



## Programação Web Front-End

### Aula 5 - JavaScript

Profa. Rosangela de Fátima Pereira Marquesone  
romarquesone@utfpr.edu.br

**Proposta:** apresentar as funcionalidades da API Web Storage utilizando objetos JSON, via JavaScript.

**Objetivos:** espera-se que após essa aula, você tenha habilidade para compreender os seguintes tópicos:

1. [Aprender a manipular objetos JSON na API Web Storage](#)
2. [Atividade prática - Utilizando objetos JSON em localStorage](#)
3. [Atividade prática - Utilizando localStorage na lista de tarefas](#)

#### Dicas de aprendizado:

- Execute todos os passos com atenção, compreendendo o que está sendo realizado;
- Procure não copiar código, para ter a prática de digitar o código desenvolvido;
- Pergunte quando tiver alguma dúvida;
- Mantenha um histórico dos códigos desenvolvidos, seja no github ou em algum outro meio de armazenamento (e-mails, google drive, etc.);
- Tenha curiosidade e explore os recursos apresentados.

#### Tópicos anteriores:

- Compreender o que é HTML
- Compreender o que são tags HTML básicas
- Criar um arquivo .html no Visual Studio (VS) Code
- Abrir o arquivo .html em um navegador
- Visualizar o código-fonte de uma página em um navegador
- Inspeccionar a página em um navegador
- Utilizar o Live Server no VS Code
- Aprender a utilizar tags semânticas
- Aprender a inserir links
- Aprender a inserir listas
- Aprender a criar uma página com seu Curriculum Vitae (CV) (atividade prática)
- Aprender a inserir figuras
- Aprender a utilizar a tag semântica <figure>
- Inserir figuras em seu Curriculum Vitae (CV) (atividade prática)
- Aprender a criar formulários
- Criar um formulário (atividade prática)
- Descobrir o que é CSS
- Aprender a sintaxe do CSS

- Aprender os tipos de seletores CSS
- Aprender as formas de inclusão de CSS
- Aprender a definir cores
- Aprender a alterar as propriedades de texto
- Aprender o conceito de modelo de caixa do CSS
- Aprender a trabalhar com a margem
- Aprender a trabalhar com a borda
- Aprender a trabalhar com o preenchimento (padding)
- Aprender a usar a propriedade display
- Aprender a utilizar a propriedade float
- Aprender a utilizar a propriedade overflow
- Estruturar páginas por meio do modelo de caixa (atividade prática)
- Aprender o conceito de flex-box
- Aprender as propriedades do elemento pai (flex container)
- Aprender as propriedades dos elementos filhos (flex items)
- Descobrir a história da linguagem JavaScript
- Compreender as formas de uso da linguagem JavaScript
- Conhecer as características da linguagem JavaScript
- Atividade prática com JavaScript
- Aprender a utilizar JavaScript com DOM HTML
- Atividade prática com JavaScript
- Aprender a utilizar o método `querySelector()` e `querySelectorAll()`
- Aprender a utilizar o método `classList.add()`
- Aprender a utilizar o método `classList.remove()`
- Criar uma lista de tarefas com HTML, CSS e JavaScript
- Atividade prática com JavaScript - Alteração da lista de tarefas
- Aprender o que é a API Web Storage
- Aprender as funcionalidades do `localStorage`
- Aprender a utilizar o `localStorage` em um cadastro de login
- Atividade prática - Remoção de dados do armazenamento local

---

# Passo 1 - Aprender a manipular objetos JSON na API Web Storage

Vimos no tutorial anterior que a API Web Storage provê dois mecanismos para realizar o armazenamento dos dados no lado cliente: o `localStorage` e o `sessionStorage`. Como vimos também, cada um deles oferece os seguintes métodos para realizar esse gerenciamento dos dados:

- `setItem()`
- `getItem()`
- `removeItem()`
- `clear()`

Além desses métodos, é importante que saibamos também como trabalhar com objetos arrays nos valores a serem armazenados. Para isso, devemos lembrar que o `localStorage` e o `sessionStorage` permitem o armazenamento dos dados no formato chave/valor. Esse formato é muito utilizado na estrutura de dados JavaScript Object Notation, mais conhecido por sua sigla, JSON.

