

Overview example

```
{ "sequence": 52634,
  "employees": [
    { "first": "Anna", "last": "Smith", "loc": 1 },
    { "first": "John", "last": "Doe", "loc": 1 },
    { "first": "Sandra", "last": "Jones", "loc": 2 }
  ],
  "locations": [
    { "city": "New York", "type": "HQ", "key": 1 },
    { "city": "Los Angeles", "type": "Branch", "key": 2 }
  ]
}
```

This is a virtual example demonstrating JSON usage. The main object contains a sequence number ("sequence") with a value of 52634, an array of employees and an array of locations.

Values

"string" see "Strings" box for details

1234 integer number

1234.5678 floating point number

1.234e-3 floating point with exponent⁽¹⁾

true boolean "True"

false boolean "False"

null denoting "empty"

object a value can be an object

array a value can be an array of values

(1) Exponent prefix is case insensitive (e or E) and could be followed by a sign, mandatory for negative, optional for positive

Object

{ } empty object

{ "key": value } single value object

{ "key1": value, "key2": value, ... } multiple value object

(1) An object is the preferred top level structure for JSON

(2) The key can be any string (see "string" for details)

(3) See "Values" box for information on possible values

Array

[] empty array

[value] single element array

[value, value, ...] multiple value array

(1) See "Values" box for information on possible values

Strings

"" empty string

"some string characters" string⁽¹⁾

(1) see "String characters" box for details on allowed characters and coding

String character

non-special characters Any unicode character except " (quotes), \ (backslash) or any control character

\ " double quotes

\\ backslash

\/ slash

\b backspace

\f formfeed

\n newlinw

\r carriage return

\t horizontal tab

\uXXXX where XXXX is the 4 digit hexadecimal unicode code for the character