Raw Files

This lesson focuses on raw files. It gives a complete explanation about how to read data from raw files using Python.

Introduction to raw files

Sometimes you get data in strange formats and you have to roll your own Python code to process the data. Fortunately, doing this is simple.

For this, we will assume that you have data in some type of text file. Each row of data corresponds to a row in your text file.

For example, you might have a file delimited by a pipe (|). It could look something like this:

```
James|22|M
Sarah|31|F
Mindy|25|F
```

Above we have 3 rows of data each with 3 values separated by a pipe (|) - name, age, and gender.

We can create this same file in Python to process using tempfile.

```
import tempfile

tmp = tempfile.NamedTemporaryFile()

# Open the file for writing. And write the data.
with open(tmp.name, 'w') as f:
    f.write("James|22|M\n")
    f.write("Sarah|31|F\n")
    f.write("Mindy|25|F")

# Read in the data from our file, line by line
with open(tmp.name, "r") as f:
    for line in f:
      print(line)
```

Above, we used `tempfile` (**line 6**) to create a file that contains three rows of data. The **n** at the end of the first 2 rows of data (**line 7-8**) tells the program to create new lines.

Reading in raw files with Python

The part of the code we care about is:

```
with open(tmp.name, "r") as f:
    for line in f:
      print(line)
```

We use the `open()` command to read the file. We first pass the *name* of the file we want to process and then an **r** which stands for *read*.

Since we used the `with as` functionality in Python, our filehandle is now represented as the variable `f` and is only relevant within this scope. This is good practice for reading files because you don't have to worry about closing the filehandle. Once you have a filehandle, Python makes it very easy to access the lines. The variable `f` is now an iterable where each iterable is a line of the file. Thus, when we do

```
for line in f
```

We are just looping over the lines of our file. The code above only prints out those lines, but since we have direct access to the lines of our file, we can do whatever we want in our processing. For example, here is code that takes only the first value of each row and appends them to a list.

```
import tempfile

tmp = tempfile.NamedTemporaryFile()

# Open the file for writing and write our data
# Open the file for writing and write our data
with open(tmp.name, 'w') as f:
    f.write("James|22|M\n")
    f.write("Sarah|31|F\n")
    f.write("Mindy|25|F")

first_values = []  # Define a list to store the first values of each row
with open(tmp.name, "r") as f:  # Open the file to read
    for line in f:  # Loop over each line
      row_values = line.split("|")  # Split each line by the | character into a list
      first_values.append(row_values[0])  # Add the first value to our list

print(first_values)
```

That’s it on reading files in Python using support from different libraries. In the next lesson, there’s a challenge for you to solve.