Como o próprio nome diz, JSON é o formato baseado em um padrão de texto para representar dados estruturados com base na sintaxe do objeto JavaScript. Os dados no formato JSON devem ser estruturados por meio de uma coleção de pares chave-valor, contendo:

- **chave:** referente ao identificador do conteúdo. Deve ser uma string delimitada por aspas duplas
- **valor:** representa o conteúdo correspondente à chave e pode conter os seguintes tipos de dados: string, array, object, number, boolean ou null.

Para ficar mais claro, podemos verificar a seguir um exemplo da estrutura JSON com um objeto.

```
{
  "titulo": "Aula de JavaScript",
  "resumo": "continuação sobre webstorage",
  "carga_horaria": 2,
  "topicos": ["webstorage", "localStorage", "json"]
}
```

E também a estrutura JSON com mais de um objeto

```
[
  {
    "titulo": "Aula de CSS",
    "resumo": "seletores CSS",
    "carga_horaria": 2,
    "topicos": ["css", "seletores", "regras css"]
  },
  {
    "titulo": "Aula de JavaScript",
    "resumo": "continuação sobre webstorage",
    "carga_horaria": 2,
    "topicos": ["webstorage", "localStorage", "json"]
  }
]
```

```
}  
]
```

Caso queira, você pode validar o arquivo JSON a partir deste [link](#).

Para atuar com o armazenamento dos dados com a API WebStorage a partir de um objetivo JSON, podemos utilizar dois métodos disponíveis:

- **JSON.stringify()**: transforma um objeto JavaScript em uma string JSON. Usado antes de armazenar um dados em localStorage.
- **JSON.parse()**: transforma um objeto JSON em um objeto JavaScript. Usado para recuperar um dado armazenado em localStorage.

Veja o exemplo a seguir:

```
const meuArray = ['ana', 'bia', 'carlos', 'dani'];  
  
// converter objeto JavaScript em string JSON  
const jsonArray = JSON.stringify(meuArray);  
  
// salvar os dados usando "nomes" como chave e jsonArray como valor  
localStorage.setItem('nomes', jsonArray);  
  
// obter a string JSON de localStorage  
const str = localStorage.getItem('nomes');  
  
// converter string JSON em objeto JavaScript  
const novoArray = JSON.parse(str);  
  
console.log(novoArray);
```

Perceba nesse exemplo que ocorreram as seguintes ações:

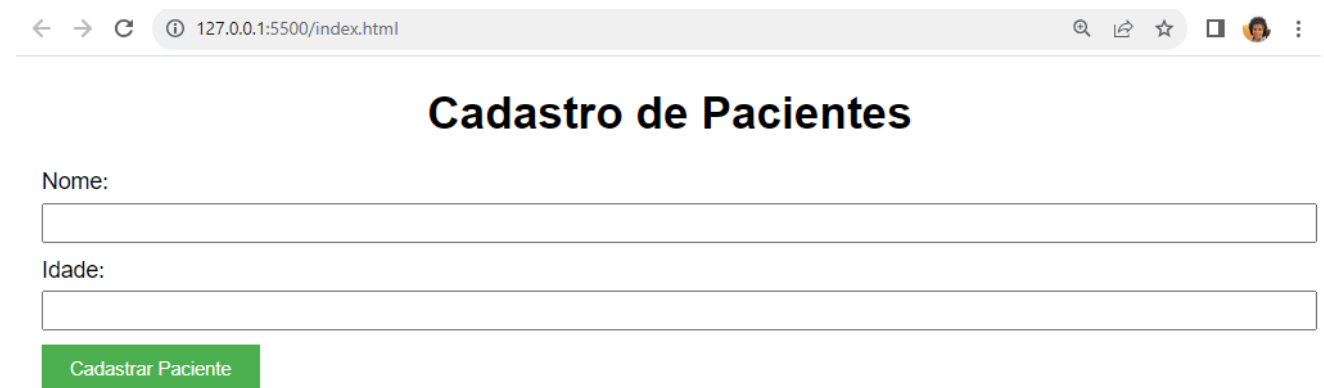
- Os itens do array são inicialmente convertidos em uma string JSON por meio do método `JSON.stringify()`.
- A partir disso, os itens são armazenados em pares de chave-valor no localStorage (poderia ser na sessionStorage também).
- Quando os itens são recuperados do armazenamento, a string JSON é analisada usando `JSON.parse()`.

