**Jeecg-Boot(v2.4.6） 单体升级为微服务**

**(本地开发用)**

1. **参考文档**

|  |  |
| --- | --- |
| 官方文档 | <http://doc.jeecg.com/2043906> |
| JeecgBoot项目 | [https://github.com/jeecgboot/jeecg-boot.git v2.4.6](https://github.com/jeecgboot/jeecg-boot.git%20v2.4.6)  <https://gitee.com/jeecg/jeecg-boot.git> v2.4.6 |

1. **部署说明(windows)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **组件** | **安装位置** | **启动方式** | **占用端口** | **URL** |
| Redis | Docker | Docker | 6379,16379 |  |
| Mysql | Docker | Docker | 3306,13306 |  |
| rabbitmq | Docker | Docker | 5672,15672 |  |
| ant-design-vue-jeecg | 宿主机 | Yarn | 3000 | <http://localhost:3000> admin/123456 |
| Nacos | 宿主机 | 使用IDEA启动 | 8848,9848 | <http://localhost:8848/nacos> nacos/nacos |
| JeecgSystemCloudApplication | 宿主机 | 使用IDEA启动 | 7001 | <http://localhost:7001/doc.html> |
| JeecgDemoCloudApplication | 宿主机 | 使用IDEA启动 | 7101 | http://localhost:7101/doc.html |
| JeecgMonitorApplication | 宿主机 | 使用IDEA启动 | 9111 | <http://localhost:9111/> admin/admin |
| JeecgGatewayApplication | 宿主机 | 使用IDEA启动 | 9999 |  |
| JeecgSentinelDashboardApplication | 宿主机 | 使用IDEA启动 | 8087 | <http://localhost:8087> sentinel/sentinel |

1. **预先下载**

|  |
| --- |
| git clone <https://github.com/jeecgboot/jeecg-boot>.git  docker pull node:16.8.0  docker pull redis:6.2.1  docker pull mysql:5.7.31  docker pull rabbitmq:3.7.7-management |

1. **配置host**

使用*管理员*权限 在 cmd中 运行 notepad C:\Windows\System32\drivers\etc\HOSTS

添加如下内容

|  |
| --- |
| 127.0.0.1 jeecg-boot-redis  127.0.0.1 jeecg-boot-mysql  127.0.0.1 jeecg-boot-nacos  127.0.0.1 jeecg-boot-gateway  127.0.0.1 jeecg-boot-system  127.0.0.1 jeecg-boot-xxljob  127.0.0.1 jeecg-boot-rabbitmq |

1. **安装 Redis**
2. 安装 Redis

|  |
| --- |
| docker rm -f jeecg-boot-redis  docker run -d --name jeecg-boot-redis --privileged=true --restart always -p 6379:6379 redis:6.2.1 |

1. **安装Rabbitmq**
2. 安装Rabbitmq

|  |
| --- |
| docker rm -f jeecg-boot-rabbitmq  docker run -d --restart always --hostname jeecg-boot-rabbitmq --name jeecg-boot-rabbitmq -e RABBITMQ\_DEFAULT\_USER=guest -e RABBITMQ\_DEFAULT\_PASS=guest -p 15672:15672 -p 5672:5672 rabbitmq:3.7.7-management |

1. 访问rabbitmq的管理端

|  |
| --- |
| http://localhost:15672/ |

1. 安装延时队列插件

* 参考:

<https://www.cnblogs.com/geekdc/p/13549613.html>

* 插件下载地址:

<https://github.com/rabbitmq/rabbitmq-delayed-message-exchange/releases/download/v3.8.0/rabbitmq_delayed_message_exchange-3.8.0.ez>

* 命令:

|  |
| --- |
| # 下载插件  wget <https://github.com/rabbitmq/rabbitmq-delayed-message-exchange/releases/download/v3.8.0/rabbitmq_delayed_message_exchange-3.8.0.ez>  #拷贝插件到rabbitmq容器  docker cp rabbitmq\_delayed\_message\_exchange-3.8.0.ez jeecg-boot-rabbitmq:/plugins  #进入容器  docker exec -it jeecg-boot-rabbitmq bash  #进入容器并启用插件  rabbitmq-plugins enable rabbitmq\_delayed\_message\_exchange  #查看  rabbitmq-plugins list  #重新启动容器  docker restart jeecg-boot-rabbitmq |

