02

Browser possui banco de dados? Conheça o IndexedDB!

Transcrição

Antes de integrarmos o IndexedDB na nossa aplicação, vamos estudar a API separadamente e entender o seu funcionamento. Depois, faremos a integração. Dentro da pasta `CLIENT`, nós criaremos o arquivo `aprendendo_indexeddb.html`. Usaremos o plugin Emmet, e digitando `!`, o código básico do HTML será preenchido automaticamente:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Aprendendo IndexedDB</title>
</head>

<body>
</body>
</html>
```

Primeiramente, veremos o funcionamento da API, depois pensaremos na questão de organização e boas práticas...

Voltaremos para o navegador, e no Console, digitaremos `window`.

```
Window {external:Object, document: document, negociacaoController: negociacaoController, speechS
```

Conseguimos ver o escopo global do JavaScript. Em seguida, usaremos a seguinte instrução:

```
window.indexedDB
```

E a saída será:

```
IDBFactory {}
```

Trata-se de uma "fábrica" para criarmos bancos no IndexedDB - temos a opção de acessá-lo também diretamente.

De volta ao VS Code, adicionaremos a tag `<script>` e abriremos uma conexão com o banco. Na verdade, pediremos uma requisição de abertura - talvez, não funcione:

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
</head>
<body>
   <script>
```

```
        var openRequest = window.indexedDB.open("aluraframe",1);

    </script>
</body>
```

O método `indexedDB.open()` retorna uma instância de IDBOpenDBRequest, ou seja, uma requisição de abertura do banco. Precisaremos lidar com um tríade de eventos disparados quando tentarmos acessar um banco no IndexedDB. A seguir, adicionaremos a tríade:

- openRequest.onupgradeneeded;
- openRequest.onsucess;
- openRequest.onerror;

Vamos começar a trabalhar com `openRequest.onupgradeneeded` :

```
<body>
    <script>

        var openRequest = window.indexedDB.open('aluraframe',1);

        openRequest.onupgradeneeded = function(e) {

                console.log('Cria ou altera um banco já existente');
        };

    </script>
</body>
```

Passamos para `onupgradeneeded` uma função que recebe um evento (`e`). E no `console` , adicionamos uma mensagem que informa sua utilidade: `Cria ou altera um banco já existente` . Pediremos para abrir pela primeira vez o `aluraframe` , que será criado no meu navegador. O código de atualização também será no `onupgradeneeded` . Minha sugestão é que você não execute o código ainda. Primeiro, vamos terminar de escrevê-lo.

Em seguida, trabalharemos com o `onsuccess` , que sempre será quando conseguirmos obter uma conexão.

```
openRequest.onsuccess = function(e) {

        console.log('Conexão obtida com sucesso');
};
```

E o `onerror` será executado se tivermos algum tipo de problema ao tentarmos nos conectar com o banco.

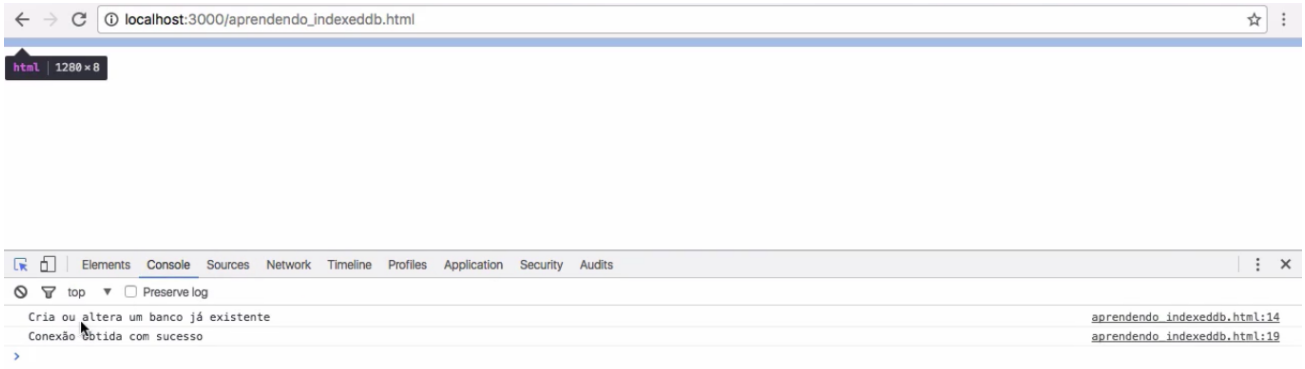
```
openRequest.onerror = function(e) {

        console.log(e.target.error);
};
```

Observe que, no `console`, vamos imprimir `e.target.error` - e assim, conseguimos descobrir qual erro ocorreu. Mas nos cursos anteriores, aprendemos a utilizar *arrow functions*, então, é por elas que substituiremos as funções anônimas.

