Jorge Costa da Silva

Manual do Vue.js

Openbox

www.openbox.pt

1. HELLO WORLD!

Vamos começar a nossa exploração do Vue.js. E como pensamos que há tradições que vale a pena manter, o primeiro passo será desenvolver uma aplicação Hello World!

Para isso, neste primeira capítulo abordaremos os seguintes temas:

- o que é o Vue;
- as duas formas possíveis de usar o Vue;
- quais as ferramentas para o desenvolvimento local;
- como se instala o Vue;
- a nossa primeira aplicação;
- propriedade “el”;
- método $mount;
- propriedade “data”;
- interpolação.

O QUE É O VUE?

O Vue é um framework de front-end. É mais uma solução ao nosso dispor para resolver o problema que temos constantemente da relação entre o interface de uma aplicação e os seus dados.

Sempre precisamos de criar ou adaptar o interface com base nos dados, e de actualizar os dados com base na interacção do utilizador com esse interface. A grande vantagem do Vue é que resolve este problema de um modo especialmente elegante e simples.

DUAS FORMAS DE UTILIZAÇÃO

O Vue pode ser usado de duas formas. Talvez a forma mais resumida de nos referirmos a essas duas formas seja “sem processo de build” e “com processo de build”. Ou seja, podemos usar o Vue sem recorrer a um processo de build, ou usar o Vue utilizando um processo de build.

Para o leitor menos habituado a esta terminologia, os processos de build, que se tornaram comuns nos últimos anos no desenvolvimento de aplicações web, significam fundamentalmente que trabalhamos com código-fonte que não podemos (ou, nalguns

casos, não queremos) entregar directamente ao browser. Necessitamos de passar esse código por um processo de build através do qual são construídos os ficheiros finais. São esses que colocamos no servidor, para serem entregue ao browser, quando este o solicitar.

Ao longo de praticamente todo o livro, a forma de utilização do Vue que se pressupõe é a primeira, sem processo de build. Isto acontece por várias razões. Por um lado, é o processo mais simples, que pode ser usado em qualquer contexto. Também é o modo usado na documentação oficial do Vue. Mas a razão mais importante será, possivelmente, que usar o Vue com ou sem processo de build muda muito pouco o modo concreto de lidar com o framework.

Embora as diferenças técnicas entre essas duas formas se possam considerar bastante acentuadas, na prática do dia-a-dia, saber usar o Vue tem quase as mesmas exigências seja uma ou outra a forma escolhida.

Por isso, apenas os dois último capítulo abordam o uso do Vue com processo de build. E isso não acontece porque o tópico seja pouco importante. Acontece simplesmente porque para usarmos o Vue com processo de build temos também de dominar todos os tópicos dos capítulos anteriores.

FERRAMENTAS PARA O DESENVOLVIMENTO LOCAL

Já nos vamos aproximando do Hello Word, mas ainda precisamos de nos referir às ferramentas que utilizaremos para o desenvolvimento local das aplicações.

Naturalmente, as versões de produção das aplicações onde utilizamos o Vue terão de ser instaladas nalgum servidor, através das múltiplas opções existentes. São exemplos um servidor em sentido estrito, uma máquina virtual ou um serviço na nuvem. Oportunamente, abordaremos também esta questão do deployment. Porém, para lá chegarmos, temos de passar primeiro pelo desenvolvimento. E, para isso, o mais simples é utilizar apenas um ambiente local.

A necessidade mais imediata é dispormos de um servidor local de http, de modo a executarmos as aplicações desenvolvidas. Há diversas possibilidades, mas a opção que preferimos será usar o Node.js. Isto porque, além de outras eventuais vantagens, mais tarde vamos utilizar uma importante aplicação de linha de comandos do Vue, o vue-cli, que funciona em Node.

Não está no âmbito deste livro abordar em detalhe o Node.js. Além disso, caso seja necessário, o leitor não terá dificuldade em encontrar recursos adequados para fazer a instalação do Node.

