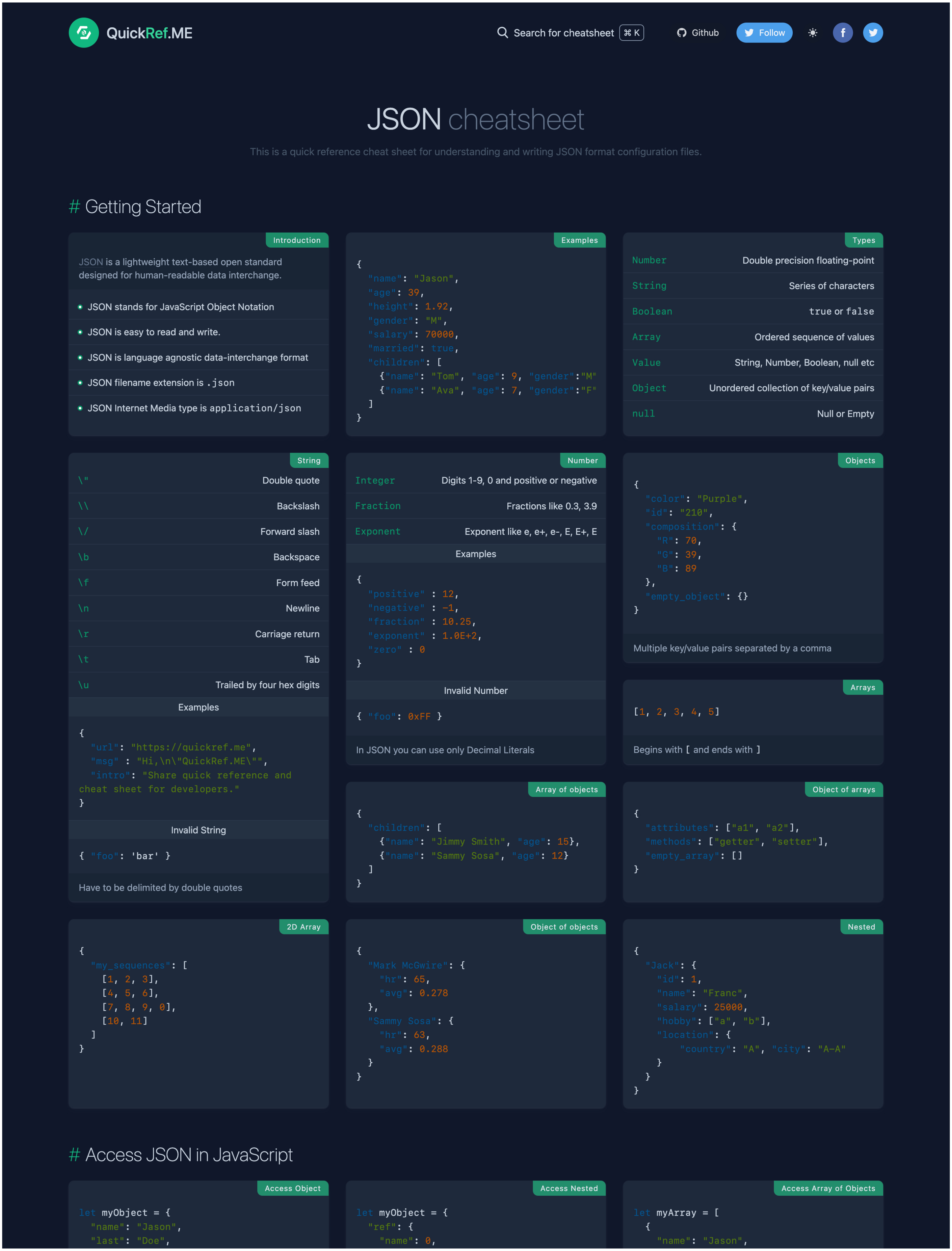

JSON cheatsheet

This is a quick reference cheat sheet for understanding and writing JSON format configuration files.

Getting Started

Introduction

JSON is a lightweight text-based open standard designed for human-readable data interchange.

- JSON stands for JavaScript Object Notation
- JSON is easy to read and write.
- JSON is language agnostic data-interchange format
- JSON filename extension is `.json`
- JSON Internet Media type is `application/json`

Examples

```
{
  "name": "Jason",
  "age": 39,
  "height": 1.92,
  "gender": "M",
  "salary": 70000,
  "married": true,
  "children": [
    {"name": "Tom", "age": 9, "gender":"M"
    {"name": "Ava", "age": 7, "gender":"F"
  ]
}
```

Types

`Number`	Double precision floating-point
`String`	Series of characters
`Boolean`	`true` or `false`
`Array`	Ordered sequence of values
`Value`	String, Number, Boolean, null etc
`Object`	Unordered collection of key/value pairs
`null`	Null or Empty

