**第01章：vue的基本概念**

**一、vue概述**

**vue的基本概念**

在学习vue之前，同学们已经基本掌握了jQuery，那么这里就用vue与jQuery比较，来说明什么是vue。

* vue和jQuery都可以使用script标签引入到html文件中（但是vue并不推荐用script标签引入，后续课程会做详细讲解）。
* 一般一个项目中，不会同时使用vue和jQuery。
* 虽然vue和jQuery不是同一类型的库，但是vue可以完全替代jQuery。
* jQuery适合做网页效果，vue适合做web应用。例如，如果我们想做一个公司的网站，大部分内容都是展示型的效果，我们可以选择使用jQuery；如果我们想完成一个博客类型的网站，那么使用vue会更适合。
* 在处理数据的时候，jQuery需要手动将数据更新到DOM中，vue让我们把关注点放在数据本身，数据变更后，vue会自动更新DOM。

**vue与jQuery对比** 我们可以使用cnpm下载vue，在命令行工具中执行下面代码。

cnpm install vue

在node\_modules > vue > dist 目录中赋值vue.js文件，然后拷贝到我们的项目中。然后可以像引入jQuery一样，通过script标签引入vue。

我们在demo01和demo02中对比jQuery和vue如何实现数据绑定。我们在body中加入一个h1标签，然后分别通过jQuery和vue将"hello world"字符串在h1标签中显示。

*<!-- demo01 -->*

<body>

<h1></h1>

<script src="../../script/jquery.js"></script>

<script>

document.querySelector('h1').innerHTML **=** "hello world";

</script>

</body>

*<!-- demo02 -->*

<body>

<div id="app">

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

</div>

<script src="../../script/vue.js"></script>

<script>

**new** Vue({

el**:**"#app",

data**:**{

message**:**"hello world"

}

})

</script>

</body>

虽然使用vue完成这个例子，代码相对于jQuery会多一些，但是随着web应用功能的增加，数据会越来越复杂，数据操作的频率越来越好，jQuery的代码就会越来越难以维护（因为我们布置要管理数据，还要操作DOM），而vue会让这一切变得简单，因为使用vue开发web应用，我们关注点主要放在数据上，不必关注如何操作DOM。

我们再来看一下demo2中的代码，这是包含了一个vue程序最基本的结构： *在html中需要有一个div作为一个容器，我们在这里将他的id设置成了"app"方便选择。*在html中使用“双大括号”语法绑定数据。 *在js中，使用new Vue()创建一个vue的实例，这里的Vue()构造函数是vue库中提供的。*在构造函数Vue中，我们需要传入一个“选项对象”来进一步描述vue实例的行为。 *el属性是选择器，对应html中的容器*data属性是数据，可以指定vue实例中的数据，并将其绑定到html中

**二、vue的基本功能**

**绑定属性**

在上面的例子中，我们将vue中的数据绑定在html的文本节点上，我们其实还可以绑定html的属性节点，代码如下（demo03.html）:

<body>

<div id="app">

<p title="未绑定的title属性">我的title属性没有绑定数据</p>

<p v-bind:title="tit">我的title属性绑定了数据</p>

</div>

<script src="../../script/vue.js"></script>

<script>

**new** Vue({

el**:**"#app",

data**:**{

tit**:**"绑定的title属性"

}

})

</script>

</body>

* 通过v-bind可以将data中的数据绑定到html元素属性上
* 【v-bind：】冒号后面可以设置需要绑定的属性，我们可以使用它绑定任何属性。

下面我们做一个练习，用v-bind设置img标签的src属性（demo04.html）。

**绑定事件**

我们可以通过vue为元素绑定事件，方法如下（demo05.html）

<body>

<div id="app">

<button v-on:click="myEvent">按钮</button>

</div>

<script src="../../script/vue.js"></script>

<script>

**new** Vue({

el**:**"#app",

methods**:**{

myEvent**:function**(){

alert("hello world")

}

}

})

</script>

</body>

* 在html中，为需要绑定事件的元素添加【v-on:click=""】可以为元素绑定点击事件。
* 如果感觉这样写比较麻烦，可以简写成【@click=""】，效果与上面相同。

