Introducing JSON

Български Český Dansk Nederlands English Esperanto Français Deutsch Ελληνικά עברית Magyar Indonesia Italiano Polski Português Română Русский Српско-хрватски Slovenščina Español Svenska Türkçe Українська Tiếng Việt

ECMA-404 The JSON Data Interchange Standard.

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

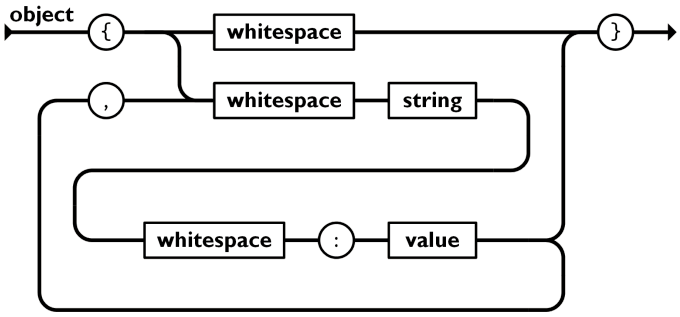
> A collection of name/value pairs. In various languages, this is realized as an *object*, record, struct, dictionary, hash table, keyed list, or associative array.
> An ordered list of values. In most languages, this is realized as an *array*, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures.

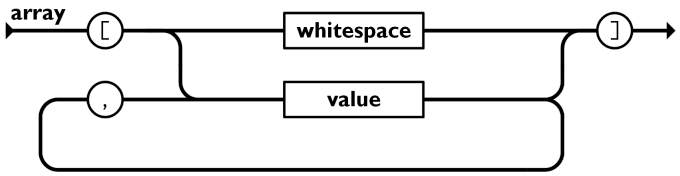
In JSON, they take on these forms:

An *object* is an unordered set of name/value pairs. An object begins with `{` *left brace* and ends with `}` *right brace*. Each name is followed by `:` *colon* and the name/value pairs are separated by `,` *comma*.

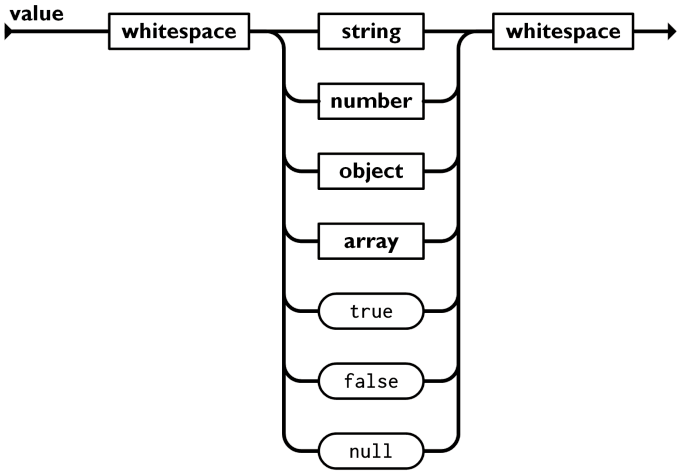
```
json
    element

value
    object
    array
    string
    number
    "true"
    "false"
    "null"

object
    '{' ws '}'
    '{' members '}'

members
    member
    member ',' members

member
    ws string ws ':' element

array
    '[' ws ']'
    '[' elements ']'

elements
    element
    element ',' elements

element
    ws value ws

string
    '"' characters '"'
```



An *array* is an ordered collection of values. An array begins with `[` *left bracket* and ends with `]` *right bracket*. Values are separated by `,` *comma*.

