

Spring REST Sample

功能介绍

以Spring mvc为核心的REST风格的Web服务基础样例。

技术栈

- Java SDK 1.7
- Maven 4
- Spring 4, Spring MVC 4
- Spring Security 4
- JPA 2.0, Hibernate 5
- MySQL
- Rest API, Json
- JUnit
- Swagger 2

开发工具推荐

- IDEA 2017: J2EE项目开发
- Postman: 测试开发API接口
- Markdown工具: 文档编写
- StartUML: UML设计工具

开发环境准备

1 Java SDK

建议使用 JDK 7 或 JDK 8，请在官方网站下载，并根据官方指引安装配置。

- [官方下载地址](#)
- [官方安装说明](#)

2 Tomcat

建议使用 8.0.x Core，请在官方网站下载。

- [官方下载地址](#)

3 Maven

- [官方地址](#)
- [官方安装说明](#)

编译

mvn成功执行之后，会在根目录的target下生成war包

```
$ mvn clean install -Pdev
```

Bash

直接部署到Tomcat

部署Tomcat

执行以下命令将war包部署到tomcat下

```
$ cp target/spring-rest-sample-1.0.0-SNAPSHOT.war /path/to/tomcat/webapps/sample.war
```

Bash


启动tomcat

```
$ cd /path/to/tomcat  
$ sudo ./bin/shutdown.sh # 关闭  
$ sudo ./bin/startup.sh # 启动
```

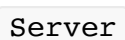
Bash


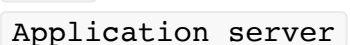
使用IDEA部署

**** 1. 创建Tomcat服务器 ****

依次点击菜单 Run->Edit Configurations ->  号 -> Tomcat Server -> Local

**** 2. 配置Tomcat服务器 ****

 项的配置如下：

- ，设服务器名称
- ，点击Configure，设置Tomcat服务器

- `Open browser` , 打开的URL地址默认是<http://localhost:8080/>。一般情况下, 在Deployment设置工程发布目录, 就会自动改变。
- `VM Option` , 如果出现乱码问题, 设置参数 `-Dfile.encoding=UTF8`

`Deployment` 项的配置如下:

- `+` 号 -> `Artifact...` -> 选择发布的工程 `spring-test-sample`
- `Application context` , 设置为 `/sample` 。

Mac启用80端口说明

Mac默认禁止1024以下的端口, 需要通过PF防火墙配置, 添加80端口。

创建防火墙配置文件

```
$ sudo vim /etc/pf.anchors/devel
```

Bash

添加80端口对8080端口的访问许可, 内容如下:

```
rdr pass on lo0 inet proto tcp from any to any port 80 -> 127.0.0.1 port 8080
```

修改配置文件

```
$ sudo vim /etc/pf.conf
```

Bash

修改内容如下

```
scrub-anchor "com.apple/*"  
nat-anchor "com.apple/*"  
rdr-anchor "com.apple/*"  
rdr-anchor "devel" #添加配置  
dummynet-anchor "com.apple/*"  
anchor "com.apple/*"  
load anchor "com.apple" from "/etc/pf.anchors/com.apple"  
load anchor "devel" from "/etc/pf.anchors/devel" #添加配置
```

立即生效。生效后, 无需配置Nginx。

```
$ sudo pfctl -ef /etc/pf.conf
```

Bash

如果希望长期有效, 可以按以下方式修改配置文件。由于苹果安全限制, 最新版本Sierra不支持以下修改。

```
$ sudo vim /System/Library/LaunchDaemons/com.apple.pfctl.plist
```

Bash

修改内容如下

```
<array>
  <string>pfctl</string>
  <!-- 下面就是修改的启动参数 -->
  <string>-ef</string>
  <string>/etc/pf.conf</string>
</array>
```

Markup

工程配置文件

所有配置文件都依据Maven一般规则放在resources目录下

app.properties

```
# 运行环境配置
spring.profiles.active=@spring.profiles.active@
```

Ini

database.properties

数据库配置文件。主要配置数据库连接，hibernate。

log4j.properties

log4j的日志配置文件。

编译指南

开发环境

考虑到方便性和安全性，项目默认配置为开发环境。可以直接使用IDEA进行开发、调试。

```
$ mvn clean install -Pdev
```

Bash

测试环境

项目没有配置测试环境的资源配置目录，所以需要复制一份开发环境的资源文件目录 `resources`，改

为 `resources-staging`

```
$ mvn clean install -Pstaging
```

Bash

生产环境

项目没有配置测试环境的资源配置目录，所以需要复制一份开发环境的资源文件目录 `resources`，改为 `resources-prod`

```
$ mvn clean install -Pprod
```

Bash

快速开发指南

本项目是为了方便快速建立一个Web工程。所以如果想基于此新建一个工程，可以参考如下步骤。

1. 下载工程

```
git clone git@github.com:lordking/spring-rest-sample.git
```

Bash

2. 用文本工具修改pom.xml

将下面两项修改成符合自己项目的配置。特别注意，请先不要用IDEA直接导入或打开工程。防止由于提前导入，导致工程名称和相关IDEA配置不是自己需要的。

```
<groupId>vip.maxhub.web</groupId>
<artifactId>spring-rest-sample</artifactId>
```