Aqui, referimos apenas que o Node.js é multi-plataforma, podendo ser instalado em diferentes sistemas operativos. Para fazer o download do instalador para o seu sistema, poderá visitar a seguinte página:

https://nodejs.org/en/download

Com a instalação do Node, é também instalado o npm, ou Node Package Manager, que iremos utilizar desde já.

Assim, executamos o comando:

> npm install http-server -g

Trata-se de usar a aplicação npm, com três argumentos. O primeiro argumento indica simplesmente que queremos fazer uma instalação. O segundo refere qual o pacote (package) que queremos instalar. Finalmente, o terceiro argumento, "-g", indica que pretendemos instalar o pacote globalmente, isto é, ficando disponível para qualquer aplicação em que estejamos a trabalhar. A alternativa seria fazer uma instalação local, que se aplica apenas a uma aplicação específica.

O modo mais simples de usarmos o http-server e executarmos o comando na pasta em que temos a nossa aplicação:

> http-server

Podem ver-se mais informações sobre esta aplicação em:

https://www.npmjs.com/package/http-server

A última ferramenta que mencionamos é um editor de código. De entre as diversas possibilidades, iremos utilizar o Visual Studio Code, ou VS Code. Trata-se de um editor open source, promovido pela Microsoft, que tem grande popularidade. Tal como em outros editores podemos instalar plugins que facilitam o trabalho. Atempadamente, falaremos dessa situação. Além disso, tem uma linha de comandos incorporada, que é bastante útil.

O download do VS Code pode fazer-se em:

https://code.visualstudio.com

INSTALAÇÃO DO VUE.JS

Convém começar por referir que há diversas versões do Vue. Mas, para já, não nos convém entrar nesses detalhes. É melhor evitar essas distrações por agora. Podemos só adiantar que, como acontece com frequência, há versões de desenvolvimento e versões de produção. Para já, é preferível usarmos uma versão de desenvolvimento, pois dá-nos algumas ajudas, como seja vermos na consola as mensagens para os erros mais frequentes.

Por outro lado, como vamos usar o Vue sem processo de build, faremos a instalação através de um CDN (content delivery network).

Para isso, basta usarmos esta linha no nosso html:

```
<script src="https://unpkg.com/vue/dist/vue.js"></script>
```

PRIMEIRA APLICAÇÃO

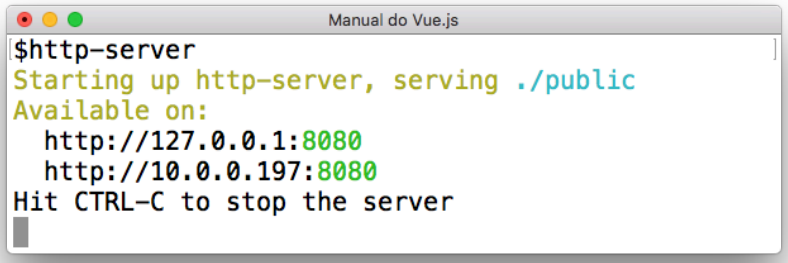
A nossa primeira aplicação Vue só terá um ficheiro, index.html, com o código:

```
<html>
<head>
    <script src="https://unpkg.com/vue/dist/vue.js"></script>
</head>
<body>
    <div id="app">
        <h1>{{ message }}</h1>
    </div>
    <script>
        var app = new Vue({
            el: "#app",
            data: {
                message: "Hello World!"
            }
        });
    </script>
</body>
</html>
```

