JavaScript Object Notation (json) Cheat Sheet

by Emre Coltu (emrecoltu) via cheatography.com/121590/cs/22330/

General Information

- Data is in **key/value** pairs
- Data groups by **colon** "**:**" and separates by **comma** "**,**"
- Objects are surrounded with **curly braces** "**{**" and "**}**"
- Each **key** in objects should be **unique**
- Arrays are surrounded with **square brackets** "**[**" and "**]**"
- Light, human-readable **data-interchange** format
- Can be used with **modern** programming languages
- Language **independent**
- The trailing commas and leading zeros are **prohibited**
- Numbers can't have **octal** or **hexadecimal** format
- **Easy** to use and doesn't use full markup like XML
- Popular due to it's **asynchronous** speed
- Extension for Json files is *".json"*
- Mime type for Json texts is *"**application/json**"*
- Official media type for Json is *"**application/json**"*
- Json internet media type is *"**application/json**"*
- The uniform type identifier for Json is *"**public.json**"*

Data Types

Json	Python	Possible Values
boolean	bool	true | false
number	float	123 | 123.456 | 1234e-3
string	str	"abcdef" | ""
array	list	["a", "b", "c", "d"]
object	dict	{"id": 7}
null	None	null

Escape Sequences

Name	Sequence
Double Quote	\"
Backslash	\\
Slash	\/
Backspace (BS)	\b
Form Feed (FF)	\f
Horizontal Tab (HT)	\t
Newline (NL)	\n
Carriage Return (CR)	\r
Hexadecimal	\uXXXX

Arrays

Syntax	Description
arr = []	Empty Array
arr = ["James"]	Single Element Array
`arr = [`	Multi Element Array
`    "James",`	Firstname
`    "Bond",`	Lastname
`    42,`	Age
`    "M",`	Sex
`    {"Germany": "Berlin"},`	Location
`    true`	Available
`]`	

Objects

Syntax	Description
obj = {}	Empty Object
obj = {"key": value}	Single Element Object

`obj = {`
`    "Firstname": "Jane",`
`    "Lastname": "Doe",`
`    "Age": 39,`
`    "Sex": "F",`
`    "Location": {"DE": "Berlin"},`
`    "Availability": false`
`}`
	Multi Element Object
	Firstname
	Lastname
	Age
	Sex
	Location
	Availability

Nested Object and Arrays

```
nested = {
    "indexes": {
        "Firstname": 0,
        "Lastname": 1,
        "Age": 2,
        "Sex": 3,
        "Country": 4,
        "Available": 5
    },
    "user": [
        "James",
        "Bond",
        42,
        "M",
        {"Germany": "Berlin"},
        true
    ]
}
```

Nested Objects and Arrays

```
nested = {
  "defaults": {
    "First": 0,
    "Last": 1,
    "Age": 2,
```

Nested Objects and Arrays (cont)

```
    "Sex": 3,
    "Country": 4,
    "Available": 5
  },
  "husband": [
    "James",
    "Bond",
    "42",
    "T",
    "Germany",
    true
  ],
  "wife": [
    "Jane",
    "Honda",
    "42",
    "T",
    "Germany",
    false
  ]
}
```

Arrays

array = []	Empty Array
array = ["James"]	Single Element Array
array = [	Multi Element Array
"James",	
"Bond",	
"42",	
"T",	
"Germany",	
true	
]	

Objects

myobject = {}	Empty Object
myobject = {"key": value}	Single Element Object
myobject = {	Multi Element Object
"First": "James",	
"Last": "Bond",	
"Age": "42",	
"Sex": "T",	
"Country": "Germany",	
"Available": true	
}	

Syntax	Description
arr = []	Empty Array
arr = ["Username"]	Single Element Array

`arr = [`
`    "James",`
`    "Bond",`
`    42,`
`    "M",`
`    {"Germany": "Berlin"},`
`    true`
`]`
	Multi Elements Array

Literals

null
true
false

Data Types

boolean	true, false
number	123, 123.456, 1.234e-3
string	"abcdef", ""
array	["a", "b", "c", "d"]
object	{"id": 7}
null	null

General Information

- Doesn't have comments. Everything in it is considered data
- Data is in name/value pairs. Grouped by a colon ":" and separated by comma ","
- Objects are surrounded with curly braces. Each key within should be unique and enclosed within the double quotes. An object begins with a left brace "{" and ends with a right brace "}"
- Arrays are surrounded with square brackets. An array begins with a left bracket "[" and ends with a right bracket "]"
- The trailing commas and leading zeros are prohibited
- Numbers can't have octal or hexadecimal format
- Lightweight and human-readable data-interchange format
- Can be used with modern programming languages
- Language independent
- Very easy to parse and generate and doesn’t use a full markup structure like a XML
- Popular alternative to XML format for its fast asynchronous client–server communication
- Extension for json files is ".json"
- Mime type for json texts is "application/json"
- The official media type for json is "application/json".
- json internet media type is "application/json".
- The uniform type identifier is "public.json"

Escape Sequences

Double Quote	\"
Backslash	\\
Slash	\/
Backspace (BS)	\b
Form Feed (FF)	\f
Horizontal Tab (HT)	\t
Newline ()	\n
Carriage Return	\r
Hexadecimal	\uXXXX