MVVM 前端框架 Vue.js 学习报告

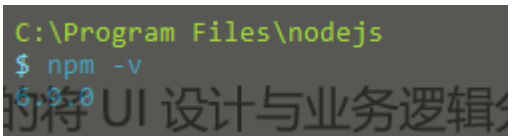
16302010050 马兵

MVVM 模式即 Model、View、ViewModel 是一种基于 MVC 的设计，实现了数据双向绑定，保证了数据一致性，就能在 Model 和 ViewModel 保持不变的情况下，很方便的将 UI 设计与业务逻辑分离，从而大大的减少繁琐的 DOM 操作。而 vue.js 是应用 MVVM 模式的移动 web 框架中的佼佼者，是一个非常优秀、高效的轻量级框架，本报告正是是关于 Vue.js 的学习。

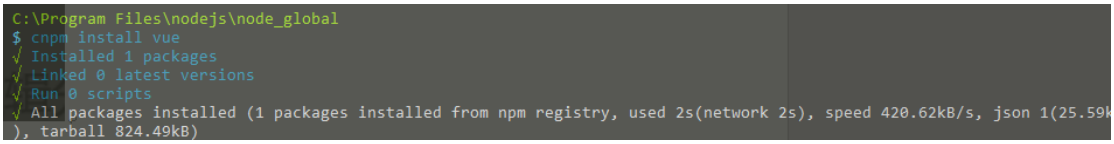
一、Vue.js 的安装

2.1 nmp 安装，在 node.js 官网下载对应操作系统的 LTS 安装包，安装后打开命令行，运行 npm -v 查看是否成功。并且把下载镜像源设置为更快的淘宝镜像：

```
npm config set registry=http://registry.npm.taobao.org
```



2.2 vue，vue-cli，webpack 安装均使用-g 参数安装在全局模块下



本地安装：将安装包放在 ./node_modules 下（运行 npm 命令时所在的目录），如果没有 node_modules 目录，会在当前执行 npm 命令的目录下生成 node_modules 目录。可以通过 require() 来引入本地安装的包。

全局安装：将安装包放在 ./node_global 下（运行 npm 命令时所在的目录），如果没有 node_global 目录，会在当前执行 npm 命令的目录下生成 node_modules

目录，可以直接在命令行里使用。

2.3 把 node_global 文件夹添加到系统环境变量 path 的尾端，这样就可以在任何地方使用 vue 指令了

二、vue 项目

1. 项目创建，

```
vue init webpack vue_demo
```

2. 项目依赖库添加，在项目根目录下，执行下列指令

```
npm install
```

安装完所需依赖后可以使用 type package.json 查看配置信息：

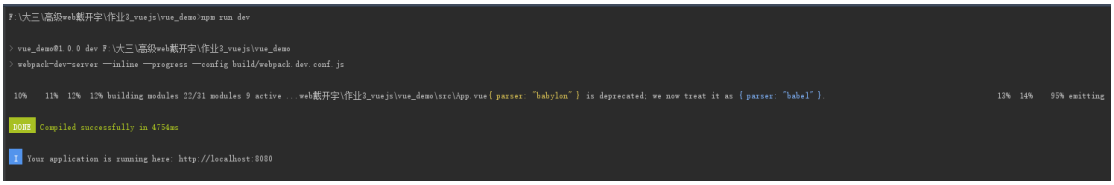
```
F:\大三\高级web戴开宇\作业3_vuejs\vue_demo  (vue_demo@1.0.0)
$ type package.json
{
  "name": "vue_demo",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "vue": "^2.6.10",
    "vue-router": "^3.0.6"
  },
  "devDependencies": {
    "css-loader": "^2.1.1",
    "file-loader": "^4.0.0",
    "style-loader": "^0.23.1",
    "url-loader": "^2.0.0",
    "vue-loader": "^15.7.0",
    "vue-template-compiler": "^2.6.10",
    "webpack": "^4.33.0",
    "webpack-dev-server": "^3.7.1"
  }
}
```