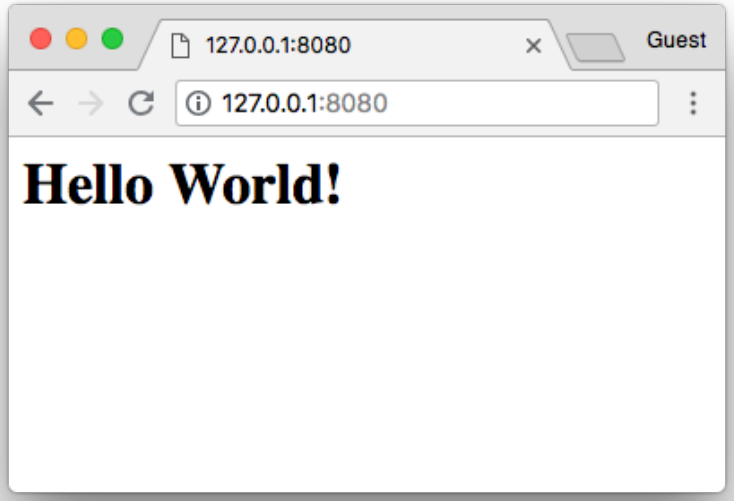
Para executar a aplicação bastará iniciarmos o nosso servidor de http com o seguinte comando, executado na raiz da pasta da aplicação:

> http-server

Usando a linha da comandos do VS Code, vemos:



Utilizando um dos dois links disponíveis, o browser mostra-nos algo muito simples, mas que é já uma aplicação Vue:



Este momento, como bem sabemos, é sempre um passo significativo. Vamos analisar o código com detalhe.

Na secção <header>, fazemos a instalação do Vue do modo acima referido, recorrendo ao CDN.

A seguir vem a parte mais interessante. Vemos que a secção <body> tem duas partes. Na primeira, temos apenas uma com o id de "app". Na segunda temos um <script>, cujo conteúdo repetimos:

```
var app = new Vue({
    el: "#app",
    data: {
        message: "Hello World!"
    }
});
```

Como podemos observar, uma aplicação Vue inicia-se com a instanciação de um objecto, recorrendo a uma função construtora de nome Vue. Essa função recebe apenas um parâmetro. Trata-se de um objecto com diversas opções, das quais aqui só utilizamos duas: a propriedade “el” e a “data”, que analisaremos de seguida.

PROPRIEDADE “EL”

A propriedade “el” serve para especificarmos qual o elemento do DOM em que o Vue vai ficar montado (“mounted”). O conceito de “moutend” ou “mount” é importante. O Vue, enquanto framework de front-end, é responsável por construir parte do nosso html. Às vezes, essa parte será quase tudo, outras vezes será menos. De qualquer forma, é necessário montar esse html criado pelo Vue. Usar a propriedade “el” é um dos modos de o fazer, embora não o único, como veremos.

O valor colocado em “el” pode ser de dois tipos. Ou um selector CSS ou um HTMLElement. Utilizámos aqui a primeira dessas possibilidades. Com o selector “#app” escolhemos o elemento do DOM com id “app”, ou seja, a que criámos no <body>.

Para testarmos a segunda possibilidade, podemos substituir o código anterior por:

```
var appElement = document.getElementById(“app”);
var app = new Vue({
    el: appElement,
    data: {
        message: “Hello World!”
    }
});
```

Neste caso, em lugar de um selector CSS usámos o próprio elemento, que previamente guardámos na variável appElement.

MÉTODO $MOUNT

Ainda podemos fazer de outro modo. A propriedade “el” não é estritamente necessária no momento da instanciação do Vue; podemos omiti-la. Se assim for, teremos de usar o método $mount(), em cujo argumento (que é opcional) podemos colocar também directamente um elemento ou um selector de CSS.

Deste modo, o código original ainda poderia ser substituído por:

```
var app = new Vue({
    data: {
        message: "Hello World!"
    }
});
app.$mount('#app');
```

Ou, então, por:

```
var appElement = document.getElementById("app");
var app = new Vue({
    data: {
        message: "Hello World!"
    }
});
app.$mount(appElement);
```

Vale a pena pensar se há alguma conveniência em usar "el" ou $mount? De facto, as duas técnicas conduzam exactamente ao mesmo resultado final. A única diferença é que o método $mount dá-nos um acréscimo de flexibilidade, pois separa o momento da instanciação do Vue do momento do mount. Pode haver cenários em que isso nos interesse.

