python json 模块

简介

JSON(JavaScript Object Notation, JS 对象标记) 是一种轻量级的数据交换格式。

任何支持的类型都可以通过 JSON 来表示，例如字符串、数字、对象、数组等。但是对象和数组是比较特殊且常用的两种类型：

- 对象表示为键值对
- 数据由逗号分隔
- 花括号保存对象
- 方括号保存数组

在 python 中，有专门处理 json 格式的模块—— json 和 picle 模块

Json 模块提供了四个方法： dumps、dump、loads、load

pickle 模块也提供了四个功能：dumps、dump、loads、load

dumps 和 dump

dumps 和 dump 序列化方法

dumps 只完成了序列化为 str，

dump 必须传文件描述符，将序列化的 str 保存到文件中

```
import json
print(json.dumps([]))# dumps 可以格式化所有的基本数据类型为字符串
print(json.dumps(1))# 数字
print(json.dumps('1'))    # 字符串
dict = {"name":"Tom", "age":23}
```

```
print(json.dumps(dict))        # 字典


a = {"name":"Tom", "age":23}
with open("test.json", "w", encoding='utf-8') as f:
    # indent 超级好用，格式化保存字典，默认为 None，小于 0 为零个空格
    f.write(json.dumps(a, indent=4))
    # json.dump(a,f,indent=4)     # 和上面的效果一样
```

保存的文件效果：

```
{
    "name": "Tom",
    "age": 23
}
```

loads 和 load

loads 和 load 反序列化方法

loads 只完成了反序列化，

load 接收文件描述符，完成了读取文件和反序列化

```
cc = json.loads('{"name":"Tom", "age":23}')
print(cc['name'])
with open("test.json", "r", encoding='utf-8') as f:
    aa = json.loads(f.read())
    f.seek(0)
    bb = json.load(f)       # 与 json.loads(f.read())
print(aa)
print(bb)
```

json 和 picle 模块

json 模块和 picle 模块都有 dumps、dump、loads、load 四种方法，而且用法一样。

不用的是 json 模块序列化出来的是通用格式，其它编程语言都认识，就是普通的字符串，而 picle 模块序列化出来的只有 python 可以认识，其他编程语言不认识的，表现为乱码

不过 picle 可以序列化函数，但是其他文件想用该函数，在该文件中需要有该文件的定义

python 对象（obj） 与 json 对象的对应关系

Python	JSON
dict	object
list, tuple	array
str	string
int, float	number
True	true
False	false
None	null

总结

1. json 序列化方法：

    dumps：无文件操作　　dump：序列化+写入文件

2. json 反序列化方法：

    loads：无文件操作　　load： 读文件+反序列化

3. json 模块序列化的数据 更通用

picle 模块序列化的数据 仅 python 可用，但功能强大，可以序列号函数

4. json 模块可以序列化和反序列化的 数据类型 见 python 对象（obj） 与 json 对象的对应关系表

5. 格式化写入文件利用 indent = 4