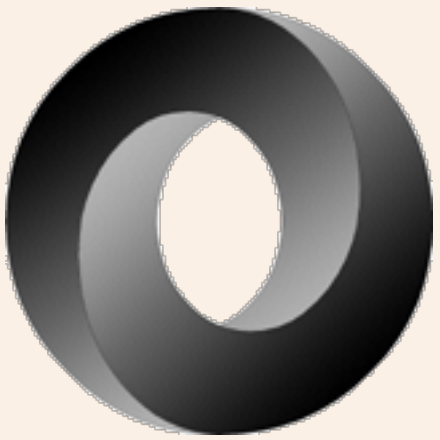


Introducción a JSON

How JavaScript Works by Douglas Crockford.

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

ECMA-404 The JSON Data Interchange Standard.

JSON (JavaScript Object Notation - Notación de Objetos de JavaScript) es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. Está basado en un subconjunto del Lenguaje de Programación JavaScript, Standard ECMA-262 3rd Edition - Diciembre 1999. JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituído por dos estructuras:

- Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un *objeto*, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o sequencias.

Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de

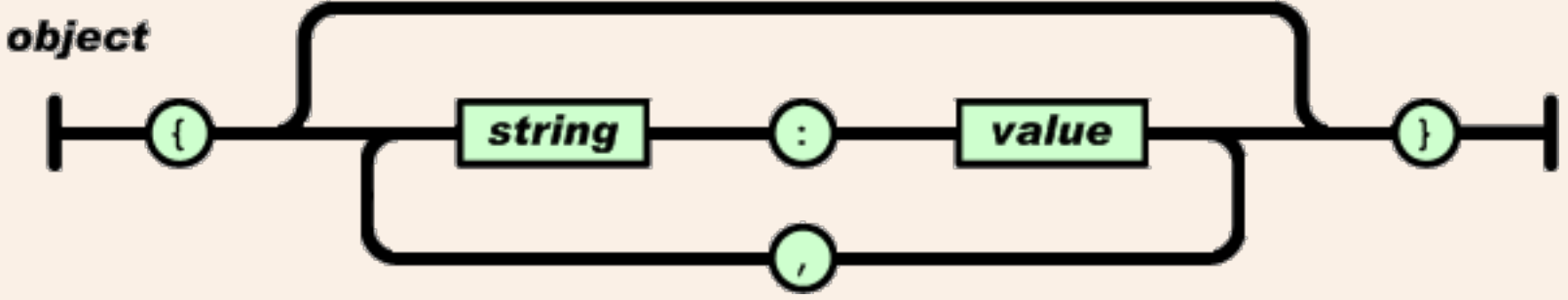
```
json
    element

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 '"'
characters
    ""
    character characters
```

intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras.

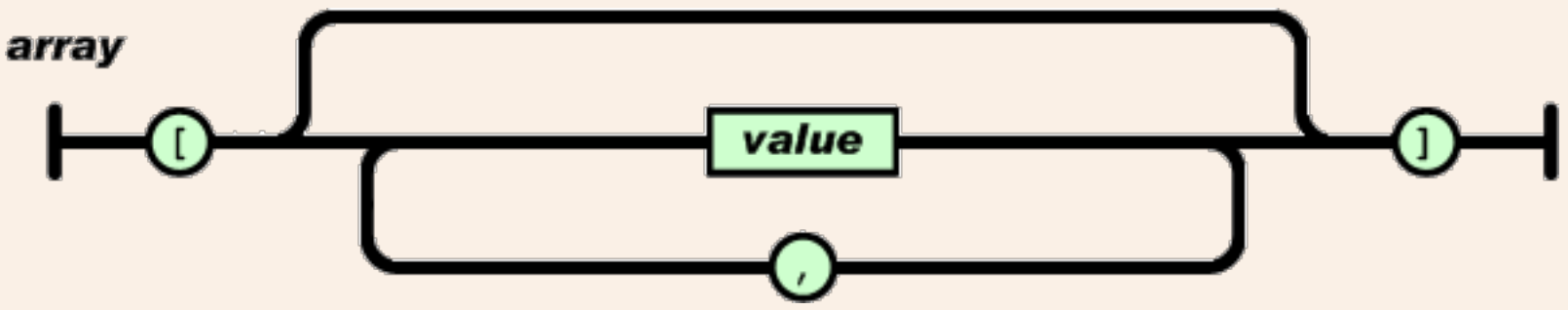
En JSON, se presentan de estas formas:

Un *objeto* es un conjunto desordenado de pares nombre/valor. Un objeto comienza con `{` (llave de apertura) y termine con `}` (llave de cierre). Cada nombre es seguido por `:` (dos puntos) y los pares nombre/valor están separados por `,` (coma).