Aliás, a flexibilidade é ainda maior pois o argumento do $mount, como referimos acima, é opcional. Quando não se utiliza qualquer argumento, a instância do Vue fica num estado "fora do documento", podendo nele ser inserido usando algum método apropriado da API do DOM, como por exemplo:

```
document.getElementById('app').appendChild(app.$el)
```

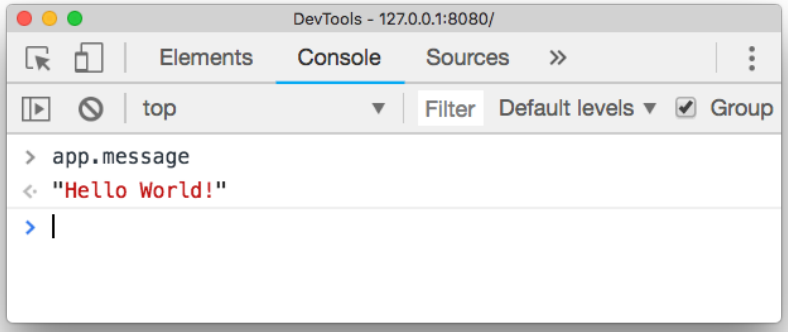
Aqui, "app" é a instância do Vue e $el é uma variável que guarda o elemento DOM criado pela instância.

PROPRIEDADE "DATA"

A outra propriedade que utilizámos foi "data" e iremos utilizá-la extensivamente. O valor desta propriedade deverá ser um objecto ou uma função que devolva um objecto.

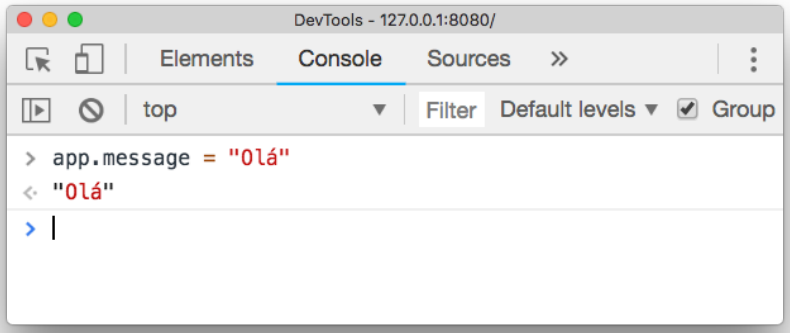
Esse objecto contém as diversas variáveis que pretendemos usar. O objecto original pode ser acedido através da instância do Vue, usando a propriedade "$data". Podemos comprovar isto através da consola do browser:

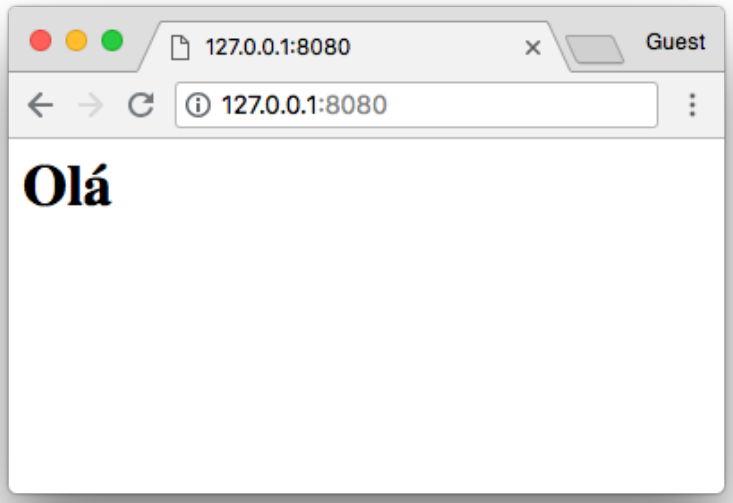
Por outro lado, as variáveis também pode ser acedidas directamente através da instância:



As diferentes variáveis do objecto “data” passam a ser reactivas, ou seja, sempre que o seu valor for alterado o interface será actualizado. Este é um aspecto-chave do Vue enquanto framework de front-end.

