Vue.js 动手实验

主讲人： 和凌志

准备工作

Vue.js 概述

- https://cn.vuejs.org/

JavaScript框架

Vue.js是一个用于创建用户界面的开源JavaScript框架

数据双向绑定

采用数据劫持结合发布者-订阅者模式的方式，通过Object.defineProperty()来劫持各个属性的setter，getter，在数据变动时发布消息给订阅者，触发相应的监听回调。

前后端分离

后端只给前端提供数据，前端负责HTML渲染和用户交互。

准备工作

- VUE 官网： https://cn.vuejs.org/
- 引入 VUE 组件库
- <!-- 开发环境版本，包含了有帮助的命令行警告 -->

```
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
```

Vue 常用指令

- v-bind
- v-model
- v-on
- v-if
- v-for
- v-click

第一个 Demo （Hello World）

创建一个 Vue 实例

- 每个 Vue 应用都是通过用 Vue 函数创建一个新的 **Vue 实例**开始的：

```
var vm = new Vue({
  // 选项
})
```

- 在文档中经常会使用 vm (ViewModel 的缩写) 这个变量名表示 Vue 实例。

第一个 Vue Demo

```
<!-- 引入 VUE 组件库 -->
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
```

```
<div id="app">
    <p> {{ message }}</p>
</div>
<script>
    var vm = new Vue({
        el: '#app',
        data: {
            message: "Hello World",
        }
    });
</script>
```

代码： vue-case-01

第一个 Vue Demo （完整代码）

```
<!-- 引入 VUE 组件库 -->
  <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"><
/script>


  <div id="app">
      <p> {{ message }}</p>
    </div>

    <script>
      var vm = new Vue({
        el: '#app',
        data: {
          message: "Hello World",
        }
      });
    </script>
```

条件语句 （ v-if ）

条件 v-if

```
<div id="app-3">
    <p v-if="seen">现在你看到我了</p>
</div>
```

```
<script>
    var app3 = new Vue({
        el: '#app-3',
        data: {
            seen: true
        }
    })
```

前提条件： 引入 Vue.js 库

代码： vue-case-02

条件 v-if

前提条件： 引入 Vue.js 库

```
<div id="app-3">
    <p v-if="seen">现在你看到我了</p>
 </div>

    <script>
        var app3 = new Vue({
            el: '#app-3',
            data: {
                seen: true
            }
        })
    </script>
```

循环语句 （ v-for ）

循环语句 v-for

v-for 指令： 可以绑定数组的数据来渲染一个项目列表

```
<div id="app-4">
    <ul>
        <li v-for="todo in todos">
            {{ todo.text }}
        </li>
    </ul>
</div>
```

```
<script>
    var app4 = new Vue({
        el: '#app-4',
        data: {
            todos: [
                { text: '学习 JavaScript' },
                { text: '学习 Vue' },
                { text: '整个牛项目' }
            ]
        }
    });
</script>
```

- 学习 JavaScript
- 学习 Vue
- 整个牛项目

循环语句 v-for（完整代码）

```
<div id="app-4">
    <ul>
        <li v-for="todo in todos">
            {{ todo.text }}
        </li>
    </ul>
</div>

<script>
    var app4 = new Vue({
        el: '#app-4',
        data: {
            todos: [
                { text: '学习 JavaScript' },
                { text: '学习 Vue' },
                { text: '整个牛项目' }
            ]
        }
    });
</script>
```

前提条件： 引入 Vue.js 库

代码： vue-case-02

循环语句 （ v-for ）

循环语句 v-for （迭代数组）

```
<div id="app-4">
    <ul>
        <li v-for="todo in todos">
            {{ todo.text }}
        </li>
    </ul>
</div>
```

```
<script>
    var app4 = new Vue({
        el: '#app-4',
        data: {
            todos: [
                { text: '学习 JavaScript' },
                { text: '学习 Vue' },
                { text: '整个牛项目' }
            ]
        }
    });
</script>
```

v-for 指令： 可以绑定数组的数据来渲染一个项目列表

- 学习 JavaScript
- 学习 Vue
- 整个牛项目

循环语句 v-for（迭代数组）

```
<div id="app-4">
    <ul>
        <li v-for="todo in todos">
            {{ todo.text }}
        </li>
    </ul>
</div>

<script>
    var app4 = new Vue({
        el: '#app-4',
        data: {
            todos: [
                { text: '学习 JavaScript' },
                { text: '学习 Vue' },
                { text: '整个牛项目' }
            ]
        }
    });
</script>
```

