Einführung in 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

ECMA-404 The JSON Data Interchange Standard.

JSON (JavaScript Object Notation) ist ein schlankes Datenaustauschformat, das für Menschen einfach zu lesen und zu schreiben und für Maschinen einfach zu parsen (Analysieren von Datenstrukturen) und zu generieren ist. Es basierd auf einer Untermenge der JavaScript Programmiersprache, Standard ECMA-262 dritte Edition - Dezember 1999.

Bei JSON handelt es sich um ein Textformat, das komplett unabhängig von Programmiersprachen ist, aber vielen Konventionen folgt, die Programmieren aus der Familie der C-basierten Sprachen (inklusive C, C++, C#, Java, JavaScript, Perl, Python und vielen anderen) bekannt sind. Diese Eigenschaften machen JSON zum idealen Format für Datenaustausch.

JSON baut auf zwei Strukturen auf:

- **Name/Wert Paare**. In verschiedenen Sprachen wird dies realisiert als ein Objekt (object), Satz (record), Struktur (struct), Wörterbuch bzw. Verzeichnis (dictionary), Hash-Tabelle (hash table), Schlüssel-Liste (keyed list) oder als ein assoziatives Array (associative array).

- **Eine geordnete Liste von Werten.** In den meisten Sprachen wird das als Array (array), Vektor (vector), Liste (list) oder Sequenz (sequence) realisiert.

Hierbei handelt es sich um universelle Datenstrukturen, die von so gut wie allen modernen Programmiersprachen in der einen oder anderen Form unterstützt werden. Es macht also Sinn, dass ein zwischen Programmiersprachen austauschbares Datenformat auch auf diesen Strukturen aufbaut.

In JSON gibt es:

Ojekte: Ein Objekt ist eine ungeordnete Menge von Name/Wert Paaren. Ein Objekt beginnt mit `{` (geschwungene Klammer auf) und endet mit `}` (geschwungene Klammer zu). Jedem Namen folgt ein `:` (Doppelpunkt) gefolgt vom Wert und die einzelnen Name/Wert Paare werden durch `,` (Komma) voneinander getrennt.

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

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

pair
- *string* `:` *value*

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

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

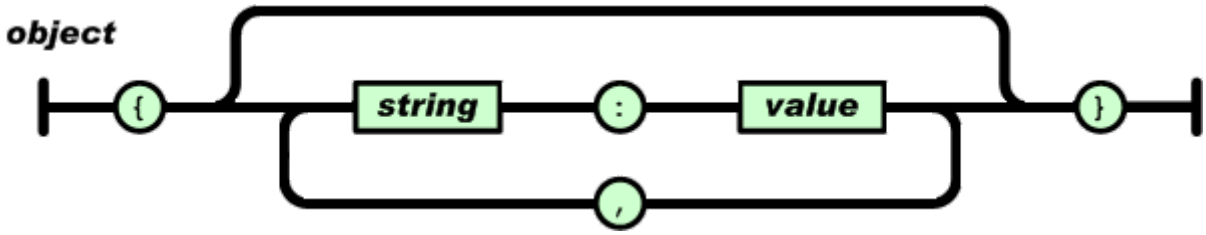
value
- *string*
- *number*
- *object*
- *array*
- `true`
- `false`
- `null`

