# Vue1

Vue是一套用于构建用户界面的渐进式框架。

Vue的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与现代化的工具链以及各种支持类库结合使用时，Vue 也完全能够为复杂的单页应用提供驱动。

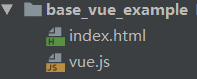
官方网站：https://cn.vuejs.org

## Vue基础

### 安装

#### 直接引入

**方式一**：下载vue.js文件并用 <script> 标签引入，Vue 会被注册为一个全局变量。



index.html

|  |
| --- |
| <head>  <meta charset="UTF-8">  <title>Title</title>  <script src="vue.js" type="text/javascript" charset="utf-8"></script> </head> |

**方式二**：通过CDN的方式来安装

|  |
| --- |
| <head>  <meta charset="UTF-8">  <title>Title</title>  <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script> </head> |

#### NPM

在用 Vue 构建大型应用时推荐使用 NPM 安装[1]。NPM 能很好地和诸如 webpack 或 Browserify 模块打包器配合使用。同时 Vue 也提供配套工具来开发单文件组件。

|  |
| --- |
| # 最新稳定版  $ npm install vue |

#### 命令行工具(CLI)

Vue 提供了一个官方的 CLI，为单页面应用 (SPA) 快速搭建繁杂的脚手架。它为现代前端工作流提供了 batteries-included 的构建设置。只需要几分钟的时间就可以运行起来并带有热重载、保存时 lint 校验，以及生产环境可用的构建版本。

|  |
| --- |
| <body>  <!-- id标识vue作用的范围 -->  <div id="app">  <!-- {{}} 插值表达式，绑定vue中的data数据 -->  {{ message }}111{{message}}  </div>  </body>  <script src="../vue.min.js"></script>  <script>  // 创建一个vue对象  new Vue({  el: '#app',//绑定vue作用的范围  data: {//定义页面中显示的模型数据  message: 'Hello Vue!'  }  })  </script> |

这就是声明式渲染：Vue.js 的核心是一个允许采用简洁的模板语法来声明式地将数据渲染进 DOM 的系统这里的核心思想就是没有繁琐的DOM操作，例如jQuery中，我们需要先找到div节点，获取到DOM对象，然后进行一系列的节点操作。

## Vue基本语法

### 基本数据渲染和指令

v-bind 特性被称为指令，指令带有前缀 v- ，除了使用插值表达式{{}}进行数据渲染，也可以使用 v-bind指令，它的简写的形式就是一个冒号（:）。

|  |
| --- |
| <body>  <div id="app">  <!-- 如果要将模型数据绑定在html属性中，则使用 v-bind 指令  此时title中显示的是模型数据-->  <h1 v-bind:title="message">我是标题</h1>  <!-- v-bind 指令的简写形式： 冒号（:） -->  <h1 :title="message">我是标题</h1>  </div>  </body>  <script src="../vue.min.js"></script>  <script>  new Vue({  el: '#app',  data: {  message: '页面加载于 ' + new Date().toLocaleString()  }  })  </script> |

### 双向数据绑定

双向数据绑定和单向数据绑定：使用 v-model 进行双向数据绑定。

|  |
| --- |
| <body>  <div id="app">  <!-- v-bind:value只能进行单向的数据渲染 -->  <input type="text" v-bind:value="searchMap.keyWord">  <!-- v-model 可以进行双向的数据绑定 -->  <input type="text" v-model="searchMap.keyWord">  <p>您要查询的是：{{searchMap.keyWord}}</p>  </div>  </body>  <script src="../vue.min.js"></script>  <script>  new Vue({  el: '#app',  data: {  searchMap: {  keyWord: 'Hello World'  }  }  })  </script> |

### 事件

|  |
| --- |
| <body>  <div id="app">  <!-- v-on 指令绑定事件，click指定绑定的事件类型，事件发生时调用vue中methods节点中定义的方法 -->  <button v-on:click="search()">查询1</button>  <!-- v-on 指令的简写形式 @ -->  <button @click="search()">查询2</button>  <p><a v-bind:href="result.site" target="\_blank">{{result.title}}</a></p>  </div>  </body>  <script src="../vue.min.js"></script>  <script>  new Vue({  el: '#app',  data: {  //查询结果  result: {}  },  methods: {  search() {  console.log('search')  this.result = {  "title": "百度",  "site": "https://www.baidu.com"  }  }  }  })  </script> |