1. **安装 Mysql**

|  |
| --- |
| docker rm -f jeecg-boot-mysql  docker run --name jeecg-boot-mysql -d --restart always -p 3306:3306 \  -e MYSQL\_ROOT\_HOST='%' \  -e MYSQL\_ROOT\_PASSWORD=root \  mysql:5.7.31 \  --character-set-server=utf8mb4 `# 设置字符编码` \  --collation-server=utf8mb4\_unicode\_ci `# 设置字符编码` \  --lower\_case\_table\_names=1 `# 表名不区分大小写` |

1. **将 JEECG-Boot 的数据库导入**
2. 数据的Sql 文件

|  |
| --- |
| https://github.com/jeecgboot/jeecg-boot/blob/v2.4.6/jeecg-boot/db/jeecgboot-mysql-5.7.sql  https://github.com/jeecgboot/jeecg-boot/raw/v2.4.6/jeecg-boot/db/jeecgboot-mysql-5.7.sql |

1. Nacos的配置Sql 文件

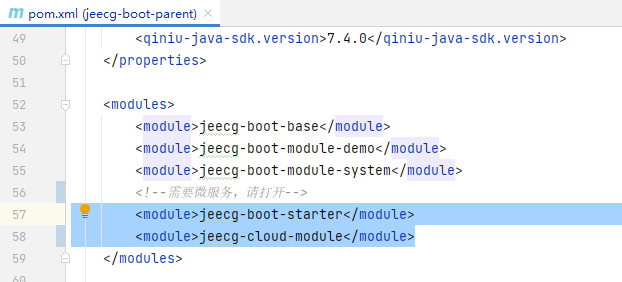
|  |
| --- |
| https://github.com/jeecgboot/jeecg-boot/blob/v2.4.6/jeecg-boot/db/tables\_nacos.sql  https://github.com/jeecgboot/jeecg-boot/raw/v2.4.6/jeecg-boot/db/tables\_nacos.sql |

1. xxl\_job的配置Sql 文件

|  |
| --- |
| https://github.com/jeecgboot/jeecg-boot/blob/v2.4.6/jeecg-boot/db/tables\_xxl\_job.sql https://raw.githubusercontent.com/jeecgboot/jeecg-boot/v2.4.6/jeecg-boot/db/tables\_xxl\_job.sql |