3.项目目录：使用 tree 指令查看项目目录结构

```
├─ dist
├─ node_modules
|    ├─ .bin
|    └─ ...
├─ src
|    ├─ assets
|    |    ├─ imgs
|    |    |    └─ logo.png
|    |    └─ styles
```

```
|   |       └─ base.css /基础样式表
|   ├─ components /公共组件 header
|   |       └─ header.vue
|   ├─ views   /前端界面
|   |   ├─ about.vue
|   |   └─ home.vue
|   ├─ App.vue /项目入口文件
|   ├─ main.js   /全局配置文件
|   └─ routes.js /路由配置文件
├─ index.html   /首页入口文件
├─ package.json
└─ webpack.config.js
```

4. 在 intellj 打开我们创建的项目，并安装 vue.js 插件即可进行开发
5．在命令行执行 npm run dev 即可运行项目,运行成功在浏览器可通过对应端口和路由访问。



三、 简单 demo 实现

实现一个 demo，体现 MVVM 模式的特点 v-model 和 view 的绑定关系，这里没有修改路由建立新界面，直接在 index.html 上修改做的 demo

1. 修改 html 界面，添加了两个按钮并绑定了事件，用于操作 msg 的跑动和停止，再添加一个 form 用于修改 msg，这里可以看到 vue 相关的@click，@submit，v-model。前两个和 js 绑定函数差不多，后一个 v-model 则是把 view 和·viewmodel 绑定的关键字。

```
<div id="test">
  <input type="button" value="run" id="btn1" @click="lang">
  <input type="button" value="stop" id="btn2" @click="stop">
  <form @submit.prevent="change">
    <input type="text" name="text1" v-model="text">
    <input type="submit">
  </form>
  <br>
  <h4>{{msg}}</h4>
</div>
```

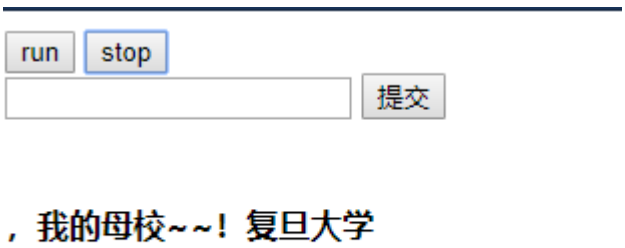
2. Js 相关代码

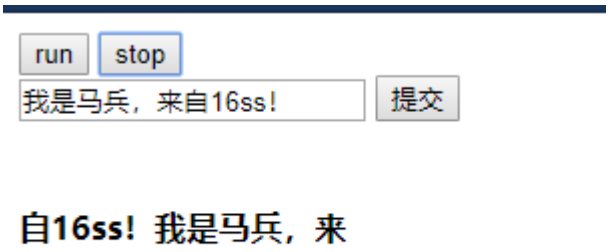
这里声名一个 vue，在里面用 el 与我们的 html 的 div test 绑定，数据绑定是由 html 中的{{}}绑定的 msg 默认为 “复旦大学，我的母校·~~!” 以及刚刚说明的 v-model 绑定的 text；

接下去是函数绑定 lang 和 stop，使得 msg 跑动和停止，change 把 msg 内容修改成我们自行输入的文字。

```
<script>
  let test = new Vue({
      el: "#test",
      data: {
        msg: "复旦大学，我的母校~~！",
        intervalId: null,
        text: ""
      },
      methods: {
        lang() {
          if (this.intervalId !== null) return;
          this.intervalId = setInterval(() => {
            let start = this.msg.substring(0, 1);
            let end = this.msg.substring(1);
            this.msg = end + start;
          }, 400)
        },
        stop(){
          clearInterval(this.intervalId);
          this.intervalId = null;
        },
        change(e){
            console.log(e);
          this.msg = this.text;
        }
      }
    }
  );
```

3. 运行结果：





四、 总结

Vue.js 的核心思想是数据驱动和组件化。组件化让 vue.js 开发非常灵活，在项目根目录下有一个 components 文件夹存储 vue 文件即各个组件，可以把公共的头部尾部提取出来做成组件，把单个界面的不同内容也可以单独做成一个插件，最后再在 router 里面配置路由并选取所需要的组件进行展示即可得到我们想要的界面，大大减少了冗余的代码，提高了代码复用性和灵活性。而数据驱动关键在 view 和 view-model 的绑定，封装了数据个 dom 对象操作的映射，当数据发生变化的时候，用户界面自动发生相应的变化，不需要我们手动的去修改 dom，使得界面开发变得更简单，比较适合入门学习 web 前端框架。

附上 github 地址：https://github.com/GeassAthena/vue_demo