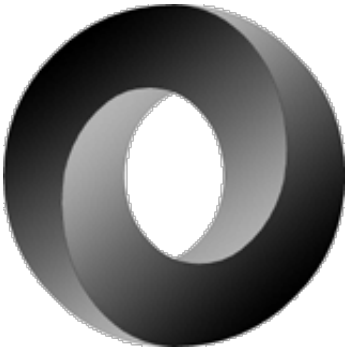


Introducing JSON

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

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).

object
- `{}`
- `{` *members* `}`

members
- *pair*
- *pair* `,` *members*

pair
- *string* `:` *value*

array
- `[]`
- `[` *elements* `]`

elements
- *value*
- *value* `,` *elements*

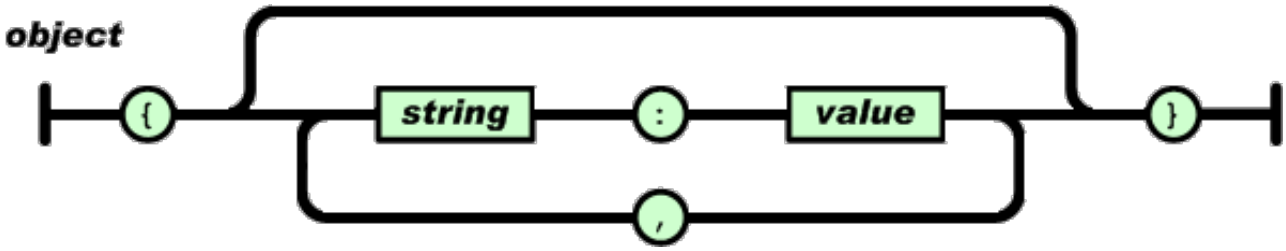
value
- *string*
- *number*
- *object*
- *array*
- `true`
- `false`
- `null`

string
- `""`
- `"` *chars* `"`

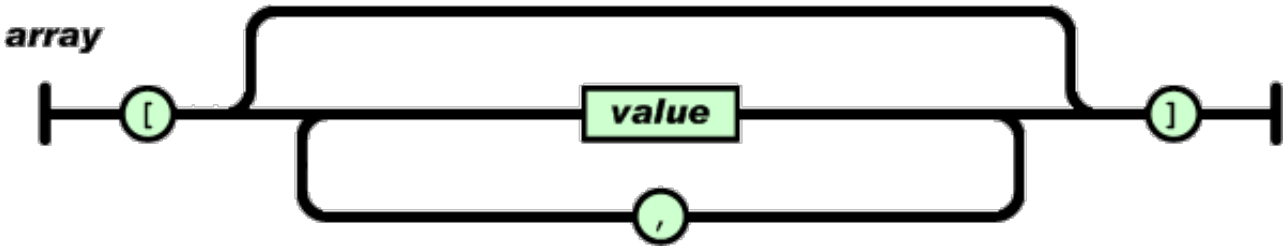
chars
- *char*
- *char chars*

char
- *any-Unicode-character-except-"-or-\-or-control-character*
- `\"`
- `\\`
- `\/`
- `\b`
- `\f`
- `\n`