# Vue路由

Vue.js 路由允许我们通过不同的 URL 访问不同的内容。通过 Vue.js 可以实现多视图的单页Web应用（single page web application，SPA）。

|  |
| --- |
| <body>  <div id="app">  <h1>Hello App!</h1>  <p>  <!-- <router-link> 默认会被渲染成一个 `<a>` 标签 -->  <!-- 通过传入 `to` 属性指定链接. -->  <router-link to="/">首页</router-link>  <router-link to="/student">会员管理</router-link>  <router-link to="/teacher">讲师管理</router-link>  </p>  <!-- 路由出口 -->  <!-- 路由匹配到的组件将渲染在这里 -->  <router-view></router-view>  </div>  </body>  <script src="vue.min.js"></script>  <script src="vue-router.min.js"></script>  <script>  // 1. 定义（路由）组件，复杂的组件也可以从独立的vue文件中引入  const Welcome = { template: '<div>欢迎</div>' }  const Student = { template: '<div>student list</div>' }  const Teacher = { template: '<div>teacher list</div>' }  // 2. 定义路由，每个路由应该映射一个组件。  const routes = [  { path: '/', redirect: '/welcome' }, //设置默认指向的路径  { path: '/welcome', component: Welcome },  { path: '/student', component: Student },  { path: '/teacher', component: Teacher }  ]  // 3. 创建 router 实例，然后传 `routes` 配置  const router = new VueRouter({  routes // （缩写）相当于 routes: routes  })  // 4. 创建和挂载根实例,从而让整个应用都有路由功能  new Vue({  el: '#app',  router  })  </script> |

# axios请求

axios是独立于vue的一个项目，可以用于浏览器和node.js中发送ajax请求。

|  |
| --- |
| <body>  <div id="app">  <button @click="getTeacherList()">访问</button>  <table>  <tr v-for="item in teacherList">  <td>{{item.id}}</td>  <td>{{item.name}}</td>  <td>{{item.career}}</td>  </tr>  </table>  </div>  </body>  <script src="vue.min.js"></script>  <script src="axios.min.js"></script>  <script>  new Vue({  el: '#app',  data: {  teacherList: []  },  methods: {  getTeacherList() {  axios.get('http://localhost:8001/teacher/list')  .then(response => {  console.log(response)  this.teacherList = response.data.data.items  })  }  }  })  </script> |

# Vue组件库

## element-ui

element-ui 是饿了么前端出品的基于 Vue.js的 后台组件库，方便程序员进行页面快速布局和构建。

### 使用示例

1、引入脚本库，将element-ui/lib目录引入到项目中。

2、在页面中引入CSS和JS。

|  |
| --- |
| <link rel="stylesheet" href="element-ui/lib/theme-chalk/index.css" />  <script src="element-ui/lib/index.js"></script>  <script src="vue.min.js"></script> |

3、编写HTML

|  |
| --- |
| <div id="app">  <el-button @click="visible = true">Button</el-button>  <el-dialog :visible.sync="visible" title="Hello world">  <p>Try Element</p>  </el-dialog>  </div> |

4、编写JS

|  |
| --- |
| <script>  new Vue({  el: '#app',  data: function () {  return { visible: false }  }  })  </script> |

5、在浏览器中运行

## Easy-ui

# Vue小技巧

## 访问显示Invalid Host header异常

出现这种情况是因为新版的webpack-dev-server出于安全考虑，默认检查hostname，如果hostname不是配置内的就不能访问。这样有2中方法，一种是设置跳过host检查，一种是直接host设置成你的地址。

**方式一**

可以在build目录下的webpack.dev.conf.js文件，devServer下添加disableHostCheck: true，跳过检查。

|  |
| --- |
| devServer: {  ...  quiet: true, // necessary for FriendlyErrorsPlugin  disableHostCheck: true,  watchOptions: {  poll: config.dev.poll  }  }, |

同样的原理，可以在package.json文件修改scripts命令:webpack-dev-server --disableHostCheck=true。

|  |
| --- |
| "scripts": {  "dev": "webpack-dev-server --disableHostCheck=true --inline --progress --config build/webpack.dev.conf.js --host 192.168.0.9",  ...  } |

**方式二**

设置host，假如host是xxx.com，同样2种方法，修改配置文件和script命令。

在config目录下修改index.js文件的host，这个默认是localhost，可修改成xxx.com。

|  |
| --- |
| dev: {  ...  host: 'localhost',  port: 9000,  ...  }, |

package.json的script语句: webpack-dev-server --host=xxx.com或者--public=xxx.com。

## 关闭严格检查

将src/config目录下的index.js文件中的useEslint的值改为false。

|  |
| --- |
| // Use Eslint Loader?  // If true, your code will be linted during bundling and  // linting errors and warnings will be shown in the console.  useEslint: false, |

## 手机访问VUE项目无法发送axios请求

VUE项目的axios请求的BaseURL设置的是localhost:port，访问的是当前机器的IP地址，因此需要将其设置为真实的后端的IP地址，这样手机才能够访问的到。

还要在package.json中进行设置：

|  |
| --- |
| "scripts": {  "dev": "webpack-dev-server --disableHostCheck=true --inline --progress --config build/webpack.dev.conf.js --host 192.168.0.9",  "start": "npm run dev",  "build": "node build/build.js",  "build:report": "npm\_config\_report=true npm run build",  "lint": "eslint --ext .js,.vue src",  "test": "npm run lint",  "svgo": "svgo -f src/icons/svg --config=src/icons/svgo.yml"  }, |