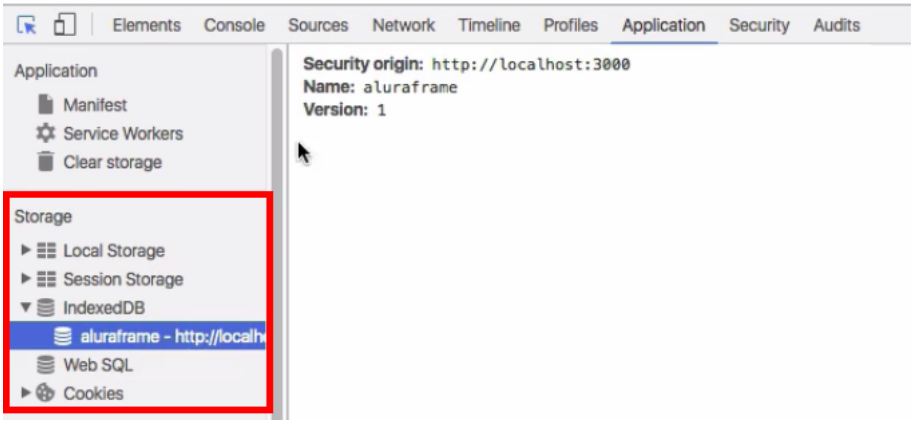
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
</head>
<body>
    <script>

        var openRequest = window.indexedDB.open('aluraframe',1);

        openRequest.onupgradeneeded = e => {

            console.log('Cria ou altera um banco já existente');
        };

        openRequest.onsuccess = e => {

            console.log('Conexão obtida com sucesso');
        };

        openRequest.onerror = e => {

            console.log(e.target.error);
        };

    </script>
</body>
```

Nosso código ficou menos verboso. Em resumo: nós pedimos para o IndexedDB abrir o `aluraframe`, que será criado pela primeira vez. Logo, o `onupgradeneeded` será executado, depois, será a vez do `onsuccess`. E se tivermos algum problema, será executado o `onerror`. A seguir, testaremos no navegador se está tudo certo. Se tudo correr bem, as duas primeiras mensagens serão impressas no Console. Na segunda vez, se executarmos o mesmo programa e o banco já tiver sido criado, ele só imprimirá a mensagem do `onsuccess`.

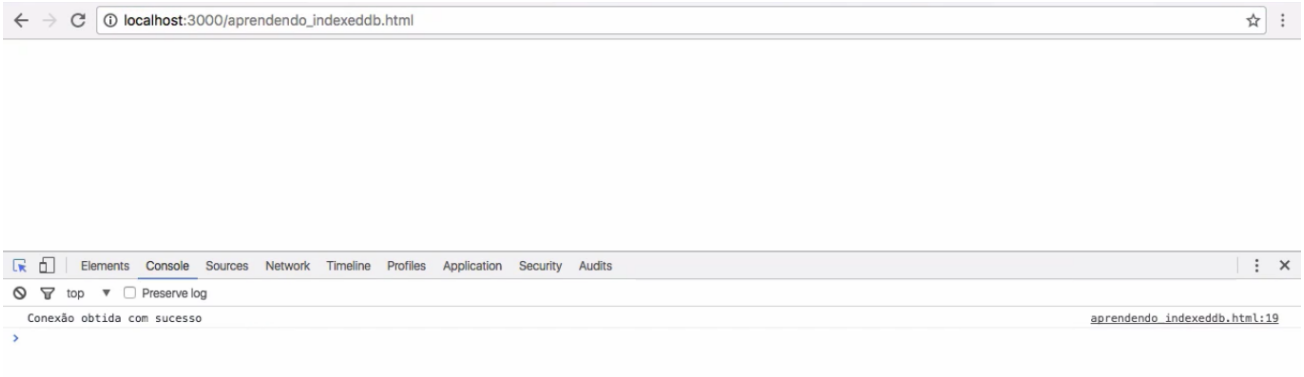
Na primeira vez, veremos as seguintes mensagens:



Agora, veremos se o banco foi criado:



Na aba "Application", é possível ver que o banco foi criado. Se recarregarmos a página, apenas a mensagem de sucesso na conexão será exibida no Console:



Nós vimos como abrir uma conexão com o banco e já entendemos a tríade de eventos disparados com o IndexedDB. Mais adiante, veremos com o interagir com ele.