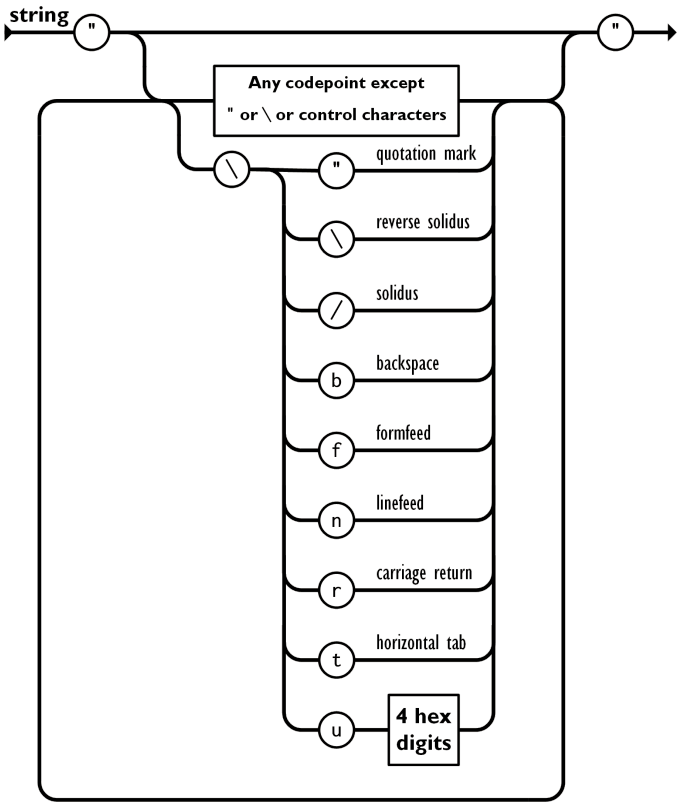


A *value* can be a *string* in double quotes, or a *number*, or `true` or `false` or `null`, or an *object* or an *array*. These structures can be nested.



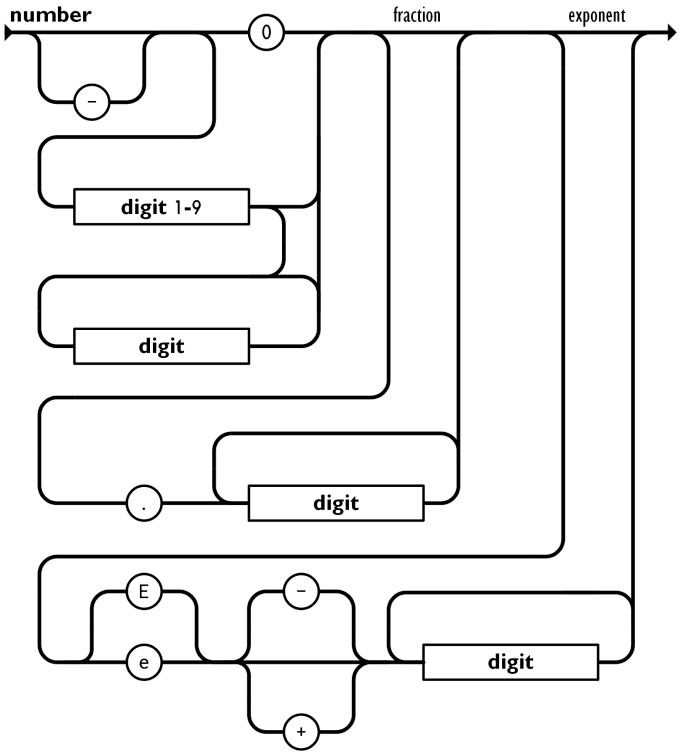
A *string* is a sequence of zero or more Unicode characters, wrapped in double quotes, using backslash escapes. A character is represented as a single character string. A string is very much like a C or Java string.

```
characters
    ""
    character characters

character
    '0020' . '10FFFF' - '"' - '\'
    '\' escape

escape
    '"'
    '\'
    '/'
    'b'
    'f'
    'n'
    'r'
    't'
    'u' hex hex hex hex

hex
    digit
    'A' . 'F'
    'a' . 'f'

number
    integer fraction exponent

integer
    digit
    onenine digits
    '-' digit
    '-' onenine digits

digits
    digit
    digit digits

digit
    '0'
    onenine

onenine
    '1' . '9'

fraction
    ""
    '.' digits

exponent
    ""
    'E' sign digits
    'e' sign digits
```

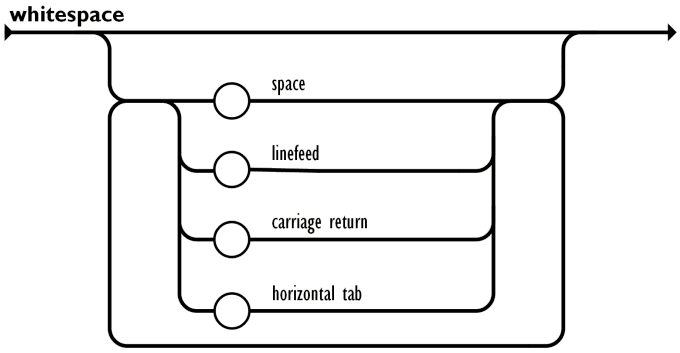



```
sign
    ""
    '+'
    '-'

ws
    ""
    '0020' ws
    '000A' ws
    '000D' ws
    '0009' ws
```

A *number* is very much like a C or Java number, except that the octal and hexadecimal formats are not used.



