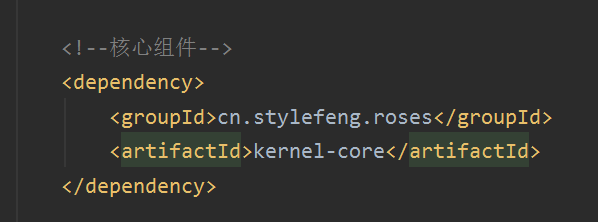
[2.1 了解Guns](https://www.stylefeng.cn/doc/guns/#/dev/?id=_21-%e4%ba%86%e8%a7%a3guns)

**用Guns开发常备如下几个工具:**

* mybatis-plus文档：<http://mp.baomidou.com/>
* beetl文档：<http://ibeetl.com/guide/#beetl>
* layui文档：<https://www.layui.com/doc/>
* Spring Boot文档：<https://docs.spring.io/spring-boot/docs/current/reference/html/>

[2.1.1 模块结构](https://www.stylefeng.cn/doc/guns/#/dev/?id=_211-%e6%a8%a1%e5%9d%97%e7%bb%93%e6%9e%84)

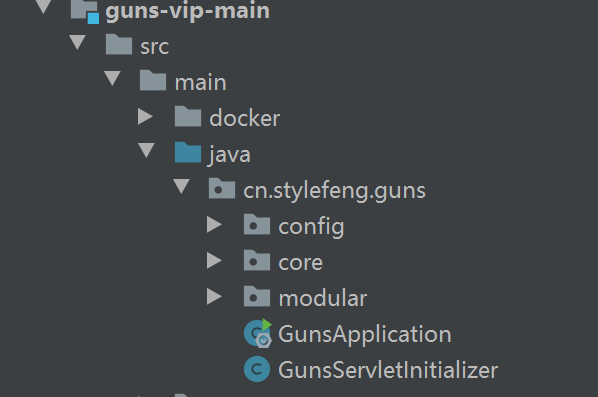
Guns最新版的Guns结构，以最全的高级企业版为例，如下： 

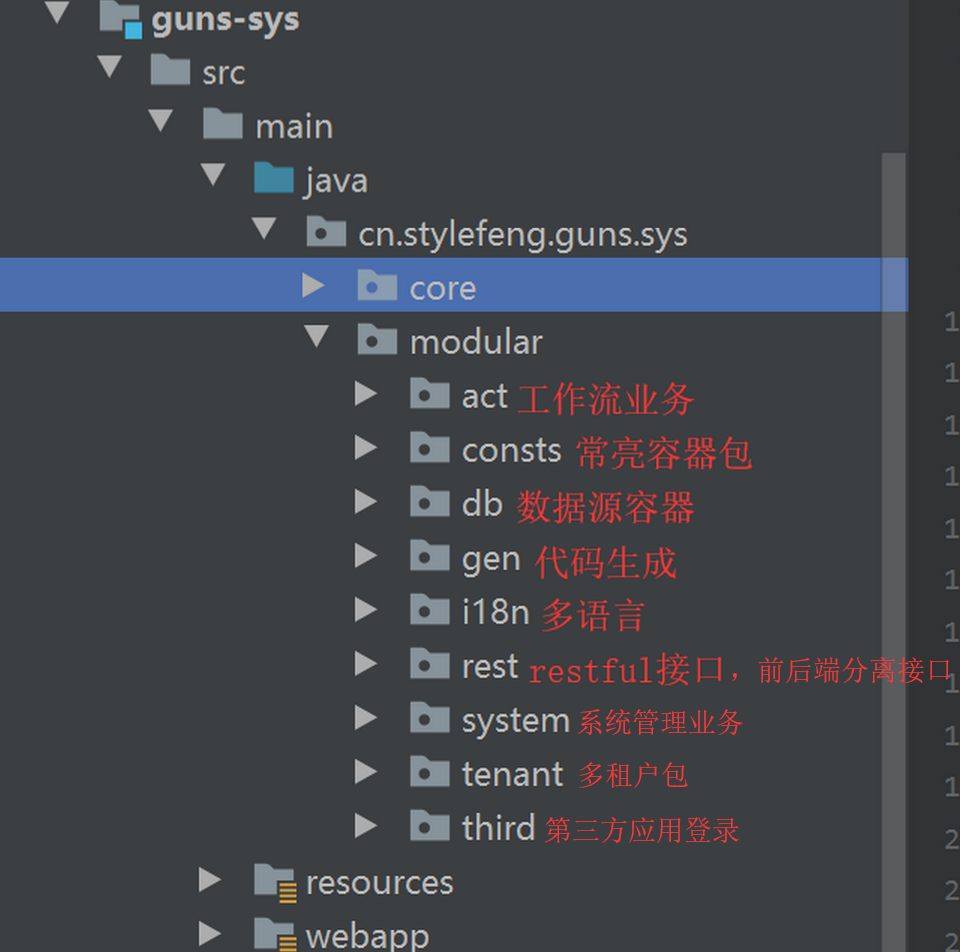
但是pom中还是依赖了作者开发的一个其他模块， 

该模块作者已经上传到maven的中央仓库中（[https://search.maven.org/search?q=cn.stylefeng）](https://search.maven.org/search?q=cn.stylefeng%EF%BC%89)

kernel-core模块为抽象出的核心（通用）模块，以供其他模块调用，此模块主要封装了一些通用的工具类，公共枚举，常量，配置等等，此模块的gitee地址是<https://gitee.com/stylefeng-Roses/roses-kernel>

[2.1.2 包结构说明](https://www.stylefeng.cn/doc/guns/#/dev/?id=_212-%e5%8c%85%e7%bb%93%e6%9e%84%e8%af%b4%e6%98%8e)

