

DESIGN DE INTERFACE PARA WEB

AULA 10 – JSON E MECANISMOS PARA ARMAZENAMENTO DE DADOS

WALTER TRAVASSOS SARINHO

@WALTEROPROFESSOR

WALTER.TRAVASSOS@UNIPE.EDU.BR



Mapa de Estudos

AULA 01

Arquitetura de Sistemas Web e sua evolução em camadas

AULA 03

Formulários HTML

AULA 05

Tipos de dados JavaScript. Principais operadores e introdução ao DOM com getElementById

AULA 06

Estruturas de decisão, repetição e arranjos em JavaScript.

AULA 02

Introdução ao HTML, principais TAGS

AULA 04

Introdução ao JavaScript. Funções declaradas e eventos HTML

Avaliação A1

Mapa de Estudos

AULA 07

Funções:
addEventListener,
createElement(),
appendChild() e
removeChild().
Hoisting e expressões
de função

AULA 09

Template
Strings, for in e
typeof

AULA 11
API JQuery
e AJAX

AULA 12, 13, 14...
NodeJS

AULA 08

Orientação a Objetos de
forma prototipada (antes do
ECMA Script 6)
Criação de classes (depois do
ECMA Script 6)

AULA 10
API JSON e
API Web Storage

Avaliação A2



JS

Aula de Hoje

- Introdução ao JSON.
- Convertendo String para JSON e JSON para String.
- Armazenamento de dados com Web Storage.
- Exercícios.

Introdução ao JSON

- JSON significa JavaScript Object Notation.
- Então isso significa que só podemos utilizar JSON no JavaScript?
- Na verdade não.
- Ele foi desenvolvido originalmente para JavaScript mas atualmente tem suporte para outras linguagens.

<https://www.json.org/json-pt.html>

8th	ColdFusion	Net.Data
json	SerializeJSON	netdata-json
ActionScript	D	Nim
ActionScript3	std.json	Module json
Ada	asdf	Objective C
GNATCOLL.JSON	vibe.data.json	NSJSONSerialization
AdvPL	Dart	json-framework
JSON-ADVPL	json library	JSONKit
APL	Delphi	yajl-objc
□JSON	Delphi Web Utils	TouchJSON
ASP	JSON Delphi Library	OCaml
JSON for ASP	E	jsonm
JSON ASP utility class	JSON in TermL	PascalScript
AWK	Erlang	JsonParser
JSON.awk	erl-json	Perl
rhawk	Fantom	CPAN
BlitzMax	Json	Photoshop
bmx-tjson	FileMaker	JSON Photoshop Scripting
C	JSON	PHP
JSON_checker	Fortran	PHP 5.2
YAJL	json-fortran	PicoLisp
LibU	YAJL-Fort	picolisp-json
json-c	jsonff	Pike
json-parser	Go	Public.Parser.JSON
jsonsl	package json	Public.Parser.JSON2
WJElement	Groovy	PL/SQL
M's JSON parser	groovy-io	pljson
cJSON	Haskell	PureBasic
Jansson	RJson package	JSON
jsmn	json package	Puredata
parson	Java	PuRestJson
ujson4c	JSON-java	Python
frozen	JSONUtil	The Python Standard Library
microison	jsonp	simplejson

Na página inicial do JSON vemos uma gama de sugestões de APIs para uso em diversas linguagens. Não existe exclusividade para JavaScript.

Introdução ao JSON

- É uma sintaxe para armazenamento e transferência de dados.
- JSON encapsula dados textuais, escritos via notação de objetos JavaScript.
- Pode-se converter qualquer arquivo JSON em objetos JavaScript, e vice-versa.
- Não há necessidade de traduções ou parsing complicados!

Formato simples



“O JSON além de ser um formato leve para troca de dados é também **muito simples de ler**. Mas quando dizemos que algo é simples, é comparando a algo mais complexo, como por exemplo, o formato XML.”

XML

JSON

```
1 <note>
2 <to>Tove</to>
3 <from>Jani</from>
4 <heading>Reminder</heading>
5 <body>Don't forget me this weekend!</body>
6 </note>
```

```
1 {
2   "id":1,
3   "nome":"Alexandre Gama",
4   "endereco":"R. Qualquer"
5 }
```


Cuidado com as Fake News!

<https://www.infoq.com/br/news/2013/11/xml-json-performance/>

Checagem dos fatos é um método simples que averigua a veracidade da informação.