Whitespace can be inserted between any pair of tokens. Excepting a few encoding details, that completely describes the language.



- 8th
  - json
- ActionScript
  - ActionScript3
- Ada
  - GNATCOLL.JSON
- AdvPL
  - JSON-ADVPL
- APL
  - JSON
- ASP
  - JSON for ASP
  - JSON ASP utility class
- AWK
  - JSON.awk
  - rhawk
- BlitzMax
  - bmx-rjson
- C
  - JSON_checker
  - YAJL
  - LibU
  - json-c
  - json-parser
  - jsonsl
  - WJElement
  - M's JSON parser
  - cJSON
  - Jansson
  - jsmn
  - parson
  - ujson4c
  - frozen
  - microjson
  - mjson
  - progbase
  - lwjson
  - cisson
- C++
  - JSONKit
  - jsonme--
  - ThorsSerializer
- ColdFusion
  - SerializeJSON
- D
  - std.json
  - asdf
  - vibe.data.json
- Dart
  - json library
- Delphi
  - Delphi Web Utils
  - JSON Delphi Library
- E
  - JSON in TermL
- Erlang
  - erl-json
- Fantom
  - Json
- FileMaker
  - JSON
- Fortran
  - json-fortran
  - YAJL-Fort
  - jsonff
- Go
  - package json
- Groovy
  - groovy-io
- Haskell
  - RJson package
  - json package
- Java
  - JSON-java
  - JSONUtil
  - jsonp
  - Json-lib
  - Stringtree
  - SOJO
  - json-taglib
  - Flexjson
  - Argo
  - jsonij
  - fastjson

- JsonBox
- jvar
- rapidjson
- JSON for Modern C++
- minijson
- jsoncons
- jsoncpp
- univalue
- ArduinoJson
- QJson
- CAJUN
- libjson
- nosjob
- JSON library for IoT
- qmjson
- JSON Support in Qt
- JsonWax for Qt
- progbase
- Qentem-Engine

C#
- fastJSON
- JSON_checker
- Json.NET
- JSON for .NET
- Manatee Json
- FastJsonParser
- LightJson
- Liersch.Json
- Liersch.JsonSerialization
- progbase
- JSON Essentials

Clojure
- data.json

Cobol
- Redvers COBOL JSON Interface

Net.Data
- netdata-json

Nim
- Module json

Objective C
- NSJSONSerialization
- json-framework
- JSONKit
- yajl-objc
- TouchJSON

OCaml
- jsonm

PascalScript
- JsonParser

Perl
- CPAN

Photoshop
- mjson
- jjson
- json-simple
- json-io
- google-gson
- FOSS Nova JSON
- Corn CONVERTER
- Apache johnzon
- Genson
- cookjson
- progbase
- jackson
- MOXy

JavaScript
- JSON
- json2.js
- clarinet
- Oboe.js
- progbase

LabVIEW
- flatten

Lisp
- Common Lisp JSON

LiveCode
- mergJSON

LotusScript
- JSON LS

Lua
- JSON Modules

M
- DataBallet

Matlab
- JSONlab
- 20565
- 23393

- JSON Photoshop Scripting
- PHP
  - PHP 5.2
- PicoLisp
  - picolisp-json
- Pike
  - Public.Parser.JSON
  - Public.Parser.JSON2
- PL/SQL
  - pljson
- PureBasic
  - JSON
- Puredata
  - PuRestJson
- Python
  - The Python Standard Library
  - simplejson
  - pyson
  - Yajl-Py
  - ultrajson
  - metamagic.json
  - progbase
- R
  - rjson
  - jsonlite
- Racket
  - json-parsing
- Rebol
  - json.r
- RPG
  - JSON Utilities
- Rust
  - Serde JSON
  - json-rust
- Ruby
  - yajl-ruby
  - json-stream
  - progbase
- Scala
  - circe
- Scheme
  - MZScheme
  - JSON-struct
- Shell
  - Jshon
  - JSON.sh
  - jwalk
- Squeak
  - Squeak
- Tcl
  - JSON
- Visual Basic
  - VB-JSON
  - PW.JSON

- .NET-JSON-Transformer
- progbase

Visual FoxPro

- fwJSON
- JSON
- vfpjson

- Videos about JSON
- Videos about the JSON Logo
- Heresy & Heretical Open Source: A Heretic's Perspective
- *How JavaScript Works* by Douglas Crockford