Working with Json files

**JSON** - JavaScript Object Notation

* JSON is a format for structuring data.
* It is mainly used for storing and transferring data between the browser and the server.
* Why JSON?
  + The fact that whenever we declare a variable and assign a value to it, it’s not the variable that holds the value but rather the variable just holds an address in the memory where the initialized value is stored. Further explaining, take for example:

let age=21;

when we use age, it gets replaced with 21, but that does not mean that age contains 21, rather what it means is that the variable age contains the address of the memory location where 21 is stored.

* + you might think what is the problem, how is JSON helpful?

well, yes, you are right! it is fine here till now but imagine you have to transfer the data and use it somewhere else (like an API maybe), so how will we share this? One way could be to send your computers entire memory along with the address of the locations that is required, as you might have understood now that this is not at all a good way to do it, also it is risky to send your entire computer memory. Here comes JSON to the rescue, JSON serializes the data and converts it into human-readable and understandable format, which also makes it transferal and to be able to communicate.

* Characteristics of JSON
  + It is **Human-readable and writable**.
  + It is **light weight text-based data interchange format** which means, it is simpler to read and write when compared to XML.
  + It is widely used as **data storage and communication** format on the web.
  + Though it is derived from a subset of JavaScript, yet it is **Language independent**. Thus, the code for generating and parsing JSON data can be written in any other programming language.

**Functions**

* **json.dump(obj, fileObj)**: Serializes *obj* as a JSON formatted stream to *fileObj*.
* **json.dumps(obj)** : Serializes *obj* as JSON formatted string.
* **json.load(JSONfile)**: De-serializes *JSONfile* to a Python object.
* **json.loads(JSONfile)** : De-serializes *JSONfile*(type: string) to a Python object.

**Classes**

* **JSONEncoder:** An encoder class to convert Python objects to JSON format.
* **JSONDecoder:** A decoder class to convert JSON format file into Python obj.
* Convert CSV to JSON using Python

CSV (or Comma Separated Value) files represent data in a tabular format, with several rows and columns.

JSON (or JavaScript Object Notation) is a dictionary-like notation.

#csv->json

import csv

import json

def csv\_to\_json(csvFilePath):

jsonArray = []

#read csv file

with open(csvFilePath, encoding='utf-8') as csvf:

#load csv file data using csv library's dictionary reader

csvReader = csv.DictReader(csvf)

#convert each csv row into python dict

for row in csvReader:

#add this python dict to json array

jsonArray.append(row)

#convert python jsonArray to JSON String and write to file

with open(r"D:\UPES\Project\Project Auto\_Dub\Transcribe\all\_words.json", 'w', encoding='utf-8') as jsonf:

jsonString = json.dumps(jsonArray, indent=4)

jsonf.write(jsonString)

csvFilePath = r"D:\UPES\Project\Project Auto\_Dub\Transcribe\all\_words.csv"

csv\_to\_json(csvFilePath)