一般情况，根目录包结构分为三个，config，core和modular，例如guns-vip模块中的包， 

另外modular包下又可细分子包，每个自包代表一个业务含义，如下： 

[2.2 实战开发](https://www.stylefeng.cn/doc/guns/#/dev/?id=_22-%e5%ae%9e%e6%88%98%e5%bc%80%e5%8f%91)

**Guns开发三部曲 -> 1.建表 2.代码生成 3.添加菜单 4.适配业务代码**

旗舰版的代码生成需要购买企业版获取，经典版5.1包含免费的代码生成 新的代码生成请购买授权版获取，购买地址：<https://www.stylefeng.cn/>

下面以一个订单业务为例，实战演练如何用Guns编写简单的增删改查业务

[2.2.1 建表](https://www.stylefeng.cn/doc/guns/#/dev/?id=_221-%e5%bb%ba%e8%a1%a8)

新建订单表如下: 

DROP TABLE IF EXISTS `biz\_order`;

CREATE TABLE `biz\_order` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT '主键',

`goods\_name` varchar(255) DEFAULT NULL COMMENT '商品名称',

`place` varchar(255) DEFAULT NULL COMMENT '下单地点',

`create\_time` datetime DEFAULT NULL COMMENT '下单时间',

`user\_name` varchar(255) DEFAULT NULL COMMENT '下单用户名称',

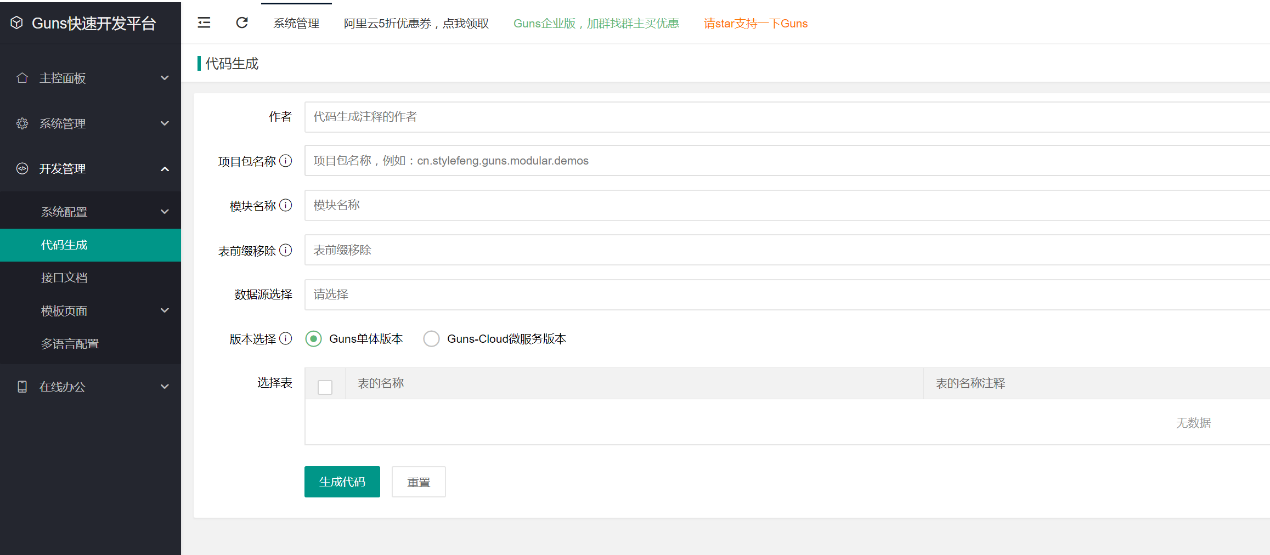
`user\_phone` varchar(255) DEFAULT NULL COMMENT '下单用户电话',

