Session 3:

Strings and APIs

Andreas Bjerre-Nielsen

Agenda

1. Strings: manipulation, combination etc.
2. Containers - key based
3. Interacting with the web
4. Loading and saving files (input-output, IO)

Strings

Strings recap

What are strings? What do they consist of?

Strings are sequential containers of characters

Python characters can be:

- Unicode (UTF)
    - Characters from European and Asian language and much more
    - 16 bit information
    - Python 3 default and newer web, e.g. [møn.dk (https://møn.dk)]
- American Standard Code (`ascii`)
    - Characters from English alphabet, numbers, symbols for writing
    - 8 bit information
    - (Python 2 default, faster)

String concatenation

How can I combine strings?

Strings can be added together:

In [ ]:
```
s1 = 'police'
s2 = 'officer'
s1 + ' ' + s2

# s = '\n'

# print(s.join([s1, s2, 'arrests']))
```

String changing case

Can I alter the sentence-case of strings?

- Yes using the string methods `upper`, `lower`, `capitalize`. Example:

In [ ]:
```
s1.upper()
```

Substrings (1)

How can I check if a substring is contained in the string?

- in/not in

In [ ]:
```
'pol' not in s1
```

Substrings (2)

How can I replace a specific substring?

- replace

In [ ]:
```
s1.replace('po', 'ma')
```

Substrings (3)

Can I also access a string via indices? (in the sequence of characters)

- sequence form - slicing/indexing

In [ ]:
```
s1[2:5]
```

Strings quiz

Which Python object do strings remind you of?

- Lists work like strings.
    - Concatention (+, *) works the same way.
    - We check if element/character is contained with `in`.
    - We can slice and use indices for.

More about strings

There are many things about strings which we have not covered:

- Methods for splitting or combining strings etc.
- String formatting (http://www.python-course.eu/python3_formatted_output.php) is exceptionally useful, e.g for making URLs, printing etc.

Containers - key based

Containers recap

What are containers? Which have we seen?

-

Dictionaries (1)

How can we make a container which is accessed by arbitrary keys?

By using a dictionary, `dict` . Try executing the code below:

In [2]:
```
my_dict = {'Andreas': 'Assistant Professor in Econ and SODAS',
           'Snorre': 'PhD student in Socioligy',
           'David': 'Professor of Economics'}

print(my_dict['Snorre'])
```

```
PhD student in Socioligy
```

Dictionaries (2)

Dictionaries can also be constructed from two associated lists. These are tied together with the `zip` function. Try the following code:

In [ ]:
```
keys = ['a', 'b', 'c']
values = [1, 2]
key_value_pairs = list(zip(keys, values, ))

my_dict2 = dict(key_value_pairs)
my_dict2['a']
```

Storing containers

Does there exist a file format for easy storage of containers?

Yes, the JSON file format.

- Can store lists and dictionaries.
- Syntax is the same as Python lists and dictionaries - only add quotation marks.
    - Example: `'{"a":1,"b":1}'`

Storing containers (2)

Why is JSON so useful?

- Standard format that looks exactly like Python.
- Extreme flexibility:
    - Can hold any list or dictionary of any depth which contains only float, int, str.
    - Does not work well with other formats, but normally holds any structured data.
        - Extension to spatial data: GeoJSON

Interacting with the web

The internet as data

When we surf around the internet we are exposed to a wealth of information.

- What if we could take this and analyze it?

Well, we can. And we will.

Examples: Facebook, Twitter, Reddit, Wikipedia, Airbnb etc.

The internet as data (2)

Some times we get lucky. The data is served to us.

- The data is provided as an `API` service (today)
- The data can extracted by queries on underlying tables (Session 8)

However, often we need to do the work ourselves (Session 8, 10)

- We need to explore the structure of the webpage we are interested in
- We can extract relevant elements
    - Requires knowledge of `HTML`, possibly Javascript and RegEx to search data

The web protocol

What is `http` and where is it used?