1. **基础配置**
2. 在parent的pom中添加cloud组件

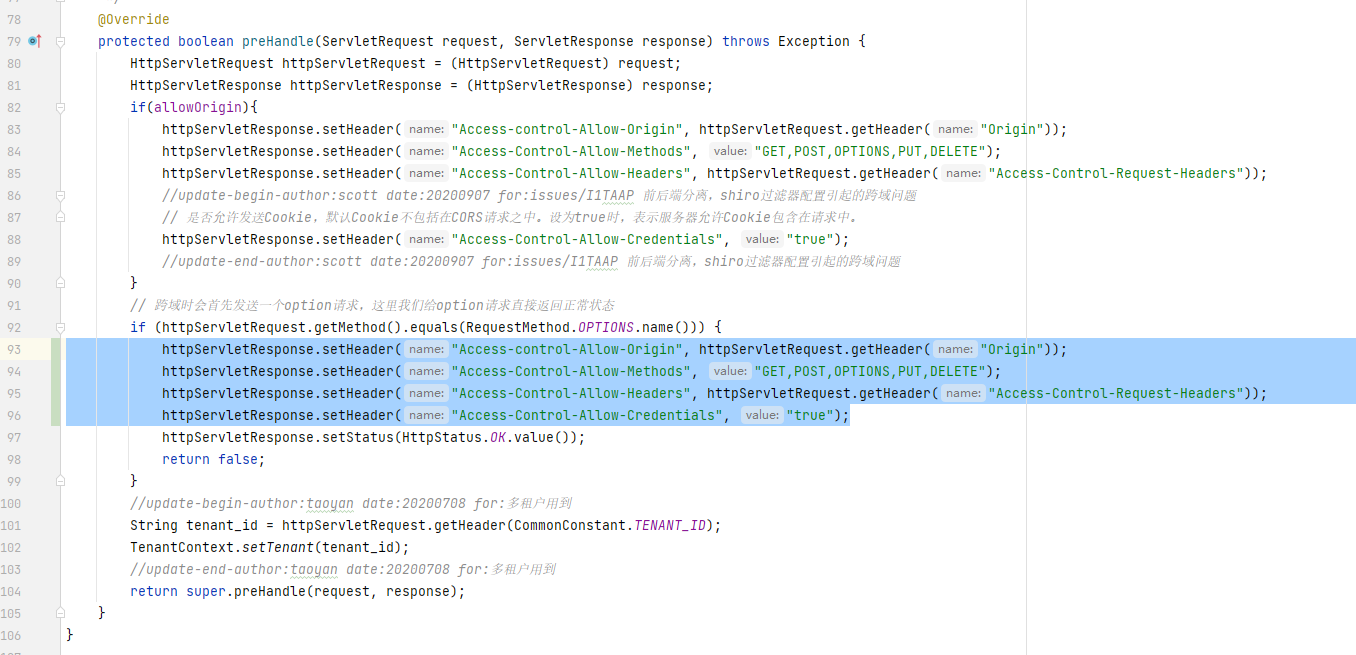
打开 jeecg-boot/jeecg-boot/pom.xml 中的注释



1. 接收到OPTIONS时允许跨域

在 jeecg-boot\jeecg-boot-base\jeecg-boot-base-core\src\main\java\org\jeecg\config\shiro\filters\JwtFilter.java 中添加代码

|  |
| --- |
| httpServletResponse.setHeader("Access-control-Allow-Origin", httpServletRequest.getHeader("Origin")); httpServletResponse.setHeader("Access-Control-Allow-Methods", "GET,POST,OPTIONS,PUT,DELETE"); httpServletResponse.setHeader("Access-Control-Allow-Headers", httpServletRequest.getHeader("Access-Control-Request-Headers")); httpServletResponse.setHeader("Access-Control-Allow-Credentials", "true"); |



1. 设置全局跨域

在jeecg-boot\jeecg-boot-base\jeecg-boot-base-core\src\main\java\org\jeecg\config\WebMvcConfiguration.java中添加方法

|  |
| --- |
| @Override public void addCorsMappings(CorsRegistry registry) {  WebMvcConfigurer.super.addCorsMappings(registry);  registry.addMapping("/\*\*")  .allowedHeaders("\*")  .allowedMethods("\*")  .allowedOrigins("\*"); } |



1. 清除开发环境缓存

* Reload jeecg-boot/pom.xml
* 在 IDEA 中运行 File->Invalidate Caches

1. 安装到Maven仓库

|  |
| --- |
| mvn install |

1. **启动SpringCloud 组件**
2. 运行Nacos

|  |
| --- |
| jeecg-boot\jeecg-cloud-module\jeecg-cloud-nacos\src\main\java\com\alibaba\nacos\JeecgNacosApplication.java |

访问Nacos <http://localhost:8848/nacos>

1. 运行monitor(SpringCloud Admin)

|  |
| --- |
| jeecg-boot\jeecg-cloud-module\jeecg-cloud-monitor\src\main\java\org\jeecg\monitor\JeecgMonitorApplication.java |

访问Monitor http://localhost:9111

1. 运行Gateway

|  |
| --- |
| jeecg-boot\jeecg-cloud-module\jeecg-cloud-monitor\src\main\java\org\jeecg\monitor\JeecgMonitorApplication.java |

1. 运行Sentinel

|  |
| --- |
| jeecg-boot\jeecg-cloud-module\jeecg-cloud-sentinel\src\main\java\com\alibaba\csp\sentinel\dashboard\JeecgSentinelDashboardApplication.java |

访问 <http://localhost:8087/> sentinel/sentinel

1. **启动jeecg-system服务**
2. 删除system项目的pom中demo的依赖

jeecg-boot\jeecg-boot\jeecg-boot-module-system\pom.xml



1. 修改打包配置(修改为true)

jeecg-boot\jeecg-boot-module-system\pom.xml



1. 启动system微服务 (jeecg-cloud-system-start)

|  |
| --- |
| jeecg-boot\jeecg-cloud-module\jeecg-cloud-system-start\src\main\java\org\jeecg\JeecgSystemCloudApplication.java |

1. 访问后台 api

|  |
| --- |
| http://localhost:7001/doc.html |

1. 验证nacos注册



1. **启动jeecg-demo服务**
2. 添加微服务启动依赖

jeecg-boot\jeecg-boot-module-demo\pom.xml

|  |
| --- |
| <!--引入微服务启动依赖 starter --> <dependency>  <groupId>org.jeecgframework.boot</groupId>  <artifactId>jeecg-boot-starter-cloud</artifactId> </dependency>  <dependency>  <groupId>org.jeecgframework.boot</groupId>  <artifactId>jeecg-boot-starter-job</artifactId>  </dependency> |



