链滴

vue-cli4.0+springboot 项目 12h 开发记录

作者：ChenforCode
原文链接：https://ld246.com/article/1583124266260
来源网站：链滴

<img src="https://ld246.com/images/img-loading.svg" alt="" data-src="https://b3logfile
com/bing/20191219.jpg?imageView2/1/w/960/h/540/interlace/1/q/100">
一、最近这一段时间，一直有两个兄弟在和我一起学习，是不是的会督促一下，问一下今天学了
么。于是我就想到一个 idea，为啥不做一个 todolist 去记录下每天要做的事情，完成了今天才算过
，大家也可以公开透明看到，互相督促。于是就开始着手做。
二、环境准备，首先本地开发环境为 vue4.0 JDK1.8 maven3.6.2。服务器上的环境为 JDK1.8 M
SQL8.0 前端开发用的是 vue，后端开发用 SpringBoot
三、前端开发：

vue create project_name 创建项目
vue ui 启动图形化界面 安装相关依赖
app 组件中将内容改成如下，即根据路由显示页面 ```

创建 router.js，里边写入需要导航的页面，我们目前只有一个 home 页面
在 main.js 里边导入相关依赖，使用 router，主要的依赖如下 ```
"dependencies": {
"axios": "^0.19.2",
"core-js": "^3.6.4",
"qs": "^6.9.1",
"stylus": "^0.54.7",
"stylus-loader": "^3.0.2",
"vue": "^2.6.11",
"vue-axios": "^2.1.5",
"vue-router": "^3.1.6"
}
<pre><code class="highlight-chroma"><span class="highlight
cl">
</code></pre>

然后创建 pages 目录，正式开始代码编写：里边的主要内容是一个标题，一个输入框（输入今
要做的事情），一个按钮（提交事情），一个 ul 列表（展示 list），每个 list 后又一个按钮（完成该
情）。
后端交互逻辑：页面初始化发送请求得到所有的 list 并在页面展示；点击提交事情按钮，将该时
信息发送至后台处理，并重新执行初始化函数；点击完成按钮，将该事件 id 发送至后台处理，并重
执行初始化函数。

四、后端开发

编写 SQL 文件，通过 navicat 远程连接服务器数据库并执行 SQL 文件完成数据库的创建。
在 IDEA 中创建 SpringBoot 项目，新建 application.yml 文件代替 application.properties 文
，写入相关配置，如项目路径，数据源，启动端口等等。
按照顺序编写 entity，repository，service 和 controller 层。并伴随着统一返回结果（由于时
问题，没有做统一异常处理），对 service 层和 repository 进行单元测试
创建 config.Cors.java 用来解决跨域问题（比较习惯用后端解决跨域），该类的核心在于，设置
个 fliter，允许你指定的某个或者某些域可以访问，例如允许你本地访问就是加入 http://localhost:po
t，以及你前端项目部署之后的域，一般只要设置这两个就可以了。注意这个 origin ，对任何一个字
都敏感，例如我写成了 http://localhost:port/，仅仅是多了一个 “/ “，他就告诉我这个请求是跨域
。这一点要注意。

以上是第一个难点，第二个难点是自己对后端接受参数了解好不够深入。
@PathVariable，这个比较简单，请求路径写成 xxx/id，这样就能把 id 注入
@RequestParam，这个接受的 application/x-www-from-urlencoded 自动根据传过来的数据根据名字自动注入，一般前端默认的也都是这个类型
@RequestBody，这个接受的是一个 JSON 字符串，不是 JSON 对象，注意前端传过来的类型是 application/json，但是前端不能直接传入{"key":"val"}的对象，必须通过 JSON Stringify()转换成 JSON 字符串，然后后端会根据名字注入相应的参数，这个一般用于注入一个类</li

不加注解，接受 application/x-www-from-urlencoded。会自动根据名进行参数注入，注意如果是一个类的话，里边的字段名和参数名一定要保持一致。

<p>五、前端向后端传递的参数问题</p>

传递 application/x-www-from-urlencoded 类型时候，要用 qs 库将 JS N 对象变成表单形式，注意不要将该对象再封装一层，如 data = {"key":"val"}，然后再参数中 stringi y({"data": data})。这样写是错的，自己调试了好长时间。。。。
<pre><code class="highlight-chroma">this.$qs.stringify({
 conte t: this.inputValue,
 isFini h: false,
 date: ate.year + '-' + date.month + '-' + date.date
 })
</code></pre>

