WORKING WITH CSV AND JSON DATA

CS 3030: Python

Instructor: Damià Fuentes Escoté

University of Colorado Colorado Springs

Working with word documents from last lesson

WORKING WITH CSV

Working with CSV

- CSV stands for “comma-separated values,” and CSV files are simplified spreadsheets stored as plaintext files.
  - *csv module*
- CSV files are simple, lacking many of the features of an Excel spreadsheet. For example, CSV files
  - *Don’t have types for their values—everything is a string*
  - *Don’t have settings for font size or color*
  - *Don’t have multiple worksheets*
  - *Can’t specify cell widths and heights*
  - *Can’t have merged cells*
  - *Can’t have images or charts embedded in them*

Advantages of CSV

- The advantage of CSV files is simplicity.
- CSV files are widely supported by many types of programs, can be viewed in text editors (including IDLE’s file editor), and are a straightforward way to represent spreadsheet data.
- The CSV format is exactly as advertised: It’s just a text file of comma-separated values.

How a CSV file looks

example.csv

```
4/5/2014 13:34,Apples,73
4/5/2014 3:41,Cherries,85
4/6/2014 12:46,Pears,14
4/8/2014 8:59,Oranges,52
4/10/2014 2:07,Apples,152
4/10/2014 18:10,Bananas,23
4/10/2014 2:40,Strawberries,98
```

Reader Objects

```
import csv

exampleFile = open('example.csv’)
exampleReader = csv.reader(exampleFile)
exampleData = list(exampleReader)
print(exampleData)
# [['4/5/2015 13:34', 'Apples', '73'], ['4/5/2015 3:41', 'Cherries', '85’],
# ['4/6/2015 12:46', 'Pears', '14'], ['4/8/2015 8:59', 'Oranges', '52’],
# ['4/10/2015 2:07', 'Apples', '152'], ['4/10/2015 18:10', 'Bananas', '23’],
# ['4/10/2015 2:40', 'Strawberries', '98']]
```

Reader Objects

```
print(exampleData[0][0])       # '4/5/2015 13:34’
print(exampleData[0][1])       # 'Apples’
print(exampleData[0][2])       # '73’
print(exampleData[1][1])       # 'Cherries’
print(exampleData[6][1])       # 'Strawberries'
```

Reading Data from Reader Objects in a for Loop

```
exampleFile = open('example.csv’)
exampleReader = csv.reader(exampleFile)
for row in exampleReader:
    print('Row #' + str(exampleReader.line_num) + ' ' + str(row))

# Row #1 ['4/5/2015 13:34', 'Apples', '73’]
# Row #2 ['4/5/2015 3:41', 'Cherries', '85’]
# Row #3 ['4/6/2015 12:46', 'Pears', '14’]
# Row #4 ['4/8/2015 8:59', 'Oranges', '52’]
# Row #5 ['4/10/2015 2:07', 'Apples', '152’]
# Row #6 ['4/10/2015 18:10', 'Bananas', '23’]
# Row #7 ['4/10/2015 2:40', 'Strawberries', '98']
```

Writer Objects

```
outputFile = open('output.csv', 'w', newline='')
# if you forget to set the newline argument, the rows in output.csv
# will be double-spaced
outputWriter = csv.writer(outputFile)
outputWriter.writerow(['spam', 'eggs', 'bacon', 'ham'])
outputWriter.writerow(['Hello, world!', 'eggs', 'bacon', 'ham'])
outputWriter.writerow([1, 2, 3.141592, 4])
outputFile.close()
```

Writer Objects

output.csv

```
spam,eggs,bacon,ham
"Hello, world!",eggs,bacon,ham
1,2,3.141592,4
```

```
outputFile = open('output.csv', 'w', newline='')
# if you forget to set the newline argument, the rows in output.csv
# will be double-spaced
outputWriter = csv.writer(outputFile)
outputWriter.writerow(['spam', 'eggs', 'bacon', 'ham'])
outputWriter.writerow(['Hello, world!', 'eggs', 'bacon', 'ham'])
outputWriter.writerow([1, 2, 3.141592, 4])
outputFile.close()
```

WORKING WITH JSON

Working with JSON

- JSON (pronounced “JAY-sawn” or “Jason”—it doesn’t matter how because either way people will say you’re pronouncing it wrong) is a format that stores information as JavaScript source code in plaintext files.
- JSON is short for JavaScript Object Notation.
  - *You don’t need to know the JavaScript programming language to use JSON files*
- JSON format is useful to know because it’s used in many web applications.