- `http` stands for HyperText Transfer Protocol.
- `http` is good for transmitting the data when a webpage is visited:
    - the visiting client sends request for URL or object;
    - the server returns relevant data if active.

The web protocol (2)

Should we care about `http` ?

- In this course we don't care explicitly about `http`.
- We use a Python module called `requests` as a `http` interface.
- However... Some useful advice - you should **always**:
    - use the encrypted version, `https`;
    - use authenticated connection, i.e. private login, whenever possible.

Markup language (1)

What is `html` and where is it used?

- HyperText Markup Lanugage
- `html` is a language for communicating how a webpage looks like and behaves.
    - That is, `html` contains: content, design, available actions.

Markup language (2)

Should we care about `html` ?

- Yes, `html` is often where the interesting data can be found.
- Sometimes, we are lucky, and instead of `html` we get a JSON in return.
- Getting data from `html` will the topic of the subsequent scraping sessions (session 8,10).

Web APIs (1)

So when do we get lucky, i.e. when is `html` not important?

- When we get an Application Protocol Interface, i.e. `API`
- What does this mean?
    - We send a query to the Web API
    - We get a response from the Web API with data back in return, typically as JSON.

Web APIs (2)

So where is the API?

- Usually on separate sub-domain, e.g. `api.github.com`

So how do we know how the API works?

- There usually is some documentation. E.g. google "api github com" (https://www.google.com/search?q=api+github)

Web APIs (3)

So is data free? As in free lunch?

- Most commercial APIs require authentication and have limited free usage
    - e.g. Google Maps, various weather services
    - public Danish APIs:
        - Danish statistics (DST)
        - Danish weather data (DMI, this fall)
        - Danish spatial data (DAWA, danish addresses)
    - global free APIs:
        - OpenStreetMaps, Wikipedia
- If no authentication is required the API may be delimited.
    - This means only a certain number of requests can be handled per second or per hour from a given IP address.

Web APIs (4)

So how do make the URLs?

- An `API` query is a URL consisting of:
    - Server URL, e.g. `https://api.github.com`
    - Endpoint path, `/users/abjer/repos`
    - Query parameters,

Web APIs in Python (1)

How do make a simple query?

In [ ]:
```
server_url = 'https://api.github.com/'
endpoint_path = 'users/abjer/repos'
url = server_url + endpoint_path
print(url)
```

Web APIs in Python (2)

How can we send a query with the `requests` module?

In [ ]:
```
import requests # import the module

response = requests.get(url) # submit query with `get` and save response
response.ok
```

Web APIs in Python (3)

How do extract something from the response?

In [ ]:
```
print(len(response.text))
print(response.text[:500])
```

Web APIs in Python (4)

Can we get something more meaningful or structured?

In [ ]:
```
response_json = response.json()
response_json
```

Web APIs in Python (5)

And how can we see it even more clearly?

In [ ]:
```
import pprint
pprint.pprint(response.json())
```

Loading and saving files

How to do input-output (IO) operations in Python

Text files

How can we save a string as a text file?

In [ ]:
```
my_str = '\nThis is important...'

with open('my_file.txt', 'a') as f:
    f.write(my_str)
```

How can we load a string from a text file?

In [ ]:
```
with open('my_file.txt', 'r') as f:
    my_str_load = f.read()
print(my_str_load)
```

JSON files

How can we save a JSON file?

The trick is to convert the JSON file to a string. This can be done with `dumps` in the module `json` :

In [ ]:
```
import json

with open('my_file.json', 'w') as f:
    response_json_str = json.dumps(response_json)
    f.write(response_json_str)
```

We can convert a string to JSON with `loads` .

File handling

How can we remove a file?

The module `os` can do a lot of file handling tasks, e.g. removing files:

In [ ]:
```
import os

os.remove('my_file.json')
```

The end

Return to agenda