De forma resumida, é importante você observar o seguinte padrão:

- Você deve usar "`JSON.stringify();`" antes de armazená-lo em localStorage.
- Você deve usar "`JSON.parse();`" após recuperá-lo do localStorage.

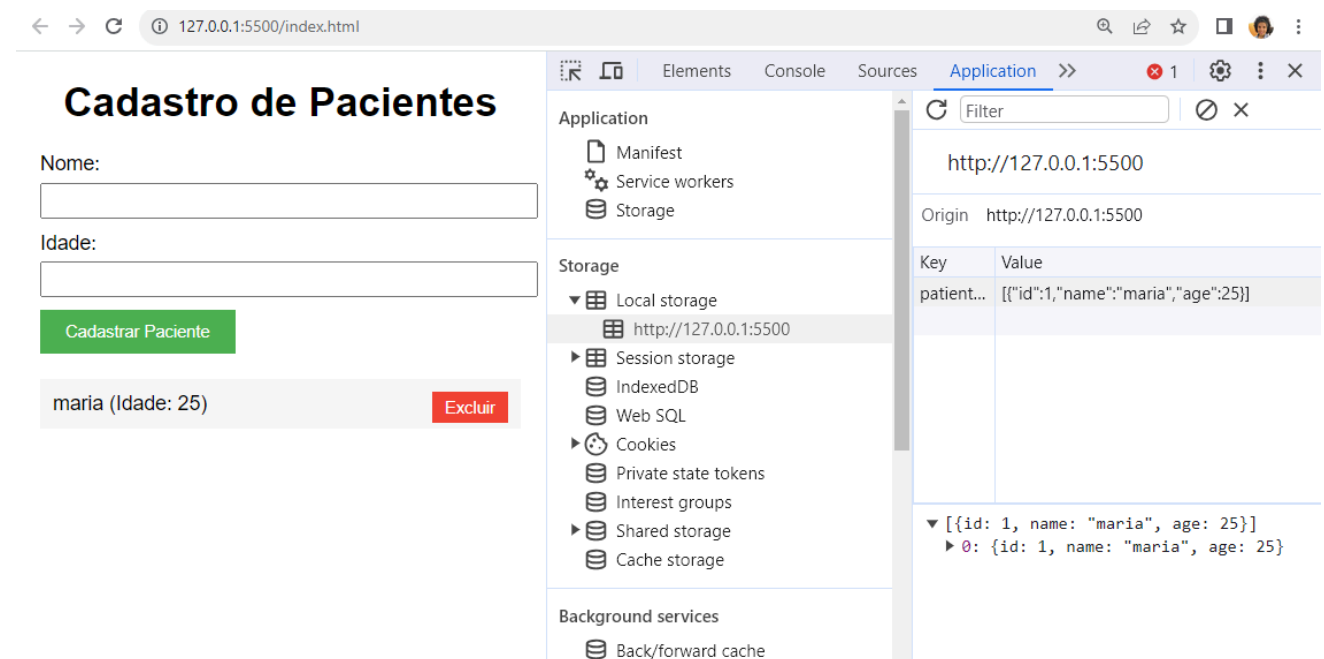
## Passo 2 - Atividade prática - Utilizando objetos JSON em localStorage

Para essa atividade, descompacte o arquivo "**aula5-javascript.zip**", e abra a pasta descompactada no Visual Studio Code. Após isso, abra no navegador o arquivo `index.html`, onde deverá aparecer uma página similar à figura a seguir:



The screenshot shows a web browser at `127.0.0.1:5500/index.html` displaying a form titled "Cadastro de Pacientes". The form has two input fields: "Nome:" and "Idade:". Below the "Idade:" field is a green button labeled "Cadastrar Paciente".

