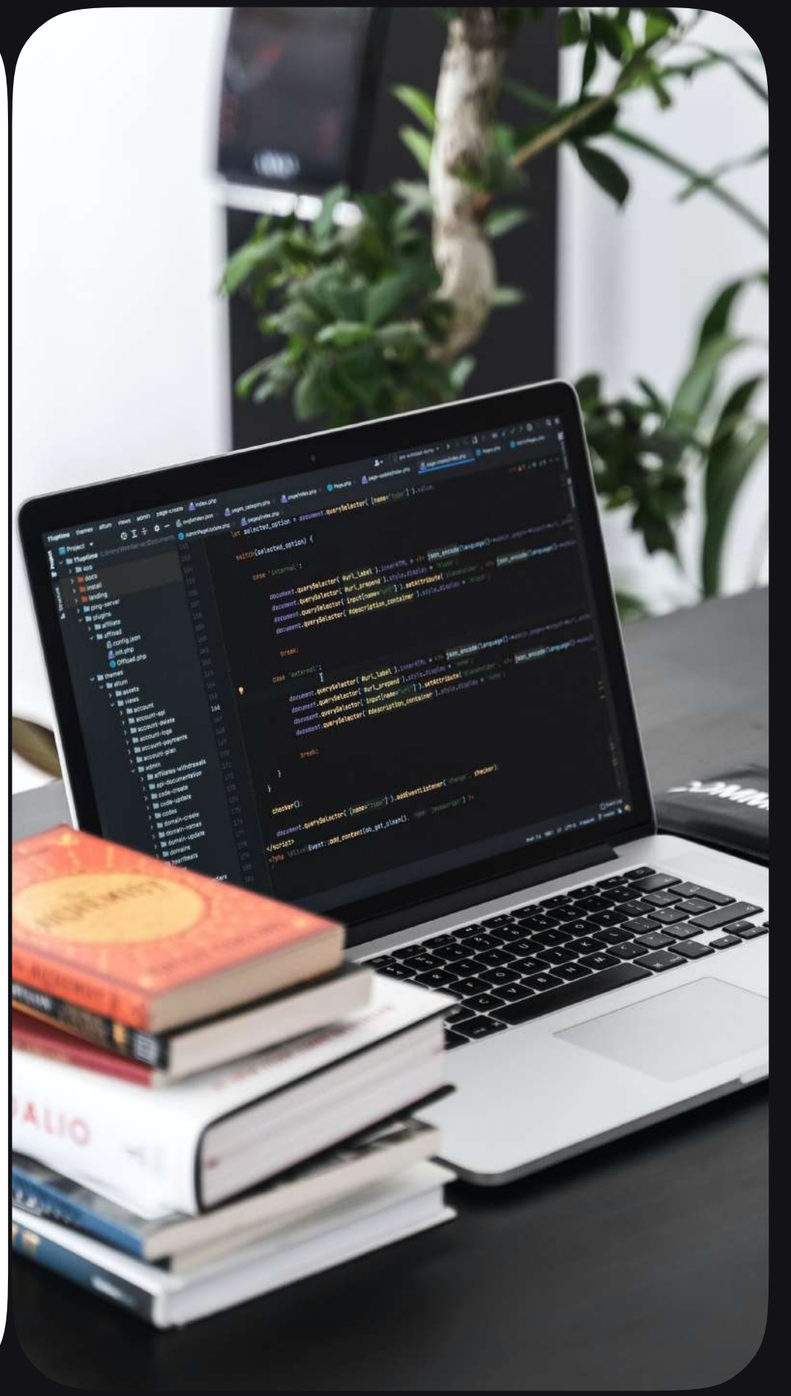
Модуль 3 Занятие 3

Работа с форматами JSON и CSV



JSON

Чтение JSON

Запись JSON

CSV

Чтение CSV

Запись CSV

JSON

JSON (JavaScript Object Notation) — стандартный текстовый формат обмена данными. JSON применяется для обмена данными в интернете между веб-приложениями. Изначально был основан на JavaScript, но сейчас считается независимым от языка и может использоваться с любым языком программирования.

