JavaScript Moderno para front-end e back-end

Turma js8374 do curso JS-12

caelum
ensino e inovação

Sumário

Capítulo 1

EXERCÍCIO: IÇAMENTO/HOISTING DE DECLARAÇÕES

1.1 COMENTÁRIOS

O código abaixo que cria a variável depois de usá-la:

```
$inputEndereco.value = paginaInicial
$janelaPrincipal.src =  paginaInicial
var paginaInicial = 'http://google.com'
```

É a mesma coisa que esse próximo código:

```
var paginaInicial = undefined
$inputEndereco.value = paginaInicial
$janelaPrincipal.src =  paginaInicial
paginaInicial = 'http://google.com'
```

O JS manda a declaração da variável pra cima, faz o **hoisting**. Esse código da página inicial vai executar sem erros porque `undefined` é um valor válido.

> "hoisting foi um detalhe de implementação" – Brendan Eich (criador do JavaScript)

Em outras linguagens usar uma variável antes de criá-la resultaria num erro que impediria que o código seguisse com a execução!

ECMAScript não vai mudar o comportamento de algo. ECMAScript não quer quebrar códigos que já usem o `var` com hoisting de propósito.

ECMAScript fez uma nova forma de declarar variáveis que segue as outras linguagens:

```
const paginaInicial = 'http://google.com'
let paginaInicial = 'http://google.com'
```

1.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`index.html`** na pasta **`raíz do projeto`** faça as seguintes alterações:

 # index.html

```
 <meta charset="utf-8">

 <link rel="stylesheet" href="styles/cake-2afdf04e92.css"/>

 <header>
     <input type="image" src="images/libCake/recarregar.svg">
-    <input type="text" value="blank">
+    <input type="text" id="$inputEndereco" value="blank">
 </header>

-<iframe src="blank" frameborder="0"></iframe>
+<iframe src="blank" frameborder="0" id="$janelaPrincipal"></iframe>

-<script src="scripts/cake-8709f3abc4.js"></script>
+<script src="scripts/cake-8709f3abc4.js"></script>
+
+<script src="scripts/pedePaginaInicial.js"></script>
```

2. Nesse código, testaremos o hoisting para ver ele acontencendo. Acesse o console do navegador apertando `F12` e execute o código abaixo nele.

   Com o seguinte código, veja como nossa página inicial se torna **undefined** e como que no *console* (`F12`) não há erros de JavaScript por termos usado a variável antes de criá-la.

   # console.js

```
+
+$janelaPrincipal.src =  paginaInicial
+$inputEndereco.value = paginaInicial
+
+var paginaInicial = prompt("Escolha a página inicial")
```

3. Nesse código, veremos como o ECMAScript 2015 tratou o hoisting com suas novas variáveis **`let`** e **`const`** . Acesse o console do navegador apertando `F12` e execute o código abaixo nele.

   Com o seguinte código, veja como o console apresenta um erro de JavaScript dizendo que a variável não existe.

   # console.js

```
+
+$janelaPrincipal.src =  paginaInicial
+$inputEndereco.value = paginaInicial
+
+const paginaInicial = prompt("Escolha a página inicial")
```

4. Crie o arquivo **`pedePaginaInicial.js`** na pasta **`scripts`** com o seguinte código:

   # scripts/pedePaginaInicial.js

```
+
+const paginaInicial = prompt("Escolha a página inicial")
+
+$janelaPrincipal.src =  paginaInicial
+$inputEndereco.value = paginaInicial
+
```

Capítulo 2

EXERCÍCIO: TERMOS DE USO, JAVASCRIPT E AS APIS DO BROWSER QUE NÃO SÃO JAVASCRIPT

2.1 COMENTÁRIOS

Se a pessoa não aceitar os termos de uso, precisamos impedir que ela continue usando o Cake.

Para isso, usaremos diversas API's do navegador que nos permitem controlá-lo com JavaScript:

- `prompt`
- `confirm`
- `alert`
- `window.close`
- `localStorage`

Apesar de escrevermos o código com JavaScript, essas API's não fazem parte da especifação *ECMAScript*.

Já os valores retornados por essas API's e os valores que passamos como parâmetros para elas, são todos especifcados pela ECMA:

- `template strings` para strings multi-linha e interpolação de variáveis com `\` Um texto aqui ${nome} ``
- `booleanos` valores *true* e *false*
- `null` representa algo vazio

2.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`index.html`** na pasta **`raíz do projeto`** faça as seguintes alterações:

 # index.html

```
<meta charset="utf-8">

<link rel="stylesheet" href="styles/cake-2afdf04e92.css"/>
```

```
 <header>
     <input type="image" src="images/libCake/recarregar.svg">
     <input type="text" id="$inputEndereco" value="blank">
 </header>

 <iframe src="blank" frameborder="0" id="$janelaPrincipal"></iframe>

 <script src="scripts/cake-8709f3abc4.js"></script>

+<script src="scripts/termosDeUso.js"></script>
 <script src="scripts/pedePaginaInicial.js"></script>
```

2. Crie o arquivo **`termosDeUso.js`** na pasta **`scripts`** com o seguinte código:

 # scripts/termosDeUso.js

```
+if(localStorage.getItem('aceitouTermos') === null){
+    // CONST é Variável. Dinâmico
+    const nome = prompt('Qual o seu nome')
+
+    const aceitouTermos = confirm(`
+        Olá ${nome}!
+
+        Antes de usar o Cake, precisamos que
+        você aceite nossos termos de uso:
+
+        • Você aceita que este software foi
+            feito por pessoas que participaram
+            do curso de JavaScript.
+        • Você aceita que o código dessas
+            pessoas pode acessar tudo o que
+            você digitar aqui.
+    `)
+
+    if(!aceitouTermos) {
+        alert(nome + ', não podemos continuar juntos')
+        window.close()
+    } else {
+        localStorage.setItem('aceitouTermos', aceitouTermos)
+    }
+}
```

CAPÍTULO 3

EXERCÍCIO: ESCOLHENDO UMA PÁGINA INICIAL - VARIÁVEIS COM `LET` E COMPARAÇÕES

3.1 COMENTÁRIOS

Seguimos fazendo a funcionalidade de pedir a página inicial para a pessoa, deixando ela digitar um endereço sem `http://` no começo. Nós completamos o endereço para a pessoa caso ela não digite.

Comparações de igualdade e diferença no JavaScript iguais as de outras linguagens são com três caractéres: `===` e `!==` .

3.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`pedePaginaInicial.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/pedePaginaInicial.js

```
-const paginaInicial = prompt("Escolha a página inicial")
+let paginaInicial = prompt("Escolha a página inicial")
+
+if (
+    paginaInicial.substring(0, 7) !== 'http://' &&
+    paginaInicial.substring(0,8) !== 'https://'
+) {
+    // Assignement Atribuição
+    paginaInicial = 'http://' + paginaInicial
+}

 $janelaPrincipal.src =  paginaInicial
 $inputEndereco.value = paginaInicial
```

CAPÍTULO 4

EXERCÍCIO: TYPE COERCING

4.1 COMENTÁRIOS

Nesse ponto, deixaremos que a pessoa possa digitar nada ou cancelar o `prompt` que pede pela página inicial.

O seguinte código:

```
if (paginaInicial !== null && paginaInicial == '') {}
```

Pode ter o mesmo efeito que esse código:

```
if (paginaInicial) {}
```

Quando digo "pode" ter o mesmo efeito é porque o que está acontecendo alí é a conversão de `paginaInicial` para booleano:

```
Boolean(paginaInicial)
```

Esse código pode ter o resultado `false` para os seguintes valores:

```
Boolean(null)
Boolean(undefined)
Boolean('')
Boolean(0)
Boolean(NaN)
```

Ou seja, fazer `if (paginaInicial)` significa muito mais condições do que `if (paginaInicial !== null && paginaInicial == '')`

Essa conversão de tipos que é feita pelo JavaScript e não pelo nosso código é chamada de *Implicit Type Coercion*

Ela sempre vai acontecer quando algum tipo específico é esperado, no caso do `if()`, um Booleano.

