SI 507 Lab #5

September 27

Today's Plan

Lab #5

- Classes, inheritance, super()
- APIs and JSON
- Homework time

Classes

Create a class using the keyword `class`

All classes have a function called `__init__()` that is executed **automatically** when the class is being initiated. Use this to assign values to object properties.

Other functions that belong to the class are called “methods”.

The `self` parameter is a reference to the current instance of the class. It is used to access the variables that belong to the class.

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def myfunc(self):
    print("Hello my name is " + self.name)

p1 = Person("John", 36)
p1.myfunc()
```

Classes

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def myfunc(self):
    print("Hello my name is " + self.name)

p1 = Person("John", 36)
p1.myfunc()
```

Creates an object of class `Person`, called `p1`

Executes the `myfunc()` method on the object `p1`

Class Inheritance

Inheritance allows us to define a class that inherits all the methods and properties from another class.

1. Navigate to the link - https://bit.ly/3fncvWK
2. Let's sort properties first!
3. Refer to the reading on class inheritance, if needed
4. Create a superclass called "Media"
5. Create *at least* one subclass for either:
    a. Movies
    b. Songs
    c. Albums
6. Navigate to a blank page in the Jamboard and share your code!
    a. Keep clicking on the right arrow to add more pages

Possible Solutions

Songs	Albums	Movies
Title	Title	Title
Artist	Artist	Director
Length	Length	Length
Year	Year	Year
Content Warning	Content Warning	Rating
	Number of Tracks	

Possible Solutions

```
2   class Media:
3       def __init__(self, title, year, length):
4           self.title = title
5           self.year = year
6           self.mins = length
7
8       def string(self):
9           print(f'{self.title} was released in {self.year}.')
10
11  jp = Media('Jurassic Park', 1993, 77)
12  jp.string()
13
```

```
Jurassic Park was released in 1993.
```

Possible Solutions

```
14  class Song(Media):
15      def __init__(self, title, year, artist, album, content_warning="None", length="None"):
16          super().__init__(title, year, length)
17          self.artist = artist
18          self.album = album
19          self.cw = content_warning
20
21  jt = Song('Cry Me A River', 2002, 'Justin Timberlake', 'Justified')
22  jt.string()
23  print(jt.album)
24  print(jt.cw)
25
```

```
Cry Me A River was released in 2002.
Justified
None
```

Possible Solutions

```
26  class Song(Media):
27      def __init__(self, title, year, artist, album, content_warning="None", length="None"):
28          super().__init__(title, year, length)
29          self.artist = artist
30          self.album = album
31          self.cw = content_warning
32
33      def string(self):
34          print(f'{self.title} is a track on the album "{self.album}" by {self.artist}.')
35
36  jt = Song('Cry Me A River', 2002, 'Justin Timberlake', 'Justified')
37  jt.string()
38  print(jt.album)
39
```

```
Cry Me A River is a track on the album "Justified" by Justin Timberlake.
Justified
```

What is an API?

- **A**pplication **P**rogramming **I**nterface
- Acts as a communication layer that allows different systems to talk to each other without having to understand exactly what each other does.
- How they work:
  - You make a **request** for information
  - The API returns a **response** with what you requested

json.loads()

Parses a JSON string and returns a dictionary:

```
import json

person = '{"name": "Bob", "languages": ["English", "Fench"]}'
person_dict = json.loads(person)

# Output: {'name': 'Bob', 'languages': ['English', 'Fench']}
print( person_dict)

# Output: ['English', 'French']
print(person_dict['languages'])
```

json.load()

```
{"name": "Bob",
"languages": ["English", "Fench"]
}
```

person.json

Reads a file containing a JSON object:

```
import json

with open('path_to_file/person.json') as f:
  data = json.load(f)

# Output: {'name': 'Bob', 'languages': ['English', 'Fench']}
print(data)
```

json.dumps()

Converts a dictionary to a JSON string:

```
import json

person_dict = {'name': 'Bob',
'age': 12,
'children': None
}
person_json = json.dumps(person_dict)

# Output: {"name": "Bob", "age": 12, "children": null}
print(person_json)
```

json.dump()

Converts a dictionary to a JSON file:

```
import json

person_dict = {"name": "Bob",
"languages": ["English", "Fench"],
"married": True,
"age": 32
}

with open('person.txt', 'w') as json_file:
  json.dump(person_dict, json_file)
```

person.txt

```
{"name": "Bob", "languages": ["English", "Fench"], "married": true, "age": 32}
```

JSON Practice

Open Colab Notebook link and make a copy for yourself to complete:

- https://bit.ly/3BOWaBL

Homework

HW2 (Schelling): Due 10/2

HW3 (Unittesting) : Due 10/10

Project 1 (iTunes): Due 10/17

Sources

https://www.w3schools.com/python/python_classes.asp

https://www.w3schools.com/python/python_inheritance.asp

https://realpython.com/python-api/#getting-to-know-apis

https://pynative.com/python-json-exercise/