

jQuery & JSON



O que é JSON?

- **JSON (JavaScript Object Notation).**
- Trata-se de um modelo para armazenamento e transmissão de informações no formato texto. Tem sido utilizado por vários tipos de aplicações devido a sua capacidade de estruturar informações de uma forma bem mais compacta do que a conseguida pelo modelo XML, tornando mais rápido o acesso dessas informações.
- Google, Yahoo, Facebook, Amazon usam o JSON para saída de suas APIs. (<https://developer.twitter.com/>)
- Além de ser um formato leve para troca de dados é também muito simples de ler.

Sintaxe do JSON

Para cada valor representado, atribui-se um nome (ou rótulo) que descreve o seu significado.

Por exemplo, para representar o curso JavaScript, utiliza-se a seguinte sintaxe:

```
"curso": "JavaScript"
```

```
"duracao": 30
```

O par nome/valor deve ser representado pelo nome entre aspas duplas, seguido de dois pontos, seguido do valor.

Os valores podem possuir apenas 3 **tipos básicos**:

- * numérico (inteiro ou real),
- * booleano
- * string.

Sintaxe do JSON

A partir dos tipos básicos, é possível construir **tipos complexos**: array e objeto.

Os **arrays** são delimitados por colchetes, com seus elementos separados entre vírgulas:

```
“estado”: [“RJ”, “SP”, “MG”, “ES”, “GO”]
```

Sintaxe do JSON

Um objeto JSON pode representar, virtualmente, qualquer tipo de informação:

```
{  
  "titulo": "Forrest Gump",  
  "resumo": "Mesmo com o raciocínio lento, Forrest Gump nunca se sentiu desfavorecido...",  
  "anoLancamento": 1994,  
  "genero": "drama"  
}
```

Sintaxe do JSON

É possível representar mais de um objeto ou registro de uma só vez. No próximo exemplo, teremos 2 cursos representados em um array.

```
[ {  
  "titulo": "JAVA",  
  "resumo": "XXXXXXXXXXXXX",  
  "ano": 2019,  
  "duracao": 60  
},  
{  
  "titulo": "JAVASCRIPT",  
  "resumo": "yyyyyyyyyyy",  
  "ano": 2019,  
  "duracao": 30  
} ]
```

Sintaxe do JSON

Pode-se escrever numa única linha:

```
[  
  { "titulo": "JAVA", "resumo": "XXXXXXXXXXXXX", "ano": 2019, "duracao": 60 },  
  { "titulo": "JAVASCRIPT", "resumo": "yyyyyyyyyyy", "ano": 2019, "duracao": 30 }  
]
```

Sintaxe do JSON

Para representações de valores nulos, usar a palavra-chave “null”:

```
“idade”: null
```


HTML, XML e JSON

A resposta a uma solicitação Ajax geralmente é dada em um dos 3 formatos: HTML, XML e JSON. Segue uma comparação entre esses formatos:

HTML	XML	JSON
Para atualizar uma seção de uma página Web, essa é a maneira mais simples de inserir dados em uma página.	XML é parecida com HTML, mas os nomes das tags são diferentes porque descrevem os dados que essas tags contém.	Usa uma sintaxe semelhante à notação de objeto literal. var hotel = { nome: 'Royal Palm Plaza', qtdequartos: 890, endereco: 'Rua Xpto, 123' }
VANTAGENS: é fácil de escrever, solicitar e exibir. Os dados enviados do servidor vão diretamente para a página. Sem processamento do navegador.	VANTAGENS: é um formato de dados flexível e pode representar estruturas complexas. É processado usando os mesmos métodos DOM que os da HTML.	VANTAGENS: é mais conciso que o HTML/XML. É comum usá-lo com Javascript e aplicações Web.
DESVANTAGENS: O servidor gera HTML num formato que esteja pronto para uso da página. Não tem portabilidade de dados. A solicitação deve vir do mesmo domínio.	DESVANTAGENS: linguagem prolixa pq as tags adicionam gde qtde de caracteres extras. A solicitação tem de vir do mesmo domínio da página. Ela pode exigir muito código para processar o resultado.	DESVANTAGENS: a sintaxe não é tolerante. Uma aspa, vírgula ou dois-pontos ausente pode quebrar o código. Usar json que foi produzido por fontes confiáveis.

Vantagens de usar JSON

- Leitura mais simples
- Analisador sintático (parsing) mais fácil
- JSON suporta objetos
- Velocidade maior na execução e transporte de dados
- Arquivo com tamanho reduzido

<https://www.json.org/>

EXEMPLO

<https://jsoneditoronline.org/#/>

The screenshot displays the JSON Editor Online interface. The left pane shows the raw JSON text, and the right pane shows a hierarchical tree view of the same data.