PRIMARY KEY (`id`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW\_FORMAT=DYNAMIC COMMENT='订单表';

SET FOREIGN\_KEY\_CHECKS = 1;

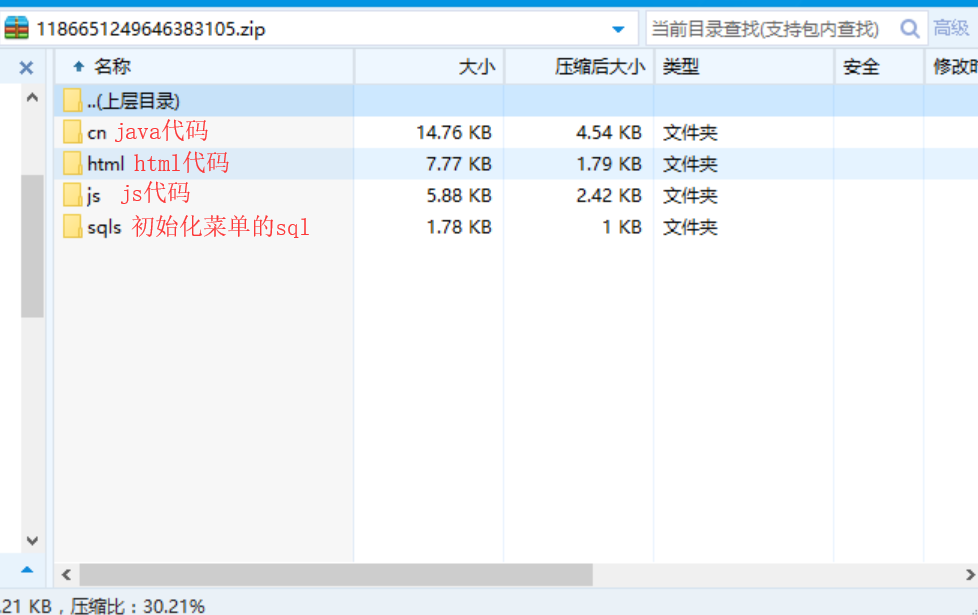
[2.2.2 代码生成](https://www.stylefeng.cn/doc/guns/#/dev/?id=_222-%e4%bb%a3%e7%a0%81%e7%94%9f%e6%88%90)

