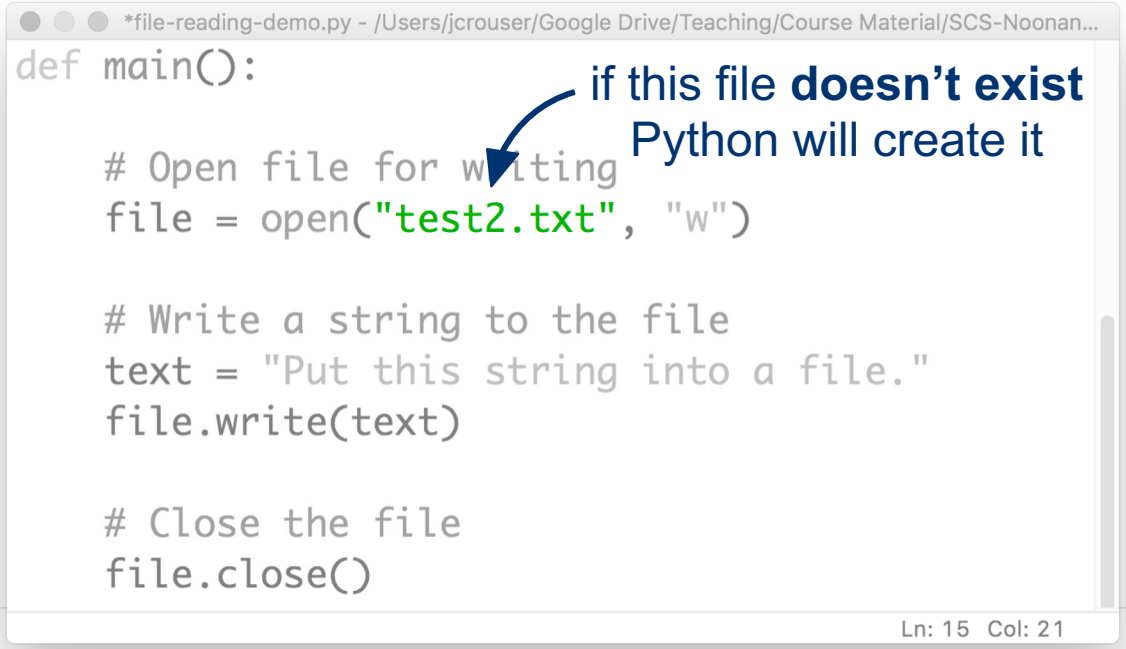
```
def main():

    # Open file for writing
    file = open("test2.txt", "w")

    # Write a string to the file
    text = "Put this string into a file."
    file.write(text)

    # Close the file
    file.close()
```