JSON Text (Left Pane):

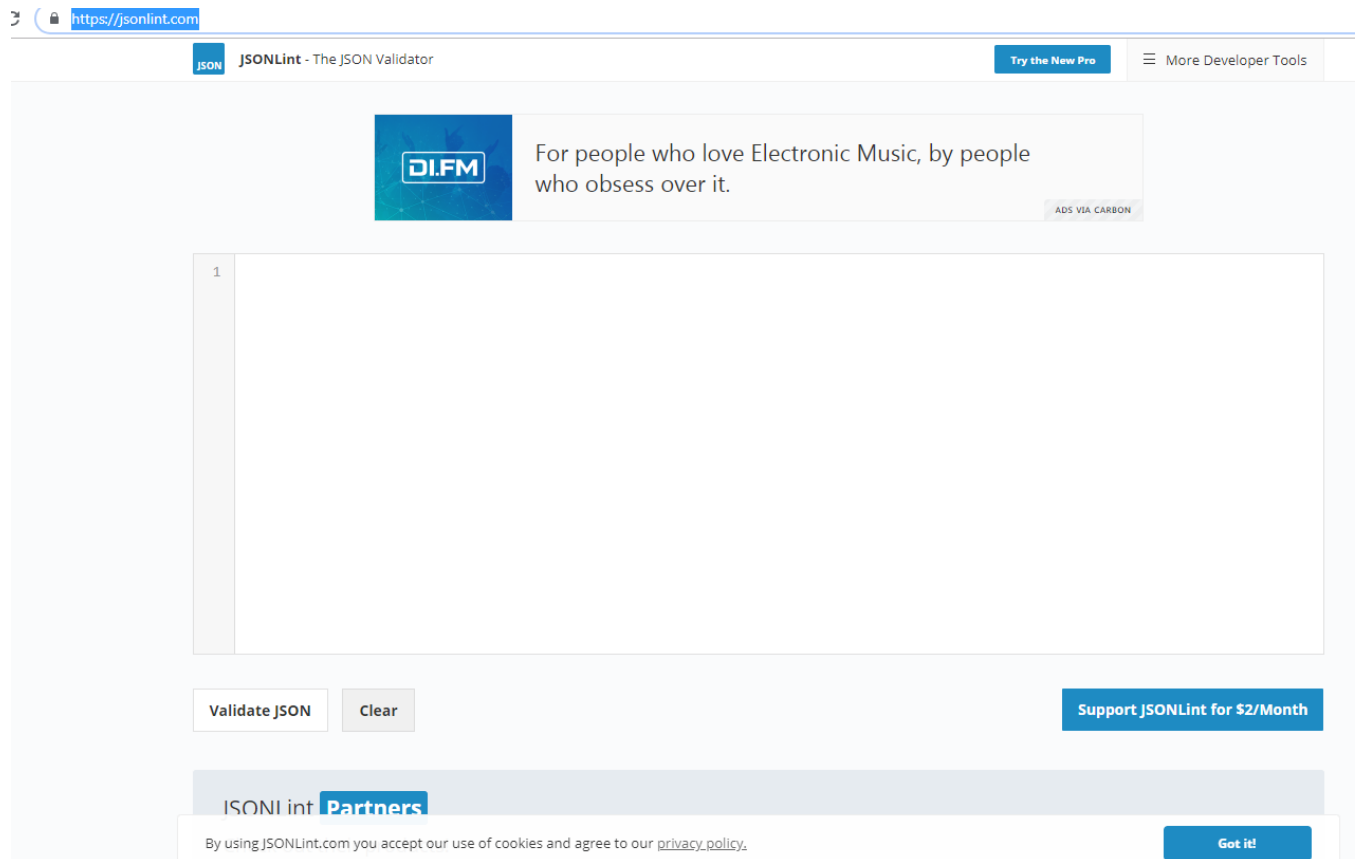
```
1- [
2- {
3-   "nome" : "Marcia",
4-   "endereco" : "Rua XPT0, 345 - apto 99",
5-   "email" : "marciac@unicamp.br",
6-   "celular" : "(19)98987-5656",
7-   "hobbies" : ["dormir", "andar", "cantar", "costurar"]
8- },
9- {
10-  "nome" : "Vitor",
11-  "endereco" : "Rua FuiEVoltei, 1234",
12-  "email" : "vitor@ig.com.br",
13-  "celular" : "(11)99874-9999",
14-  "hobbies" : ["cantar", "estudar", "tocar violao"]
15- }
16- ]
```

Tree View (Right Pane):

- array ▶ 1 ▶ hobbies ▶
 - ▼ array [2]
 - ▼ 0 {5}
 - nome : Marcia
 - endereco : Rua XPT0, 345 - apto 99
 - email : marciac@unicamp.br
 - celular : (19)98987-5656
 - hobbies [4]
 - 0 : dormir
 - 1 : andar
 - 2 : cantar
 - 3 : costurar
 - ▼ 1 {5}
 - nome : Vitor
 - endereco : Rua FuiEVoltei, 1234
 - email : vitor@ig.com.br
 - celular : (11)99874-9999
 - hobbies [3]
 - 0 : cantar
 - 1 : estudar
 - 2 : tocar violao

EXEMPLO

<https://jsonlint.com/>



The screenshot shows the JSONLint website interface. At the top, the browser address bar displays <https://jsonlint.com/>. The website header includes the JSONLint logo, the text "JSONLint - The JSON Validator", a "Try the New Pro" button, and a "More Developer Tools" link. Below the header is a large advertisement for D.I.F.M. with the text "For people who love Electronic Music, by people who obsess over it." and a small "ADS VIA CARBON" label. The main content area is a large, empty text box for pasting JSON code. Below the text box are two buttons: "Validate JSON" and "Clear". To the right of these buttons is a blue button that says "Support JSONLint for \$2/Month". At the bottom of the page, there is a footer section with the text "JSONLint int Partners" and a small "Got it!" button. A privacy policy notice is also visible at the bottom, stating "By using JSONLint.com you accept our use of cookies and agree to our [privacy policy](#)."