Para comprovarmos esta situação, façamos um teste, também utilizando a consola. Alteramos o valor de “message” para “Olá!”. Verificamos que, imediatamente, o conteúdo mostrado no browser se actualiza:





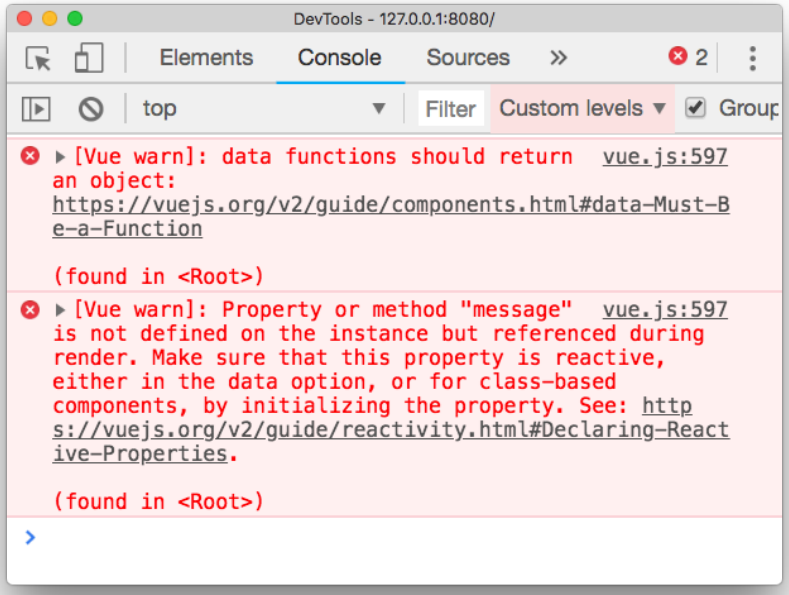
O objecto “data” pode ser obtido através de uma função que devolva esse objecto. Por isso, uma versão alternativa do código será:

```
var app = new Vue({
    el: '#app',
    data: function () {
        return {
            message: 'Hello World!'
        }
    }
});
```

Chamamos a atenção que é mesmo necessário que a função devolva o objecto. O código abaixo não é válido, pois neste caso a propriedade “data” não teria um objecto mas sim uma função (é certo que em JavaScript uma função é um objecto, mas o Vue exige um objecto em sentido estrito):

```
var app = new Vue({
    el: '#app',
    data: function() {
        {
            message: 'Hello World!'
        }
    }
});
```

Usando este código seríamos informados do erro cometido:



INTERPOLAÇÃO

Para terminar a nossa análise do Hello World falta apenas referir a interpolação. É com esta técnica que, a maior parte das vezes (embora não exclusivamente), mostramos a informação nos templates.

Foi o que fizemos em:

```
<h1>{{ message }}</h1>
```

Basta envolver num duplo par de chavetas a expressão desejada. Podemos usar as diferentes variáveis presentes em "data", bem como as "computed properties" e os métodos, que veremos em breve. Podemos ainda usar expressões simples de JavaScript.

Terminámos a nossa aplicação Hello World! Já percorremos algum caminho. Preparámos o ambiente de desenvolvimento local. Aprendemos a instalar o Vue. Vimos que o primeiro passo é instanciar o objecto Vue através de uma função construtora e fizemos a sua configuração inicial através das propriedades "el" e "data" e usámos a interpolação para mostrar a variável.

No próximo capítulo vamos passar pelos restantes aspectos fundamentais do Vue.