Um caso de type coercion acontece quando fazemos uma concatenação: Se página inicial for null: 'http://' + paginaInicial

```
`paginaInicial` será con vertida para a string 'null' e terá como resultado: 'http://null'
```

Em uma operação/comparação de igualdade com 2 iguais `==` isso também acontece:

```
const paginaInicial = null
paginaInicial == 'null' // true
```

Já no caso de 3 iguais `===` , isso nãoo acontece:

```
const paginaInicial = null
paginaInicial === 'null' // true
```

4.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`index.html`** na pasta **`raíz do projeto`** faça as seguintes alterações:

 # index.html

```
  <meta charset="utf-8">

  <link rel="stylesheet" href="styles/cake-2afdf04e92.css"/>

  <header>
-     <input type="image" src="images/libCake/recarregar.svg">
+     <input type="image" src="images/libCake/recarregar.svg" onclick="window.location.reload()">
      <input type="text" id="$inputEndereco" value="blank">
  </header>

  <iframe src="blank" frameborder="0" id="$janelaPrincipal"></iframe>

  <script src="scripts/cake-8709f3abc4.js"></script>

  <script src="scripts/termosDeUso.js"></script>
  <script src="scripts/pedePaginaInicial.js"></script>
```

2. No arquivo **`termosDeUso.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/termosDeUso.js

```
-if(localStorage.getItem('aceitouTermos') === null){
+if(!localStorage.getItem('aceitouTermos')){
      // CONST é Variável. Dinâmico
      const nome = prompt('Qual o seu nome')

      const aceitouTermos = confirm(`
          Olá ${nome}!

          Antes de usar o Cake, precisamos que
          você aceite nossos termos de uso:

          • Você aceita que este software foi
              feito por pessoas que participaram
              do curso de JavaScript.
          • Você aceita que o código dessas
              pessoas pode acessar tudo o que
              você digitar aqui.
      `)

      if(!aceitouTermos) {
          alert(nome + ', não podemos continuar juntos')
          window.close()
      } else {
          localStorage.setItem('aceitouTermos', aceitouTermos)
      }
  }
```

3. No arquivo **`pedePaginaInicial.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/pedePaginaInicial.js

```
-let paginaInicial = prompt("Escolha a página inicial")
+let paginaInicial = localStorage.getItem('paginaInicial')

-if (
-    paginaInicial.substring(0, 7) !== 'http://' &&
-    paginaInicial.substring(0,8) !== 'https://'
-) {
-    // Assignement Atribuição
-    paginaInicial = 'http://' + paginaInicial
+if(!paginaInicial) {
+    paginaInicial = prompt("Escolha a página inicial")
 }

-$janelaPrincipal.src =  paginaInicial
-$inputEndereco.value = paginaInicial
+if(paginaInicial) {
+    if (
+        paginaInicial.substring(0, 7) !== 'http://' &&
+        paginaInicial.substring(0,8) !== 'https://'
+    ) {
+        // Assignement Atribuição
+        paginaInicial = 'http://' + paginaInicial
+    }

+    $janelaPrincipal.src =  paginaInicial
+    $inputEndereco.value = paginaInicial
+
+    localStorage.setItem('paginaInicial', paginaInicial)
+}
```

Capítulo 5

EXERCÍCIO: MÓDULOS NO NAVEGADOR COM IIFES

5.1 COMENTÁRIOS

Os navegadores pegam os arquivos nas tags `<script>` e fazem com que todos rodem no mesmo escopo. Toda variável que esteja 'solta' no arquivo fica global!

Por que não deixar global?

Para poder da o nome das suas variáveis do jeito que quiser. Sem impedir que outros códigos do sistema usem um nome de variável igual. Cada arquivo fica na sua.

Declarar sua variável com `var` permite que você re-declare a variável inicializando ela com outro valor. Se dois arquivos usam o mesmo nome de `var`:

```
// Arquivo A
var contador = 2
===
// Arquivo B
var contador = 4
```

O valor da variável contador será `4` em todos os scripts, inclusive no Arquivo A, que queria que a variável tivesse o valor `2`.

Outros códigos não relacionados com o seu podem modificar variáveis no seu código, muitas vezes sem querer. Em casos de nomes de variáveis comuns como `contador` e `i` utilizados em loops, isso pode ser bem ruim.

Uma solução seria declarar suas variáveis apenas com `const` e `let` pois elas não podem ser re-declaradas. O seguinte caso resutlaria em um erro que encerraria a execução do código:

```
// Arquivo A
const numero = 2
===
// Arquivo B
const numero = 4
```

O Arquivo B lançará um erro dizendo que não é possível redeclarar a `const numero`. A mesma coisa aconteceria com o `let`, porém outros códigos ainda poderiam alterar o valor da variável.

Para evitar esse tipo de problemas, passamos a usar ***IIFEs***:

Immediately **I**nvoked **F**unction **E**xpression

Traduzido: Função Anônima Imediatemente Invocada Vulgo: *IIFE*

Como funciona?

Todo `var` dentro de função, só existe dentro daquela função, enquanto ela é executada. Tirando isso, ela sempre estará global, mesmo dentro de blocos de código em `if`, `else`, `while` e `for`. Dizemos que o escopo da `var` é dinâmico.

Já, `const` e `let` têm escopo léxico. Traduzindo: Elas ficam contidas em blocos de código. Variáveis com `const` e `let` não ficam globais se estiverem dentro de um `if`, `else`, `while`, `for` ou `function`.

A solução encontrada para todos os casos:

```
(function() {
        // Qualquer `var`, `const` e `let` aqui dentro não será colocada no escopo global
})()
```

Uma função sem nome que é executada assim que é criada. Ninguém mais consegue chamar ela depois. Executa uma vez e acabou. As variáveis ali dentro estõ presas alí dentro. Porém.... se esquecer de colocar o `var` antes do nome da variável:

```
(function() {
        numero = 2
})()
```

O processo de **hoisting** faz com que a variável seja declarada assim, por debaixo dos panos do JS:

```
var numero = undefined
(function() {
        numero = 2
})()
```

Variável `numero` está global! A solução foi dar um jeito de isso dar erro nas versões novas do JS, mas sem quebrar versões antigas. Não podiam fazer esse código dar erro porque muita gente já usava esse código dessa maneira, de propósito.

A solução, uma string `'use strict'` solta no meio do código:

```
(function() {
        "use strict"
        numero = 2
})()
```

O código acima resultará num erro, pois o modo 'strict' do JavaScript foi ativado quando ele encontra a srting `"use strict"`.

Chamamos isso de módulo.

Cada módulo pode fazer as variáveis que quiser sem interferir sem querer com o escopo global.

Mas se for de propósito, ainda consegue:

```
let numero
(function() {
        "use strict"
        numero = 2
})()
```

O código roda sem erros!

Se quiser exportar alguma variável para o escopo global, basta retorná-la e armazená-la em uma variável global:

```
const variavelGlobal = (function() {
        "use strict"
        const numero = 2
        const variavelExportada = numero + 365
        return variavelExportada
})()
```

Criamos uma `variavelGlobal` que tem o valor `367`, resultado de ter rodado o código do nosso módulo.

Esse já foi o melhor jeito de criar módulos no ECMAScript antigo.

5.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`index.html`** na pasta **`raíz do projeto`** faça as seguintes alterações:

   # index.html

   ```
     <meta charset="utf-8">

     <link rel="stylesheet" href="styles/cake-2afdf04e92.css"/>

     <header>
         <input type="image" src="images/libCake/recarregar.svg" onclick="window.location.reload()">
         <input type="text" id="$inputEndereco" value="blank">
     </header>

     <iframe src="blank" frameborder="0" id="$janelaPrincipal"></iframe>

     <script src="scripts/cake-8709f3abc4.js"></script>

     <script src="scripts/termosDeUso.js"></script>
    +<script src="scripts/aceitouAnteriormente.js"></script>
     <script src="scripts/pedePaginaInicial.js"></script>
   ```

2. Crie o arquivo **`aceitouAnteriormente.js`** na pasta **`scripts`** com o seguinte código:

   # scripts/aceitouAnteriormente.js

   ```
    +const aceitouSalvar = (function() {
    +    "use strict"
    +    const aceitouAnteriormente = localStorage.getItem("aceitouSalvar")
    +
   ```

```
+    let aceitouSalvar
+
+    if(!aceitouAnteriormente){
+        aceitouSalvar = confirm('Você aceita que a gente salve suas informações?')
+        if(!aceitouSalvar) {
+            alert('Você pode mudar isso na página de configurações')
+        }
+
+        localStorage.setItem("aceitouSalvar", aceitouSalvar)
+    }
+
+    return aceitouSalvar
+})()
```