Muitas das suposições sobre o quão lento, dispendioso e "gordo" o [XML](#) é se comparado à leveza do [JSON](#), não foram sustentadas pelo teste feito por [David Lee](#), engenheiro líder na [Marklogic](#), após a execução de um experimento "crowd sourcing" com 33 documentos diferentes e aproximadamente 1200 testes, em uma grande quantidade de navegadores e sistemas operacionais mais utilizados. David [afirma](#) ter descoberto que o desempenho total da experiência de usuário – transferência, análise (parsing) e consulta (querying) de um documento – é quase idêntico em ambos os formatos: XML e JSON.

E para finalizar, David deixa um conselho:

“

Não confie em ninguém.

Não acredite cegamente no que dizem. Faça experiências, teste seus próprios dados e codifique com os seus próprios usuários e dispositivos. O que "parece óbvio" nem sempre é verdade.

Compreendendo a estrutura do JSON

Criando um objeto JSON

- Para inserir um objeto você vai colocar entre chaves um par de propriedade (string) e valor, separado por **dois pontos**.

```
{ "nome" : "Walter" }
```

- Caso queira adicionar mais um atributo, separa-se por **vírgula**:

```
{ "nome" : "Walter" , "profissão" : "Professor" }
```

Criando um array no JSON

- Se quisermos representar vários objetos, utilizamos um array.
- Ele é delimitado por **colchetes** e cada elemento é separado por uma vírgula.

```
[ { "nome": "Walter" } , { "telefone" : "(83) 999999999" } ]  
  { "cores" : [ "azul" , "amarelo" , "vermelho" ] }
```


Questão 1

Resposta: D

A Figura apresenta um documento de texto em um formato utilizado para troca de dados entre cliente e servidor.

Que formato é esse?

- a) CSS.
- b) CSV.
- c) HTML.
- d) JSON.
- e) XML.

```
{  
  "titulo": "A Internet das Coisas",  
  "area": "Tecnologia",  
  "palavras-chave": ["internet", "tecnologia", "dispositivo"],  
  "ano": 2021  
}
```

Questão 2

Resposta: E

O script JSON corretamente formado é:

- a) {loja : "ImportCar", ranking:null, carros : ["Audi","BMW"]}
- b) {"loja" : "ImportCar", "ranking" : null, "carros" : {"Audi","BMW"}}
- c) {loja : ImportCar, ranking:null, carros : ["Audi","BMW"]}
- d) {loja : ImportCar,ranking:null, carros:[Audi;BMW]}
- e) {"loja" : "ImportCar" , "ranking" : null, "carros" : ["Audi","BMW"]}

Vantagens do JSON

- Leitura mais simples.
- Analisador (parsing) mais fácil.
- JSON suporta objetos! Sim, ele é tipado!
- Velocidade maior na execução e transporte de dados.
- Arquivo com tamanho reduzido.
- Quem utiliza? Google, Facebook, Yahoo!, Twitter...

Convertendo Objetos JavaScript para JSON

E vice-versa.

Método **JSON.stringify**

O `JSON.stringify()` é um dos vários métodos nativos da linguagem de programação JavaScript. Sua principal função é **converter** valores e objetos denotados na linguagem Javascript em uma String JSON.

```
const pessoa = {  
    nome: "Walter",  
    idade: 42  
};  
const strJSON = JSON.stringify(pessoa);
```

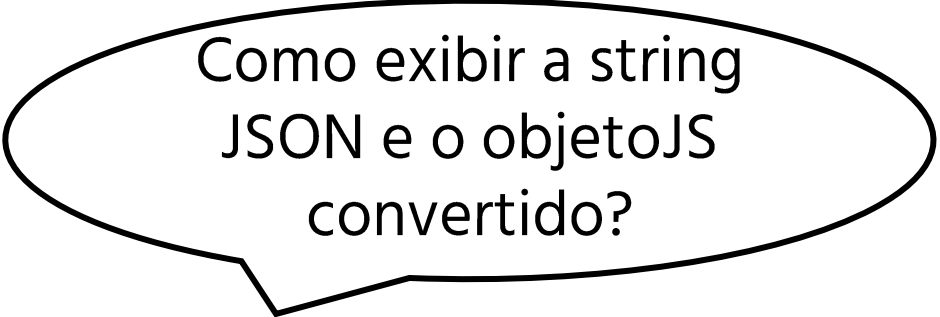

Método **JSON.parse**

O método `JSON.parse()` analisa uma String JSON e, por sua vez, a transforma em um objeto em JavaScript.

```
const pessoa = { nome: "Walter", idade: 42 };
```

```
const strJSON = JSON.stringify(pessoa); //objeto para JSON
```

```
const objetoJS = JSON.parse(strJSON); //JSON para objeto
```



Como exibir a string
JSON e o objetoJS
convertido?