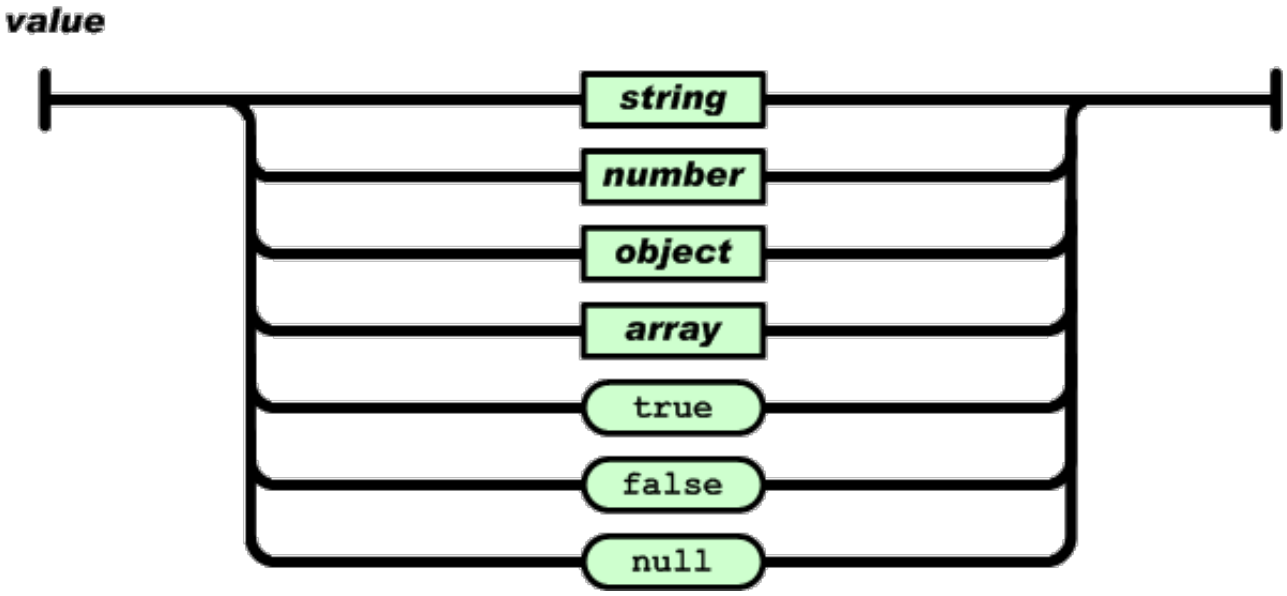
```
        \r
        \t
        \u four-hex-digits
number
        int
        int frac
        int exp
        int frac exp
int
        digit
        digit1-9 digits
        - digit
        - digit1-9 digits
frac
        . digits
exp
        e digits
digits
        digit
        digit digits
e
        e
        e+
        e-
        E
        E+
        E-
```



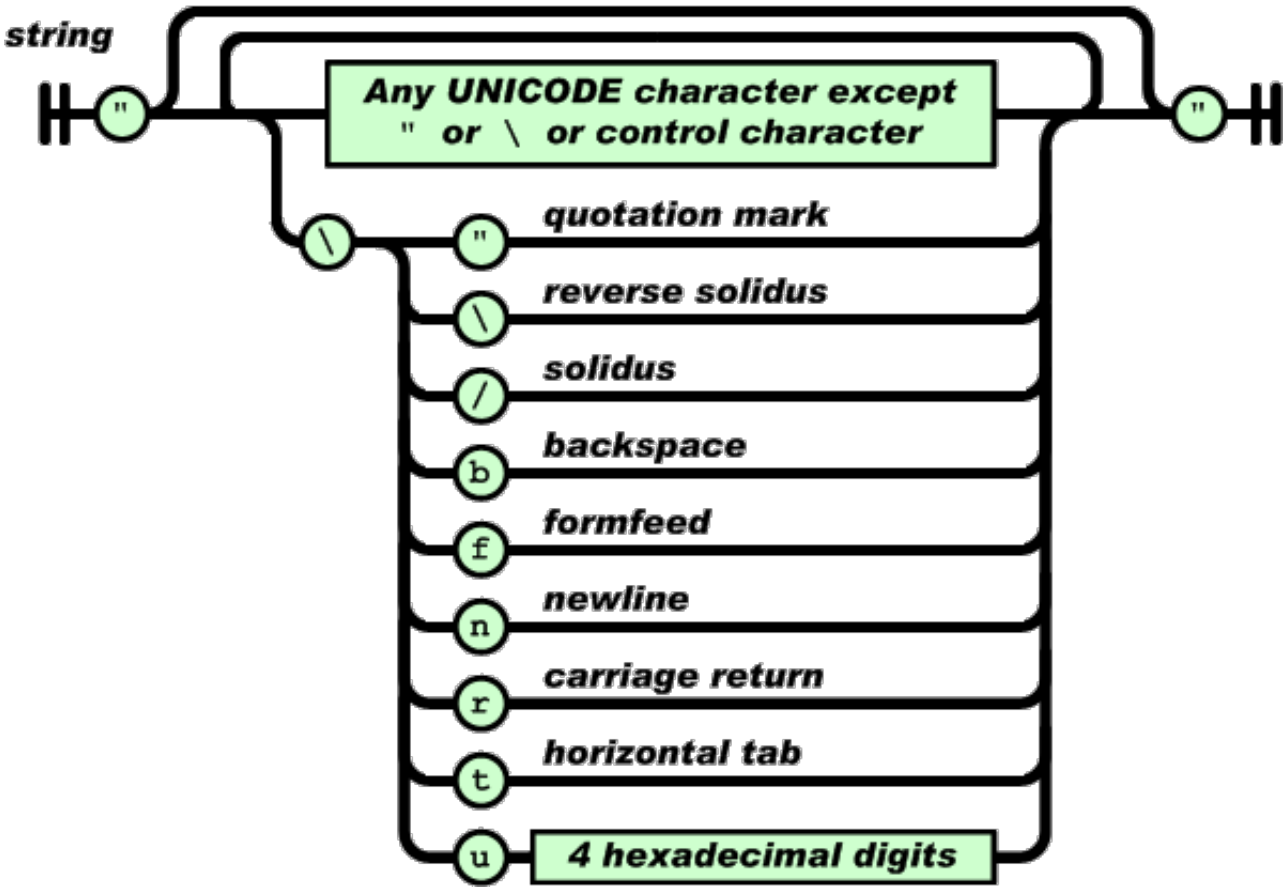
An *array* is an ordered collection of values. An array begins with `[` (left bracket) and ends with `]` (right bracket). Values are separated by `,` (comma).