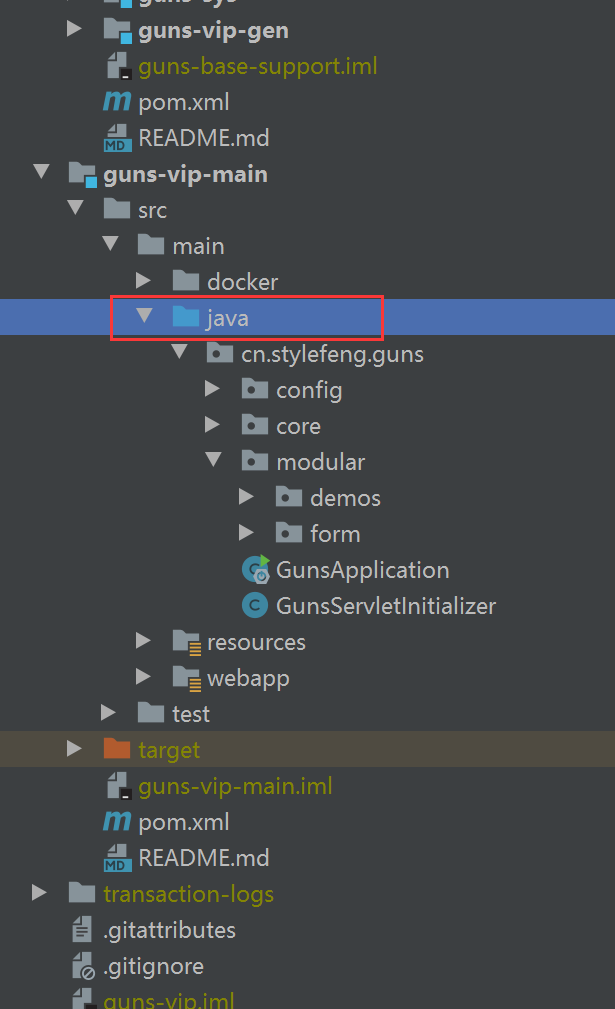
打开localhost，进入Guns的主页，找到代码生成菜单，如下 

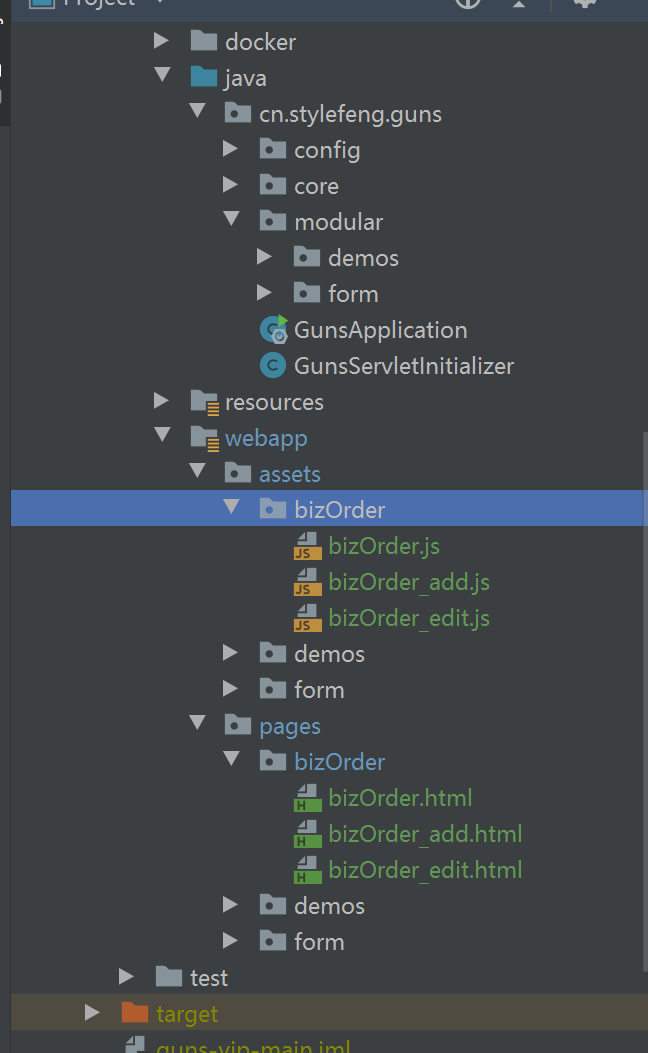
代码生成功能介绍:

1. **Guns单体版本-代码生成** 生成传统单体，前后端不分离的代码
2. **Guns-Cloud微服务版本代码生成** 修改或删除数据源信息

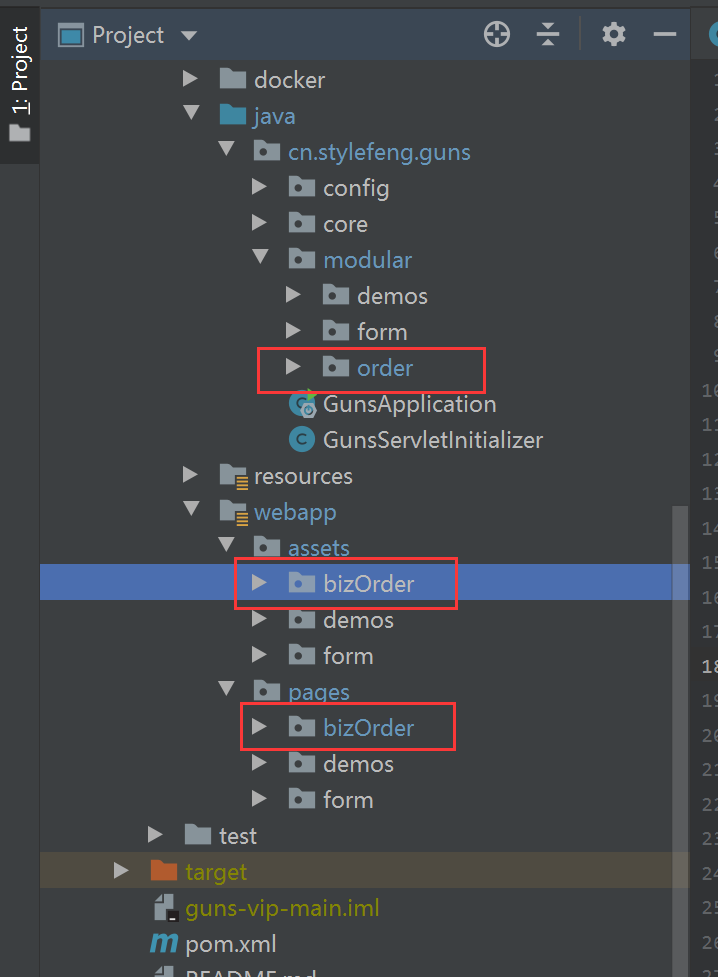
代码生成的过程： 打开代码生成页面 

点击成成后，我们得到一个下载的压缩包，如下： 

java代码，复制cn目录扔到java目录下： 

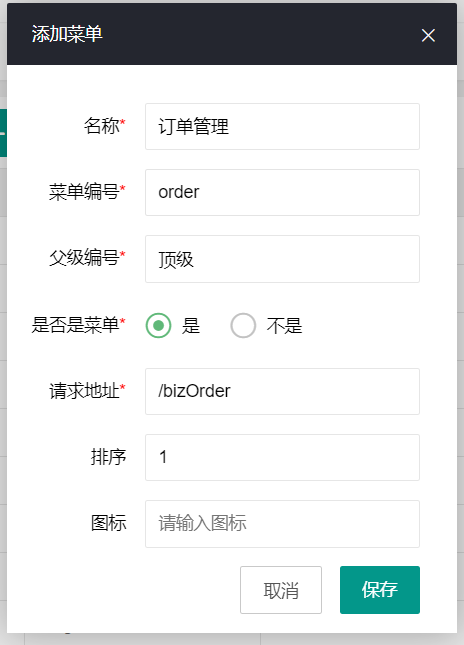
把html文件夹下的文件夹复制到pages/modular，把js文件夹下的文件夹复制到assets/modular 

在数据库中执行sqls文件夹中的sql文件 

至此我们添加了新生成的java代码，html，js到我们的项目中 

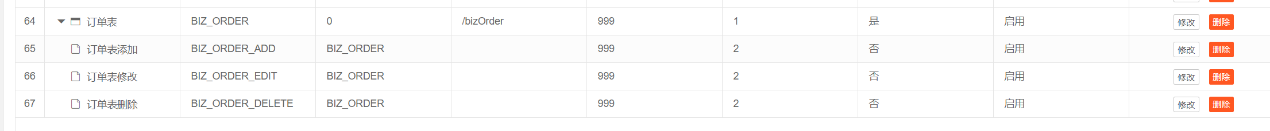
我们重新启动一下guns项目，进行后续步骤

[2.2.3 添加菜单与分配权限](https://www.stylefeng.cn/doc/guns/#/dev/?id=_223-%e6%b7%bb%e5%8a%a0%e8%8f%9c%e5%8d%95%e4%b8%8e%e5%88%86%e9%85%8d%e6%9d%83%e9%99%90)

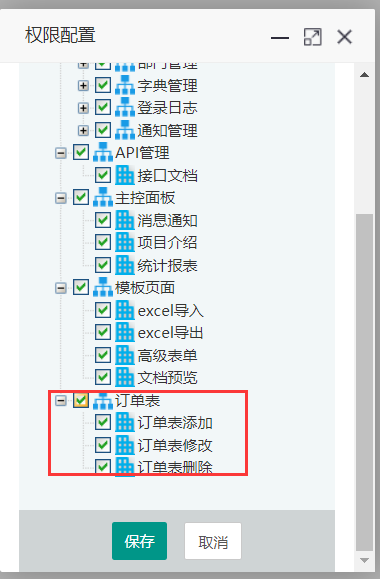
生成代码之后，需要为管理系统添加菜单，才可以让新增加的业务显示到页面上，添加菜单有两种方式: **第一种**为手动添加菜单，依次点击系统管理->菜单管理->点击添加，打开添加页面，如下 