Ln: 15 Col: 21

as before, after you `.write()` to a file remember to `.close()` it

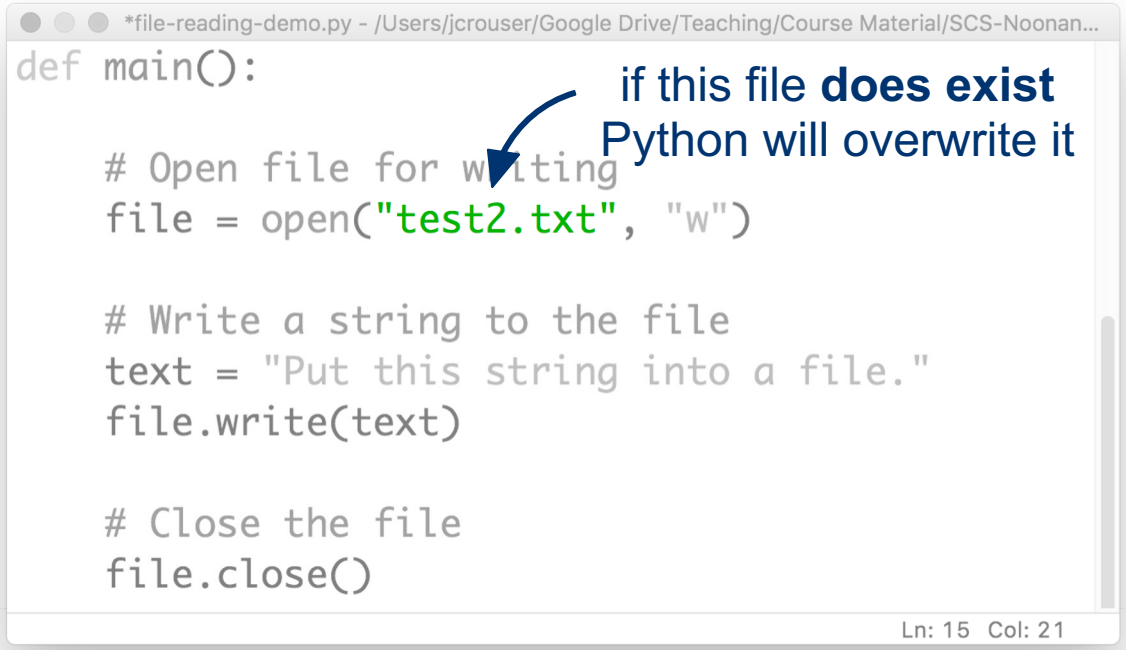
Writing data to a text file

- Some quirks to be aware of when **writing** to files:



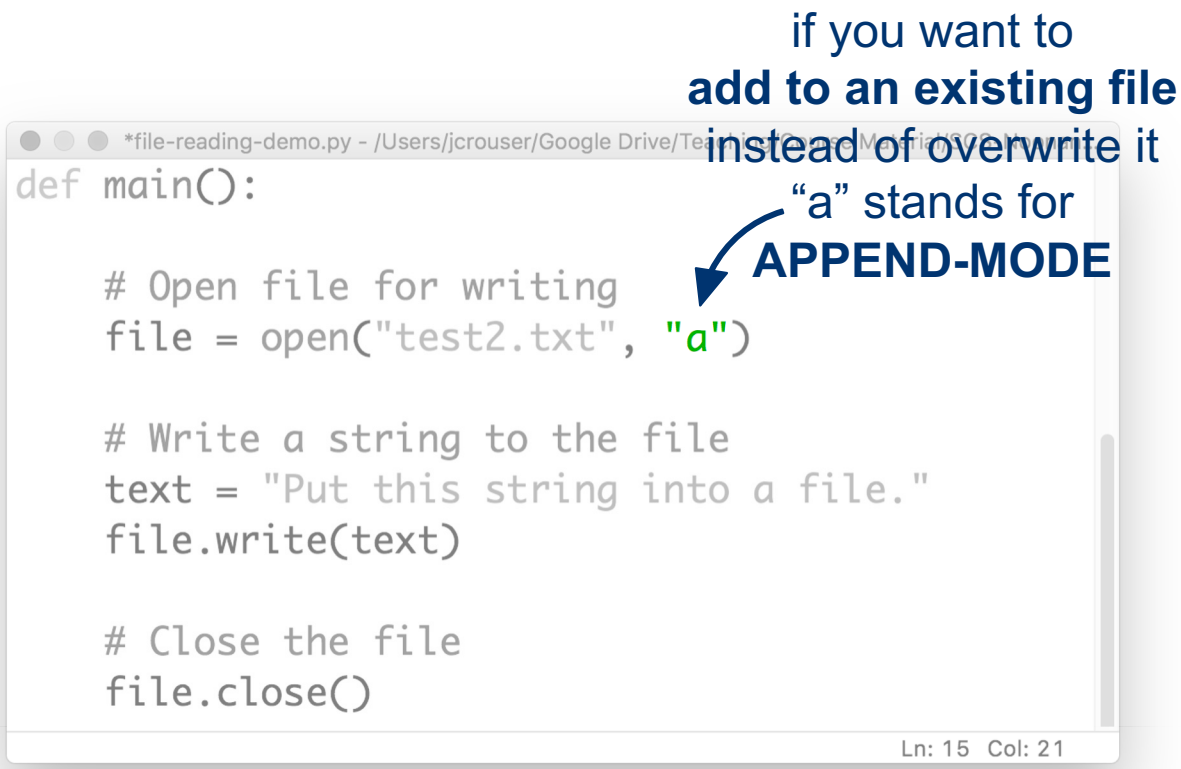
Writing data to a text file

- Some quirks to be aware of when **writing** to files:



Writing data to a text file

- Some quirks to be aware of when **writing** to files:



Key points for **writing** to files

- Three-step process:
  1. `.open()`
  2. `.write()`
  3. `.close()`
- Unlike `.read()`, you can `.write()` to an `.open()` file as many times as you want (appending each time)

```
file = open("test2.txt", "w")
file.write("Hello")
file.write("there!")
file.close()
```

- If you want a new line, you have to add it yourself! (`\n`)

15-minute exercise: read and write

Open the files.py demo on repl.it

Write a program that:

1. reads the file **`horizontal.txt`**
2. breaks it into individual words
3. and writes the words to a new file **`vertical.txt`**, each one on its own line

Discussion

What did you come up with?