JSON по структуре похож на словарь в Python, но более стандартизирован.

https://swapi.dev/api/people/1/?format=json


```
{
    "name": "Luke Skywalker",
    "height": "172",
    "mass": "77",
    "hair_color": "blond",
    "skin_color": "fair",
    "eye_color": "blue",
    "birth_year": "19BBY",
    "gender": "male",
    "homeworld": "https://swapi.dev/api/planets/1/",
    "films": [
            "https://swapi.dev/api/films/1/",
            "https://swapi.dev/api/films/2/",
            "https://swapi.dev/api/films/3/",
            "https://swapi.dev/api/films/6/"
    ],
    "species": [],
    "vehicles": [
            "https://swapi.dev/api/vehicles/14/",
            "https://swapi.dev/api/vehicles/30/"
    ],
    "starships": [
            "https://swapi.dev/api/starships/12/",
            "https://swapi.dev/api/starships/22/"
    ],
    "created": "2014-12-09T13:50:51.644000Z",
    "edited": "2014-12-20T21:17:56.891000Z",
    "url": "https://swapi.dev/api/people/1/"
}
```

Чтение JSON

Для работы с файлами JSON существует стандартный модуль json. Он позволяет преобразовать данные из JSON файла в программе в объекты языка Python, а также способен выполнять обратную операцию для записи Python объектов в JSON файл.

Для чтения в модуле json есть два метода:

- **load(file)** — метод принимает в качестве обязательного аргумента файловый объект в формате JSON и возвращает объекты Python
- **loads(s)** — метод принимает в качестве обязательного аргумента строку в формате JSON и возвращает объекты Python

Пример

```python
import json

with open("data.json", encoding="UTF-8") as file:
    data = json.load(file)
print(type(data))
print((data))
```

Вывод:

```
<class 'dict'>
{'name': 'Luke Skywalker', 'height': '172', 'films':
['https://swapi.dev/api/films/1/', 'https://swapi.dev/api/films/2/',
'https://swapi.dev/api/films/3/', 'https://swapi.dev/api/films/6/'],
'starships': ['https://swapi.dev/api/starships/12/',
'https://swapi.dev/api/starships/22/'], 'created':
'2014-12-09T13:50:51.644000Z', 'edited':
'2014-12-20T21:17:56.891000Z',
'url': 'https://swapi.dev/api/people/1/'}
```

Считываем файл в формате JSON с помощью метода load().

data.json — Блокнот

Файл Правка Формат Вид Справка

```
{
  "name": "Luke Skywalker",
  "height": "172",
  "films": [
    "https://swapi.dev/api/films/1/",
    "https://swapi.dev/api/films/2/",
    "https://swapi.dev/api/films/3/",
    "https://swapi.dev/api/films/6/"
  ],
  "starships": [
    "https://swapi.dev/api/starships/12/",
    "https://swapi.dev/api/starships/22/"
  ],
  "created": "2014-12-09T13:50:51.644000Z",
  "edited": "2014-12-20T21:17:56.891000Z",
  "url": "https://swapi.dev/api/people/1/"
}
```

Пример

```
import json

s = '''{
  "name": "Luke Skywalker",
  "height": "172",
  "films": [
    "https://swapi.dev/api/films/1/",
    "https://swapi.dev/api/films/2/",
    "https://swapi.dev/api/films/3/",
    "https://swapi.dev/api/films/6/"
  ],
  "starships": [
    "https://swapi.dev/api/starships/12/",
    "https://swapi.dev/api/starships/22/"
  ],
  "created": "2014-12-09T13:50:51.644000Z",
  "edited": "2014-12-20T21:17:56.891000Z",
  "url": "https://swapi.dev/api/people/1/"
}'''

data = json.loads(s)
print(type(data))
print((data))
```

Считываем строку в формате JSON с помощью метода loads().

Вывод:

```
<class 'dict'>
{'name': 'Luke Skywalker', 'height': '172',
'films': ['https://swapi.dev/api/films/1/',
'https://swapi.dev/api/films/2/',
'https://swapi.dev/api/films/3/',
'https://swapi.dev/api/films/6/'],
'starships':
['https://swapi.dev/api/starships/12/',
'https://swapi.dev/api/starships/22/'],
'created': '2014-12-09T13:50:51.644000Z',
'edited': '2014-12-20T21:17:56.891000Z',
'url': 'https://swapi.dev/api/people/1/'}
```

Запись JSON

Для записи данных в формате JSON в модуле json есть также два метода:

- **json.dump**(obj, file) — метод принимает в качестве обязательных аргументов Python объект и файловый объект и сохраняет объект Python в файл в формате JSON.
- **json.dumps**(obj) — метод принимает в качестве обязательного аргумента Python объект и возвращает строку в формате JSON.

