Curso Técnico Superior Profissional em Tecnologias e Programação de Sistemas de Informação

Unidade Curricular: Desenvolvimento Web – Front-End **1 º Ano/2º Semestre**

Docente: Marco Miguel Olival Olim **Data** 06/04/2018

ESTE EXERCÍCIO ABORDA CONCEITOS COMO A FILTRAGEM COM PIPES NUMA APLICAÇÃO COM CARRINHO DE COMPRAS

Depois da introdução de produtos no Firebase com o formulário criado, voltamos ao ficheiro index.vue correspondente à primeira página do site:

index.vue ● header.vue users.vue default.vue

```
<template>
  <div class="app">
      <app-ficha-produto v-for="(artigo, index) in artigos"
                         :key="index"
                         :produto="artigo.produto"
                         :valor="artigo.valor" />
  </div>
</template>
<script>
import axios from 'axios';
import appFichaProduto from '@/components/app-ficha-produto';
export default {
  components:{
      appFichaProduto
  },
  asyncData() {
    // retornar uma Promessa
    return axios.get('https://umartigos.firebaseio.com/.json')
      .then((res) => {
        return { artigos: res.data }
      })
  },
}
</script>
<style scoped>
.app{
    margin-top: 4.5rem;
    display:flex;
    align-items: center;
    flex-direction: column;
}
</style>
```

Faremos a seleção de produtos para um carrinho de compras, pelo que serão aplicados novos estilos:

```
<template>
<div class="app">
  <div class="listaProdutos">
      <app-ficha-produto v-for="(artigo, index) in artigos"
                         :key="index"
                         :produto="artigo.produto"
                         :valor="+artigo.valor"  />
  </div>
    <div class="listaCarrinhoCompras">

    </div>
  </div>
</template>
```

Cofinanciado por:

MADEIRA 14-20 · REGIÃO AUTÓNOMA DA MADEIRA · PORTUGAL 2020 · UNIÃO EUROPEIA Fundo Social Europeu

As listas ficarão lado a lado, pelo que o CSS necessário resulta em duas novas classes:

```
<style scoped>
.app{
    margin-top: 4.5rem;
    display:flex;
    flex-direction: row;
    justify-content: space-around;
}
.listaProdutos{
    display:flex;
    align-items: center;
    flex-direction: column;
}
.listaCarrinhoCompras{
    display:flex;
    min-width: 500px;
    align-items: center;
    flex-direction: column;
    border-radius: 5px;
    border: 1px solid #06c4d1;
    background-color: #eee;
}
</style>
```

Deverá obter um resultado semelhante:

Cofinanciado por:
MADEIRA 14-20
REGIÃO AUTÓNOMA DA MADEIRA
PORTUGAL 2020
UNIÃO EUROPEIA
Fundo Social Europeu

Os artigos selecionados para o carrinho de compras serão colocados num novo Array, devendo este ser inicializado em **data()**. Também será registada a data da última compra, sendo esta inicializada a **false**

```
<script>
import axios from 'axios';
import appFichaProduto from '@/components/app-ficha-produto';
export default {
  components:{
      appFichaProduto
  },
  data () {
      return {
          carrinhoCompras:[],
          ultimaCompra: false
      }
  },
  asyncData() {
    // retornar uma Promessa
    return axios.get('https://umartigos.firebaseio.com/.json')
      .then((res) => {
        return { artigos: res.data }
      })
  },
```

À ficha do produto será então associada uma função que transfere a informação desse produto para o carrinho de compras (sempre que essa ficha for clicada)

```
<div class="listaProdutos">
    <app-ficha-produto v-for="(artigo, index) in artigos"
                       :key="index"
                       :produto="artigo.produto"
                       :valor="+artigo.valor"
                       v-on:click.native="carregaCarrinho(artigo)"
                       style="cursor: pointer" />

</div>
```

Esta função é criada em **methods** e colocará o artigo no início do Array **carrinhoCompras**

```
<script>
import axios from 'axios';
import appFichaProduto from '@/components/app-ficha-produto';
export default {
  components:{
      appFichaProduto
  },
  data () {
      return {
          carrinhoCompras:[],
          ultimaCompra: false
      }
  },
    methods:{
      carregaCarrinho(artigo){
          this.carrinhoCompras.unshift(artigo)
      },
```