EXEMPLO

<https://mholt.github.io/json-to-go/>

← → ↻ 🔒 https://mholt.github.io/json-to-go/ 📄 ☆ 👤 ⋮


JSON-to-Go



Convert JSON to Go struct

This tool instantly converts JSON into a Go type definition. Paste a JSON structure on the left and the equivalent Go type will be generated to the right, which you can paste into your program. The script has to make some assumptions, so double-check the output!

For an example, try converting JSON from the [SmartyStreets API](#) or the [GitHub API](#).

© 2015 [Matt Holt \(@mholt6\)](#) • [View on GitHub](#) • [Dark mode](#)

JSON to 



JSON → Go ☒ Inline type definitions

Paste JSON here

Go will appear here

EXEMPLO

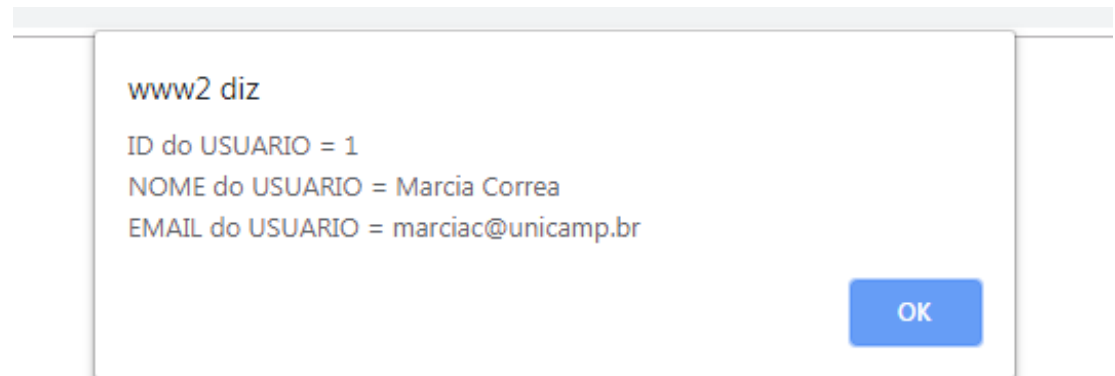
<https://chrome.google.com/webstore/detail/json-formatter/bcjindcccaagfpapjjmafapmmgkkhgoa?hl=pt-BR>

- 8th:
 - json.
- 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.
 - microjson.
 - mjson.
 - progbase.
- C++:
 - JSONKit.
 - jsonme--.
 - ThorsSerializer.
- 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.
 - Stringtree.
 - SOJO.
 - json-taglib.
 - Flexjson.
 - Argo.
 - jsonij.
 - fastjson.
 - mjson.
- 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.
- Prolog:
 - Jekejeke.
- PureBasic:
 - JSON.
- Puredata:
 - PuRestJson.
- Python:
 - The Python Standard Library.
 - simplejson.
 - pyson.
 - Yajl-Py.
 - ultrajson.
 - metamaclic.json.
 - progbase.
- R:

Linguagens com Bibliotecas que criam formato JSON para entrada/saída. Transformam dados nativos e transformam numa representação de formato JSON.

EXEMPLO

```
<html>
<head><title></title>
<script>
  var request = new XMLHttpRequest();
  var url = 'http://www2/marcia/capitulo%208/usuario.json';
  request.open('GET',url,true);
  request.onload = function() {
    var dados = JSON.parse(request.responseText);
    var mensagem = "ID do USUARIO = " + dados[0].idUsuario + "\n";
    mensagem += "NOME do USUARIO = " + dados[0].nomeUsuario + "\n";
    mensagem += "EMAIL do USUARIO = " + dados[0].email;
    alert(mensagem);
  }
  request.send();
</script></head>
<body>
</body></html>
```



EXEMPLO

```
var xhr = new XMLHttpRequest();
xhr.onload = function() {
  if (xhr.status === 200) {
    var objeto = JSON.parse(xhr.responseText);
    var dados = "";
    for (var i = 0; i < objeto.length; i++) {
      dados += '<div class="quadro">';
      dados += '<b>Cidade:' + objeto[i].cidade + '</b><br>';
      dados += '';
    }
    document.getElementById('resultado').innerHTML = dados;
  }
};
xhr.open('GET','http://www2/marcia/capitulo%208/exemplo%202/cidades.json',true);
xhr.send();
```

Cidade:Araxá



Cidade:Rio de Janeiro



Cidade:São Paulo



Cidade:Campinas