Insira os dados de um paciente e, após isso, verifique, a partir da ferramenta inspecionar, se os dados foram armazenados em formato JSON, similar a figura a seguir:



The screenshot shows the same "Cadastro de Pacientes" form, but now with a patient record displayed below the inputs: "maria (Idade: 25)" with a red "Excluir" button. The Chrome DevTools "Application" panel is open on the right, showing the "Storage" tab. Under "Local storage", the entry for `http://127.0.0.1:5500` is selected. The "Key" is "patient..." and the "Value" is `[{"id":1,"name":"maria","age":25}]`. The expanded view at the bottom shows the array structure: `[{"id": 1, name: "maria", age: 25}]` with an index `0` pointing to the object `{id: 1, name: "maria", age: 25}`.

Ao inserir um novo cadastro, perceba que um novo objeto será adicionado ao array, conforme a figura a seguir.

← → ↻ 127.0.0.1:5500/index.html

# Cadastro de Pacientes

Nome:

Idade:

Cadastrar Paciente

maria (Idade: 25)

Excluir

dani (Idade: 57)

Excluir

Application

- Manifest
- Service workers
- Storage

Storage

- Local storage
  - http://127.0.0.1:5500
- Session storage
- IndexedDB
- Web SQL
- Cookies
- Private state tokens
- Interest groups
- Shared storage
- Cache storage

Background services

- Back/forward cache
- Background fetch
- Background sync

Filter

http://127.0.0.1:5500

Origin http://127.0.0.1:5500

Key	Value
patientList	[{"id":1,"name":"maria","age":25},{"id":2,"name":"dani","age":57}]

▼ [{"id": 1, name: "maria", age: 25}, {"id": 2, name: "dani", age: 57}]

- 0: {"id": 1, name: "maria", age: 25}
- 1: {"id": 2, name: "dani", age: 57}

Por fim, verifique, também, que ao remover o item da lista, o objeto referente a esse item também será removido.

← → ↻ 127.0.0.1:5500/index.html

# Cadastro de Pacientes

Nome:

Idade:

Cadastrar Paciente

dani (Idade: 57)

Excluir

Application

- Manifest
- Service workers
- Storage

Storage

- Local storage
  - http://127.0.0.1:5500
- Session storage
- IndexedDB
- Web SQL
- Cookies
- Private state tokens
- Interest groups
- Shared storage
- Cache storage

Background services

- Back/forward cache
- Background fetch
- Background sync

Filter

http://127.0.0.1:5500

Origin http://127.0.0.1:5500

Key	Value
patientList	[{"id":2,"name":"dani","age":57}]

▼ [{"id": 2, name: "dani", age: 57}]

- 0: {"id": 2, name: "dani", age: 57}

---

## Passo 3 - Atividade prática - Utilizando localStorage na lista de tarefas

Chegou o momento de colocar em prática o conteúdo apresentado.

Para essa atividade, você deverá retornar ao projeto realizado no tutorial "Aula 3 - JavaScript", referente à criação de uma lista de tarefas (TODO list). Você agora deverá realizar a seguinte atividade nesse projeto:

- Altere o projeto referente à lista de tarefas e faça a inclusão dos dados da tarefa (ID e descrição) em um localStorage, a partir de uma estrutura JSON.
- Exclua o item armazenado caso alguma tarefa seja excluída.

---

## Considerações finais

Caso tenha chegado até aqui, você conseguiu completar o conteúdo do quinto tutorial de JavaScript, referente à API Web Storage. A partir desses recursos, você passa a compreender cada vez mais a base para o desenvolvimento de funcionalidades via JavaScript. Nas aulas seguintes veremos ainda mais funcionalidades para tornar as páginas ainda mais dinâmicas.

Bom estudo!