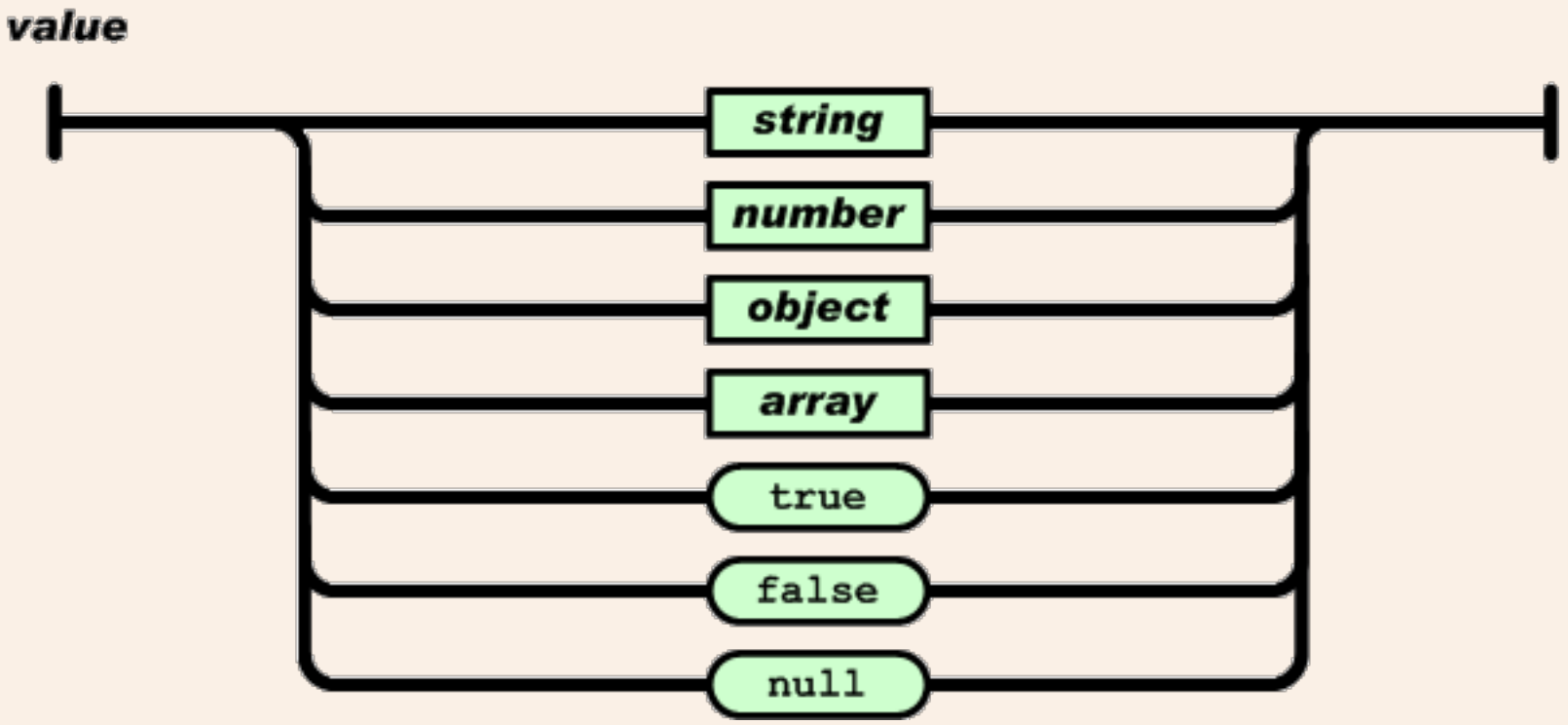
```
character
    '0020' . '10ffff' - '"' - '\'
    '\' escape
escape
    '"'
    '\'
    '/'
    'b'
    'n'
    'r'
    't'
    'u' hex hex hex hex

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

number
    int frac exp

int
    digit
    onenine digits
    '-' digit
    '-' onenine digits

digits
    digit
    digit digits

digit
    '0'
    onenine

onenine
    '1' . '9'

frac
    ""
    '.' digits

exp
    ""
    'E' sign digits
    'e' sign digits

sign
    ""
    '+'
    '-'

ws
    ""
    '0009' ws
    '000a' ws
    '000d' ws
    '0020' ws
```