Markup

3. 用IDEA打开工程

IDEA打开工程时会自动提示是否是Maven、是否导入JPA配置文件等多个确认，根据提示点击确认。

打开后，IDEA会根据pom.xml配置自动导入依赖包。

4. 在IDEA下修改包名

在IDEA下的目录树中，右键选择包 `vip.maxhub.web.sample`，弹出右键菜单。点击 `Refactor` - `> Rename...` 修改成自己需要的名称。

编辑 `Constants.java` 文件，修改包名。一般情况下，上面操作成功后，下面的配置会自动修改。

```
public static final String BASE_PACKAGE = "vip.maxhub.web.sample";
```

5. 开发文档

- UML文档用StartUML编写。原始文件在doc_src目录。可直接阅读的格式是JPG图片，在doc目录。不输出PDF，是由于StartUML输出的PDF显示中文乱码。
- API文档用使用Postman导出的文件。`docs_src/spring-rest-sample.postman_collection.json`

6. 运行环境配置

6.1 在代码中使用 `@Profile` 注解配置

本项目只配置了一条，可以参考 `swaggerConfig.java` 。下面含义是只支持开发和测试环境。

```
@Profile(value = {"dev", "staging"})
```

6.2 在IDEA选择不同的运行环境

一般情况下，我们需要根据不同的运行环境，做不同的资源文件配置。所以在 `pom.xml` 有如下配置：

```
<profiles>

  <profile>
    <id>dev</id>
    <activation>
      <activeByDefault>true</activeByDefault>
    </activation>
    <properties>
      <log4j.level>DEBUG</log4j.level>
      <spring.profiles.active>dev</spring.profiles.active>
    </properties>
    <build>
      <resources>
        <resource>
          <directory>src/main/resources</directory>
          <filtering>true</filtering>
        </resource>
      </resources>
    </build>
  </profile>

  <profile>
```

```

<id>staging</id>
<properties>
  <log4j.level>DEBUG</log4j.level>
  <spring.profiles.active>dev</spring.profiles.active>
</properties>
<build>
  <resources>
    <resource>
      <directory>src/main/resources-staging</directory>
      <filtering>true</filtering>
    </resource>
  </resources>
</build>
</profile>

<profile>
  <id>prod</id>
  <properties>
    <log4j.level>INFO</log4j.level>
    <spring.profiles.active>dev</spring.profiles.active>
  </properties>
  <build>
    <resources>
      <resource>
        <directory>src/main/resources-prod</directory>
        <filtering>true</filtering>
      </resource>
    </resources>
  </build>
</profile>
</profiles>

```

6.3 在IDEA选择不同的运行环境

打开编辑栏最右侧的导航栏，一般被缩略成标签样式。其中有一个标签是 `Maven Projects`。打开后，就可以发现配置好的profiles，可以选择自己需要的profile。

Swagger编辑指南

项目内置Swagger，可以自动生成API文档。

不推荐Swagger作为API设计工具

1. Swagger与Postman的结合非常差。Postman虽然支持导入Swagger文件，但不完全兼容Swagger，尤其是解析不了body。

2. Swagger Editor的文档编辑功能也是非常差，有代码提示，但没有GUI界面。编辑起来，非常得费力。工作效率低下。
3. Swagger可以作为补充，在没有编写接口协议文档或接口协议文档与实际差异比较大的情况下，可以导出观看实际接口。

建议步骤如下:

1. 在项目中配置Swagger

编辑 `config/SwaggerConfig.java` 。注意，考虑安全性的原因，只允许开发环境、测试环境使用swagger。

2. 查看自动生成的API

在开发环境、测试环境情况下启动项目后，自动就可以查看到API文档。查看地址是:

```
http://127.0.0.1:8080/sample/v2/api-docs
```

3. 使用Swagger Editor编辑生成的文档

环境准备

- NodeJS 6.x
- NPM 3.x
- Chrome (不支持Safari)

安装Swagger Editor命令如下

```
$ npm install -g http-server
$ git clone https://github.com/swagger-api/swagger-editor.git
$ cd swagger-editor
$ npm install
$ npm run build
$ http-server .
```

Bash

用Chrome 打开地址。由于Tomcat占用了8080端口，本机启动时会自动改为8081端口。

```
http://127.0.0.1:8081/
```

点击 `File` -> `import URL` ->输入如下地址打开API文档。


```
http://127.0.0.1:8080/sample/v2/api-docs
```

打开后，即可编辑、导出。

4. 使用Swagger UI直接打开浏览

安装Swagger UI命令如下

```
$ npm install -g http-server
$ git clone https://github.com/swagger-api/swagger-ui.git
$ cd swagger-ui
$ npm install
$ npm run build
$ http-server dist
```

Bash

本项目已经编辑完成，所以安装完成后输入地址：

```
https://raw.githubusercontent.com/lordking/spring-rest-sample/master/doc_src/接口协议
```

如果不想安装，可以使用Swagger官方提供的演示网站打开：

```
http://petstore.swagger.io
```

用Chrome 打开地址。由于Tomcat占用了8080端口，本机启动时会自动改为8081端口。