Cofinanciado por:

MADEIRA 14-20 · REGIÃO AUTÓNOMA DA MADEIRA · PORTUGAL 2020 · UNIÃO EUROPEIA · Fundo Social Europeu

Como é pretendido registar também a data da compra, adicionamos esta nova propriedade (data do sistema) ao objeto que contém os dados do produto com o **spread** operator (…)

```
this.carrinhoCompras.unshift({...artigo, dataCompra: new Date()});
```

Sempre que adicionamos um novo artigo ao carrinho, este fica no início do Array porque foi usado o unshift. Sendo assim, usamos a data desse artigo na propriedade **ultimaCompra**

```
40      methods:{
41          carregaCarrinho(artigo){
42              this.carrinhoCompras.unshift({...artigo, dataCompra: new Date()});
43              this.ultimaCompra = this.carrinhoCompras[0].dataCompra;
44          },
```

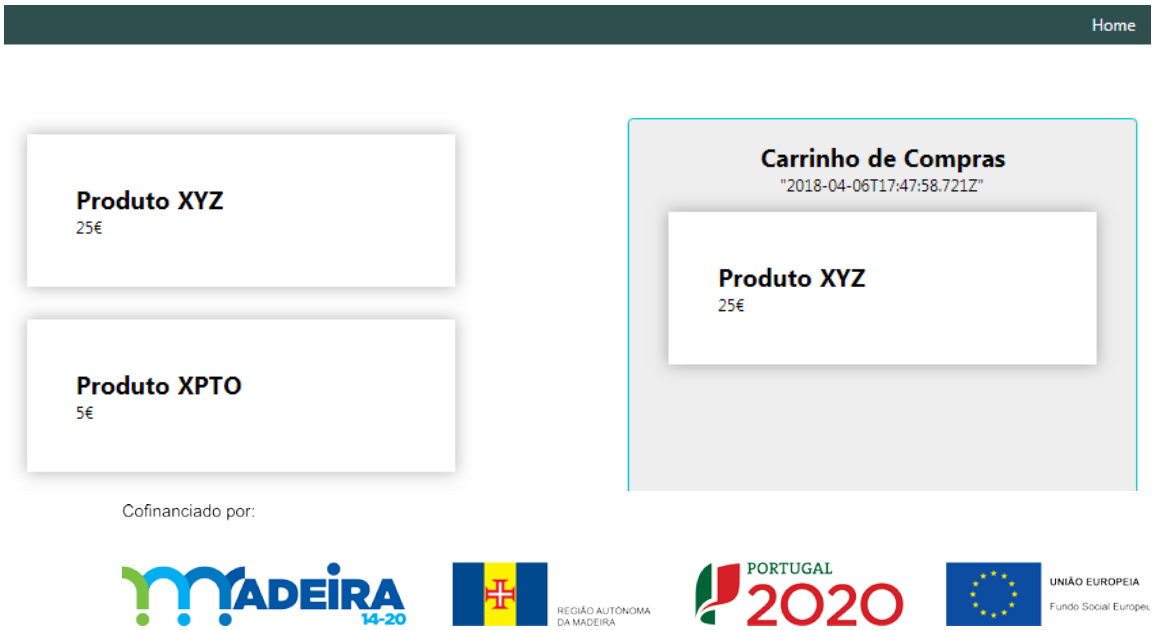
Usamos a interpolação de string para apresentar o resultado no carrinho de compras

```
12      <div class="listaCarrinhoCompras">
13        <br>
14        <h2>Carrinho de Compras</h2>
15        {{ultimaCompra}}
16
```

Acrescentamos uma nova lista de produtos, desta vez com os artigos do carrinhoCompras

```
1   <template>
2   <div class="app">
3     <div class="listaProdutos">
4         <app-ficha-produto v-for="(artigo, index) in artigos"
5                            :key="index"
6                            :produto="artigo.produto"
7                            :valor="+artigo.valor"
8                            v-on:click.native="carregaCarrinho(artigo)"
9                            style="cursor: pointer" />
10    </div>
11      <div class="listaCarrinhoCompras">
12        <br>
13        <h2>Carrinho de Compras</h2>
14        {{ultimaCompra}}
15        <app-ficha-produto v-for="(artigo, index) in carrinhoCompras"
16                           :key="index"
17                           :produto="artigo.produto"
18                           :valor="+artigo.valor"/>
19      </div>
20    </div>
21  </template>
```

