Python JSON

JSON is a syntax for storing and exchanging data.

JSON is text, written with JavaScript object notation.

Python has a built-in package called json, which can be used to work with JSON data.

import json  
  
# some JSON:  
x =  '{ "name":"John", "age":30, "city":"New York"}'  
  
# parse x:  
y = json.loads(x)  
  
# the result is a Python dictionary:  
print(y["age"])

## **Convert from Python to JSON**

If you have a Python object, you can convert it into a JSON string by using the json.dumps() method.

import json  
  
# a Python object (dict):  
x = {  
  "name": "John",  
  "age": 30,  
  "city": "New York"  
}  
  
# convert into JSON:  
y = json.dumps(x)  
  
# the result is a JSON string:  
print(y)

You can convert Python objects of the following types, into JSON strings:

* dict
* list
* tuple
* string
* int
* float
* True
* False
* None

### **Example**

import json  
  
print(json.dumps({"name": "John", "age": 30}))  
print(json.dumps(["apple", "bananas"]))  
print(json.dumps(("apple", "bananas")))  
print(json.dumps("hello"))  
print(json.dumps(42))  
print(json.dumps(31.76))  
print(json.dumps(True))  
print(json.dumps(False))  
print(json.dumps(None))

### **Example**

Convert a Python object containing all the legal data types:

import json  
  
x = {  
  "name": "John",  
  "age": 30,  
  "married": True,  
  "divorced": False,  
  "children": ("Ann","Billy"),  
  "pets": None,  
  "cars": [  
    {"model": "BMW 230", "mpg": 27.5},  
    {"model": "Ford Edge", "mpg": 24.1}  
  ]  
}  
  
print(json.dumps(x))

## **Format the Result**

The example above prints a JSON string, but it is not very easy to read, with no indentations and line breaks.

The json.dumps() method has parameters to make it easier to read the result:

import json

x = {

"name": "John",

"age": 30,

"married": True,

"divorced": False,

"children": ("Ann","Billy"),

"pets": None,

"cars": [

{"model": "BMW 230", "mpg": 27.5},

{"model": "Ford Edge", "mpg": 24.1}

]

}

# use four indents to make it easier to read the result:

print(json.dumps(x, indent=4))

import json

x = {

"name": "John",

"age": 30,

"married": True,

"divorced": False,

"children": ("Ann","Billy"),

"pets": None,

"cars": [

{"model": "BMW 230", "mpg": 27.5},

{"model": "Ford Edge", "mpg": 24.1}

]

}

# use . and a space to separate objects, and a space, a = and a space to separate keys from their values:

print(json.dumps(x, indent=4, separators=(". ", " = ")))

## **Order the Result**

The json.dumps() method has parameters to order the keys in the result:

### **Example**

Use the sort\_keys parameter to specify if the result should be sorted or not:

import json

x = {

"name": "John",

"age": 30,

"married": True,

"divorced": False,

"children": ("Ann","Billy"),

"pets": None,

"cars": [

{"model": "BMW 230", "mpg": 27.5},

{"model": "Ford Edge", "mpg": 24.1}

]

}

# sort the result alphabetically by keys:

print(json.dumps(x, indent=4, sort\_keys=True))