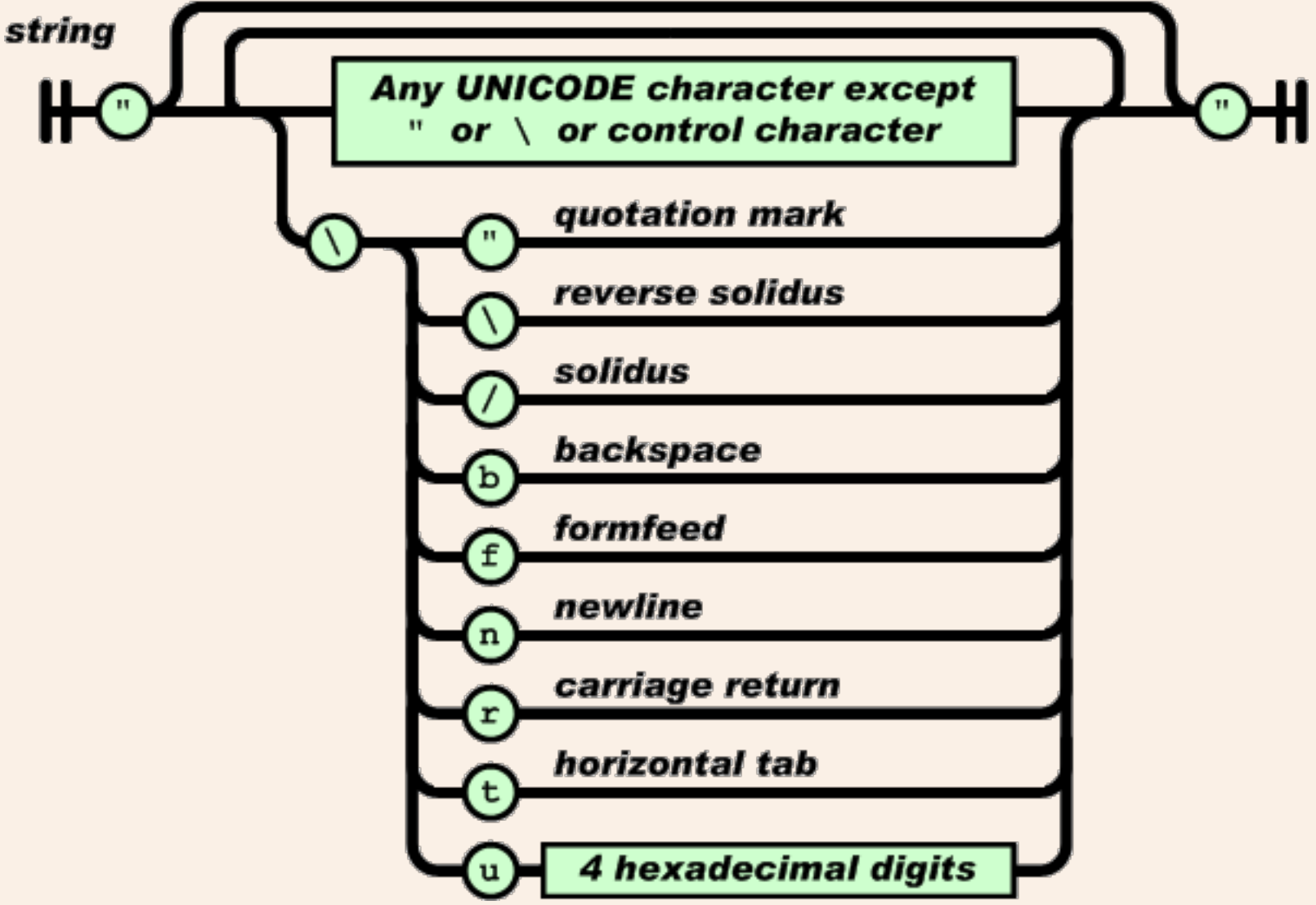
Un *arreglo* es una colección de valores. Un arreglo comienza con `[` (corchete izquierdo) y termina con `]` (corchete derecho). Los valores se separan por `,` (coma).



