1. 概述

Vue.js是当下很火的一个JavaScript MVVM库，它是以数据驱动和组件化的思想构建的。**ViewModel是Vue.js的核心，它是一个Vue实例。**

<body>

<div id="app"> <!-- 这是我们的View -->

{{ message }}

</div>

</body>

<script src="js/vue.js"></script> <!-- 引包 -->

<script>

var exampleData = { //这是Model

message: 'Hello World!'

}

// new出一个 Vue 实例或 "ViewModel"，里面需要传入一个选项对象（包含数据、模板、挂载元素、方法、模生命周期钩子等）。

new Vue({

el: '#app', //指向View，表示该Vue实例将挂载到哪个元素上

data: exampleData //Model

})

</script>

1. Vue的数据绑定的语法

|  |  |
| --- | --- |
| 文本插值 | <div id="app">  {{ message }}  </div> |
| 表单元素双向数据绑定 | <div id="app-6">  <input v-model = "message">  </div> |
|  | <div id="app-2">  <span v-bind:title="message">  鼠标悬停几秒钟查看此处动态绑定的提示信息！  </span>  </div> |

1. Vue的指令

|  |  |  |
| --- | --- | --- |
| 条件渲染 v-if，根据条件表达式的值来执行元素的插入或删除 | | <div id="app-3">  <p v-if = "seen">现在你看到我了</p>  </div> |
| v-else | v-else 元素必须紧跟在 v-if 或者 v-else-if 元素的后面——否则它将不会被识别 | |
| v-show | 使用v-show指令的元素始终会被渲染，它只是简单地为元素设置style属性。  <h4 id="box" style="display: none;">Love</h4> | |
|  |  | |
| 循环 v-for  从**1**开始循环 | <li v-for = "item in items">  {{ item.text }}  </li>  <tr v-for = "(value,index) in persons"> | |
| 事件监听  v-on / @ | <div id="app-5">  <p>{{ message }}</p>  <button v-on:click="greet">Greet</button>  <button v-on:click="say**(**‘Hi’**)**">Hi!</button>  </div>  函数中的this指的是当前的Vue实例 | |
| v-bind / ： | v-bind:argument = "expression" //冒号和参数之间不能有空格  argument通常是HTML元素的属性，如title / class | |
| v-model |  | |

1. 组件
2. 注册组件：

Vue.component('todo-item', {

template: '<li>apple</li>'

})

1. 使用Vue.component() 注册组件时，需要提供2个参数，第1个参数是
2. 其他

<script>

var vm = new Vue({

el: '#app',

data: {

yes: true,

no: false,

age: 28,

name: 'keepfool'

}

})

</script>

可以在控制台直接输入：vm.age = 24，问：age是定义在选项对象的data属性中的，为什么Vue实例可以直接访问它呢？因为**每个Vue实例都会代理其data对象里所有的属性**。