JSON.stringify()

Objeto em JavaScript para String JSON.



{j s o n}



JSON.parse()

String JSON para objeto em JavaScript.

Exercício 1

- Converter um objeto para uma string JSON:
 - Converta um objeto JavaScript em uma string JSON usando `JSON.stringify`.
 - Em seguida, exiba a string JSON no console.
-
- `const objeto = { nome: "Alice", idade: 25 };`

Exercício 1 - Resposta

```
const objeto = { nome: "Alice", idade: 25 };  
const jsonString = JSON.stringify(objeto);  
console.log(jsonString);
```

Exercício 2

- Converter uma string JSON em um objeto:
- Converta uma string JSON de volta para um objeto JavaScript usando `JSON.parse`.
- Em seguida, acesse propriedades do objeto e exiba-as no console.
- `const jsonString = '{"nome": "Bob", "idade": 30}';`

Exercício 2 - Resposta

```
const jsonString = '{"nome": "Bob", "idade": 30}';  
const objeto = JSON.parse(jsonString);  
  
for (const propriedade in objeto) {  
    console.log(propriedade + ": " + objeto[propriedade]);  
}
```

Conversão de texto no HTML para JSON

- O próximo exercício tem como objetivo praticar a conversão de valores inseridos em campos de texto (nome e e-mail) em uma representação JSON usando JavaScript.
- Você deve criar uma página HTML que permita aos usuários inserir seu nome e e-mail em campos de texto separados, e, em seguida, converter esses valores em uma representação JSON quando um botão for clicado.

Exercício 3

Exercício de Conversão para JSON - Nome e E-mail

Nome:

E-mail:

Usuário em JSON: {"nome":"Walter","email":"travassoswalter@gmail.com"}

Crie uma página HTML com dois campos de texto, um para o nome e outro para o e-mail, um botão e uma área de exibição de resultados. Quando o botão "Converter para JSON" for clicado, o JavaScript deve:

- a) Pegar o valor inserido nos campos de texto de nome e e-mail.
- b) Criar um objeto JavaScript que contenha o nome e o e-mail.
- c) Converter o objeto JavaScript em uma string JSON usando `JSON.stringify`.
- d) Exibir a representação JSON do nome e do e-mail.

Exercício 3 – Resposta HTML

```
10 v <body>
11   <h1>Exercício de Conversão para JSON - Nome e E-mail</h1>
12   <label for="nome">Nome:</label>
13   <input type="text" id="nome" placeholder="Seu nome"><br><br>
14   <label for="email">E-mail:</label>
15   <input type="text" id="email" placeholder="Seu e-mail"><br><br>
16   <button id="converter">Converter para JSON</button>
17   <div id="resultado"></div>
```

Exercício 3 – Resposta JavaScript

```
19 <script>
20     // Função que é acionada quando o botão é clicado
21 document.getElementById("converter").addEventListener("click", function() {
22     // Obtém os valores do campo de nome e campo de e-mail
23     const nome = document.getElementById("nome").value;
24     const email = document.getElementById("email").value;
25
26     // Cria um objeto JavaScript com os valores
27     const usuario = { nome, email };
28
29     // Converte o objeto JavaScript em uma string JSON
30     const jsonUsuario = JSON.stringify(usuario);
31
32     // Apresenta o resultado no HTML
33     const resultadoDiv = document.getElementById("resultado");
34     resultadoDiv.innerHTML = `<p>Usuário em JSON: ${jsonUsuario}</p>`;
35
36     // Agora você pode usar jsonUsuario como desejado
37 });
38 </script>
```


Mecanismos para Armazenar Dados no Cliente

JavaScript oferece 3 mecanismos: Local Storage, Session Storage e Cookies

Mecanismos para Armazenar Dados

Local Storage

- O armazenamento local é o mecanismo mais recente. Permite armazenar maiores quantidades de dados, mas os dados não são excluídos quando o navegador é fechado. O armazenamento local é útil para armazenar dados que o usuário precisará acessar posteriormente, como dados offline.
- Muito usado para manter o usuário logado no site. A partir do login, geramos um token de acesso que é armazenado no local storage.
- Outro exemplo é o tema que o usuário escolheu: dark ou light. Ao final, essa preferência pode ser salva para quando o usuário retornar ao site.
- Outra possibilidade é quando navegamos muito numa determinada página de um produto, quando voltamos, essa preferência pode ser utilizada para oferecer o produto.