JSON example

```
{"name": "Zophie", "isCat": true, "miceCaught": 0, "napsTaken": 37.5, "felineIQ":
null}
```

API: Application Programming Interface

- JSON is useful to know, because many websites offer JSON content as a way for programs to interact with the website.
- This is known as providing an application programming interface (API) . Accessing an API is the same as accessing any other web page via a URL. The difference is that the data returned by an API is formatted (normally with JSON).
- Many websites make their data available in JSON format.
  - *Facebook, Twitter, Yahoo, Google, Tumblr, Wikipedia, Flickr, Data.gov, Reddit, IMDb, Rotten Tomatoes, LinkedIn, and many other popular sites offer APIs for programs to use.*

API: Application Programming Interface

- Using APIs, you could write programs that do the following:
  - *Scrape raw data from websites. (Accessing APIs is often more convenient than downloading web pages and parsing HTML with Beautiful Soup.)*
  - *Automatically download new posts from one of your social network accounts and post them to another account. For example, you could take your Tumblr posts and post them to Facebook.*
  - *Create a “movie encyclopedia” for your personal movie collection by pulling data from IMDb, Rotten Tomatoes, and Wikipedia and putting it into a single text file on your computer.*

The json Module

- *import json*

- JSON can’t store every kind of Python value. It can contain values of only the following data types: strings, integers, floats, Booleans, lists, dictionaries, and NoneType.
- JSON cannot represent Python-specific objects.

Reading JSON with the loads() Function

```
import json


stringOfJsonData = '{"name": "Zophie", "isCat": true,
"miceCaught": 0, "felineIQ": null}’


jsonDataAsPythonValue = json.loads(stringOfJsonData)
print(jsonDataAsPythonValue)
# {'isCat': True, 'miceCaught': 0, 'name': 'Zophie', 'felineIQ':
None}
```

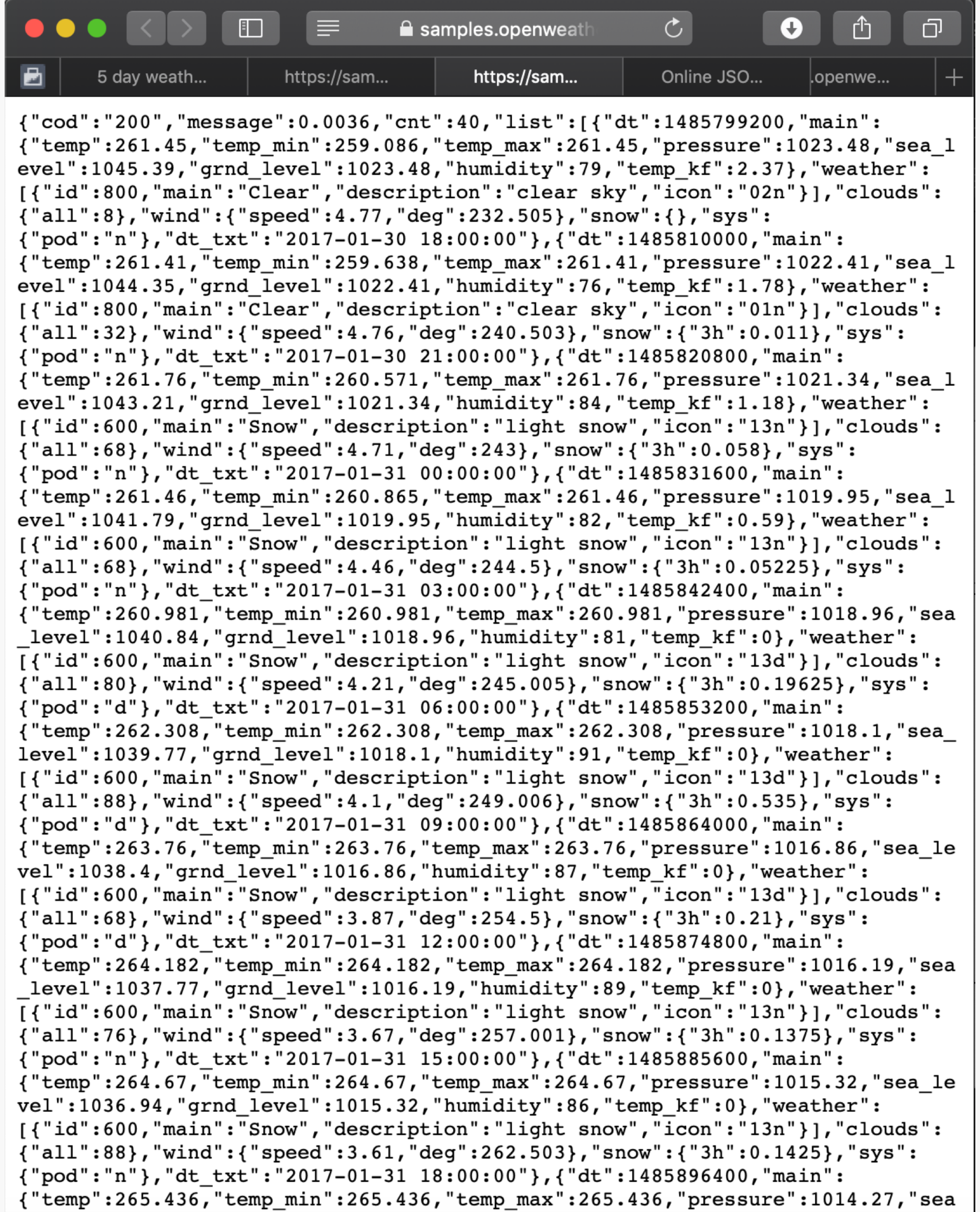
Writing JSON with the dumps() Function

```
import json


pythonValue = {'isCat': True, 'miceCaught': 0, 'name':
'Zophie', 'felineIQ': None}


stringOfJsonData = json.dumps(pythonValue)
print(stringOfJsonData)
# '{"isCat": true, "felineIQ": null, "miceCaught": 0, "name":
"Zophie" }'
```

