Advanced files and objects

Olatz Perez de Viñaspre Garralda
Ander Soraluze Irureta

Advanced files: CSV

CSV files

Coma Separated Values

For example, the following table:

name	age
Mikel	85
Amaia	20
Ane	5

We will represent it as:

```
name,age
Mikel,85
Amaia,20
Ane,5
```

We will use the very powerful pandas library

How to use libraries?

- You need to import it into your code (pandas, e.g.):

```
import pandas
```

- If you want to give it a shorter name:

```
import pandas as pd
```

- Now you can use any function implemented in the library (pandas in this case)

Reading CSV files with pandas

```
import pandas as pd
df = pd.read_csv(fin,header=None)
for index,row in df.iterrows():
        print(index,row[0],row[1])
```

Output:

```
0 name age
1 Mikel 85
2 Amaia 20
3 Ane 5
```

- We do not have to set the number of columns
- The content of the file is stored in a data frame (pandas object)
- You can iterate over the rows to work with its content

Reading CSV files with pandas

```
import pandas as pd
df = pd.read_csv(fin,header=0)
for index,row in df.iterrows():
        print(index,row["name"],row["age"])
```

Output:

```
0 Mikel 85
1 Amaia 20
2 Ane 5
```

- You can determine the header line (default is 0)
- If the header is set, you can use the column names to access the row's value

Let’s try it

06-01-csv_pandas.py

Instead of commas, any character can be used

```
import pandas as pd
df = pd.read_csv(“example.csv”,sep=”:”)
for index,row in df.iterrow():
    print(row[0],row[1])
```

Writing CSV files

With the pandas library also, we can write them easily

We can write the whole data frame with the function **to_csv(filename)**

```
import pandas pd
rows = []
rows.append([“Hello”, “world”])
rows.append([“bye”]*2)

df = pd.DataFrame(rows)
df.to_csv(“example.csv”)
```

Let’s try it

06-02-csv_write.py

Append rows

You can even add rows to a previous CSV file

```
import pandas as pd

df = pd.read_csv(“example.csv”)
new_row = {“name”:”Olatz”, “age”:34}

df =
df.append(new_row,ignore_index=True)

df.to_csv(“example_out.csv”)
```

Let’s try it

06-03-csv_write.py

Objects

Python is object oriented

- Object = collection of data (variables) and methods (functions)
- Class = blueprint of an object
- Example: class is the sketch (prototype) of a house: details of floors, doors, windows,... We build a house based on that sketch. House will be the object (we can build many houses from the same sketch)

Defining a Class

Similar to functions, we will use the keyword **class** instead of **def**

```
class House:
        windows = 4
        floors = 2
        def message(self):
                print("This is a house.")

print(House.windows)

print(House.message)
```

Creating an Object in Python

Definition of the Class:

```
class House:
        windows = 4
        floors = 2
        def message(self):
                print(“This is a house”)
```

Creation of object

```
my_house = House()
```

Constructors

__init__() function

Functions that begin with double __ are called special functions

```
class House:
    def __init__(self,pW=4,pF=2):
        self.windows = pW
        self.floors = pF
    def get_info(self):
        print("This is a",
self.windows, "windows and",
self.floors, "floors house.")

house1 = House()
house2 = House(2,1)
house3 = House(6,3)
____
```

Let’s try it

06-04-objects.py


That's all!
http://www.parentesis.com