3. No arquivo **`pedePaginaInicial.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/pedePaginaInicial.js

```
+if (aceitouSalvar === null || aceitouSalvar === true) {
     let paginaInicial = localStorage.getItem('paginaInicial')

     if(!paginaInicial) {
         paginaInicial = prompt("Escolha a página inicial")
     }

     if(paginaInicial) {
         if (
             paginaInicial.substring(0, 7) !== 'http://' &&
             paginaInicial.substring(0,8) !== 'https://'
         ) {
             // Assignement Atribuição
             paginaInicial = 'http://' + paginaInicial
         }

         $janelaPrincipal.src =  paginaInicial
         $inputEndereco.value = paginaInicial

         localStorage.setItem('paginaInicial', paginaInicial)
     }
+}
```

CAPÍTULO 6

EXERCÍCIO: MÓDULOS COM ESMODULES

6.1 COMENTÁRIOS

Apesar das IIFEs e do `'use strict'` existirem e nos ajudarem bastante, ainda temos um problema em aplicaçoes com muitos scripts que dependem um do outro.

Nesse casos, ficamos dependentes da ordem em que incluímos os arquivos nas tags `<script>` no HTML. Por exemplo, no nosso caso, como `pedePaginaInicial.js` precisa de uma variável que é criada em `aceitouAnteriormente.js`, precisamos que a tag `<script>` de `aceitouAnteriormente.js` seja incluída antes da de `pedePaginaInicial.js`:

```
<script src="scripts/aceitouAnteriormente.js"></script>
<script src="scripts/pedePaginaInicial.js"></script>
```

Em páginas com mais scripts, isso pode se tornar bstante confuso, ilegível e também difculta casos em que haja scripts que dependam mutuamente um do outro.

Com todos esse pontos de atencão, o *Node.js* resolveu não seguir a situação dos módulos no navegador e adotou um padrão chamado "CommonJS". Nele, todo arquivo já é um módulo. E nenhuma variável é colocada no escopo global automaticamente.

Para o comparttilhamento de valores entre módulos, foi criado um mecanismo de importação e exportação de valores.

Quando algum arquivo quer acessar informações de outro:

```
// Arquivo a.js
const valorDeOutroArquivo = require('b.js')

===

//Arquivo b.js
const valorExportado = 2

module.exports = valorExportado
```

CommonJS não funciona nos navegadores por padrão. É preciso compilar seu código para algo que funcione no navagador ou adicionar alguma biblioteca que implemente o padrão no navegador.

No mundo dos navegadores, com o tempo, outros padrões também surgiram e foram adotados em muitos projetos, com maior destaque para o AMD (Async Module Definition).

Com 2 padrões muito bem estabelecidos: AMD nos navegadores e CommonJS no Node.js. Um 3º padrão surgiu que permitia a criação de módulos que rodassem em ambos os padrões: o UMD (Universal Module Definition).

Assim, com 3 jeitos de fazer módulos no JS, a ECMA elaborou os ESModules, a especifacação oficial de módulos que estamos usando nesse código.

Os *ESModules* ainda não são suportados por padrão no *Node.js*, ainda há muita discussão sobre como tudo será implementado e se será mantida ou não a compatibilidade com módulos antigos feitos com CommonJS.

Os navegadores mais novos já conseguem usar esses módulos se identificarmos nosso arquivo como um módulo lá na tag `<script type="module">` .

Em qualquer outro caso, em navegadores mais antigos e no Node, se você já estiver usando `import` e `export` , você com certeza está transformando seu código em IIFEs, CommonJS, AMD, UMD ou algum outro padrão, através de alguma forma de compilador, como o Babel, por exemplo.

6.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`index.html`** na pasta **`raíz do projeto`** faça as seguintes alterações:

 # index.html

```
  <meta charset="utf-8">

  <link rel="stylesheet" href="styles/cake-2afdf04e92.css"/>

  <header>
      <input type="image" src="images/libCake/recarregar.svg" onclick="window.location.reload()">
      <input type="text" id="$inputEndereco" value="blank">
  </header>

  <iframe src="blank" frameborder="0" id="$janelaPrincipal"></iframe>

  <script src="scripts/cake-8709f3abc4.js"></script>

  <script src="scripts/termosDeUso.js"></script>
 -<script src="scripts/aceitouAnteriormente.js"></script>
 -<script src="scripts/pedePaginaInicial.js"></script>
 +<script type="module" src="scripts/pedePaginaInicial.js"></script>
```

2. No arquivo **`aceitouAnteriormente.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/aceitouAnteriormente.js

```
 -const aceitouSalvar = (function() {
 -    "use strict"
 -    const aceitouAnteriormente = localStorage.getItem("aceitouSalvar")
 -
 -    let aceitouSalvar
 -
```

```
-    if(!aceitouAnteriormente){
-        aceitouSalvar = confirm('Você aceita que a gente salve suas informações?')
-        if(!aceitouSalvar) {
-            alert('Você pode mudar isso na página de configurações')
-        }
-        localStorage.setItem("aceitouSalvar", aceitouSalvar)
-    }
-
-    return aceitouSalvar
-})()
+const aceitouAnteriormente = localStorage.getItem("aceitouSalvar")
+
+let aceitouSalvar
+
+if(!aceitouAnteriormente){
+    aceitouSalvar = confirm('Você aceita que a gente salve suas informações?')
+    if(!aceitouSalvar) {
+        alert('Você pode mudar isso na página de configurações')
+    }
+    localStorage.setItem("aceitouSalvar", aceitouSalvar)
+}
+
+export default aceitouSalvar
```

3. No arquivo **`pedePaginaInicial.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/pedePaginaInicial.js

```
+import aceitouSalvar from './aceitouAnteriormente.js'

 if(aceitouSalvar === null || aceitouSalvar === true){

     let paginaInicial = localStorage.getItem('paginaInicial')

     if(!paginaInicial) {
         paginaInicial = prompt("Escolha a página inicial")
     }

     if(paginaInicial) {
         if (
             paginaInicial.substring(0, 7) !== 'http://' &&
             paginaInicial.substring(0,8) !== 'https://'
         ) {
             // Assignement Atribuição
             paginaInicial = 'http://' + paginaInicial
         }

         $janelaPrincipal.src =  paginaInicial
         $inputEndereco.value = paginaInicial

         localStorage.setItem('paginaInicial', paginaInicial)
     }
 }
```

CAPÍTULO 7

EXERCÍCIO: MENOS *TYPE COERCING* E CENTRALIZANDO MÓDULOS COM `MAIN.JS`

7.1 COMENTÁRIOS

Neste exercício, nós centralizamos nossos código JS no módulo `main.js` que faz todos os `import` necessários para que nossa app rode.

Também deixamos de depender do *type coercing* para evitar *bugs* e passamos a verificar exatamente os tipos de valores que esperamos das variáveis nos nossos `if` s.