1. 加入打包插件

jeecg-boot\jeecg-boot\jeecg-boot-module-demo\pom.xml

|  |
| --- |
| <build>  <plugins>  <plugin>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-maven-plugin</artifactId>  </plugin>  </plugins> </build> |



记得在maven上Reload project

1. 添加application.yml

内容如下(注意： 不要有context-path):

|  |
| --- |
| server:  port: 7101  spring:  application:  name: jeecg-demo  jeecg :  # 签名密钥串(前后端要一致，正式发布请自行修改)  signatureSecret: dd05f1c54d63749eda95f9fa6d49v442a |

1. 运行

jeecg-boot\jeecg-boot-module-demo\src\main\java\org\jeecg\JeecgDemoCloudApplication.java

1. 访问后台 api

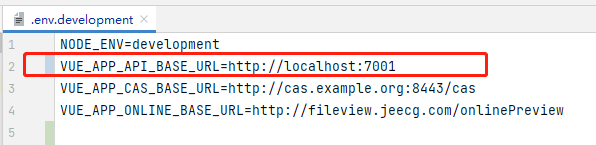
|  |
| --- |
| http://localhost:7101/ |

1. 验证nacos注册



1. **启动web**
2. 修改.env.development

ant-design-vue-jeecg\ .env.development



1. 启动web

|  |
| --- |
| cd ant-design-vue-jeecg  yarn install  yarn serve |

备选方案:也可在docker中运行

|  |
| --- |
| cd ant-design-vue-jeecg  docker run -it --rm -v $PWD:/jeecgboot -w /jeecgboot -p 3000:3000 node:16.8.0 bash -c "yarn install && yarn serve" |

1. 访问WEB

|  |
| --- |
| http://localhost:3000/ |

1. **对生成代码的修改**
2. JEECG-BOOT微服务建议用法：

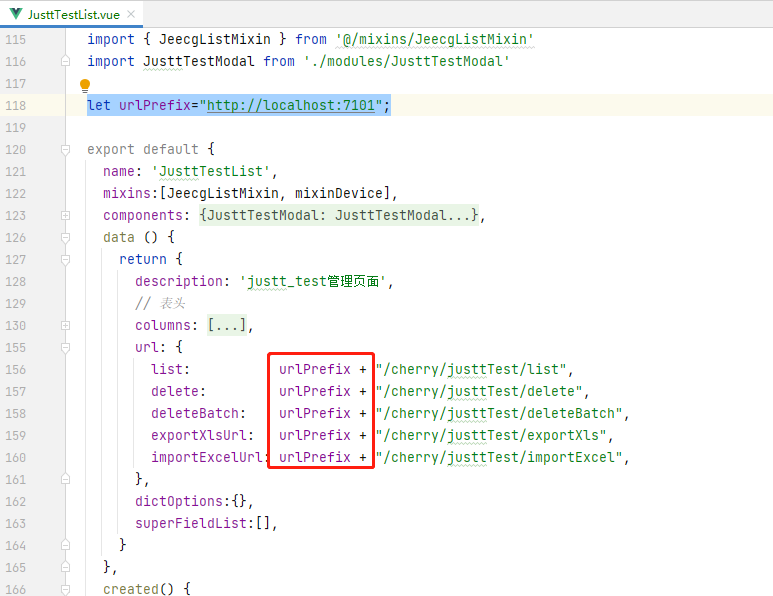
所有入口Controller都写在 jeecg-boot/jeecg-boot-module-system 项目中，被调用的组件写成各个微服务。

1. 生成的代码中前端的url修改

如果不接受建议，可以在各个文件中添加url前缀，从而直接调用各个微服务。

* 在List.vue中修改代码

|  |
| --- |
| let urlPrefix="http://localhost:7101";  ……  add: urlPrefix + "/cherry/justtTest/add", edit: urlPrefix + "/cherry/justtTest/edit", queryById: urlPrefix + "/cherry/justtTest/queryById" |



* 在 From.vue中修改代码

|  |
| --- |
| let urlPrefix="http://localhost:7101";  ……  list: urlPrefix + "/cherry/justtTest/list", delete: urlPrefix + "/cherry/justtTest/delete", deleteBatch: urlPrefix + "/cherry/justtTest/deleteBatch", exportXlsUrl: urlPrefix + "/cherry/justtTest/exportXls", importExcelUrl: urlPrefix + "/cherry/justtTest/importExcel", |