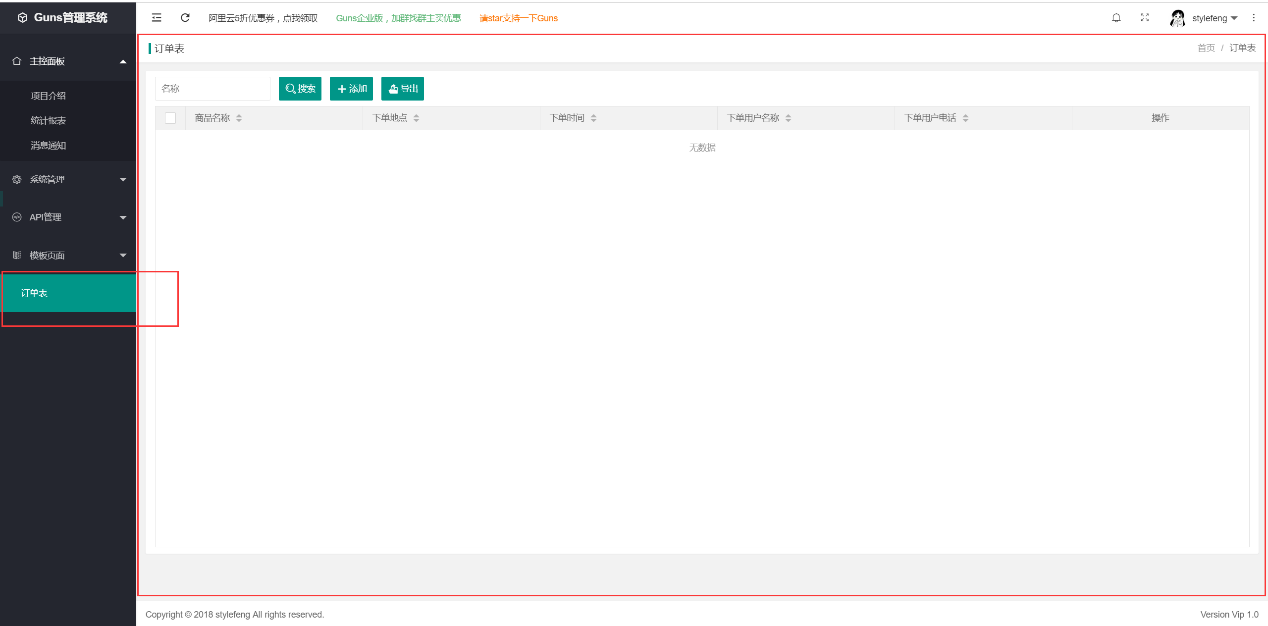
这里需要注意如下几点:

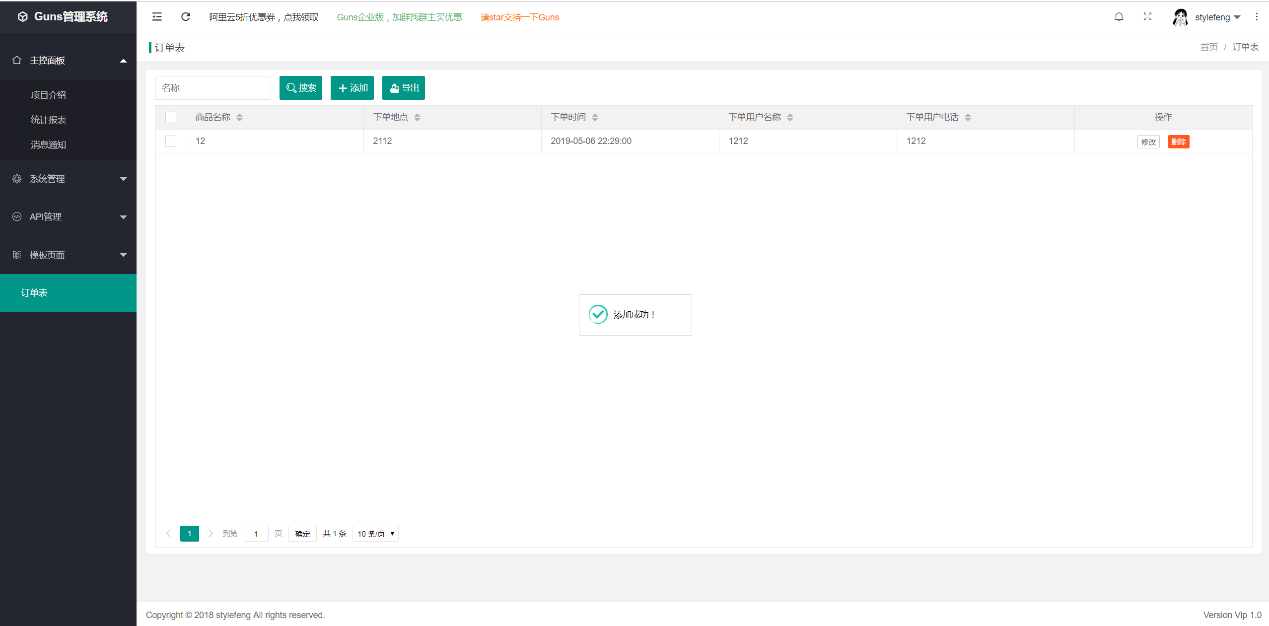
* 请求地址需要和Controller中的RequestMapping的值一致
* 排序为同层级菜单中显示菜单的顺序
* 父级编号的选择可以更改菜单插入的位置
* 因为菜单管理不单单是对管理系统中的菜单管理，也包含权限的管理，所以需要选择是否是菜单这个选项