7.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`index.html`** na pasta **`raíz do projeto`** faça as seguintes alterações:

 # index.html

```
  <meta charset="utf-8">

  <link rel="stylesheet" href="styles/cake-2afdf04e92.css"/>

  <header>
      <input type="image" src="images/libCake/recarregar.svg" onclick="window.location.reload()">
      <input type="text" id="$inputEndereco" value="blank">
  </header>

  <iframe src="blank" frameborder="0" id="$janelaPrincipal"></iframe>

  <script src="scripts/cake-8709f3abc4.js"></script>

 -<script src="scripts/termosDeUso.js"></script>
 -
 -<script type="module" src="scripts/pedePaginaInicial.js"></script>
 +<script type="module" src="scripts/main.js"></script>
```

2. Crie o arquivo **`main.js`** na pasta **`scripts`** com o seguinte código:

 # scripts/main.js

```
 +import './termosDeUso.js'
 +import './pedePaginaInicial.js'
```

3. Na pasta **`scripts`** , renomeie o arquivo **`aceitouAnteriormente.js`** para **`aceitouSalvar.js`** Faça também as seguintes modificações no código:

 # scripts/aceitouSalvar.js

```
-const aceitouAnteriormente = localStorage.getItem("aceitouSalvar")
+let aceitouSalvar = JSON.parse(localStorage.getItem("aceitouSalvar"))

-let aceitouSalvar
-
-if(!aceitouAnteriormente){
+if(aceitouSalvar === null){
     aceitouSalvar = confirm('Você aceita que a gente salve suas informações?')
     if(!aceitouSalvar) {
         alert('Você pode mudar isso na página de configurações')
     }

     localStorage.setItem("aceitouSalvar", aceitouSalvar)
 }

 export default aceitouSalvar
```

4. No arquivo **`pedePaginaInicial.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/pedePaginaInicial.js

```
-import aceitouSalvar from './aceitouAnteriormente.js'
+import aceitouSalvar from './aceitouSalvar.js'

-if(aceitouSalvar === null){
-
+if(aceitouSalvar === null || aceitouSalvar === true){
     let paginaInicial = localStorage.getItem('paginaInicial')

     if(!paginaInicial) {
         paginaInicial = prompt("Escolha a página inicial")
     }

     if(paginaInicial) {
         if (
             paginaInicial.substring(0, 7) !== 'http://' &&
             paginaInicial.substring(0,8) !== 'https://'
         ) {
             // Assignement Atribuição
             paginaInicial = 'http://' + paginaInicial
         }

         $janelaPrincipal.src =  paginaInicial
         $inputEndereco.value = paginaInicial

         localStorage.setItem('paginaInicial', paginaInicial)
     }
-
 }
```

CAPÍTULO 8

EXERCÍCIO: ORGANIZANDO A APLICAÇÃO COM MÓDULOS.

8.1 COMENTÁRIOS

Nesse exercício, fizemos uma grande refatoração para organizar melhor nossa aplicação com *ESModules*.

Criamos um módulo agregador `/scripts/pedeInfosIniciais/index.js` que importa todos os módulos que pedem informações na inicialização do navegador.

Também criamos módulos dentro de pasta `/scripts/storage/` que evitam que a gente duplique o código de acesso às informaçõe do `localStorage` .

Refatoramos todos os nossos códigos para utilizar essa estrutura nova do projeto.

8.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`main.js`** na pasta **`scripts`** faça as seguintes alterações:

   # scripts/main.js

   ```
   -import './termosDeUso.js'
   -import './pedePaginaInicial.js'
   +import '/scripts/pedeInfosIniciais/index.js'
   ```

2. Crie o arquivo **`index.js`** na pasta **`scripts/pedeInfosIniciais`** com o seguinte código:

   # scripts/pedeInfosIniciais/index.js

   ```
   +import './termosDeUso.js'
   +import './pedeAceitouSalvar.js'
   +import './pedePaginaInicial.js'
   ```

3. Mova o arquivo **`aceitouSalvar.js`** para a pasta **`scripts/pedeInfosIniciais`** e o renomeie para **`pedeAceitouSalvar.js`** . No momento o arquivo está na pasta **`scripts`** . Faça também as seguintes modificações no código:

   # scripts/pedeInfosIniciais/pedeAceitouSalvar.js

   ```
   -let aceitouSalvar = JSON.parse(localStorage.getItem("aceitouSalvar"))
   +import aceitouSalvar from '/scripts/storage/aceitouSalvar.js'
   ```

```
 if(aceitouSalvar === null){
-    aceitouSalvar = confirm('Você aceita que a gente salve suas informações?')
+    // shadowing/sombra no módulo
+    // redeclerando com o mesmo nome
+    const aceitouSalvar = confirm('Você aceita que a gente salve suas informações?')
+
     if(!aceitouSalvar) {
         alert('Você pode mudar isso na página de configurações')
     }

     localStorage.setItem("aceitouSalvar", aceitouSalvar)
-}
-
-export default aceitouSalvar
+}
```

4. Mova o arquivo **pedePaginaInicial.js** para a pasta **scripts/pedeInfosIniciais** . No momento este arquivo está na pasta **scripts** . Faça também as seguintes modificações no código:

 # scripts/pedeInfosIniciais/pedePaginaInicial.js

```
-import aceitouSalvar from './aceitouSalvar.js'
+import aceitouSalvar from '/scripts/storage/aceitouSalvar.js'
+import paginaInicial from '/scripts/storage/paginaInicial.js'

 if(aceitouSalvar === null || aceitouSalvar === true){
-    let paginaInicial = localStorage.getItem('paginaInicial')
+    // Sem shadowing
+    let paginaInicialDefault = paginaInicial

-    if(!paginaInicial) {
-        paginaInicial = prompt("Escolha a página inicial")
+    if(!paginaInicialDefault) {
+        paginaInicialDefault = prompt("Escolha a página inicial")
     }

-    if(paginaInicial) {
+    if(paginaInicialDefault) {
         if (
-            paginaInicial.substring(0, 7) !== 'http://' &&
-            paginaInicial.substring(0,8) !== 'https://'
+            paginaInicialDefault.substring(0, 7) !== 'http://' &&
+            paginaInicialDefault.substring(0,8) !== 'https://'
         ) {
             // Assignement Atribuição
-            paginaInicial = 'http://' + paginaInicial
+            paginaInicialDefault = 'http://' + paginaInicialDefault
         }

-        $janelaPrincipal.src =  paginaInicial
-        $inputEndereco.value = paginaInicial
+        $janelaPrincipal.src =  paginaInicialDefault
+        $inputEndereco.value = paginaInicialDefault

-        localStorage.setItem('paginaInicial', paginaInicial)
+        localStorage.setItem('paginaInicial', paginaInicialDefault)
     }
 }
```

5. Mova o arquivo **termosDeUso.js** para a pasta **scripts/pedeInfosIniciais** . No momento este

arquivo está na pasta **`scripts`** . Faça também as seguintes modificações no código:

 # scripts/pedeInfosIniciais/termosDeUso.js

6. Crie o arquivo **`aceitouSalvar.js`** na pasta **`scripts/storage`** com o seguinte código:

   # scripts/storage/aceitouSalvar.js

   ```
   +export default JSON.parse(localStorage.getItem('aceitouSalvar'))
   ```

7. Crie o arquivo **`paginaInicial.js`** na pasta **`scripts/storage`** com o seguinte código:

   # scripts/storage/paginaInicial.js

   ```
   +export default localStorage.getItem('paginaInicial')
   ```

CAPÍTULO 9

EXERCÍCIO: INÍCIO DA PÁGINA DE CONFIGURAÇÃO

9.1 COMENTÁRIOS

Criamos uma nova página de configuração que será acessada quando a pessoa clicar no ícone da engrenagem ao lado da barra de endereço.

Nessa página de configuração, visualizaremos as informações salvas no `localStorage`, mas não acessaremos ele diretamente. Podemos usar nossos novos módulos dentro de `storage/`.

9.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`index.html`** na pasta **`raíz do projeto`** faça as seguintes alterações:

 # index.html

```
  <meta charset="utf-8">

  <link rel="stylesheet" href="styles/cake-2afdf04e92.css"/>

  <header>
      <input type="image" src="images/libCake/recarregar.svg" onclick="window.location.reload()">
      <input type="text" id="$inputEndereco" value="blank">
+     <a href="config.html">
+         <img src="images/libCake/engrenagem.svg" alt="Ir para as configurações">
+     </a>
  </header>

  <iframe src="blank" frameborder="0" id="$janelaPrincipal"></iframe>

  <script src="scripts/cake-8709f3abc4.js"></script>

  <script type="module" src="scripts/main.js"></script>
```

2. No arquivo **`config.html`** na pasta **`raíz do projeto`** faça as seguintes alterações:

 # config.html

```
  <link rel="stylesheet" href="styles/cake-config-33d6f649ee.css"/>

  <h1>Configurações</h1>

  <label>
```

```
      Página inicial:
-     <input type="text">
+     <input type="text" id="$inputPaginaInicial">
  </label>

  <label>
-     Permissão de compartilhamento:
-     <input type="checkbox">
+     Permissão para armazenamento:
+     <input type="checkbox" id="$inputPermitiuSalvar">
  </label>

  <button>
      Salvar configurações
  </button>
  <a href="/">Voltar</a>
+
+<script type="module" src="scripts/paginaConfiguracao.js"></script>
```

3. Crie o arquivo **`paginaConfiguracao.js`** na pasta **`scripts`** com o seguinte código:

 # scripts/paginaConfiguracao.js

```
+import paginaInicial from '/scripts/storage/paginaInicial.js'
+import aceitouSalvar from '/scripts/storage/aceitouSalvar.js'
+
+$inputPaginaInicial.value = paginaInicial
+$inputPermitiuSalvar.checked = aceitouSalvar
```

CAPÍTULO 10

EXERCÍCIO: CONFIGURANDO ESMODULES NO VSCODE

10.1 COMENTÁRIOS

Nos nossos códigos, passamos a importar nossos módulos com um caminho absoluto:

```
import aceitouSalvar from '/scripts/storage/aceitouSalvar.js'
```

O VSCode por padrão acha que endereços iniciados com `/` se referem à pasta raiz do seu computador, e não do nosso projeto. Isso se deve pelo fato de que ESModules ainda não são amplamente adotados em todas as plataformas.