前提条件： 引入 Vue.js 库

代码： vue-case-03

循环语句 v-for （迭代对象）

```
<div id="app-4">
    <ul>
        <li v-for="value in object">
            {{ value }}
        </li>
    </ul>
</div>
```

```
var app4 = new Vue({
    el: '#app-4',
    data: {
        object:
        {
            name: '张三',
            age: 20,
            city: "北京"
        },
    }
});
```

- 张三
- 20
- 北京

循环语句 v-for（迭代对象）

```
<div id="app-4">
  <ul>
    <li v-for="value in object">
      {{ value }}
    </li>
  </ul>
</div>

<script>
  var app4 = new Vue({
    el: '#app-4',
    data: {
      object:
      {
        name: '张三',
        age: 20,
        city: "北京"
      },
    }
  });
</script>
```

前提条件： 引入 Vue.js 库

代码： vue-case-04

事件处理 （v-on）

处理用户点击事件 v-on

- 为了让用户和你的应用进行交互，我们可以用 v-on 指令添加一个事件监听器，通过它调用在 Vue 实例中定义的方法
- 可以用 v-on 指令监听 DOM 事件，并在触发时运行一些 JavaScript 代码。

v-on 指令 示例-1

```
<div id="app-5">
    <p>{{ message }}</p>
    <button v-on:click="reverseMessage">反转消息</button>
</div>

<script>
    var app5 = new Vue({
        el: '#app-5',
        data: {
            message: 'Hello Vue.js!'
        },
        methods: {
            reverseMessage: function () {
                this.message = this.message.split('').reverse().join('')
            }
        }
    })
</script>
```

v-on 指令

前提条件：引入 Vue.js 库　　代码：vue-case-05

```
<div id="app-5">
        <p>{{ message }}</p>
        <button v-on:click="reverseMessage">反转消息</button>
    </div>

    <script>
        var app5 = new Vue({
            el: '#app-5',
            data: {
                message: 'Hello Vue.js!'
            },
            methods: {
                reverseMessage: function () {
                    this.message = this.message.split('').reverse().join('')
                }
            }
        })
    </script>
```

v-on 指令 示例-2

```
<div id="example-3">
    <button v-on:click="say('hi')">Say hi</button>
    <button v-on:click="say('what')">Say what</button>
</div>

<script>
    new Vue({
        el: '#example-3',
        methods: {
            say: function (message) {
                alert(message)
            }
        }
    })
</script>
```

v-on 指令 –示例2

前提条件： 引入 Vue.js 库　　代码： vue-case-06

```
<div id="example-3">
        <button v-on:click="say('hi‘)“ >Say hi</button>
        <button v-on:click="say('what‘)“ >Say what</button>
    </div>

    <script>
        new Vue({
            el: '#example-3',
            methods: {
                say: function (message) {
                    alert(message)
                }
            }
        })
    </script>
```

双向数据绑定 （ v-model ）

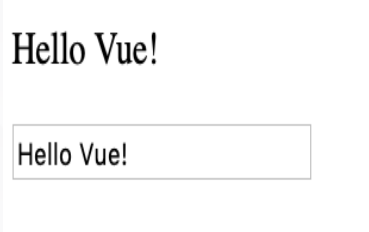
双向数据绑定 v-model

- Vue 还提供了 v-model 指令，它能轻松实现表单输入和应用状态之间的双向绑定。

双向数据绑定 v-model

```
<div id="app-6">
    <p>{{ message }}</p>
    <input v-model="message">
</div>

<script>
    var app6 = new Vue({
        el: '#app-6',
        data: {
            message: 'Hello Vue!'
        }
    })
</script>
```