Session Storage

- O armazenamento da sessão é semelhante aos cookies, mas os dados são armazenados apenas para a sessão atual. Isso significa que os dados serão excluídos quando o usuário fechar o navegador.
- O armazenamento de sessão é útil para armazenar dados confidenciais, como credenciais de login.

Cookies

- Os cookies são o mecanismo mais antigo e conhecido.
- Eles são simples de usar e bem suportados pelos navegadores.
- No entanto, eles são limitados a 4 KB de dados e são frequentemente usados para armazenar dados que não são confidenciais, como preferências do usuário.

Armazenando Dados JSON com API Web Storage

localStorage

Web Storage

A API chamada Web Storage provê dois mecanismos básicos para se guardar informações no browser do usuário: **sessionStorage** e **localStorage**.

Web Storage

- O **sessionStorage** mantém o dado gravado apenas até o término da sessão do usuário, ou seja, até o browser do usuário se fechar incluindo reloads da página ou restores.
- Já o **localStorage** mantém o dado gravado mesmo se o browser é fechado e reaberto. Isso facilita criar alguns comportamentos de interface durante o uso do usuário. É um **recurso simples de armazenamento**, portanto **não é recomendado para gravar dados sensíveis**.

4 métodos principais

- `localStorage.setItem()` - para criar um novo par de chave: valor.
- `localStorage.getItem()` - para recuperar o valor do par chave: valor.
- `localStorage.removeItem()` - para remover um par específico.
- `localStorage.clear()` - apagar TODOS os pares gravados.

Exemplo – `setItem()` e `getItem()`

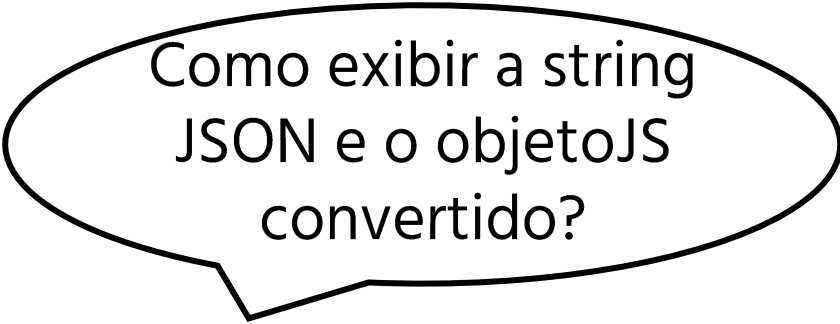
```
const pessoa = { nome: "Walter", idade: 42 };
```

```
//Armazenando o JSON
```

```
const strJSON = JSON.stringify(pessoa);  
localStorage.setItem("JSONsalvo", strJSON);
```

```
//Recuperando o JSON
```

```
const strRecuperada = localStorage.getItem("JSONsalvo");  
const objeto = JSON.parse(strRecuperada);
```



Como exibir a string
JSON e o objetoJS
convertido?

Cuidados, segurança e limitações

- Não use o localStorage para guardar dados sensíveis.
- Os dados que estão gravados não tem camada nenhuma de proteção de acesso, ou seja, todos os dados gravados ali podem ser acessados por qualquer código da sua página.
- Em qualquer browser, o localStorage é limitado a guardar apenas 5Mb de dados. Muito mais que os 4Kb dos cookies.
- Você só pode usar strings JSON no localStorage, e isso é um saco, porque limita a gravação de dados mais complexos e se você tentar fazer qualquer conversão se transforma numa gambiarra sem limites.