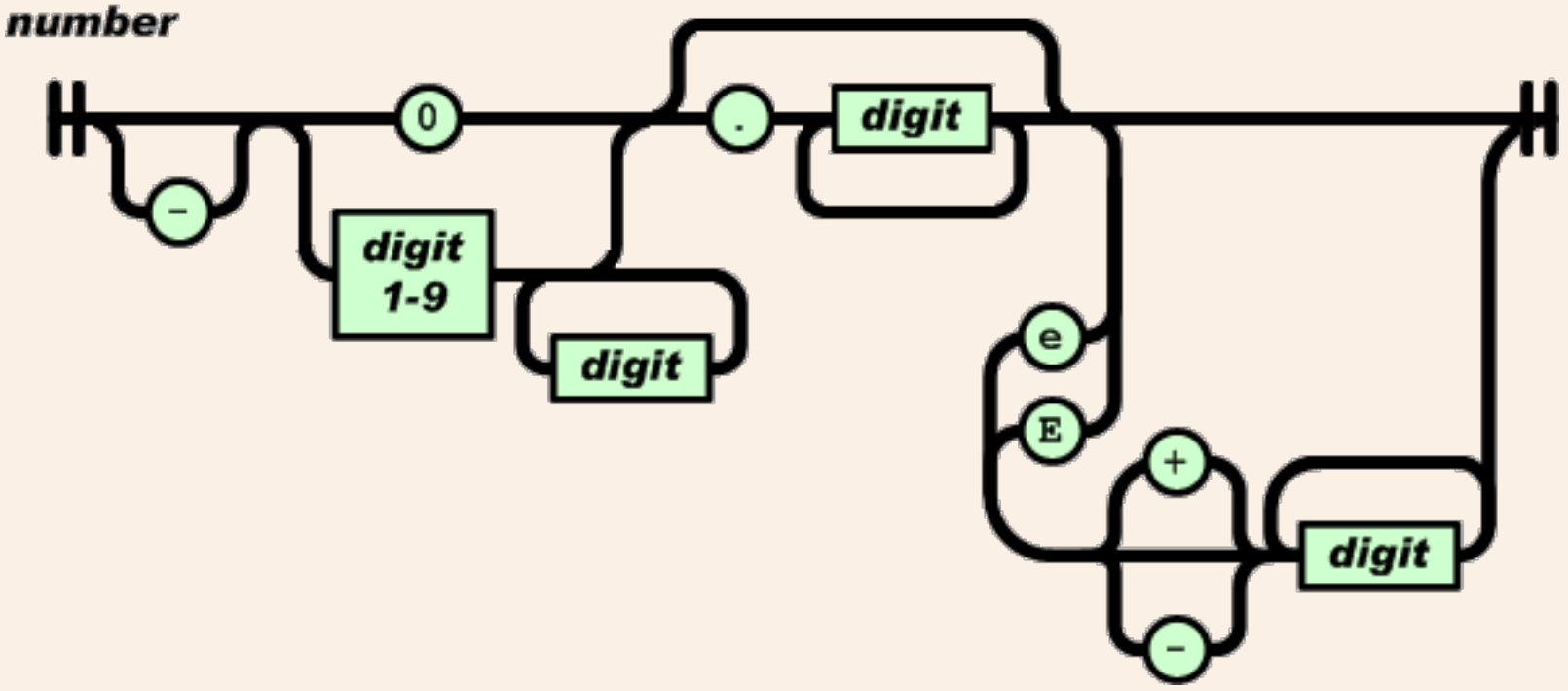
Un *valor* puede ser una *cadena de caracteres* con comillas dobles, o un *número*, o `true` o `false` o `null`, o un *objeto* o un *arreglo*. Estas estructuras pueden anidarse.



Una *cadena de caracteres* es una colección de cero o más caracteres Unicode, encerrados entre comillas dobles, usando barras divisorias invertidas como escape. Un carácter está representado por una cadena de caracteres de un único carácter. Una *cadena de carateres* es parecida a una cadena de caracteres C o Java.



Un *número* es similar a un número C o Java, excepto que no se usan los formatos octales y hexadecimales.



Los espacios en blanco pueden insertarse entre cualquier par de símbolos.

Exceptuando pequeños detalles de *encoding*, esto describe completamente el lenguaje.