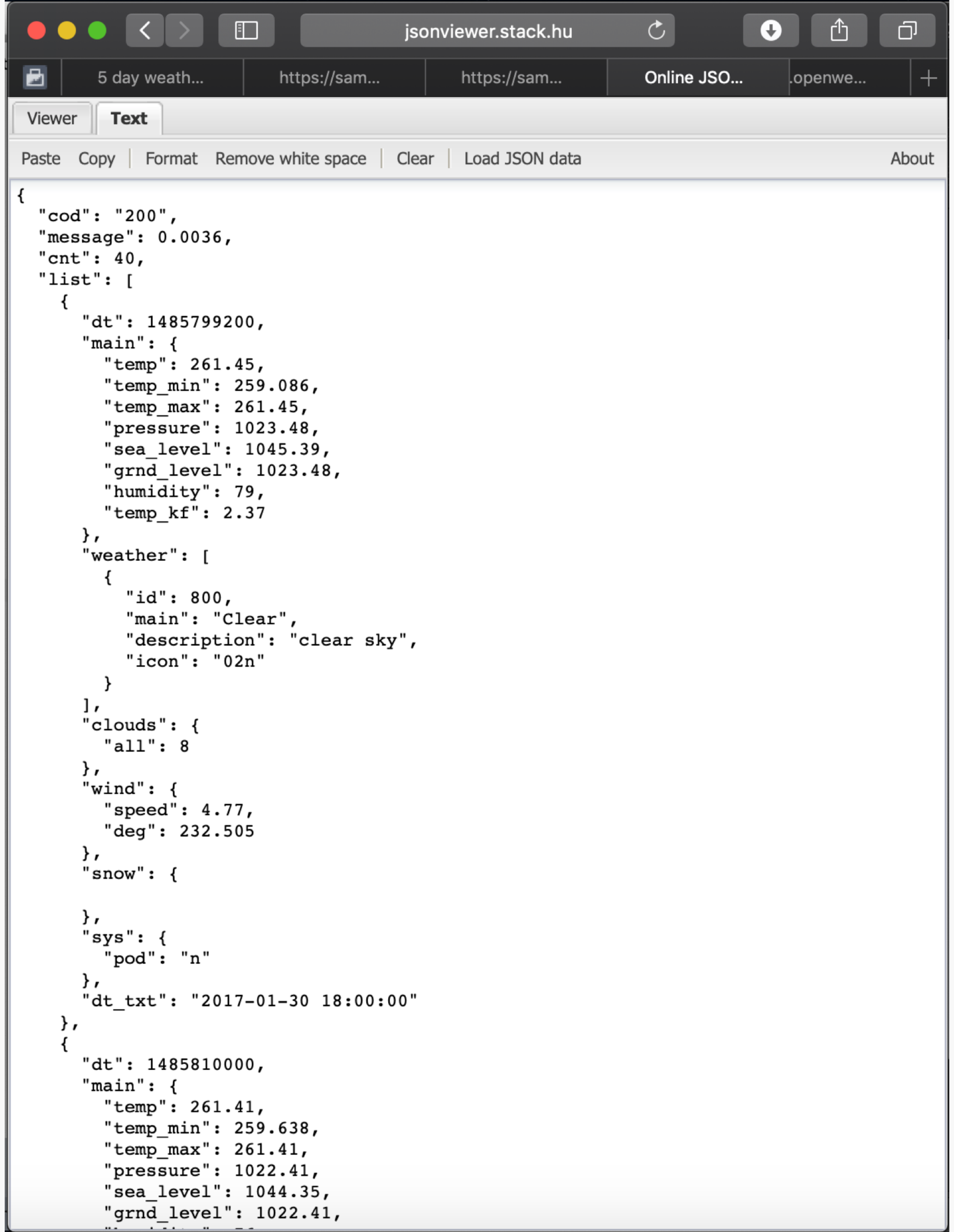
A JSON can look like:



```
{"cod":"200","message":0.0036,"cnt":40,"list":[{"dt":1485799200,"main":
{"temp":261.45,"temp_min":259.086,"temp_max":261.45,"pressure":1023.48,"sea_l
evel":1045.39,"grnd_level":1023.48,"humidity":79,"temp_kf":2.37},"weather":
[{"id":800,"main":"Clear","description":"clear sky","icon":"02n"}],"clouds":
{"all":8},"wind":{"speed":4.77,"deg":232.505},"snow":{},"sys":
{"pod":"n"},"dt_txt":"2017-01-30 18:00:00"},{"dt":1485810000,"main":
{"temp":261.41,"temp_min":259.638,"temp_max":261.41,"pressure":1022.41,"sea_l
evel":1044.35,"grnd_level":1022.41,"humidity":76,"temp_kf":1.78},"weather":
[{"id":800,"main":"Clear","description":"clear sky","icon":"01n"}],"clouds":
{"all":32},"wind":{"speed":4.76,"deg":240.503},"snow":{"3h":0.011},"sys":
{"pod":"n"},"dt_txt":"2017-01-30 21:00:00"},{"dt":1485820800,"main":
{"temp":261.76,"temp_min":260.571,"temp_max":261.76,"pressure":1021.34,"sea_l
evel":1043.21,"grnd_level":1021.34,"humidity":84,"temp_kf":1.18},"weather":
[{"id":600,"main":"Snow","description":"light snow","icon":"13n"}],"clouds":
{"all":68},"wind":{"speed":4.71,"deg":243},"snow":{"3h":0.058},"sys":
{"pod":"n"},"dt_txt":"2017-01-31 00:00:00"},{"dt":1485831600,"main":
{"temp":261.46,"temp_min":260.865,"temp_max":261.46,"pressure":1019.95,"sea_l
evel":1041.79,"grnd_level":1019.95,"humidity":82,"temp_kf":0.59},"weather":
[{"id":600,"main":"Snow","description":"light snow","icon":"13n"}],"clouds":
{"all":68},"wind":{"speed":4.46,"deg":244.5},"snow":{"3h":0.05225},"sys":
{"pod":"n"},"dt_txt":"2017-01-31 03:00:00"},{"dt":1485842400,"main":
{"temp":260.981,"temp_min":260.981,"temp_max":260.981,"pressure":1018.96,"sea
_level":1040.84,"grnd_level":1018.96,"humidity":81,"temp_kf":0},"weather":