String

`\"`	Double quote
`\\`	Backslash
`\/`	Forward slash
`\b`	Backspace
`\f`	Form feed
`\n`	Newline
`\r`	Carriage return
`\t`	Tab
`\u`	Trailed by four hex digits

Examples

```
{
  "url": "https://quickref.me",
  "msg" : "Hi,\n\"QuickRef.ME\"",
  "intro": "Share quick reference and
cheat sheet for developers."
}
```

Invalid String

```
{ "foo": 'bar' }
```

Have to be delimited by double quotes

Number

`Integer`	Digits 1-9, 0 and positive or negative
`Fraction`	Fractions like 0.3, 3.9
`Exponent`	Exponent like e, e+, e-, E, E+, E

Examples

```
{
  "positive" : 12,
  "negative" : -1,
  "fraction" : 10.25,
  "exponent" : 1.0E+2,
  "zero" : 0
}
```

Invalid Number

```
{ "foo": 0xFF }
```

In JSON you can use only Decimal Literals

Objects

```
{
  "color": "Purple",
  "id": "210",
  "composition": {
    "R": 70,
    "G": 39,
    "B": 89
  },
  "empty_object": {}
}
```

Multiple key/value pairs separated by a comma

Arrays

```
[1, 2, 3, 4, 5]
```

Begins with `[` and ends with `]`

Array of objects

```
{
  "children": [
    {"name": "Jimmy Smith", "age": 15},
    {"name": "Sammy Sosa", "age": 12}
  ]
}
```

Object of arrays

```
{
  "attributes": ["a1", "a2"],
  "methods": ["getter", "setter"],
  "empty_array": []
}
```

2D Array

```
{
  "my_sequences": [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9, 0],
    [10, 11]
  ]
}
```

Object of objects

```
{
  "Mark McGwire": {
    "hr": 65,
    "avg": 0.278
  },
  "Sammy Sosa": {
    "hr": 63,
    "avg": 0.288
  }
}
```

Nested

```
{
  "Jack": {
    "id": 1,
    "name": "Franc",
    "salary": 25000,
    "hobby": ["a", "b"],
    "location": {
        "country": "A", "city": "A-A"
    }
  }
}
```

Access JSON in JavaScript

Access Object

```
let myObject = {
  "name": "Jason",
  "last": "Doe",
```

Access Nested

```
let myObject = {
  "ref": {
    "name": 0,
```

Access Array of Objects

```
let myArray = [
  {
    "name": "Jason",
    "age": 39,
    "gender": "M",
    "salary": 70000,
    "married": true
};
```

`myObject.name`	"Jason"
`myObject["name"]`	"Jason"
`myObject.age`	39
`myObject.other`	undefined
`myObject[0]`	undefined

Access Array

```
let myArray = [
  "Jason",
  "Doe",
  39,
  "M",
  70000,
  true
];
```

`myArray[1]`	"Doe"
`myArray[5]`	true
`myArray[6]`	undefined

```
    "last": 1,
    "age": 2,
    "gender": 3,
    "salary": 4,
    "married": 5
  },
  "jdoe": [
    "Jason",
    "Doe",
    39,
    "M",
    70000,
    true
  ],
  "jsmith": [
    "Tom",
    "Smith",
    42,
    "F",
    80000,
    true
  ]
};
```

`myObject.ref.age`	2
`myObject["ref"]["age"]`	2
`myObject.jdoe`	["Jason", "Doe", 39 ...]
`myObject.jsmith[3]`	"F"
`myObject[1]`	undefined

```
    "last": "Doe",
    "age": 39,
    "gender": "M",
    "salary": 70000,
    "married": true
  },
  {
    "name": "Tom",
    "last": "Smith",
    "age": 42,
    "gender": "F",
    "salary": 80000,
    "married": true
  },
  {
    "name": "Amy",
    "last": "Burnquist",
    "age": 29,
    "gender": "F",
    "salary": 60000,
    "married": false
  }
];
```

`myArray[0]`	{"name": "Jason", ...}
`myArray[1].name`	"Tom"
`myArray[1][2]`	42
`myArray[3]`	undefined
`myArray[3].gender`	TypeError: Cannot read...

Also see

JSON (json.org)
JSON Editor Online (jsoneditoronline.org)
Convert JSON Array to Markdown Table, CSV and more (tableconvert.com)

Related Cheatsheet

ES6 Cheatsheet
Quick Reference

Kubernetes Cheatsheet
Quick Reference

TOML Cheatsheet
Quick Reference

YAML Cheatsheet
Quick Reference

Recent Cheatsheet

Google Search Cheatshee
Quick Reference

Kubernetes Cheatsheet
Quick Reference

ES6 Cheatsheet
Quick Reference

ASCII Code Cheatsheet
Quick Reference

QuickRef.ME

Share quick reference and cheat sheet for developers.

中文版 #Notes