Em projetos que usam `import` e `export`, na maioria das vezes, o código é passado para algum compilador que traduz esses módulos para módulos de padrões mais antigos, como *CommonJS* ou até mesmo *IIFEs*, que comentamos anteriormente. Esse processo de compilação geralmente ocorre na máquina de quem está desenvolvendo, e assim, caminhos iniciados em `/` são considerados também como a partir da pasta raíz do computador.

Com ESModules nos browsers, esses nossos imports são URLs relativas sempre ao endereço em que a aplicação está rodando. No nosso caso, relativas à pasta raíz do nosso projeto.

A configuração abaixo diz que a pasta raíz do nosso projeto deve ser considerada sempre que algum endereço de `import` iniciado com `/` for encontrado no nosso código.

Com essa configuração, ganhamos diversas facilidades do editor na hora de acessar, navegar e até mesmo mudar códigos entre nossos módulos.

10.2 PASSO A PASSO COM CÓDIGO

1. Crie o arquivo **`jsconfig.json`** na pasta **`raíz do projeto`** com o seguinte código:

 # jsconfig.json

```
+{
+    "compilerOptions": {
+        "baseUrl": ".",
+        "module": "esnext",
+        "paths": {
```

```
+            "/*": ["./*"]
+        }
+    }
+}
```

Capítulo 11

EXERCÍCIO: MÓDULOS COM MAIS DE UM VALOR EXPORTADO

11.1 COMENTÁRIOS

Nesse código, implementamos a lógica o botão de salvar as configurações na página de configurações.

Para adicionar comportamento ao botão de salvar criamos uma função chamada `salvar`. Nós criamos essa função para guardar e poder apontar para o pedaço de código que deve ser executado quando o evento de click acontecer no botão. Não seremos nós que executaremos essa função, mas sim o navegador, por isso passamos apenas o nome dela na propriedade onclick do botão:

```
$botaoSalvar.onclick = salvar
```

Chamamos esse tipo de função de função de callback.

Para evitar código duplicado que altera o `localStorage`, criamos funções *setter* em nossos módulos de `/scripts/storage/` que serão responsáveis por alterar o `localStorage`.

Refatoramos nosso código anterior para que não haja mais acesso ao `localStorage` fora desses módulos.

11.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`aceitouSalvar.js`** na pasta **`scripts/storage`** faça as seguintes alterações:

   # scripts/storage/aceitouSalvar.js

   ```
   -export default JSON.parse(localStorage.getItem('aceitouSalvar'))
   +export default JSON.parse(localStorage.getItem('aceitouSalvar'))
   +
   +export function setAceitouSalvar(aceitouSalvar) {
   +    localStorage.setItem("aceitouSalvar", aceitouSalvar)
   +}
   ```

2. No arquivo **`paginaInicial.js`** na pasta **`scripts/storage`** faça as seguintes alterações:

   # scripts/storage/paginaInicial.js

   ```
   -export default localStorage.getItem('paginaInicial')
   +export default localStorage.getItem('paginaInicial')
   ```

```
+
+export function setPaginaInicial(valor) {
+    localStorage.setItem('paginaInicial', valor)
+}
```

3. No arquivo **`config.html`** na pasta **`raíz do projeto`** faça as seguintes alterações:

 # config.html

```
 <link rel="stylesheet" href="styles/cake-config-33d6f649ee.css"/>

 <h1>Configurações</h1>

 <label>
     Página inicial:
     <input type="text" id="$inputPaginaInicial">
 </label>

 <label>
     Permissão para armazenamento:
     <input type="checkbox" id="$inputPermitiuSalvar">
 </label>

-<button>
+<button id="$botaoSalvar">
     Salvar configurações
 </button>
 <a href="/">Voltar</a>

 <script type="module" src="scripts/paginaConfiguracao.js"></script>
```

4. No arquivo **`paginaConfiguracao.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/paginaConfiguracao.js

```
-import paginaInicial from '/scripts/storage/paginaInicial.js'
-import aceitouSalvar from '/scripts/storage/aceitouSalvar.js'
+import paginaInicial, { setPaginaInicial } from '/scripts/storage/paginaInicial.js'
+import aceitouSalvar, { setAceitouSalvar }  from '/scripts/storage/aceitouSalvar.js'

 $inputPaginaInicial.value = paginaInicial
 $inputPermitiuSalvar.checked = aceitouSalvar
+
+
+// o que vai ser executado quando clicar
+// o que vai ser executado quando o evento de click acontecer
+$botaoSalvar.onclick = salvar
+
+// função de callback
+// hoisting
+// função é um tipo de dado
+// executada em um outro momnento do tempo
+function salvar(){
+    setAceitouSalvar($inputPermitiuSalvar.checked)
+    setPaginaInicial($inputPaginaInicial.value)
+}
```

5. No arquivo **`pedeAceitouSalvar.js`** na pasta **`scripts/pedeInfosIniciais`** faça as seguintes alterações:

 # scripts/pedeInfosIniciais/pedeAceitouSalvar.js

```
-import aceitouSalvar from '/scripts/storage/aceitouSalvar.js'
+import aceitouSalvar, {setAceitouSalvar} from '/scripts/storage/aceitouSalvar.js'

 if(aceitouSalvar === null){
     // shadowing/sombra no módulo
     // redeclerando com o mesmo nome
     const aceitouSalvar = confirm('Você aceita que a gente salve suas informações?')

     if(!aceitouSalvar) {
         alert('Você pode mudar isso na página de configurações')
     }

-    localStorage.setItem("aceitouSalvar", aceitouSalvar)
+    setAceitouSalvar(aceitouSalvar)
 }
```

6. No arquivo **`pedePaginaInicial.js`** na pasta **`scripts/pedeInfosIniciais`** faça as seguintes alterações:

 # scripts/pedeInfosIniciais/pedePaginaInicial.js

```
 import aceitouSalvar from '/scripts/storage/aceitouSalvar.js'
-import paginaInicial from '/scripts/storage/paginaInicial.js'
+import paginaInicial, {setPaginaInicial} from '/scripts/storage/paginaInicial.js'

 if(aceitouSalvar === null || aceitouSalvar === true){
     // Sem shadowing
     let paginaInicialDefault = paginaInicial

     if(!paginaInicialDefault) {
         paginaInicialDefault = prompt("Escolha a página inicial")
     }

     if(paginaInicialDefault) {
         if (
             paginaInicialDefault.substring(0, 7) !== 'http://' &&
             paginaInicialDefault.substring(0,8) !== 'https://'
         ) {
             // Assignement Atribuição
             paginaInicialDefault = 'http://' + paginaInicialDefault
         }

         $janelaPrincipal.src =  paginaInicialDefault
         $inputEndereco.value = paginaInicialDefault

-        localStorage.setItem('paginaInicial', paginaInicialDefault)
+        setPaginaInicial(paginaInicialDefault)
     }
 }
```

7. Para que não haja mais o carregamento da página *blank.html* automaticamente na inicialização do navegador, removeremos alguns atributos na página principal **`index.html`** , que está na pasta **`raíz do projeto`** :