Ao selecionarmos um produto, devemos obter o seguinte resultado



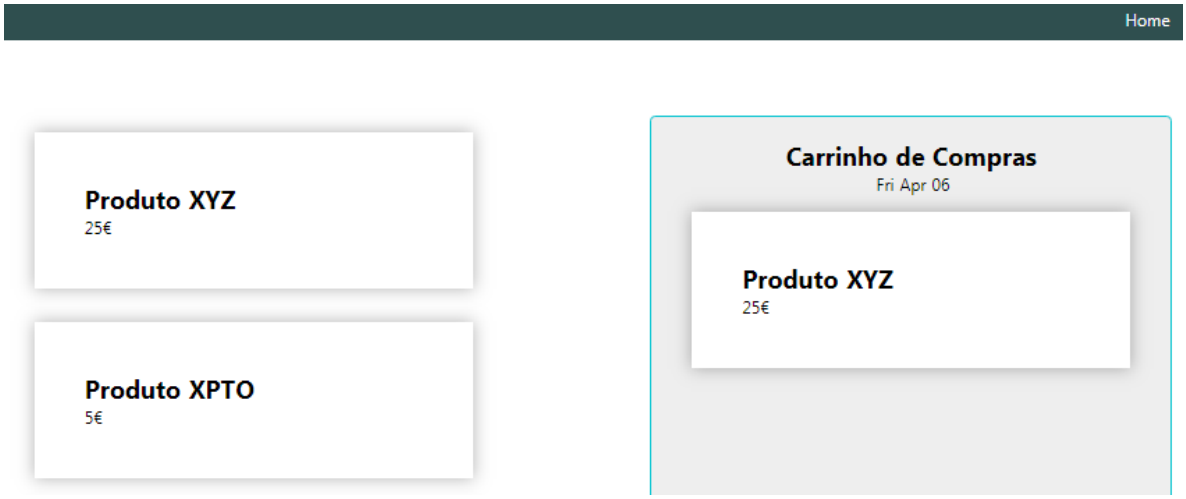
Como podemos verificar, a data fica com um aspeto pouco inteligível, mas pode ser formatada com recurso a um filtro. Este é aplicado diretamente na template com um pipe a separar o nome do filtro e a propriedade

```
{{ultimaCompra | formataData }}
```

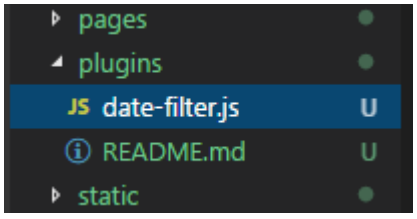
Este filtro é criado em **filters**. Inicialmente vamos apenas circunscrever a data aos primeiros 11 caracteres

```
<script>
import axios from 'axios';
import appFichaProduto from '@/components/app-ficha-produto';
export default {
    components:{
        appFichaProduto
    },
    data () {
        return {
            carrinhoCompras:[],
            ultimaCompra: false
        }
    },
    filters: {
        formataData(dados){
            return dados.toString().slice(0,10)
        }
    },
    methods:{
        carregaCarrinho(artigo){
            this.carrinhoCompras.unshift({...artigo, dataCompra: new Date()});
            this.ultimaCompra = this.carrinhoCompras[0].dataCompra;
        },
```

A data deverá ser renderizada já formatada com o novo filtro local



Esta formatação de datas poderá ser necessária noutras páginas e/ou componentes da aplicação, pelo que faz sentido criar um filtro global em vez de estar sempre a copiar este filtro local. O NUXT dá-nos essa possibilidade com PLUGINS. Neste separador vamos criar um novo ficheiro **date-filter.js**



Cofinanciado por:

Uma possível implementação deste filtro seria:

```
JS date-filter.js ●   index.vue

1   import Vue from 'vue'
2
3   const months = [
4       "Janeiro",
5       "Fevereiro",
6       "Março",
7       "Abril",
8       "Maio",
9       "Junho",
10      "Julho",
11      "Agosto",
12      "Setembro",
13      "Outubro",
14      "Novembro",
15      "Dezembro"
16  ];
17
18  const dateFilter = value => {
19      return formatDate(value);
20  };
21
22  function formatDate(inputDate) {
23      if (!!inputDate) {
24      const date = new Date(inputDate);
25      const year = date.getFullYear();
26      const month = date.getMonth();
27      const day = date.getDate();
28      const hour = date.getHours();
29      const min = date.getMinutes();
30      const formattedDate = day +
31                            " de " +
32                            months[month] +
33                            ", " +
34                            year +
35                            " às " +
36                            hour +
37                            ":" +
38                            min;
39      return formattedDate; } else { return 'Nenhuma compra efetuada'}
40  }
41  Vue.filter('date', dateFilter)
```

Note que o nome pelo que será invocado este filtro é '**date'**. Este plugin tem ainda de ser carregado no arranque da app para ficar disponível globalmente, sendo por isso necessário registá-lo em **nuxt.config.js**

```
JS date-filter.js   JS nuxt.config.js ×   index.vue

33    ** Plugins to load before mounting the App
34    */
35    plugins: [
36      '~plugins/date-filter.js'
37    ],
```

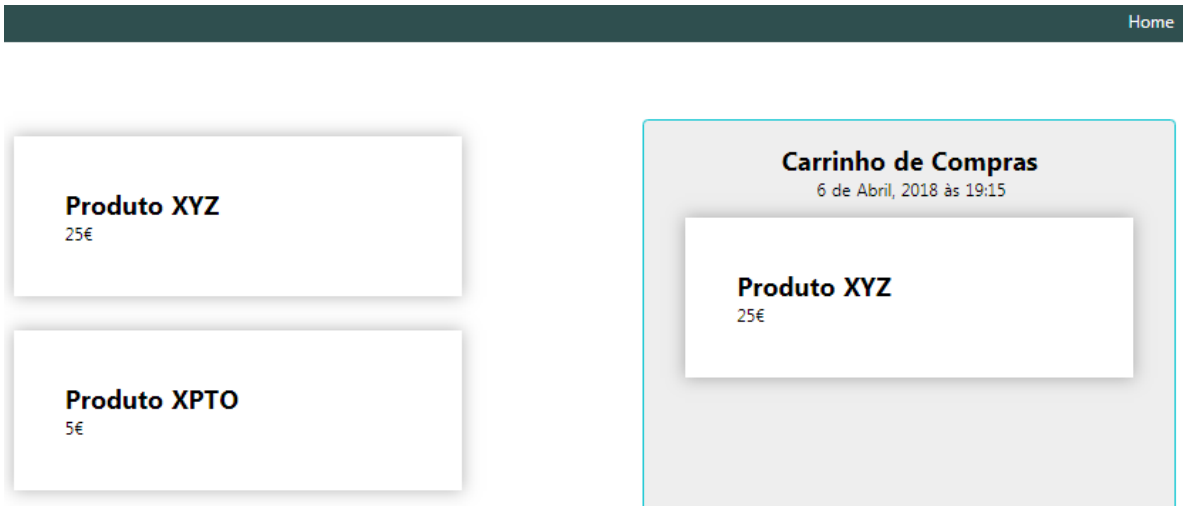
Cofinanciado por:

MADEIRA 14-20 · REGIÃO AUTÓNOMA DA MADEIRA · PORTUGAL 2020 · UNIÃO EUROPEIA · Fundo Social Europeu

A única alteração agora a fazer é mudar o nome do filtro na interpolação de string que o utiliza

```
{{ultimaCompra | date }}
```