Также методы могут принимать некоторые полезные необязательные аргументы:

- **indent** (по умолчанию None) — « задает отступ. Можно задать строку вместо None, и эта строка будет использоваться в качестве отступа. Или можно задать целое число, тогда отступ будет состоять из такого количества пробелов.
- **sort_keys** (по умолчанию False) — сортирует ключи при записи.
- **ensure_ascii** (по умолчанию True) — заменяет все не-ASCII-символы юникод последовательностями в формате \uXXXX. При использовании русских символов необходимо передать в аргумент значение False.

Пример

```python
import json

films = [
  'Империя наносит ответный удар',
  'Возвращение джедая',
  'Скрытая угроза',
  'Атака клонов',
  'Месть ситхов'
]
yoda = {
  'name': 'Йода',
  'height': '66',
  'mass': '17',
  'hair_color': 'белый',
  'skin_color': 'зеленый',
  'films': films
}

with open('data.json', 'w', encoding='utf-8') as file:
  json.dump(yoda, file, ensure_ascii=False, indent=2)
```

Сохраняем объект Python в файл в формате JSON с помощью метода dump().

data.json — Блокнот

Файл Правка Формат Вид Справка

```json
{
  "name": "Йода",
  "height": "66",
  "mass": "17",
  "hair_color": "белый",
  "skin_color": "зеленый",
  "films": [
    "Империя наносит ответный удар",
    "Возвращение джедая",
    "Скрытая угроза",
    "Атака клонов",
    "Месть ситхов"
  ]
}
```

Пример

```python
import json

films = [
  'Империя наносит ответный удар',
  'Возвращение джедая',
  'Скрытая угроза',
  'Атака клонов',
  'Месть ситхов'
]
yoda = {
  'name': 'Йода',
  'height': '66',
  'mass': '17',
  'hair_color': 'белый',
  'skin_color': 'зеленый',
  'films': films
}

result = json.dumps(yoda, ensure_ascii=False, indent=2)
print(type(result))
print(result)
```

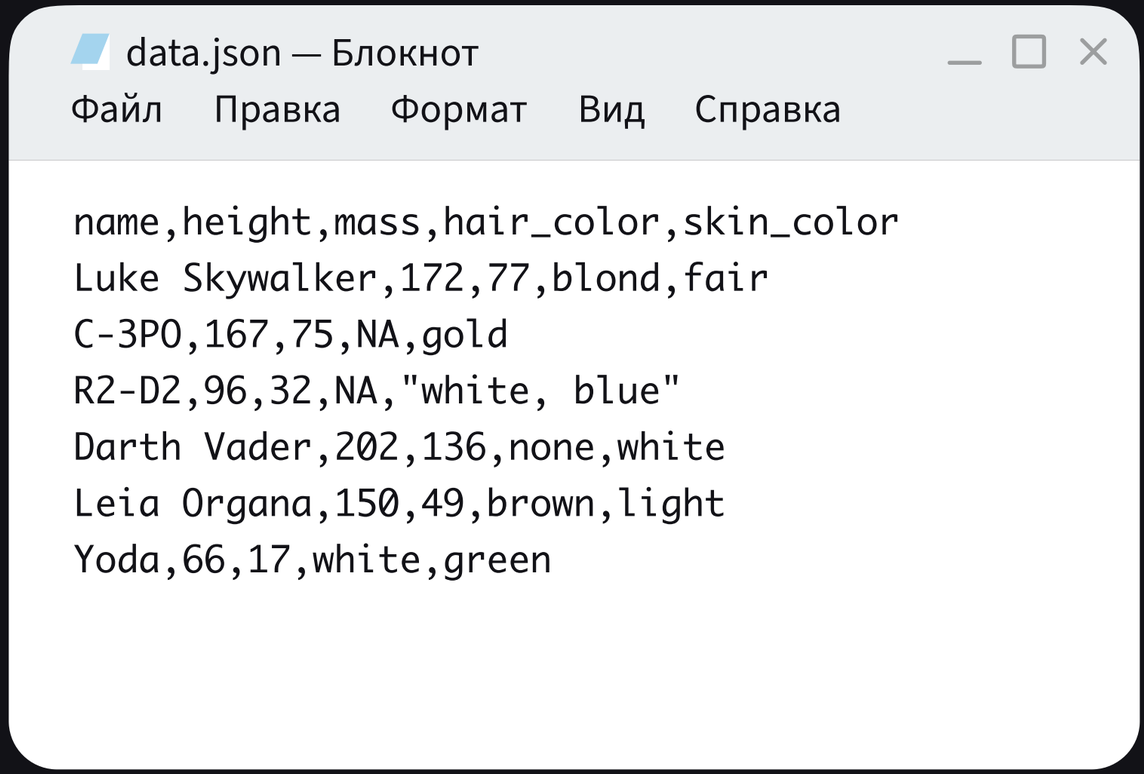
Получаем строку в формате JSON из Python объекта с помощью метода dumps().