[{"id":600,"main":"Snow","description":"light snow","icon":"13d"}],"clouds":
{"all":80},"wind":{"speed":4.21,"deg":245.005},"snow":{"3h":0.19625},"sys":
{"pod":"d"},"dt_txt":"2017-01-31 06:00:00"},{"dt":1485853200,"main":
{"temp":262.308,"temp_min":262.308,"temp_max":262.308,"pressure":1018.1,"sea_
level":1039.77,"grnd_level":1018.1,"humidity":91,"temp_kf":0},"weather":
[{"id":600,"main":"Snow","description":"light snow","icon":"13d"}],"clouds":
{"all":88},"wind":{"speed":4.1,"deg":249.006},"snow":{"3h":0.535},"sys":
{"pod":"d"},"dt_txt":"2017-01-31 09:00:00"},{"dt":1485864000,"main":
{"temp":263.76,"temp_min":263.76,"temp_max":263.76,"pressure":1016.86,"sea_le
vel":1038.4,"grnd_level":1016.86,"humidity":87,"temp_kf":0},"weather":
[{"id":600,"main":"Snow","description":"light snow","icon":"13d"}],"clouds":
{"all":68},"wind":{"speed":3.87,"deg":254.5},"snow":{"3h":0.21},"sys":
{"pod":"d"},"dt_txt":"2017-01-31 12:00:00"},{"dt":1485874800,"main":
{"temp":264.182,"temp_min":264.182,"temp_max":264.182,"pressure":1016.19,"sea
_level":1037.77,"grnd_level":1016.19,"humidity":89,"temp_kf":0},"weather":
[{"id":600,"main":"Snow","description":"light snow","icon":"13n"}],"clouds":
{"all":76},"wind":{"speed":3.67,"deg":257.001},"snow":{"3h":0.1375},"sys":
{"pod":"n"},"dt_txt":"2017-01-31 15:00:00"},{"dt":1485885600,"main":
{"temp":264.67,"temp_min":264.67,"temp_max":264.67,"pressure":1015.32,"sea_le
vel":1036.94,"grnd_level":1015.32,"humidity":86,"temp_kf":0},"weather":
[{"id":600,"main":"Snow","description":"light snow","icon":"13n"}],"clouds":
{"all":88},"wind":{"speed":3.61,"deg":262.503},"snow":{"3h":0.1425},"sys":
{"pod":"n"},"dt_txt":"2017-01-31 18:00:00"},{"dt":1485896400,"main":
{"temp":265.436,"temp_min":265.436,"temp_max":265.436,"pressure":1014.27,"sea
```