Exercício 4

- Atualizar um Valor no Objeto Armazenado no Local Storage
- Armazene um objeto JSON no localStorage usando o método `setItem`.
- Em seguida, atualize um valor da chave idade para 31 dentro do objeto armazenado e, em seguida, recupere e exiba o objeto atualizado no console.
- `const objeto = { nome: "Bob", idade: 30 };`

Exercício 4 - Resposta

// Armazenar no localStorage

```
localStorage.setItem("objetoSalvo", JSON.stringify(objeto));
```

// Atualizar um valor

```
const objetoSalvo = JSON.parse(localStorage.getItem("objetoSalvo"));
```

```
objetoSalvo.idade = 31;
```

```
localStorage.setItem("objetoSalvo", JSON.stringify(objetoSalvo));
```

// Recuperar do localStorage e exibir

```
const objetoRecuperado = JSON.parse(localStorage.getItem("objetoSalvo"));
```

```
console.log(objetoRecuperado);
```

Exercício 5

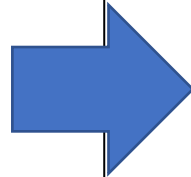
- Melhore o exercício 3 que converte nome e e-mail para JSON.
- Agora serão 2 botões, um que "Converter para JSON e Armazenar" e o outro para "Recuperar e Exibir JSON".

Exercício de Conversão para JSON - Nome e E-mail

Nome:

E-mail:

Usuário em JSON: {"nome":"Walter","email":"travassoswalter@gmail.com"}



Exercício de Conversão para JSON e Armazenamento no localStorage - Nome e E-mail

Nome:

E-mail:

Usuário em JSON: {"nome":"Walter","email":"travassoswalter@gmail.com"}

Exercício 5 – Resposta HTML

```
11 <body>
12   <h1>Exercício de Conversão para JSON e Armazenamento no localStorage - Nome e E-mail</h1>
13   <label for="nome">Nome:</label>
14   <input type="text" id="nome" placeholder="Seu nome"><br><br>
15   <label for="email">E-mail:</label>
16   <input type="text" id="email" placeholder="Seu e-mail"><br><br>
17   <button id="converter">Converter para JSON e Armazenar</button>
18   <button id="recuperar">Recuperar e Exibir JSON</button>
19   <div id="resultado"></div>
```

Exercício 5 – Resposta JavaScript – parte 1

```
21 v <script>
22     // Função que é acionada quando o botão "Converter para JSON e Armazenar" é clicado
23 v document.getElementById("converter").addEventListener("click", function() {
24     // Obtém os valores inseridos nos campos de nome e e-mail
25     const nome = document.getElementById("nome").value;
26     const email = document.getElementById("email").value;
27
28     // Cria um objeto JavaScript com os valores
29     const usuario = { nome, email };
30
31     // Converte o objeto JavaScript em uma string JSON
32     const jsonUsuario = JSON.stringify(usuario);
33
34     // Armazena a string JSON no localStorage
35     localStorage.setItem("usuarioJSON", jsonUsuario);
36
37     // Fornece feedback ao usuário
38     alert("Dados armazenados com sucesso!");
39 });
```


Exercício 5 – Resposta JavaScript – parte 2

```
41 // Função que é acionada quando o botão "Recuperar e Exibir JSON" é clicado
42 document.getElementById("recuperar").addEventListener("click", function() {
43     // Recupera a string JSON do localStorage
44     const jsonUsuario = localStorage.getItem("usuarioJSON");
45
46     // Apresenta o resultado no HTML
47     const resultadoDiv = document.getElementById("resultado");
48     if (jsonUsuario) {
49         resultadoDiv.innerHTML = `<p>Usuário em JSON: ${jsonUsuario}</p>`;
50     } else {
51         resultadoDiv.innerHTML = "<p>Nenhum dado encontrado no localStorage.</p>";
52     }
53 });
54 </script>
```

Sugestão

- Experimente também utilizar os métodos `localStorage.removeItem()` e `localStorage.clear()`.
- <https://tableless.com.br/guia-f%C3%A1cil-sobre-usar-localstorage-com-javascript/>

Lista de Exercícios

JSON e Localstorage

Referências

- <https://www.devmedia.com.br/o-que-e-json/23166>
- <https://www.json.org/json-pt.html>
- <https://www.infoq.com/br/news/2013/11/xml-json-performance/>
- <https://tableless.com.br/guia-f%C3%A1cil-sobre-usar-localstorage-com-javascript/>

DESIGN DE INTERFACE PARA WEB

AULA 10 – JSON E MECANISMOS PARA ARMAZENAMENTO DE DADOS

WALTER TRAVASSOS SARINHO

@WALTEROPROFESSOR

WALTER.TRAVASSOS@UNIPE.EDU.BR