Вывод:

```
{
 "name": "Йода",
 "height": "66",
 "mass": "17",
 "hair_color": "белый",
 "skin_color": "зеленый",
 "films": [
   "Империя наносит ответный удар",
   "Возвращение джедая",
   "Скрытая угроза",
   "Атака клонов",
   "Месть ситхов"
 ]
}
```

CSV

CSV (comma-separated value) — текстовый формат, предназначенный для представления табличных данных. Строка таблицы соответствует строке текста, которая содержит одно или несколько полей, разделенных запятыми или другими разделителями.



Чтение CSV

Для работы с файлами CSV есть модуль csv, который позволяет работать с файлами в CSV формате. Для того чтобы выполнить чтение CSV файла, можно воспользоваться методом reader() или классом DictReader().

- **reader**(file) — метод принимает в качестве обязательного аргумента файловый объект и возвращает объект итератор reader.

Также можно указать необязательные аргументы:

- **delimiter** (по умолчанию запятая ',') — символ разделитель.
- **quotechar** (по умолчанию двойная кавычка '"') — символ, используемый для окружения полей, содержащих символ разделитель.

Пример

```
import csv

with open('data.csv', newline='') as file:
    data = csv.reader(file)
    print(type(data))
    for line in data:
        print(line)
```

Вывод:

```
<class '_csv.reader'>
['name', 'height', 'mass', 'hair_color', 'skin_color']
['Luke Skywalker', '172', '77', 'blond', 'fair']
['C-3PO', '167', '75', 'NA', 'gold']
['R2-D2', '96', '32', 'NA', 'white, blue']
['Darth Vader', '202', '136', 'none', 'white']
['Leia Organa', '150', '49', 'brown', 'light']
['Yoda', '66', '17', 'white', 'green']
```

Выполним чтение файла
с помощью метода reader().

data.json — Блокнот

Файл Правка Формат Вид Справка

```
name,height,mass,hair_color,skin_color
Luke Skywalker,172,77,blond,fair
C-3PO,167,75,NA,gold
R2-D2,96,32,NA,"white, blue"
Darth Vader,202,136,none,white
Leia Organa,150,49,brown,light
Yoda,66,17,white,green
```

Чтение CSV

Модуль csv позволяет работать с файлом как со словарем с помощью класса DictReader():

DictReader() — класс принимает в качестве обязательного аргумента файловый объект и возвращает такой же объект reader, но позволяет работать с ним как со словарем, с ключами, указанными в первой строке CSV файла.

Пример

```python
import csv

with open('data.csv', newline='') as file:
    data = csv.DictReader(file)
    print(data.fieldnames)
    for line in data:
        print(line['name'], line['height'])
```

Выполним чтение файла с помощью класса DictReader(). В качестве ключей были использованы записи из первой строки. Атрибут fieldnames содержит список ключей.

Вывод:

```
['name', 'height', 'mass', 'hair_color', 'skin_color']
Luke Skywalker 172
C-3PO 167
R2-D2 96
Darth Vader 202
Leia Organa 150
Yoda 66
```

data.json — Блокнот

Файл Правка Формат Вид Справка

```
name,height,mass,hair_color,skin_color
Luke Skywalker,172,77,blond,fair
C-3PO,167,75,NA,gold
R2-D2,96,32,NA,"white, blue"
Darth Vader,202,136,none,white
Leia Organa,150,49,brown,light
Yoda,66,17,white,green
```

Пример

```python
import csv

with open('data.csv', newline='') as file:
    names = ['name', 'height', 'mass', 'hair_color', 'skin_color']
    data = csv.DictReader(file, fieldnames=names)
    print(data.fieldnames)
    for line in data:
        print(line['name'], line['height'])
```

Если в CSV файле нет первой строки с названиями столбцов, то при создании DictReader(), необходимо передать необязательный параметр fieldnames со списком ключей.