So:

http://jsonviewer.stack.hu



```
{
  "cod": "200",
  "message": 0.0036,
  "cnt": 40,
  "list": [
    {
      "dt": 1485799200,
      "main": {
        "temp": 261.45,
        "temp_min": 259.086,
        "temp_max": 261.45,
        "pressure": 1023.48,
        "sea_level": 1045.39,
        "grnd_level": 1023.48,
        "humidity": 79,
        "temp_kf": 2.37
      },
      "weather": [
        {
          "id": 800,
          "main": "Clear",
          "description": "clear sky",
          "icon": "02n"
        }
      ],
      "clouds": {
        "all": 8
      },
      "wind": {
        "speed": 4.77,
        "deg": 232.505
      },
      "snow": {

      },
      "sys": {
        "pod": "n"
      },
      "dt_txt": "2017-01-30 18:00:00"
    },
    {
      "dt": 1485810000,
      "main": {
        "temp": 261.41,
        "temp_min": 259.638,
        "temp_max": 261.41,
        "pressure": 1022.41,
        "sea_level": 1044.35,
        "grnd_level": 1022.41,
```

In-class project - Fetching Current Weather Data – HW8 Ex3

- Write a program that downloads the weather forecast of the city of your choice for the next few days and print it as plaintext.
  - *https://openweathermap.org*
  - *Look for the API and go to How to start*
  - *You will have to sign up to receive an API key*
  - *Then, you will add that API key to your urls and you can start using the API!*
    - If you are a new start, it needs ten minutes or more time before the key can be used

Time to code – Excel-to-CSV Converter – HW6 ex 4

- Write a program that reads all the Excel files in the current working directory and outputs them as CSV files.
- A single Excel file might contain multiple sheets; you'll have to create one CSV file per sheet. The filenames of the CSV files should be <excel filename>_<sheet title>.csv
- The skeleton of the program is provided in the homework paper.