# Python JSON

本章节我们将为大家介绍如何使用 Python 语言来编码和解码 JSON 对象。

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式，易于人阅读和编写。

## JSON 函数

使用 JSON 函数需要导入 json 库：**import json**。

|  |  |
| --- | --- |
| **函数** | **描述** |
| json.dumps | 将 Python 对象编码成 JSON 字符串 |
| json.loads | 将已编码的 JSON 字符串解码为 Python 对象 |

## json.dumps

json.dumps 用于将 Python 对象编码成 JSON 字符串。

### **语法**

json.dumps(obj, skipkeys=False, ensure\_ascii=True, check\_circular=True, allow\_nan=True, cls=None, indent=None, separators=None, encoding="utf-8", default=None, sort\_keys=False, \*\*kw)

### **实例**

以下实例将数组编码为 JSON 格式数据：

#!/usr/bin/pythonimport json

data = [ { 'a' : 1, 'b' : 2, 'c' : 3, 'd' : 4, 'e' : 5 } ]

json = json.dumps(data)

print (json)

以上代码执行结果为：

[{"a": 1, "c": 3, "b": 2, "e": 5, "d": 4}]

使用参数让 JSON 数据格式化输出：

>>> import json

>>> print json.dumps({'a': 'Runoob', 'b': 7}, sort\_keys=True, indent=4, separators=(',', ': ')){

"a": "Runoob",

"b": 7}

python 原始类型向 json 类型的转化对照表：

| **Python** | **JSON** |
| --- | --- |
| dict | object |
| list, tuple | array |
| str, unicode | string |
| int, long, float | number |
| True | true |
| False | false |
| None | null |

## json.loads

json.loads 用于解码 JSON 数据。该函数返回 Python 字段的数据类型。

### **语法**

json.loads(s[, encoding[, cls[, object\_hook[, parse\_float[, parse\_int[, parse\_constant[, object\_pairs\_hook[, \*\*kw]]]]]]]])

### **实例**

以下实例展示了Python 如何解码 JSON 对象：

#!/usr/bin/pythonimport json

jsonData = '{"a":1,"b":2,"c":3,"d":4,"e":5}';

text = json.loads(jsonData)print text

以上代码执行结果为：

{u'a': 1, u'c': 3, u'b': 2, u'e': 5, u'd': 4}

json 类型转换到 python 的类型对照表：

| **JSON** | **Python** |
| --- | --- |
| object | dict |
| array | list |
| string | unicode |
| number (int) | int, long |
| number (real) | float |
| true | True |
| false | False |
| null | None |

## Python 简要使用



Dump