双向数据绑定 v-model

前提条件： 引入 Vue.js 库

代码： vue-case-07

```
<div id="app-6">
    <p>{{ message }}</p>
    <input v-model="message">
</div>

<script>
    var app6 = new Vue({
        el: '#app-6',
        data: {
            message: 'Hello Vue!'
        }
    })
</script>
```

V-bind 指令

v-bind 指令

v-bind 完整语法　　<a v-bind:href="url">

v-bind 缩写语法　　<a :href ="url">

- v-bind指令的 缩写　:
- <a v-bind: href="url">

这里 href 是参数，它告诉 v-bind 指令将元素的 href 特性跟表达式 url 的值绑定

v-bind

```
<div id="app-6">
    <a v-bind:href="url"> 网站地址-可动态修改</a>
</div>

<script>
    var app6 = new Vue({
        el: '#app-6',
        data: {
            url: 'http://www.baidu.com'
        }
    })
</script>
```

网站地址-可动态修改

v-bind

前提条件： 引入 Vue.js 库

代码： vue-case-08

```
<div id="app-6">
    <a v-bind:href="url"> 网站地址-可动态修改</a>
</div>
<script>
    var app6 = new Vue({
        el: '#app-6',
        data: {
            url: 'http://www.baidu.com'
        }
    })
</script>
```

注意：

v-bind: 与属性之间不能有空格

v-bind:href

:href

v-bind: href

动手实验

- 引入 Vue.js
- 创建 Vue 组件
- 实现数据绑定

通过 Vue 改造静态页面

页面效果

第1个卡片

这里是详情描述11111, 这里是详情描述11111,

第2个卡片

这里是详情描述2222, 这里是详情描述22222,

第3个卡片

这里是详情描述3333, 这里是详情描述333,

第4个卡片

这里是详情描述4444, 这里是详情描述44,

实现思路

- 在原有静态页面基础上，引入 Vue.js
- Vue 是数据驱动的，构建数据结构
- 用Vue 语法，比如 数据绑定、事件绑定、v-for 循环、 v-if 条件判断等

代码实现—— 构建数据结构

商品列表由多个商品构成，每个商品是一个对象 （object）， 所有商品构成一个数组，

实际项目中，数据来自后台服务器 。 { } 表示对象； [ ] 表示数组

```
data: {
 list:
  [
    {
     image: "./images/photo.png",
     title: '第1个卡片',
     detail: "这里是详情描述11111, 这里是详情描述11111,"
    },

   {
    image: "./images/photo.png",
    title: '第2个卡片',
    detail: "这里是详情描述2222, 这里是详情描述22222,"
    }
    ……
 ]
```

代码实现—— HTML 数据绑定

- 图片地址， src 改为 :src = “变量”
- 文字绑定，改为 {{ 对象变量 }}， 不再是静态的内容
- 注意 v-for 的作用域， v-for 所在块（block）开始循环； 循环次数取决于数组的长度

```
<div  v-for= “item in list”  >
```

代码实现—— HTML 页面

```
<div id="app-6" class="container">
    <div class="row">
        <div class="card" style="width: 18rem;" v-for= "item in list">
        <img :src="item.image" class="card-img-top" alt="...">
        <div class="card-body">
            <h3> {{item.title}} </h3>
            <p class="card-text"> {{item.detail}}</p>
        </div>
        </div>
    </div>
</div>
```

动手： 扩展

- 把图片改为url， url 来自网络
- 添加一个价格标签，比如： 99 ￥
- 商品名称、 商品详情、商品价格等

小结

- ✓ 掌握 MVVM 前端框架理念
- ✓ 掌握 Vue.js 实例
- ✓ 掌握 Vue.js 数据绑定与事件绑定

下节课预告

- Vue.js 生命周期钩子
- Vue.js 组件的应用
- Element-UI 的引入