

JSON Schema 简介与安装

JSON Schema 是描述 JSON 数据结构的一种格式，JSON Schema 模式是一个词汇表。通过 JSON Schema 可以注释 JSON 的字段以及字段数据类型等信息。

在实际工作中，对接口返回值进行断言校验，除了常用字段的断言检测以外，还要对其他字段的类型进行检测。对返回的字段一个个写断言显然是非常耗时的，这个时候就需要一个模板，可以定义好数据类型和匹配条件，除了关键参数外，其余可直接通过此模板来断言，Json Schema 可以完美实现这样的需求。通过校验 JSON Schema 就可以判断 Response 是否符合约定。

一个 JSON 格式的数据，通常是由以下一种或多种数据类型组成的：

1. string
2. Numeric (integer、number)
3. object
4. array
5. boolean
6. null

JSON Schema 中对上面的 6 种数据类型，都有相应的属性对其进行描述。

Json Schema 官网：<http://json-schema.org/implementations.html>

环境准备

安装 JSON Schema 包：

```
1 pip install jsonschema
```

JSON Schema 的使用

JsonSchema 模板生成

1. 首先要借助于 Json Schema Tool 的网站：

<https://www.jsonschema.net/>

将返回 JSON 字符串复制到页面左边，然后点击 INFER SCHEMA，就会自动转换为 Schema JSON 文件类型，会将每个地段的返回值类型都设置一个默认类型；在 pattern 中也可以写正则进行匹配：

The image shows a web-based JSON Schema editor. The interface is split into two main panels. The left panel, titled 'JSON', contains a text area with the following JSON data:

```
{
  "lotto": {
    "lottoId": 5,
    "winnername": "allen",
    "winning-numbers": [
      2,
      45,
      34,
      23,
      7,
      5,
      3
    ]
  },
  "winners": [
    {
      "winnerId": 23,
      "numbers": [
        2,
        45,
        34,
        23,
        3,
        5
      ]
    },
    {
      "winnerId": 54,
      "numbers": [
        52,
        3,
        12,
        11,
        18,
        22
      ]
    }
  ]
}
```

Below the JSON text area are two buttons: 'RESET' and 'INFER SCHEMA'. The right panel, titled 'Schema', displays the JSON Schema inferred from the provided data. The schema is as follows:

```
{
  "definitions": {},
  "$schema": "http://json-schema.org/draft-07/schema#",
  "$id": "http://example.com/root.json",
  "type": "object",
  "title": "The Root Schema",
  "required": [
    "lotto"
  ],
  "properties": {
    "lotto": {
      "$id": "#/properties/lotto",
      "type": "object",
      "title": "The Lotto Schema",
      "required": [
        "lottoId",
        "winnername",
        "winning-numbers",
        "winners"
      ],
      "properties": {
        "lottoId": {
          "$id": "#/properties/lotto/properties/lottoId",
          "type": "integer",
          "title": "The Lottoid Schema",
          "default": 0,
          "examples": [
            5
          ]
        },
        "winnername": {
          "$id": "#/properties/lotto/properties/winnername",
          "type": "string",
          "title": "The Winnername Schema",
          "default": "",
          "examples": [
            "allen"
          ]
        },
        "pattern": "^(.*)$"
      ]
    }
  }
}
```

2. 点击“设置”按钮会出现各个类型返回值更详细的断言设置，这个就是 Schema 最常用也是最实用的功能，也可以对每种类型的字段更细化的区间值校验或者断言，例如长度，取值范围等。

Configuration

Global ^

URI Format *
JSON Pointer Fragm... ▾

Version *
draft-07 ▾

Absolute URI *
http://example.com/root.json

ABSOLUTE URI [RFC3986]