**第二种**添加菜单的方式为直接执行代码生成中的sql脚本，上小节已指明，执行完成后可以看到，菜单管理页面中已经有了新添加的订单相关的菜单和资源，如下 

**在添加完菜单只有，还需要给角色分配相关的菜单权限，才可以把新增的业务显示到菜单上**

打开系统管理->角色管理，给当前的登录的超级管理员，增加刚才新增的权限，如下图 

配置完成刷新页面即可看到，即可看到新增加的菜单，如下图，若看不到请重新登录 

到这里，基本的增删改查功能就实现了，如下图  

[2.2.4 编写业务代码](https://www.stylefeng.cn/doc/guns/#/dev/?id=_224-%e7%bc%96%e5%86%99%e4%b8%9a%e5%8a%a1%e4%bb%a3%e7%a0%81)

由于Guns的代码生成器还不能实现100%的智能，所以生成之后还需要对生成的代码做一些完善，如果有除了增删改查以外的业务，还需要手动编写。例如，上面编写的添加订单和修改订单里，下单时间默认是text文本框，这里需要手动改为laydate样式的日期框

[2.3 权限控制与校验](https://www.stylefeng.cn/doc/guns/#/dev/?id=_23-%e6%9d%83%e9%99%90%e6%8e%a7%e5%88%b6%e4%b8%8e%e6%a0%a1%e9%aa%8c)

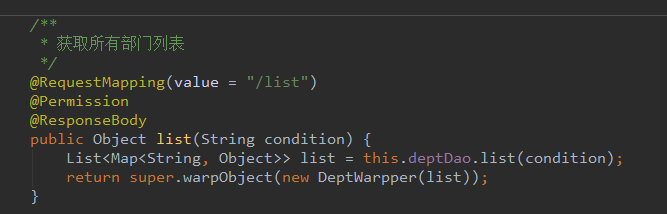
[2.3.1 用户，角色和资源](https://www.stylefeng.cn/doc/guns/#/dev/?id=_231-%e7%94%a8%e6%88%b7%ef%bc%8c%e8%a7%92%e8%89%b2%e5%92%8c%e8%b5%84%e6%ba%90)

用户、角色和资源（或者说权限），这三者的关系是用户对应角色，角色对应资源，菜单和所有的按钮都可以看做是资源(或权限)，把某一个角色赋予相应的资源，那么该角色就会有访问该资源的权限，否则，该角色访问这些被管控的资源就会被服务器返回403 没有权限，当角色绑定资源后还需要给用户赋予角色才可以让登录的用户访问相关服务器接口。

**一句话概括: 用户对应角色，角色对应资源**

[2.3.2 如何对资源进行权限控制](https://www.stylefeng.cn/doc/guns/#/dev/?id=_232-%e5%a6%82%e4%bd%95%e5%af%b9%e8%b5%84%e6%ba%90%e8%bf%9b%e8%a1%8c%e6%9d%83%e9%99%90%e6%8e%a7%e5%88%b6)

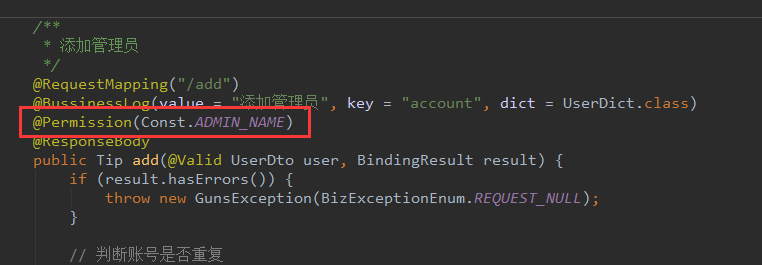
Guns系统中，通过在控制器上加@Permission注解进行权限校验，如下所示，该接口在被访问的时候，就会进行权限校验