 # index.html

```
 <meta charset="utf-8">

 <link rel="stylesheet" href="styles/cake-2afdf04e92.css"/>

 <header>
     <input type="image" src="images/libCake/recarregar.svg" onclick="window.location.reload()">
-    <input type="text" id="$inputEndereco" value="blank">
+    <input type="text" id="$inputEndereco">
     <a href="config.html">
         <img src="images/libCake/engrenagem.svg" alt="Ir para as configurações">
     </a>
 </header>

-<iframe src="blank" frameborder="0" id="$janelaPrincipal"></iframe>
+<iframe frameborder="0" id="$janelaPrincipal"></iframe>

 <script src="scripts/cake-8709f3abc4.js"></script>

 <script type="module" src="scripts/main.js"></script>
```

Capítulo 12

EXERCÍCIO: FUNÇÕES: CALLBACKS, ENCAPSULAMENTO E CLOSURES

12.1 COMENTÁRIOS

Chegamos num ponto da aplicação no qual precisamos alterar o valor e `aceitouSalvar` e `paginaInicial`. Porém, até a gora elas eram os valores padrões exportados pelos nossos módulos de `/scripts/storage/`. Vimos que esses valores exportados em nossos módulos são constantes e não podem ter seus valores alterados. Assim, transformamos esses valores em variáveis `let`:

```
-export default JSON.parse(localStorage.getItem('aceitouSalvar'))
+export let aceitouSalvar = JSON.parse(localStorage.getItem('aceitouSalvar')) //null
```

Porém, mesmo declarados como `let` vimos que outros módulos que importam essas variáveis não conseguem altterá-las. Qualquer proprieade exportada por um módulo fica constante para qualquer outro módulo que o importe.

Esse comportamento permite que a gente controle como e quem acessará e modificará essas variáveis.

Para permitir que essas variáveis possam ter seu valor alterado, alteramos nossas funçoes `setter` para que alterassem o valor dos nossos `let`:

```
 export let aceitouSalvar = JSON.parse(localStorage.getItem('aceitouSalvar'))

 export function setAceitouSalvar(aceitouSalvar) {
+    aceitouSalvar = valor
     localStorage.setItem("aceitouSalvar", aceitouSalvar)
 }
```

Aproveitamos o comportamento das funções no JS que têm acesso a todas as variáveis dentro do escopo em que foram criadas.

Se alguém chamar `setAceitouSalvar`, conseguirá alterar o valor da variável `aceitouSalvar`, por mais que ela esteja encapsulada no módulo.

Podemos chamar `setAceitouSalvar` de *Closure*. Um tipo de função que tem acesso a um ambiente/escopo privado onde foi criada.

As *closures* permitem que a gente encapsule código em nossos módulos e defina de que modo outros

códigos terão ou não acesso a variáveis e outras funções.

12.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`aceitouSalvar.js`** na pasta **`scripts/storage`** faça as seguintes alterações:

2. No arquivo **`paginaInicial.js`** na pasta **`scripts/storage`** faça as seguintes alterações:

 # scripts/storage/paginaInicial.js

```
-export default localStorage.getItem('paginaInicial')
+export let paginaInicial = localStorage.getItem('paginaInicial')

 export function setPaginaInicial(valor) {
+    paginaInicial = valor
     localStorage.setItem('paginaInicial', valor)
 }
```

3. No arquivo **`paginaConfiguracao.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/paginaConfiguracao.js

```
-import paginaInicial, { setPaginaInicial } from '/scripts/storage/paginaInicial.js'
-import aceitouSalvar, { setAceitouSalvar }  from '/scripts/storage/aceitouSalvar.js'
+import * as storagePaginaInicial from '/scripts/storage/paginaInicial.js'
+import * as storageAceitouSalvar  from '/scripts/storage/aceitouSalvar.js'

-$inputPaginaInicial.value = paginaInicial
-$inputPermitiuSalvar.checked = aceitouSalvar
+$inputPaginaInicial.value = storagePaginaInicial.paginaInicial
+$inputPermitiuSalvar.checked = storageAceitouSalvar.aceitouSalvar

 // o que vai ser executado quando clicar
 // o que vai ser executado quando o evento de click acontecer
 $botaoSalvar.onclick = salvar

 // função de callback
 // hoisting
 // função é um tipo de dado
 // executada em um outro momnento do tempo
 function salvar(){
-    setAceitouSalvar($inputPermitiuSalvar.checked)
-    setPaginaInicial($inputPaginaInicial.value)
+    storageAceitouSalvar.setAceitouSalvar($inputPermitiuSalvar.checked)
+    storagePaginaInicial.setPaginaInicial($inputPaginaInicial.value)
 }
```

4. No arquivo **`pedeAceitouSalvar.js`** na pasta **`scripts/pedeInfosIniciais`** faça as seguintes alterações:

 # scripts/pedeInfosIniciais/pedeAceitouSalvar.js

```
-import aceitouSalvar, {setAceitouSalvar} from '/scripts/storage/aceitouSalvar.js'
+import * as storageAceitouSalvar from '/scripts/storage/aceitouSalvar.js'

-if(aceitouSalvar === null){
```

```
+if(storageAceitouSalvar.aceitouSalvar === null){
     // shadowing/sombra no módulo
     // redeclerando com o mesmo nome
     const aceitouSalvar = confirm('Você aceita que a gente salve suas informações?')

     if(!aceitouSalvar) {
         alert('Você pode mudar isso na página de configurações')
     }

-    setAceitouSalvar(aceitouSalvar)
+
+    storageAceitouSalvar.setAceitouSalvar(aceitouSalvar)
 }
```

5. No arquivo **`pedePaginaInicial.js`** na pasta **`scripts/pedeInfosIniciais`** faça as seguintes alterações:

 # scripts/pedeInfosIniciais/pedePaginaInicial.js

```
-import aceitouSalvar from '/scripts/storage/aceitouSalvar.js'
-import paginaInicial, {setPaginaInicial} from '/scripts/storage/paginaInicial.js'
+import { aceitouSalvar as storageAceitouSalvar } from '/scripts/storage/aceitouSalvar.js'
+import { paginaInicial, setPaginaInicial } from '/scripts/storage/paginaInicial.js'

-if(aceitouSalvar === null || aceitouSalvar === true){
+if(storageAceitouSalvar === null || storageAceitouSalvar === true){
     // Sem shadowing
     let paginaInicialDefault = paginaInicial

     if(!paginaInicialDefault) {
         paginaInicialDefault = prompt("Escolha a página inicial")
     }

     if(paginaInicialDefault) {
         if (
             paginaInicialDefault.substring(0, 7) !== 'http://' &&
             paginaInicialDefault.substring(0,8) !== 'https://'
         ) {
             // Assignement Atribuição
             paginaInicialDefault = 'http://' + paginaInicialDefault
         }

         $janelaPrincipal.src =  paginaInicialDefault
         $inputEndereco.value = paginaInicialDefault

         setPaginaInicial(paginaInicialDefault)
     }
 }
```

6. No arquivo **`aceitouSalvar.js`** na pasta **`scripts/storage`** faça as seguintes alterações:

 # scripts/storage/aceitouSalvar.js

```
-export default JSON.parse(localStorage.getItem('aceitouSalvar'))
+// Encapsulamento
+export let aceitouSalvar = JSON.parse(localStorage.getItem('aceitouSalvar')) //null

-export function setAceitouSalvar(aceitouSalvar) {
-    localStorage.setItem("aceitouSalvar", aceitouSalvar)
-}
```

```
+
+// Ambiente onde ela foi criada
+// Acesso a variaveis do ambiente
+// function setAceitouSalvar é uma "Closure"
+export function setAceitouSalvar(valor) {
+    aceitouSalvar = valor
+    localStorage.setItem("aceitouSalvar", valor)
+}
```

Capítulo 13

EXERCÍCIO: FUNÇÕES COMO CLOSURES E TIPOS DE VARIÁVEL

13.1 COMENTÁRIOS

No exercício anterior, vimos como funções podem permitir o acesso à variáveis encapsuladas dentro dos nossos módulos.

Porém, criamos uma função `setAceitouSalvar` que recebe qulaquer tipo de valor e atribui isso à variável `aceitouSalvar`. Essa variável deveria ser `null` ou `boolean`, sempre, e não deveríamos colocar qualquer valor nela! Podemos usar as *closures* para limitar melhor o que pode ser feito com essa nossa variável. Ao invés de exportar uma função que aceita qualquer valor, exportaremos 2 funções que ou colocam o valor como `true`, ou como `false`.