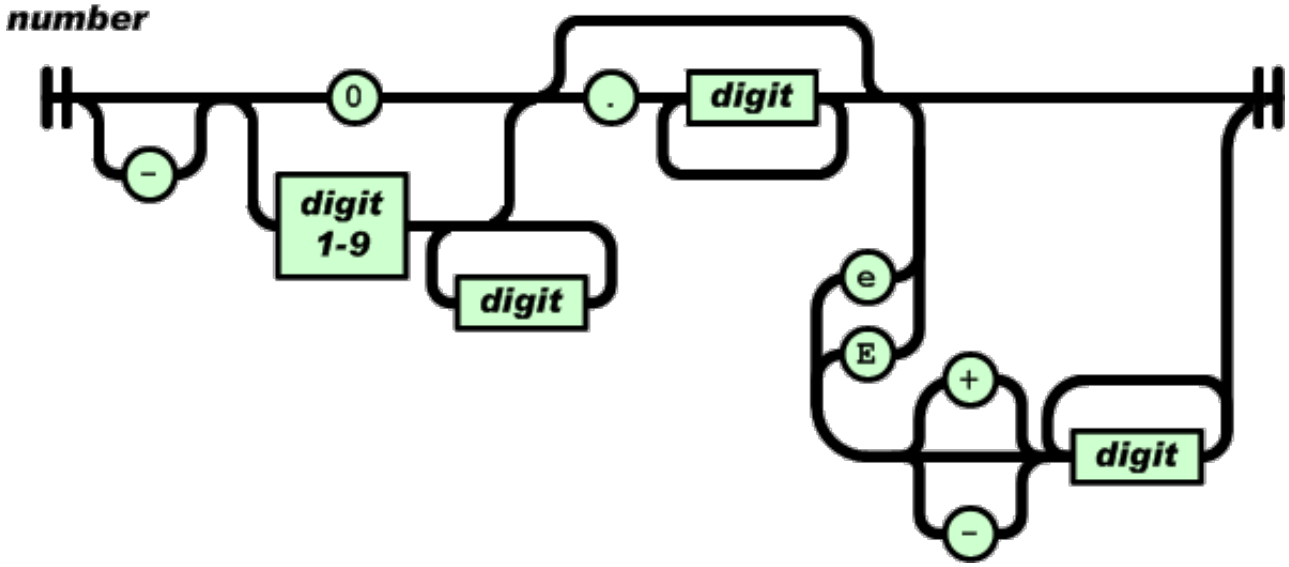
A *value* can be a *string* in double quotes, or a *number*, or `true` or `false` or `null`, or an *object* or an *array*. These structures can be nested.



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.



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.

- ASP:
  - JSON for ASP.
  - JSON ASP utility class.
- ActionScript:
  - ActionScript3.
  - JSONConnector.
- Ada:
  - GNATCOLL.JSON.
- Bash:
  - Jshon.
  - JSON.sh.
- BlitzMax:
  - bmx-rjson.
- C:
  - JSON_checker.
  - YAJL.
  - js0n.
  - LibU.
  - json-c.
  - json-parser.
  - jsonsl.
  - WJElement.
  - M's JSON parser.
  - cJSON.
  - Jansson.
  - jsmn.
  - cson.
  - parson.
- C++:
  - JSONKit.
  - jsonme--.
  - ThorsSerializer.
  - JsonBox.
  - jsoncpp.
- Delphi:
  - Delphi Web Utils.
  - JSON Delphi Library.
  - JSON Toolkit.
  - tiny-json.
- E:
  - JSON in TermL.
- Erlang:
  - ejson.
  - mochijson2.
- Fantom:
  - Json.
- Go:
  - package json.
- Haskell:
  - RJson package.
  - json package.
- haXe:
  - hxJSON.
- Java:
  - org.json.
  - org.json.me.
  - Jackson JSON Processor.
  - Json-lib.
  - JSON Tools.
  - Stringtree.
  - SOJO.
  - Jettison.
  - json-taglib.
  - XStream.
  - Flexjson.
  - JON tools.
  - Argo.
  - jsonij.