Вывод:

```
['name', 'height', 'mass', 'hair_color', 'skin_color']
Luke Skywalker 172
C-3PO 167
R2-D2 96
Darth Vader 202
Leia Organa 150
Yoda 66
```

data.json — Блокнот

Файл Правка Формат Вид Справка

```
Luke Skywalker,172,77,blond,fair
C-3PO,167,75,NA,gold
R2-D2,96,32,NA,"white, blue"
Darth Vader,202,136,none,white
Leia Organa,150,49,brown,light
Yoda,66,17,white,green
```

Запись CSV

Для записи файлов в формат CSV можно воспользоваться методами writer() или классом DictWriter().

- **writer**(file) — метод принимает в качестве обязательного аргумента файловый объект и возвращает объект записи, который имеет методы:
writerow() — позволяет добавить одну запись (список строк)
и writerows() — позволяет добавить несколько строк (двумерный список строк).

Также метод writer() может принимать необязательные аргументы:

- **delimiter** (по умолчанию запятая ',') — символ разделитель.
- **quotechar** (по умолчанию двойная кавычка '"') — символ, используемый для окружения полей, содержащих символ разделитель.

Пример

```python
import csv

names = ['name', 'height', 'mass', 'hair_color', 'skin_color']
rows = [
    ['Luke Skywalker', '172', '77', 'blond', 'fair'],
    ['C-3PO', '167', '75', 'NA', 'gold'],
    ['R2-D2', '96', '32', 'NA', 'white, blue'],
    ['Darth Vader', '202', '136', 'none', 'white'],
    ['Leia Organa', '150', '49', 'brown', 'light'],
    ['Yoda', '66', '17', 'white', 'green']
]

with open('data.csv', 'w', newline='') as file:
    data = csv.writer(file)
    data.writerow(names)
    data.writerows(rows)
```

Выполним запись с помощью методов writerow() и writerows().

data.json — Блокнот

Файл Правка Формат Вид Справка

```
name,height,mass,hair_color,skin_color
Luke Skywalker,172,77,blond,fair
C-3PO,167,75,NA,gold
R2-D2,96,32,NA,"white, blue"
Darth Vader,202,136,none,white
Leia Organa,150,49,brown,light
Yoda,66,17,white,green
```

Запись CSV

Если можно считать CSV файл в словарь, то и записать файл можно из словаря с помощью класса DictWriter(). В отличие от DictReader(), параметр fieldnames при записи является обязательным. Для записи первой строки в виде имен столбцов используется метод writeheader().

Пример

```python
import csv

names = ['name', 'height', 'mass']
rows = [
    {'name': 'Luke Skywalker', 'height': '172', 'mass': '77'},
    {'name': 'C-3PO', 'height': '167', 'mass': '75'},
    {'name': 'R2-D2', 'height': '96', 'mass': '32'},
    {'name': 'Darth Vader', 'height': '202', 'mass': '136'},
    {'name': 'Leia Organa', 'height': '150', 'mass': '49'},
    {'name': 'Yoda', 'height': '66', 'mass': '17'}
]

with open('data_out.csv', 'w', newline='') as file:
    data = csv.DictWriter(file, fieldnames=names)
    data.writeheader()
    data.writerows(rows)
```

Выполним запись словаря с помощью класса DictWriter().

data.json — Блокнот

Файл Правка Формат Вид Справка

```
name,height,mass
Luke Skywalker,172,77
C-3PO,167,75
R2-D2,96,32
Darth Vader,202,136
Leia Organa,150,49
Yoda,66,17
```

Итоги

- **JSON** — стандартный текстовый формат обмена данными
- **CSV** — текстовый формат, предназначенный для представления табличных данных
- Для работы с файлами JSON существует стандартный модуль json, который позволяет преобразовать данные из JSON файла в объекты языка Python, а также способен выполнять обратную операцию для записи Python объектов в JSON файл
- Для работы с файлами CSV есть модуль csv, который позволяет работать с файлами в CSV формате