Os códigos que forem alterar noss variável devem antes escolher qual das funções devem ser chamadas para isso.

Alteraremos nosso código anterior para isso! Note que nós escolheremos a função que deve ser executada usando um `if` ternário, salvando a função escolhida numa variável:

```
const funcaoEscolhida = $inputPermitiuSalvar.checked === true
        ? storageAceitouSalvar.setAceitou
        : storageAceitouSalvar.setNaoAceitou
```

Salvar uma função numa variável é possível pois no JavaScript, funções são um tipo de dado/variável.

Isso significa que podemos declarar e criar nossas funções de duas maneiras no JavaScript.

Como uma ***Function expression***:

```
const nomeDaFuncao = function () {

}
```

Como uma ***Function declaration***

```
function nomeDaFuncao() {

}
```

Lembrando que as ***Function expressions*** estão sujeitas aos mesmos comportamentos de *hoisting* das variáveis, ou seja: caso tenha declarado com `const` e `let`, só poderá usar a variável após a declaração

dela; já no caso do `var`, é possível usar a variável antes do ponto onde declaramos ela, porém ela terá valor `undefined` no código enquanto não for feita a atribuição.

13.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`paginaConfiguracao.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/paginaConfiguracao.js

```
import * as storagePaginaInicial from '/scripts/storage/paginaInicial.js'
import * as storageAceitouSalvar  from '/scripts/storage/aceitouSalvar.js'

$inputPaginaInicial.value = storagePaginaInicial.paginaInicial
$inputPermitiuSalvar.checked = storageAceitouSalvar.aceitouSalvar

// o que vai ser executado quando clicar
// o que vai ser executado quando o evento de click acontecer
$botaoSalvar.onclick = salvar

// função de callback
// hoisting
// função é um tipo de dado
-// executada em um outro momento do tempo
+// executada em um outro momento do tempo
+// Declaração de função
+// Function declaration
 function salvar(){
-    storageAceitouSalvar.setAceitouSalvar($inputPermitiuSalvar.checked)
+
+    // Expressão de função
+    // Function expression
+    const funcaoEscolhida = $inputPermitiuSalvar.checked === true
+        ? storageAceitouSalvar.setAceitou
+        : storageAceitouSalvar.setNaoAceitou
+
+    funcaoEscolhida()
+
     storagePaginaInicial.setPaginaInicial($inputPaginaInicial.value)
 }
```

2. No arquivo **`pedeAceitouSalvar.js`** na pasta **`scripts/pedeInfosIniciais`** faça as seguintes alterações:

 # scripts/pedeInfosIniciais/pedeAceitouSalvar.js

```
import * as storageAceitouSalvar from '/scripts/storage/aceitouSalvar.js'

if(storageAceitouSalvar.aceitouSalvar === null){
    // shadowing/sombra no módulo
    // redeclerando com o mesmo nome
    const aceitouSalvar = confirm('Você aceita que a gente salve suas informações?')

    if(!aceitouSalvar) {
        alert('Você pode mudar isso na página de configurações')
    }

-
```

```
-    storageAceitouSalvar.setAceitouSalvar(aceitouSalvar)
+    const funcaoSalvar = aceitouSalvar === true
+        ? storageAceitouSalvar.setAceitou
+        : storageAceitouSalvar.setNaoAceitou
+
+    funcaoSalvar()
 }
```

3. No arquivo **`aceitouSalvar.js`** na pasta **`scripts/storage`** faça as seguintes alterações:

 # scripts/storage/aceitouSalvar.js

```
// Encapsulamento
export let aceitouSalvar = JSON.parse(localStorage.getItem('aceitouSalvar')) //null

// Ambiente onde ela foi criada
// Acesso a variaveis do ambiente
// function setAceitouSalvar é uma "Closure"
-export function setAceitouSalvar(valor) {
+function setAceitouSalvar(valor) {
    aceitouSalvar = valor
    localStorage.setItem("aceitouSalvar", valor)
+}
+
+export function setAceitou() {
+    setAceitouSalvar(true)
+}
+
+export function setNaoAceitou() {
+    setAceitouSalvar(false)
}
```

Capítulo 14

EXERCÍCIO: UM NOVO MÓDULO PARA TRATAR ENDEREÇOS

14.1 COMENTÁRIOS

Transformar um endereço em um endereço completo iniciado em 'http://' é algo que precisaremos faaer a todo momento.

Inclusive, já estamos precisando fazer isso na página de configuração e no momento que a pessoa escolhe sua página inicial.

Para evitar duplicação de código, vamos criar um módulo que exporta uma função `formataEndereco` que faz exatamente o que o seu nome diz.

Nosso módulo exportará apenas uma função por enquanto, porém, seguiremos o padrão de exportar apenas propriedades nomeadas e não um `export default` . Assim, quem importar nosso módulo não será obrigado a escolher um nome qualquer para a função e por padrão usará o nome que já demos a ela:

```
import { formataEndereco } from '/scripts/endereco/formataEndereco.js'
```

Lembrando que damos o nome de *destructuring* pare essa sintaxe `import { formataEndereco }` , que acessa e escolhe importar apenas algumas propriedades de um módulo.

14.2 PASSO A PASSO COM CÓDIGO

1. Crie o arquivo **`formataEndereco.js`** na pasta **`scripts/endereco`** com o seguinte código:

 # scripts/endereco/formataEndereco.js

```
+function formataEndereco(enderecoPraFormatar) {
+    if (
+        enderecoPraFormatar.substring(0, 7) !== 'http://' &&
+        enderecoPraFormatar.substring(0,8) !== 'https://'
+    ) {
+        // Assignement Atribuição
+        enderecoPraFormatar = 'http://' + enderecoPraFormatar
+    }
+
+    return enderecoPraFormatar
+}
+
+export { formataEndereco }
```

2. No arquivo **`paginaConfiguracao.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/paginaConfiguracao.js

```
 import * as storagePaginaInicial from '/scripts/storage/paginaInicial.js'
 import * as storageAceitouSalvar  from '/scripts/storage/aceitouSalvar.js'

+// named export
+// destructuring
+// desestruturação, explodindo
+import { formataEndereco } from '/scripts/endereco/formataEndereco.js'
+
 $inputPaginaInicial.value = storagePaginaInicial.paginaInicial
 $inputPermitiuSalvar.checked = storageAceitouSalvar.aceitouSalvar

 // o que vai ser executado quando clicar
 // o que vai ser executado quando o evento de click acontecer
 $botaoSalvar.onclick = salvar

 // função de callback
 // hoisting
 // função é um tipo de dado
 // executada em um outro momento do tempo
 // Declaração de função
 // Function declaration
 function salvar(){

     // Expressão de função
     // Function expression
     const funcaoEscolhida = $inputPermitiuSalvar.checked === true
         ? storageAceitouSalvar.setAceitou
         : storageAceitouSalvar.setNaoAceitou

     funcaoEscolhida()

-    storagePaginaInicial.setPaginaInicial($inputPaginaInicial.value)
+    const enderecoCompleto = formataEndereco($inputPaginaInicial.value)
+    $inputPaginaInicial.value = enderecoCompleto
+
+    storagePaginaInicial.setPaginaInicial(enderecoCompleto)
 }
```

3. No arquivo **`pedePaginaInicial.js`** na pasta **`scripts/pedeInfosIniciais`** faça as seguintes alterações:

 # scripts/pedeInfosIniciais/pedePaginaInicial.js

```
 import { aceitouSalvar as storageAceitouSalvar } from '/scripts/storage/aceitouSalvar.js'
 import { paginaInicial, setPaginaInicial } from '/scripts/storage/paginaInicial.js'

+// named export
+import { formataEndereco } from '/scripts/endereco/formataEndereco.js'
+
 if(storageAceitouSalvar === null || storageAceitouSalvar === true){
     // Sem shadowing
     let paginaInicialDefault = paginaInicial

     if(!paginaInicialDefault) {
         paginaInicialDefault = prompt("Escolha a página inicial")
     }
```

```
     if(paginaInicialDefault) {
-        if (
-            paginaInicialDefault.substring(0, 7) !== 'http://' &&
-            paginaInicialDefault.substring(0,8) !== 'https://'
-        ) {
-            // Assignement Atribuição
-            paginaInicialDefault = 'http://' + paginaInicialDefault
-        }
-
-        $janelaPrincipal.src =  paginaInicialDefault
-        $inputEndereco.value = paginaInicialDefault
-
-        setPaginaInicial(paginaInicialDefault)
+        const enderecoCompleto = formataEndereco(paginaInicialDefault)
+
+        $janelaPrincipal.src = enderecoCompleto
+        $inputEndereco.value = enderecoCompleto
+
+        setPaginaInicial(enderecoCompleto)
     }
 }
```

4. [Opcional] No arquivo **`aceitouSalvar.js`** na pasta **`/scripts/storage`** , ao invés de escrever a palavra `export` variás vezes, podemos seguir o *Revealing Module Pattern*, onde com apenas um `export` no final do nosso arquivo, exportamos todas as propriedades que queremos deixar públicas no nosso módulo.