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

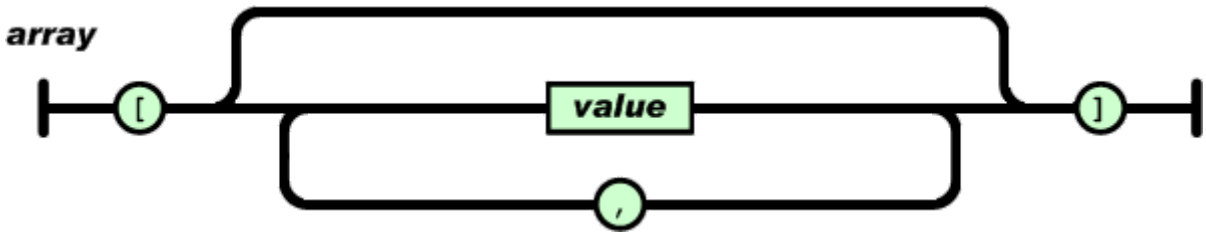
chars
- *char*
- *char chars*

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

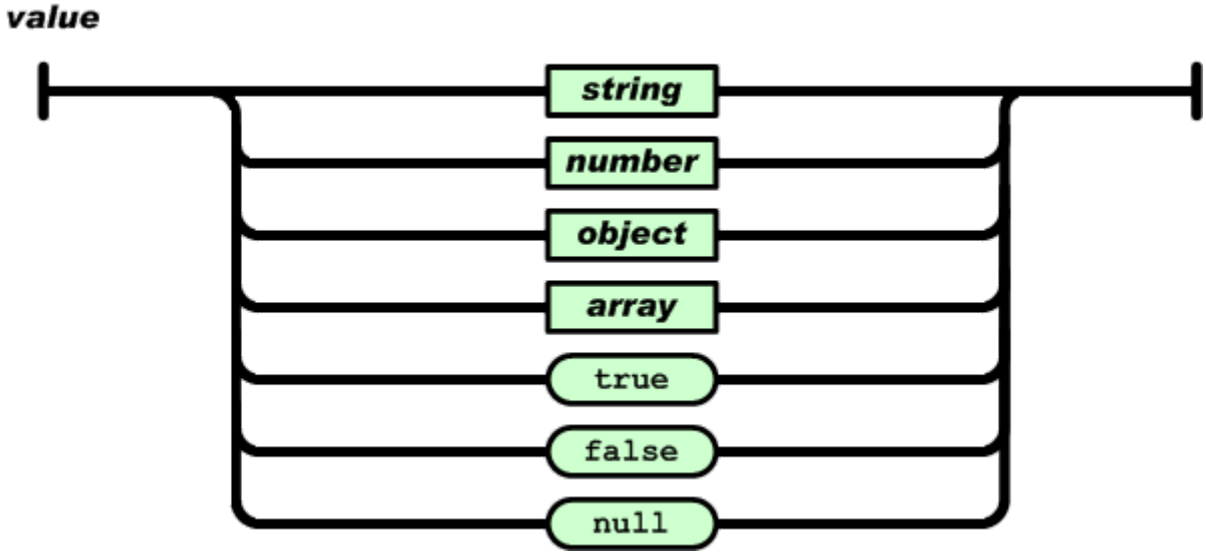
```
        \t
        \u four-hex-digits
number
        int
        int frac
        int exp
        int frac exp
int
        digit
        digit1-9 digits
        - digit
        - digit1-9 digits
frac
        . digits
exp
        e digits
digits
        digit
        digit digits
e
        e
    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



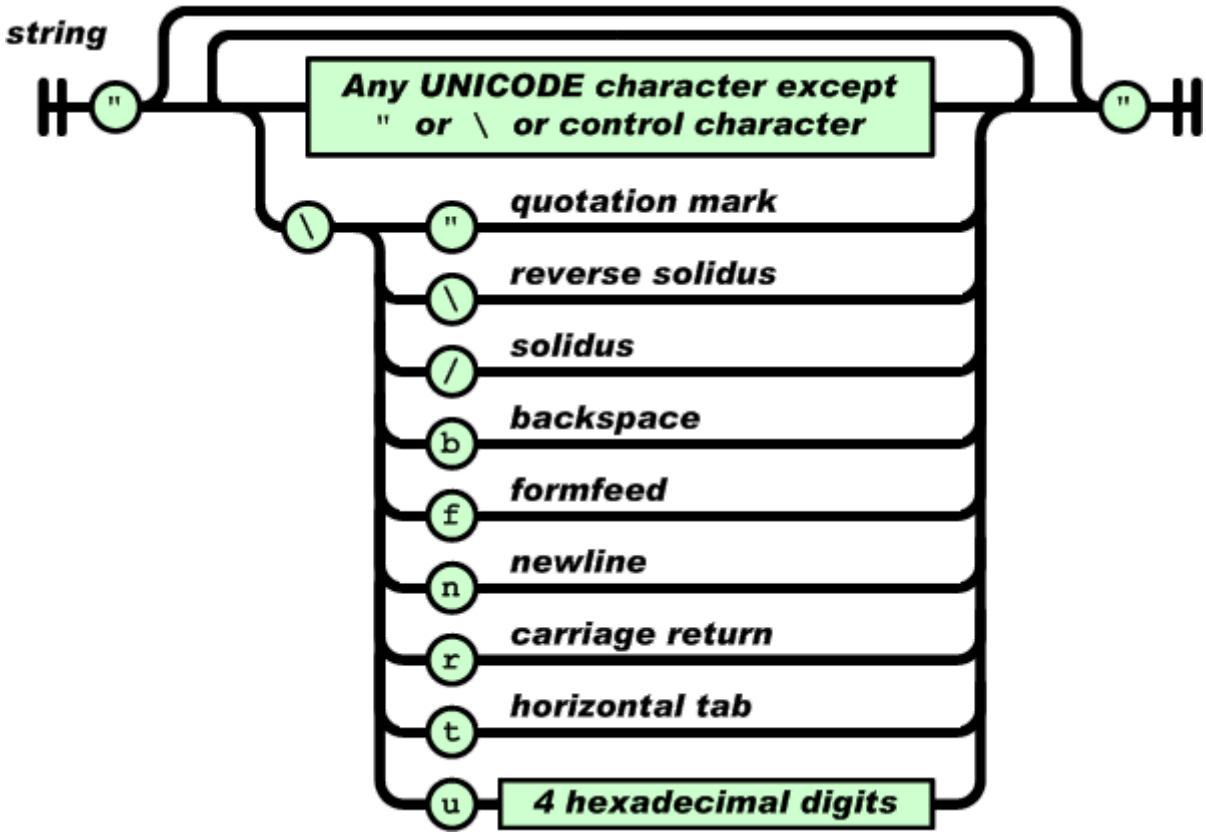
Ein **Array** ist eine geordnete Liste von Werten. Arrays beginnen mit `[` (eckige Klammer auf) und enden mit `]` (eckige Klammer zu). Werte werden durch `,` (Komma) voneinander getrennt.



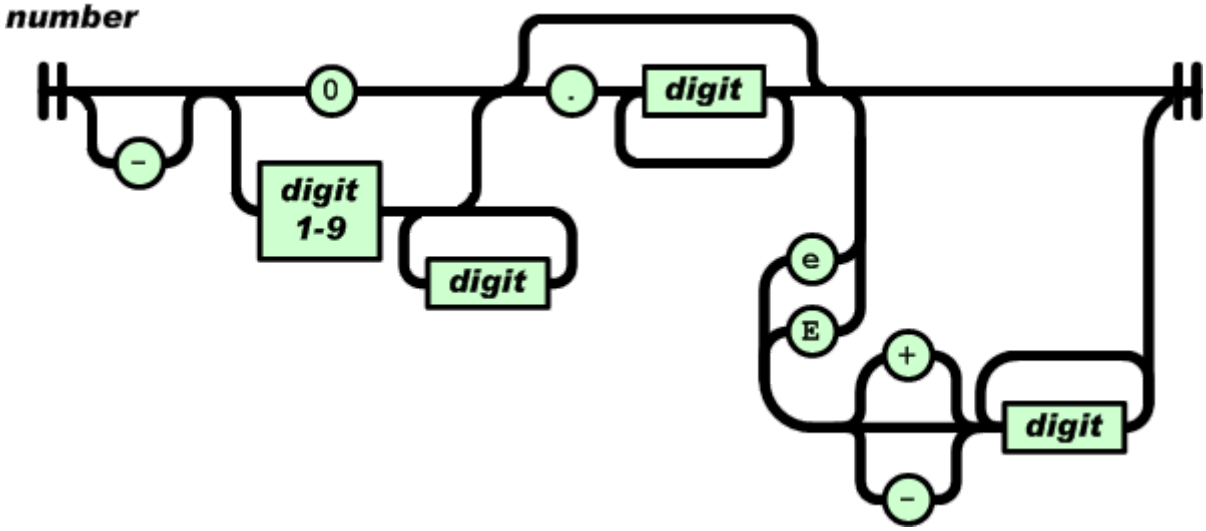
Ein **Wert** kann ein Objekt, ein Array, eine Zeichenkette (string), eine Zahl oder einer der Ausdrücke `true`, `false` oder `null` sein. Diese Strukturen können ineinander verschachtelt sein.



Eine **Zeichenkette** besteht aus keinem (leere Zeichenkette) oder mehr Unicode Zeichen und wird von doppelten Anführungszeichen umschlossen. Eine Zeichenkette kann Escape-Sequenzen mit einer besonderen Bedeutung enthalten. Ein einzelnes Zeichen wird durch eine Zeichenkette bestehend aus nur einem einzigen Zeichen dargestellt. Eine Zeichenkette (string) in JSON ist einer Zeichenkette in C oder Java sehr ähnlich.



Eine Zahl in JSON ist einer Zahl in C oder Java sehr ähnlich mit der Ausnahme, dass oktale und hexadezimale Zahlen nicht verwendet werden.



Leerzeichen können zwischen JSON-Elementen beliebig eingefügt werden.

It der Ausnahme einiger Details zur Enkodierung beschreibt das die gesamte Sprache.

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

    - 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.
- 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.
    - 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.
    - 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.
    - perl-JSON-SL.
- 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.