

Data Structures & Programmatic Thinking

Pepe García

2020-04-20

Data Structures & Programmatic Thinking

<https://slides.com/pepegar/dspt-10/live>

Plan for this session

- Learn about handling files

Reading files

```
file = open("file_path")  
  
for line in file:  
    #do something with line  
    pass
```

Reading files

- create a text file
- read all its lines

Interlude: with

Every time we use files we need to **close()** the file after use. Not closing the file would end up in an unexpected program crash.

```
fh = open("file.txt")  
# do whatever with the file here  
fh.close()
```

Interlude: with

Rewriting our previous example to use **with**

Interlude: with

In order to avoid this, Python provides the **with** keyword. Whatever we pass to **with** will be closed after the body.

```
with open("file.txt") as fh:  
    pass # do whatever with the file
```


Writing files

We can write to files using a similar approach

```
with open("file.txt", "w") as f:  
    f.write("this content will be written to the file!")
```

Writing files. modes

When opening a file, we can choose in which **mode** we open it

CSV files

Python comes with a **CSV** library that we can use out of the box. We use it by **importing** it. **Imports** are commonly added at the top of the file.

```
import csv
```

CSV files

The **csv** library is based on the idea of readers and writers. One can read all lines in a file like so:

```
with open("file.csv") as f:
    reader = csv.reader(f)
    for line in reader:
        print(line) #line will be a list here
```

first we open the file normally

Then we create a reader using **csv.reader()**

Finally, we operate with the reader

CSV files

writing is not very different from reading:

```
lines = [  
    ["asdf", "qwer"],  
    ["hello", "world"]  
]  
  
with open("file.csv", "a") as f:  
    writer = csv.writer(f)  
    for line in lines:  
        writer.writerow(line)
```

First we need some data to put in the csv file

Then we open the file with the append mode

Later, we create a **csv.writer**

CSV files. Dictionaries

We can use specific writers for dictionaries!

```
beatles = [  
    {"name": "John", "instrument": "voice"},  
    {"name": "Paul", "instrument": "guitar"},  
    {"name": "George", "instrument": "bass"},  
    {"name": "Ringo", "instrument": "drums"}  
]  
  
with open("beatles.csv", "w") as my_file:  
    writer = csv.DictWriter(my_file, ["name", "instrument"])  
    writer.writeheader()  
    for beatle in beatles:  
        writer.writerow(beatle)
```

First we need some data to put in the csv file

CSV files. Dictionaries

We can use specific readers too

```
with open("beatles.csv") as my_file:
    reader = csv.DictReader(my_file)
    for beatle in reader:
        print(beatle["name"] + " -> " + beatle["instrument"])
```

Then we open the file with the read mode (default)

Later, we create a **csv.DictReader**

Each element in the reader will be a dictionary already

Exercises

1. Without using the csv library, create a function named `parse` **for** converting from CSV to a lists of lists.
2. Without using the csv library, create a function named `to_csv` **for** converting that list of lists to CSV.