传递表单数据，上述方式，对应后端接受的就是一个没有注解的一个类。。如果只传递一个参数后端就是一个加上 @RequestParam 的参数（该注解应该不加也可以，待验证）。

<p>六、后端部署</p>

以上问题解决之后，用 mvn package 打包，然后进入 target 文件夹，将生成的 jar 包利用 scp 命令传输到服务器，利用守护线程的启动方式将 jar 包启动。例如：nohup Java -jar test.jar >out ut 2>&1 &

<p>七、前端测试真实后端接口</p>

后端上线之后，前端的请求路径要做相应的改动，从 localhot 要改成相应的后端路径如 http:// 名或 ip:port/api/xxxxxx
由于后端已经解决的跨域问题（前端的本地和线上都解决了），这个时候前端的数据应该仍然是常的。

<p>八、前端部署前的相关配置</p>

刚开始我没有进行这一步的配置，直接打包生成 dist 文件，然后打开之后发现是白屏，猜测应该资源访问不到。然后查询很多资料，基本上都是让找到 config/index.js 或者是 vue.config.js 进行 as ets 目录的路径配置，因此选择第二种。
在根路径下（注意不是 src 目录，是项目根目录）建立 vue.config.js 文件，加入如下代码。其解决了白屏问题的应该是第三个配置，指定 assertsDir，后边的这个路径是相对于 vue.config.js 的径。项目路径如下图，因此写法如下。

<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight cl">module.exports = {

```
</span></span><span class="highlight-line"><span class="highlight-cl">    // 部署应用时基本 URL
</span></span><span class="highlight-line"><span class="highlight-cl">    publicPath: process.env.NODE_ENV === 'production' ? './' : '/',
</span></span><span class="highlight-line"><span class="highlight-cl">    // build时构建件的目录 构建时传入 --no-clean 可关闭该行为
</span></span><span class="highlight-line"><span class="highlight-cl">    outputDir: 'dist',
</span></span><span class="highlight-line"><span class="highlight-cl">    // build时放置成的静态资源 (js、css、img、fonts) 的 (相对于 outputDir 的) 目录
</span></span><span class="highlight-line"><span class="highlight-cl">    assetsDir: './src/ssets',
</span></span><span class="highlight-line"><span class="highlight-cl">    // 指定生成的 index.html 的输出路径 (相对于 outputDir)。也可以是一个绝对路径。
</span></span><span class="highlight-line"><span class="highlight-cl">}
</span></span></code></pre>
<p><img src="https://ld246.com/images/img-loading.svg" alt="" title="image.png" data-src="https://b3logfile.com/file/2020/03/image-572ab08e.png?imageView2/2/interlace/1/format/pg"></p>
<p>九、前端部署</p>
<ul>
<li>首先在项目根路径下通过 npm run build 进行项目打包，之后会在项目根路径下生成 dist 文件这个文件就是可以直接打开的，就能直接用了。</li>
<li>将 gitignore 中的 dist 路径删掉（vue 默认会在 Git 中忽略 dist 文件，因为这个文件是生成的并且很大），这样让 Git 重新管理 dist，之后推送到 gitee 仓库，然后在 gitee page 服务这里部署自己某个分支下的 dist 目录，然后点击部署，完成之后会在页面上出现一个路径，这个时候已经可以线上访问你的前端了（当然后端服务也是可以的）</li>
<li>注意由于后端的接口都是 http，因此在 gitee 上部署服务的时候，不要勾选强制使用 https（因勾选了，虽然你的项目可以用 https 访问，但是就不再允许你的项目通过 http 访问后台接口），如你的后台服务也是 https，那么这里可以勾选强制开启 https</li>
</ul>
<p>十、完毕，总结</p>
<ul>
<li>自己对一些经常用的点还是了解不太深，前端稍微一变动，后端就废掉，例如 controller 接受参。</li>
<li>这十二个小时，在第 6/7 个小时的时候已经完成开发。剩下近一半的时候都是在前后端联调和前部署相关的配置。这一点自己还是有点弱。</li>
<li>这次开发算是给以后踩了很多了雷，自己还是很满足的，毕竟以后的 vue+springboot 的开发上就有参考而稍微容易些！！！</li>
<li>继续努力，加油。</li>
</ul>
```