    - zoolib.
    - JOST.
    - CAJUN.
    - libjson.
    - nosjob.
    - rapidjson.
- C#:
    - fastJSON.
    - JSON_checker.
    - Jayrock.
    - Json.NET - LINQ to JSON.
    - LitJSON.
    - JSON for .NET.
    - JsonFx.
    - JSON@CodeTitans
    - How do I write my own parser?
    - JSONSharp.
    - JsonExSerializer.
    - fluent-json
    - Manatee Json
- Ciao:
    - Ciao JSON encoder and decoder
- Clojure:
    - clojure-json.
    - API for json.
- Cobol:
    - XML Thunder.
- ColdFusion:
    - ColdFusion 8.
    - toJSON.
- D:
    - Cashew.
    - Libdjson.
- Dart:
    - json library.
    - fastjson.
    - mjson.
    - jjson.
    - json-simple.
    - json-io.
    - JsonMarshaller.
    - google-gson.
    - Json-smart.
    - FOSS Nova JSON.
- JavaScript:
    - JSON.
    - json2.js.
    - json_sans_eval.
    - clarinet.
- Lisp:
    - Common Lisp JSON.
    - Yason.
    - Emacs Lisp.
- LotusScript:
    - JSON LS.
- Lua:
    - Json4Lua.
    - LuaJSON.
    - LuaJSON C Library.
    - Lua CJSON.
    - dkjson.
- LabVIEW:
    - JSON Toolkit.
- M:
    - DataBallet.
- Matlab:
    - JSONlab.
    - JSON Parser.
    - (another) JSON Parser.
- Objective C:
    - json-framework.
    - MTJSON.
    - JSONKit.
    - yajl-objc.
    - TouchJSON.
- OCaml:
    - Yojson.
    - jsonm.
- OpenLaszlo:
    - JSON.
- Perl:
    - CPAN.
    - perl-JSON-SL.
- PHP:
    - PHP 5.2.
    - json.
    - Services_JSON.

    - Zend_JSON.
    - Solar_Json.
    - Comparison of php json libraries.
- Pike:
    - Public.Parser.JSON.
    - Public.Parser.JSON2.
- PL/SQL:
    - pljson:
    - Librairie-JSON.
- PowerShell:
    - PowerShell.
- Prolog:
    - SWI-Prolog HTTP support
- Puredata:
    - PuRestJson
- Python:
    - The Python Standard Library.
    - simplejson.
    - pyson.
    - Yajl-Py.
    - ultrajson.
    - metamagic.json.
- Qt:
    - QJson.
- R:
    - rjson.
- Racket:
    - json-parsing.
- Rebol:
    - json.r.
- RPG:
    - JSON Utilities.
- Ruby:
    - json.
    - yajl-ruby.
    - json-stream.
- Scala:
    - package json.
- Scheme:
    - MZScheme.
    - PLT Scheme.
- Squeak:
    - Squeak.
- Symbian:
    - s60-json-library.
- Tcl:
    - JSON.
- Visual Basic:
    - VB-JSON.
    - PW.JSON.
- Visual FoxPro:
    - fwJSON.
    - JSON.
    - vfpjson.

- RFC 4627 application/json.

- The JSON Group on Yahoo!

- JSLint, Syntax Checker.
- JSONLint, The JSON Validator.
- JSON shell for the browser
- JSON Formatter
- JSON Designer
- JSON Editor
- JSON Parser
- JSON Test

- JSONT.
- JSONPath.
- JSONSelect.
- JSONiq.
- Draft JSON Schema.
- json-template.
- JPath.
- jaql.
- Itemscript.
- JSPON.
- JsonML.
- BSON.
- RSON.

- CouchDB.
- MongoDB.
- DBSlayer.
- Metaweb Query Language.
- ChaiDB.
- Persevere.
- FleetDB.
- OrientDB.
- terrastore.
- MLJSON.

- JSON-RPC.
- jabsorb.
- Simple Remoting.

- XSLT and XPath for JSON.
- xml2json-xslt.
- XSLTJSON.
- x-xml2jsonphp.
- Pure.
- csv2json.
- The Fat-free Alternative to XML