Консервация обьектов в Python

Практикум на ЭВМ 2017/2018

Полетаев Всеволод Андреевич

МГУ имени М. В. Ломоносова, факультет ВМК, кафедра ММП

21 ноября 2017 г.

Что это?

- **Консервация (serializing)** — генерация последовательности байт, описывающей структуру объекта

- **Расконсервация (deserializing)** — восстановление объекта из последовательности байт, полученной при консервации

Зачем?

- Долговременное хранение на диске
- Использование и обработка вне Python
- Передача по сети

Классификация форматов

- Бинарные
  - Pickle
  - Protobuf
- Текстовые
  - JSON
  - YAML
  - XML

Pickle: использование

```
>>> import numpy as np
>>> import pickle
>>> a = np.arange(-5, 5)
>>> with open('dump.pickle', 'wb') as file:
...     pickle.dump(a, file)
...
>>> a = None
>>> with open('dump.pickle', 'rb') as file:
...     a = pickle.load(file)
...
>>> a
array([-5, -4, -3, -2, -1,  0,  1,  2,  3,  4])
```

Существует возможность сериализации в строку.

Полное описание интерфейса `https://docs.python.org/3/library/pickle.html#module-interface`

С чем может работать Pickle?

Pickle может работать с:

- None, True, False
- Любые числа
- Встроенные структуры данных
- Байтовые массивы и строки
- Функции, определенные в корне модуля (не `lambda`)
- Встроенные функции
- Экземпляры классов, чьи `__dict__` или `__getstate__()` возвращают сериализуемый результат

Pickle не может работать с:

- Кодом функций
- Метаинформацией и атрибутами самих классов
- Объектами, содержащими в себе ресурсы

Pickle: плюсы и минусы

+ Прост в базовом использовании
+ Может работать с большим количеством объектов из коробки
+ Достаточно просто может подружится с вашим классом, если он особенный

– При использовании с более старыми версиями Python нужно следить за совместимостью
– Плохо подходит для хранения объектов в развивающемся проекте
– Доступен только в Python
– Небезопасен при использовании плохих данных

JSON: пример использования

```
>>> import json
>>> article = {
...   'title': 'Text summarization using Latent ...',
...   'doi': ' 10.1177/0165551511408848',
...   'authors': [
...     'Makbule Gulcin Ozsoy', 'Ferda Nur Alpaslan',
    'Ilyas Cicekli']
... }
>>> s = json.dumps(article)
>>> s
'{"title": "Text summarization using Latent Semantic
    Analysis", "doi": " 10.1177/0165551511408848", "
    authors": ["Makbule Gulcin Ozsoy", "Ferda Nur
    Alpaslan", "Ilyas Cicekli"]}'
>>> article = None
>>> article = json.loads(s)
>>> article['title']
'Text summarization using Latent Semantic Analysis'
```

Интерфейс

docs.python.org/3/library//json.html#basic-usage

С чем может работать JSON?

- None, True, False
- Словари, списки
- Строки
- Вещественные и целые числа

JSON: плюсы и минусы

+ Прост в использовании
+ Очень распространен, за редким исключением есть библиотеки для всех ЯП
+ Можно научить работать со своими классами
+ Не зависит от версии Python и вашего кода
+ Текстовый, можно редактировать руками

– Менее компактен, чем Pickle
– Ограниченный список того, с чем может работать