下面我们来做一个练习，让页面上有一个按钮和一张图片，当点击这个按钮的时候，图片切换成另一张图片（这里需要一个小提示，在事件的函数中，通过this.dataName可以获取和设置数据）（demo06.html）。

**三、列表**

**显示列表** 在学习jQuery的时候，我们学习过水果列表的例子，如果忘记了，我们可以再练习写一下这个例子：有一个数组如下所示：

**var** fruits **=** ["香蕉","苹果","鸭梨"];

通过js，将数组中的所有元素在html的列表中显示，jQuery的代码这里不再演示。我们的目标是用vue来实现显示列表的功能，在vue中，fruits不再是一个变量，而是data中的一个属性（demo07.html）。

<body>

<div id="app">

<ul>

<li v-for="fruit in fruits">{{fruit}}</li>

</ul>

</div>

<script src="../../script/vue.js"></script>

<script>

**new** Vue({

el**:**"#app",

data**:**{

fruits**:**["香蕉","苹果","鸭梨"]

}

})

</script>

</body>

**获取索引**

* 通过v-for可以实现列表的显示
* v-for后面的值需要有in，in的后面是一个集合类型的数据，in前面的是集合中的每一个元素
* 在上面的例子中"fruit in fruits"中的fruit代表集合中的每个元素，集合中有多少个元素，就会有多少个li被生成，这些不用我们手动操作，vue都为我们做好了。

如果将代码改成下面的样子，我们还可以获取到数据元素的索引（demo08.html）

<div id="app">

<ul>

<li v-for="(val,index) in fruits">

<p>水果名字是 {{val}}</p>

<p>水果序号是 {{index}}</p>

</li>

</ul>

</div>

**列表操作**

在上一节的例子中，我们讲解了：

* 如何将数组中的元素在列表中显示
* 如何获取元素的所以

接下来我们要做一个例子，仍然是显示一个普通的水果列表，当我们点击水果列表中的元素时，可以在控制台输出这个元素的索引（demo09.html）

<body>

<div id="app">

<ul>

<li

v-on:click="showIndex(index)"

v-for="(val,index) in fruits">

{{val}}

</li>

</ul>

</div>

<script src="../../script/vue.js"></script>

<script>

**new** Vue({

el**:**"#app",

data**:**{

fruits**:**["香蕉","苹果","鸭梨"]

},

methods**:**{

showIndex**:function**(index){

console.log(index);

}

}

})

</script>

</body>

通过上面的例子我们可以知道v-onclick="showIndex(index)"代码中,事件是可以传递参数的，通过这种方式可以将数据元素的索引传递到methods中，再通过形参来获取这个索引，既然可以获取到数据元素的索引，那么我们就可以实现一个“删除列表元素的功能”（demo10.html）

<body>

<div id="app">

<ul>

<li v-for="(val,index) in fruits">

{{val}}

<span v-on:click="del(index)">删除</span>

</li>

</ul>

</div>

<script src="../../script/vue.js"></script>

<script>

**new** Vue({

el**:**"#app",

data**:**{

fruits**:**["香蕉","苹果","鸭梨"]

},

methods**:**{

del**:function**(index){

**this**.fruits.splice(index,1);

}

}

})

</script>

</body>

上面的例子在li中添加了另一个span元素，我们为span绑定单击事件，并添加了删除元素的功能。

在这里，我们不必考虑考虑如何删除DOM元素节点，如何操作DOM的文本节点，我们只需要将html的元素和属性与vue中的选项逐一对应，然后关注点放在数据上，就能实现我们需要的功能了。

**七、授课说明**

* 本章内容占12课时，学员的练习时间需要略大于授课时间，以确保学员可以充分掌握授课内容。
* 本章内容需要学员熟练掌握jQuery中DOM操作相关的内容，若学员基础较差，对jQuery中DOM操作的内容已经忘记，那么在做每个vue的案例之前，都需要先用jQuery来实现一遍，否则无法实现教学效果。