 # scripts/storage/aceitouSalvar.js

```
 // Encapsulamento
-export let aceitouSalvar = JSON.parse(localStorage.getItem('aceitouSalvar')) //null
-
+let aceitouSalvar = JSON.parse(localStorage.getItem('aceitouSalvar')) //null

 // Ambiente onde ela foi criada
 // Acesso a variaveis do ambiente
 // function setAceitouSalvar é uma "Closure"
+//  privada
 function setAceitouSalvar(valor) {
     aceitouSalvar = valor
     localStorage.setItem("aceitouSalvar", valor)
 }

-export function setAceitou() {
+function setAceitou() {
     setAceitouSalvar(true)
 }

-export function setNaoAceitou() {
+ function setNaoAceitou() {
     setAceitouSalvar(false)
+}
+
+// Revealing Module Pattern
+export {
+    aceitouSalvar,
+    setAceitou,
+    setNaoAceitou
 }
```

CAPÍTULO 15

EXERCÍCIO: INCIANDO A NAVEGAÇÃO: EVENTOS DO IFRAME PARA ATUALIZAR A BARRA DE ENDEREÇOS

15.1 COMENTÁRIOS

Nossa barra de endereços fica estática enquanto a pessoa navega usando o cake. Para que a barra de endereço seja atualizada com o endereço da página atual, criaremos uma função de `callback` que será chamada pelo navegador sempre que o evento de `load` for disparado no `<iframe>`.

Funções de `callback` são aquelas funções que criamos, mas não executamos. Quem executará elas será o próprio navegador, dado algum evento que ocorra.

O conceito de evento de `load` numa tag `<iframe>` está estritamente relacionado a JavaScript que roda em navegadores. Em qualquer outra plataforma que rode JavaScript, mas que não esteja relacionada com navegadores (como o Node.js) esse tipo de evento não existe. Por isso, é importante reforçar que o nome dos eventos e onde eles serão disparados não fazem parte do que chamamos de JavaScript e dependem totalmente da plataforma na qual estamos desenvolvendo.

O mais importante é notar que a ideia de eventos e funções de callback que executam quando esses eventos ocorrem sempre vai existir. Isso sim faz parte do JavaScript. Seja um evento de `load` num `<iframe>` do navegador, ou seja um evento `data` num `httpserver` do *Node.js*.

15.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`main.js`** na pasta **`scripts`** faça as seguintes alterações:

 # scripts/main.js

```
 import '/scripts/pedeInfosIniciais/index.js'
+import '/scripts/navegacao/barraEndereco.js'
```

2. Crie o arquivo **`barraEndereco.js`** na pasta **`scripts/navegacao`** com o seguinte código:

 # scripts/navegacao/barraEndereco.js

```
+$janelaPrincipal.onload = exibeEndereco
```

```
+
+function exibeEndereco(){
+    $inputEndereco.value = $janelaPrincipal.contentWindow.location.href
+}
```

CAPÍTULO 16

EXERCÍCIO: INCIANDO A NAVEGAÇÃO: MAIS EVENTOS PARA MELHORAR A BARRA DE ENDEREÇOS

16.1 COMENTÁRIOS

Enquanto navegamos usando o cake, a barra de endereços sempre está com o endereço completo da página atual. Em buscas do google, a url fica gigantesca. Agora que conhecemos melhor a ideia de eventos e callbacks, podemos melhorar isso e replicar o comportamento da barra de endereços de outros navegadores.

Mostraremos apenas um endereço resumido enquanto a pessoa navega e quando a barra de endereço é focada, mostramos o endereço completo para que possa ser copiado, por exemplo.

Para resumir o endereço, criaremos um objeto URL que contém cada parte da url separada como uma propriedade:

```
{
    protocol: 'http://',
    hostname: 'google.com',
    search: '/?q=teste&prop=234tghjkl'
}
```

Esse objeto será construído ao chamarmos a função construtora `URL` . Essa função já existe e está presente em quase todas as plataformas que rodam JavaScript. Para criar esse objeto, basta chamarmos a função, porém, antecedido de um novo operador, o `new` :

```
const url = new URL($janelaPrincipal.contentWindow.location.href)
```

16.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`barraEndereco.js`** na pasta **`scripts/navegacao`** faça as seguintes alterações:

 # scripts/navegacao/barraEndereco.js

```
+$inputEndereco.onfocus = exibeEnderecoCompleto

+$inputEndereco.onblur = exibeEnderecoResumido
-$janelaPrincipal.onload = exibeEndereco
+$janelaPrincipal.onload = exibeEnderecoResumido
```

```
+
-function exibeEndereco(){
+function exibeEnderecoCompleto(){
     $inputEndereco.value = $janelaPrincipal.contentWindow.location.href
 }
+
+function exibeEnderecoResumido() {
+    const url = new URL($janelaPrincipal.contentWindow.location.href)
+    const enderecoResumido = url.hostname
+
+    $inputEndereco.value = enderecoResumido
+}
```

Capítulo 17

EXERCÍCIO: ARRUMANDO O RELOAD

17.1 COMENTÁRIOS

TODO

17.2 PASSO A PASSO COM CÓDIGO

1. No arquivo **`main.js`** na pasta **`scripts`** faça as seguintes alterações:

   # scripts/main.js

   ```
    import '/scripts/pedeInfosIniciais/index.js'
    import '/scripts/navegacao/barraEndereco.js'
   +import '/scripts/navegacao/navegacao.js'
   ```

2. Crie o arquivo **`navegacao.js`** na pasta **`scripts/navegacao`** com o seguinte código:

   # scripts/navegacao/navegacao.js

   ```
   +import * as storagePaginaInicial from '/scripts/storage/paginaInicial.js'
   +import { formataEndereco } from '/scripts/endereco/formataEndereco.js'
   +
   +//TODO criar módulo pra isso
   +// Pegar pagina atual do localStorage
   +const paginaAtual = sessionStorage.getItem('paginaAtual')
   +
   +//TODO isolar funcao carrega
   +if (paginaAtual !== null) {
   +    const enderecoCompleto = formataEndereco(paginaAtual)
   +
   +    $janelaPrincipal.src = enderecoCompleto
   +    $inputEndereco.value = enderecoCompleto
   +} else {
   +    const enderecoCompleto = formataEndereco(storagePaginaInicial.paginaInicial)
   +
   +    $janelaPrincipal.src = enderecoCompleto
   +    $inputEndereco.value = enderecoCompleto
   +}
   ```

3. No arquivo **`pedePaginaInicial.js`** na pasta **`scripts/pedeInfosIniciais`** faça as seguintes alterações:

   # scripts/pedeInfosIniciais/pedePaginaInicial.js

   ```
    import { aceitouSalvar as storageAceitouSalvar } from '/scripts/storage/aceitouSalvar.js'
   ```

```
import { paginaInicial, setPaginaInicial } from '/scripts/storage/paginaInicial.js'

// named export
import { formataEndereco } from '/scripts/endereco/formataEndereco.js'

if(storageAceitouSalvar === null || storageAceitouSalvar === true){
    // Sem shadowing
    let paginaInicialDefault = paginaInicial

    if(!paginaInicialDefault) {
        paginaInicialDefault = prompt("Escolha a página inicial")
    }

    if(paginaInicialDefault) {
        const enderecoCompleto = formataEndereco(paginaInicialDefault)
-
-       $janelaPrincipal.src = enderecoCompleto
-       $inputEndereco.value = enderecoCompleto
-
        setPaginaInicial(enderecoCompleto)
    }
}
```