A data já deve surgir com a nova formatação



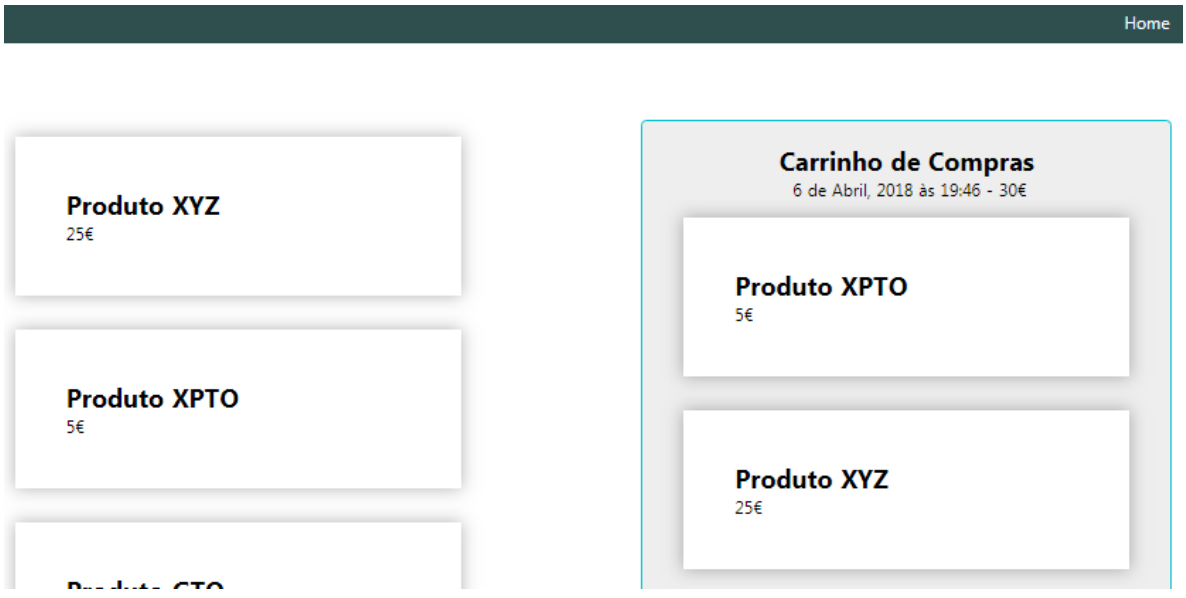
Á frente desta data vamos também interpolar o somatório do valor dos produtos a comprar

```
{{ultimaCompra | date }} - {{ sum() }}€
```

O método sum() calcula então o agregado dos valores, exceto no caso do carrinho não ter artigos

```
sum() {
    if (this.carrinhoCompras.length < 1) {
        return 0
    } else {
        return this.carrinhoCompras.map( (a) => Math.floor(a.valor))
                                   .reduce((a,b)=>{return  a + b })}
}
```

Este valor total é sempre atualizado reactivamente pelo VueJS sempre que adicionados artigos



Cofinanciado por:

Para o caso de ter sido selecionado um produto indesejado, é necessário criar um método para retirá-lo. Concluímos a aplicação implementando o **descarregaCarrinho** invocado ao clicar algum dos artigos do carrinho de compras.

index.vue

```vue
<template>
<div class="app">
  <div class="listaProdutos">
      <app-ficha-produto v-for="(artigo, index) in artigos"
                         :key="index"
                         :produto="artigo.produto"
                         :valor="+artigo.valor"
                         v-on:click.native="carregaCarrinho(artigo)"
                         style="cursor: pointer" />
  </div>
    <div class="listaCarrinhoCompras">
      <br>
      <h2>Carrinho de Compras</h2>
      {{ultimaCompra | date }} - {{ sum() }}€
      <app-ficha-produto v-for="(artigo, index) in carrinhoCompras"
                         :key="index"
                         :produto="artigo.produto"
                         :valor="+artigo.valor"
                         v-on:click.native="descarregaCarrinho(artigo)"
                         style="cursor: pointer" />
    </div>
  </div>
</template>
<script>
import axios from 'axios';
import appFichaProduto from '@/components/app-ficha-produto';
export default {
    components:{
        appFichaProduto
    },
    data () {
        return {
            carrinhoCompras:[],
            ultimaCompra: false
        }
    },
    methods:{
        carregaCarrinho(artigo){
            this.carrinhoCompras.unshift({...artigo, dataCompra: new Date()});
            this.ultimaCompra = this.carrinhoCompras[0].dataCompra;
        },
        descarregaCarrinho(index){
            this.carrinhoCompras.splice(index,1)
        },
        sum() {
            if (this.carrinhoCompras.length < 1) {
                return 0
            } else {
                return this.carrinhoCompras.map( (a) => Math.floor(a.valor))
                                           .reduce((a,b)=>{return  a + b })}
            }
    },
```

Cofinanciado por:

MADEIRA 14-20 · REGIÃO AUTÓNOMA DA MADEIRA · PORTUGAL 2020 · UNIÃO EUROPEIA Fundo Social Europeu