Annotations

Annotations ▾

Custom Annotations ▾

Assertions

Any ▾

Object ▾

Array ▾

String ▾

Number ^

multipleOf
--
NUMBER

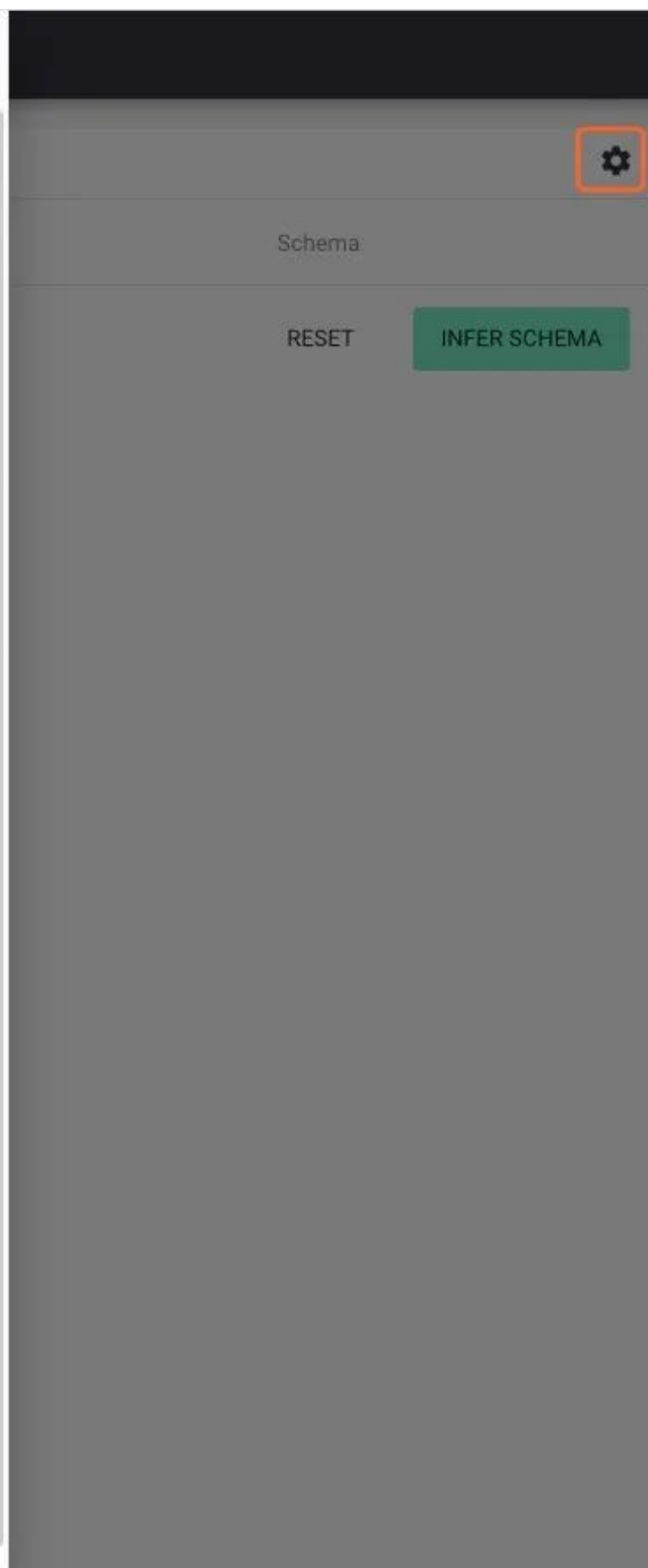
minimum
--
NUMBER

maximum
--
NUMBER

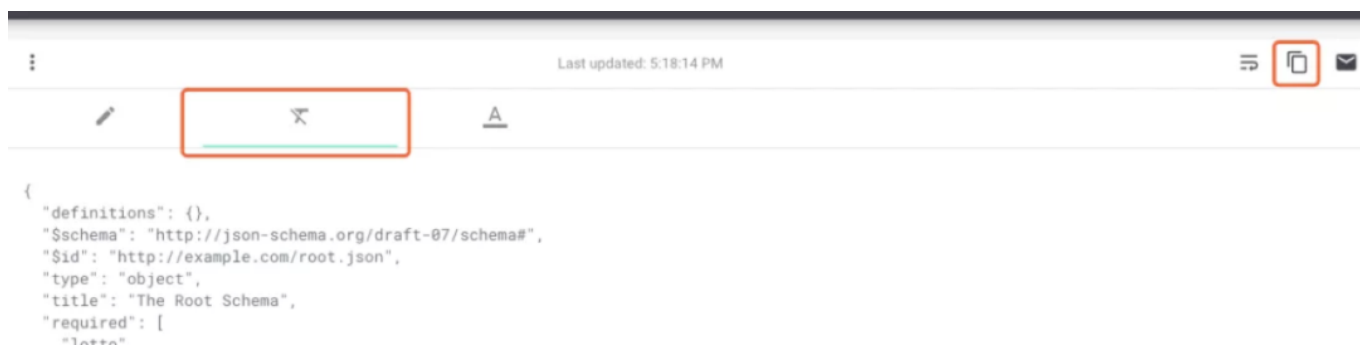
exclusiveMinimum
--
BOOLEAN

exclusiveMaximum
--
BOOLEAN

☐ Use number, not integer, for numeric instances.



3. 点击复制按钮，可以将生成的 Schema 模板保存下来。



JSON Schema 校验案例

下面有个 JSON Schema 例子，它只有两个重要字段 name 和 price。这个 schema 规定 name 必须是 string 类型，price 必须是 number 类型。使用 JSON Schema 进行校验，使用 validate 方法，输入一个 name 为 Eggs 和 price 为 34.99 的数据进行校验：

```
1 def test_schema(self):    schema = {        "type": "object",        "properties": {            "price": {"type": "number"},            "name": {"type": "string"},        },    }    validate(instance={"name": "Eggs", "price": 34.99}, schema=schema)
```

如果将 number 写成 string，则会出现报错：

```
1 from jsonschema import validate
2 >>> schema = {
3     ...     "type": "object",
4     ...     "properties": {
5     ...         "price": {"type": "string"},
6     ...         "name": {"type": "string"},
7     ...     },
8     ... }
9 >>> validate(instance={"name": "Eggs", "price": 34.99}, schema=schema)
```

返回报错信息：

```
1 Traceback (most recent call last):
2   File "<stdin>", line 1, in <module>
3   File "/Users/lixu/Library/Python/3.7/lib/python/site-packages/jsonschema/validators.py", \
4     line 934, in validate
5     raise error
6 jsonschema.exceptions.ValidationError: 34.99 is not of type 'string'
```