- 8th:
  - json>.
- ABAP:
  - EPO Connector.
- ActionScript:
  - ActionScript3.
- Clojure:
  - data.json.
- Cobol:
  - XML Thunder.
  - Redvers COBOL JSON Interface.
- ColdFusion:

- Ada:
  - GNATCOLL.JSON.
- AdvPL:
  - JSON-ADVPL.
- ASP:
  - JSON for ASP.
  - JSON ASP utility class.
- AWK:
  - JSON.awk.
  - rhawk.
- Bash:
  - Jshon.
  - JSON.sh.
- BlitzMax:
  - bmx-rjson.
- C:
  - JSON_checker.
  - YAJL.
  - LibU.
  - json-c.
  - json-parser.
  - jsonsl.
  - WJElement.
  - M's JSON parser.
  - cJSON.
  - Jansson.
  - jsmn.
  - parson.
  - ujson4c.
  - nxjson.
  - frozen.
  - microjson.
  - mjson.
- C++:
  - JSONKit.
  - jsonme--.
  - ThorsSerializer.
  - JsonBox.
  - jvar.
  - rapidjson.
  - JSON for Modern C++.
  - ArduinoJson.
  - minijson.
  - jsoncons.
  - QJson.
  - jsoncpp.
  - CAJUN.
  - libjson.
  - nosjob.
  - JSON++.
  - JSON library for IoT.
  - qmjson.
  - JSON Support in Qt.
  - JsonWax for Qt.
  - SerializeJSON.
  - toJSON.
- D:
  - Libdjson.
- Dart:
  - json library.
- Delphi:
  - Delphi Web Utils.
  - JSON Delphi Library.
- E:
  - JSON in TermL.
- Fantom:
  - Json.
- FileMaker:
  - JSON.
- Fortran:
  - json-fortran.
  - YAJL-Fort.
- Go:
  - package json.
- Groovy:
  - groovy-io.
- Haskell:
  - RJson package.
  - json package.
- Java:
  - JSON-java.
  - JSONUtil.
  - jsonp.
  - Json-lib.
  - Stringtree.
  - SOJO.
  - json-taglib.
  - Flexjson.
  - JON tools.
  - Argo.
  - jsonij.
  - fastjson.
  - mjson.
  - jjson.
  - json-simple.
  - json-io.
  - JsonMarshaller.
  - google-gson.
  - Json-smart.
  - FOSS Nova JSON.
  - Corn CONVERTER.
  - Apache johnzon.
  - Genson.
  - JSONUtil.
  - cookjson.
- JavaScript:
  - JSON.
  - json2.js.

- C#:
  - fastJSON.
  - JSON_checker.
  - Jayrock.
  - Json.NET - LINQ to JSON.
  - JSON for .NET.
  - JSONSharp.
  - fluent-json.
  - Manatee Json.
  - FastJsonParser.
  - LightJson.
  - liersch.json.
- Ciao:
  - Ciao JSON encoder and decoder.
  - clarinet.
  - Oboe.js.
- LabVIEW:
  - flatten.
- Lisp:
  - Common Lisp JSON.
  - Emacs Lisp.
- LiveCode:
  - mergJSON.
- LotusScript:
  - JSON LS.
- Lua:
  - JSON Modules.
- M:
  - DataBallet.
- Matlab:
  - JSONlab.
  - 20565.
  - 23393.
- Net.Data:
  - netdata-json.
- Nim:
  - Module json.
- Objective C:
  - NSJSONSerialization.
  - json-framework.
  - JSONKit.
  - yajl-objc.
  - TouchJSON.
- OCaml:
  - jsonm.
- PascalScript:
  - JsonParser.
- Perl:
  - CPAN.
- Photoshop:
  - 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.
- R:
    - rjson.
    - jsonlite.
- Racket:
    - json-parsing.
- Rebol:
    - json.r.
- RPG:
    - JSON Utilities.
- Rust:
    - Serde JSON.
    - json-rust.
- Ruby:
    - yajl-ruby.
    - json-stream.
- Scheme:
    - MZScheme.
    - PLT Scheme.
- Squeak:
    - Squeak.
- Symbian:
    - s60-json-library.
- Tcl:
    - JSON.
- Visual Basic:
    - VB-JSON.
    - PW.JSON.
    - .NET-JSON-Transformer.
- Visual FoxPro:
    - fwJSON.
    - JSON.
    - vfpjson.