通过我们查找用户对应的角色，并查找角色对应的资源，可以找到，当前用户(admin)有该资源的权限，如下



@Permission注解中可以带一个String数组类型的参数，如下，加上该参数，则接口被限制为只有某个或某些角色才可访问



权限的检查是通过AOP拦截@Permission注解完成的，当访问受权限控制的资源时，AOP对当前请求的servletPath和数据库中sys\_menu表的url字段进行匹配，如果当前用户所拥有的权限包含当前请求的servletPath，则访问这个接口成功

[2.3.3 前端页面对权限资源的显示](https://www.stylefeng.cn/doc/guns/#/dev/?id=_233-%e5%89%8d%e7%ab%af%e9%a1%b5%e9%9d%a2%e5%af%b9%e6%9d%83%e9%99%90%e8%b5%84%e6%ba%90%e7%9a%84%e6%98%be%e7%a4%ba)

在前端页面中，如果增删改查等按钮受权限控制，则我们需要对资源进行一个权限检查，如果有该资源的权限，才能让该按钮显示，通过beetl的shiro注册方法即可完成该项的检查

@if(shiro.hasPermission("/dept/update")){

<a class="layui-btn layui-btn-primary layui-btn-xs" lay-event="edit">修改</a>

@}

@if(shiro.hasPermission("/dept/delete")){

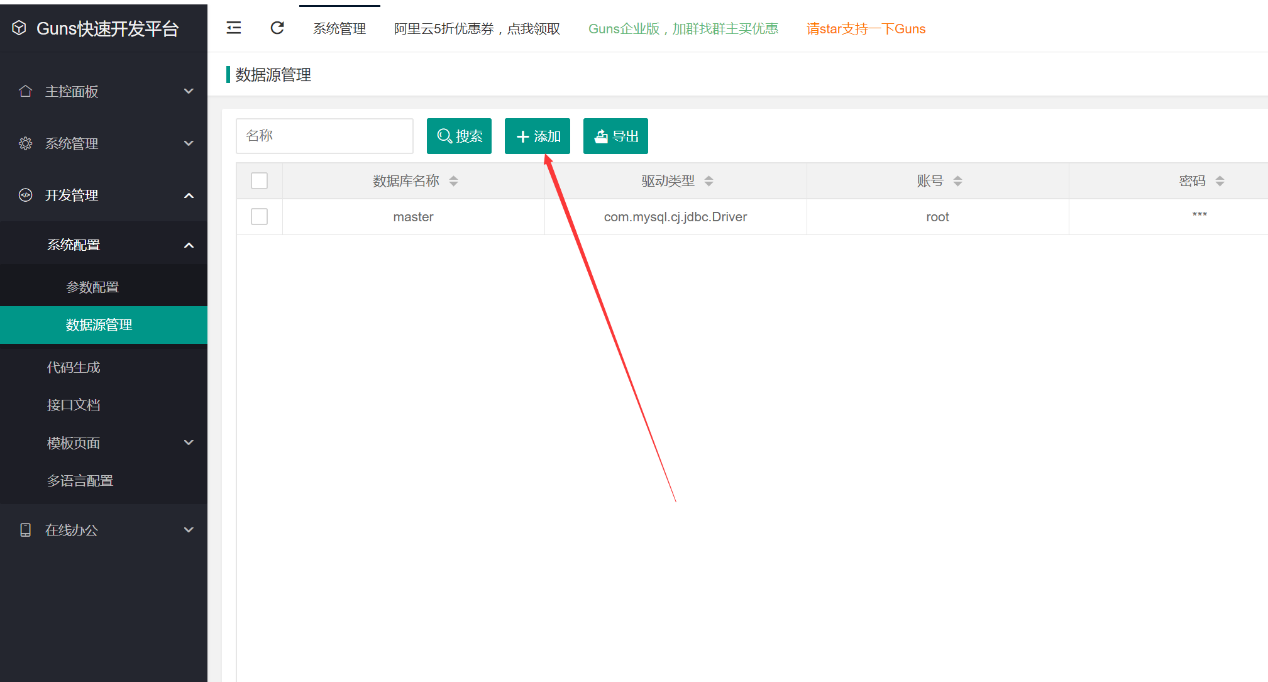
<a class="layui-btn layui-btn-danger layui-btn-xs" lay-event="delete">删除</a>

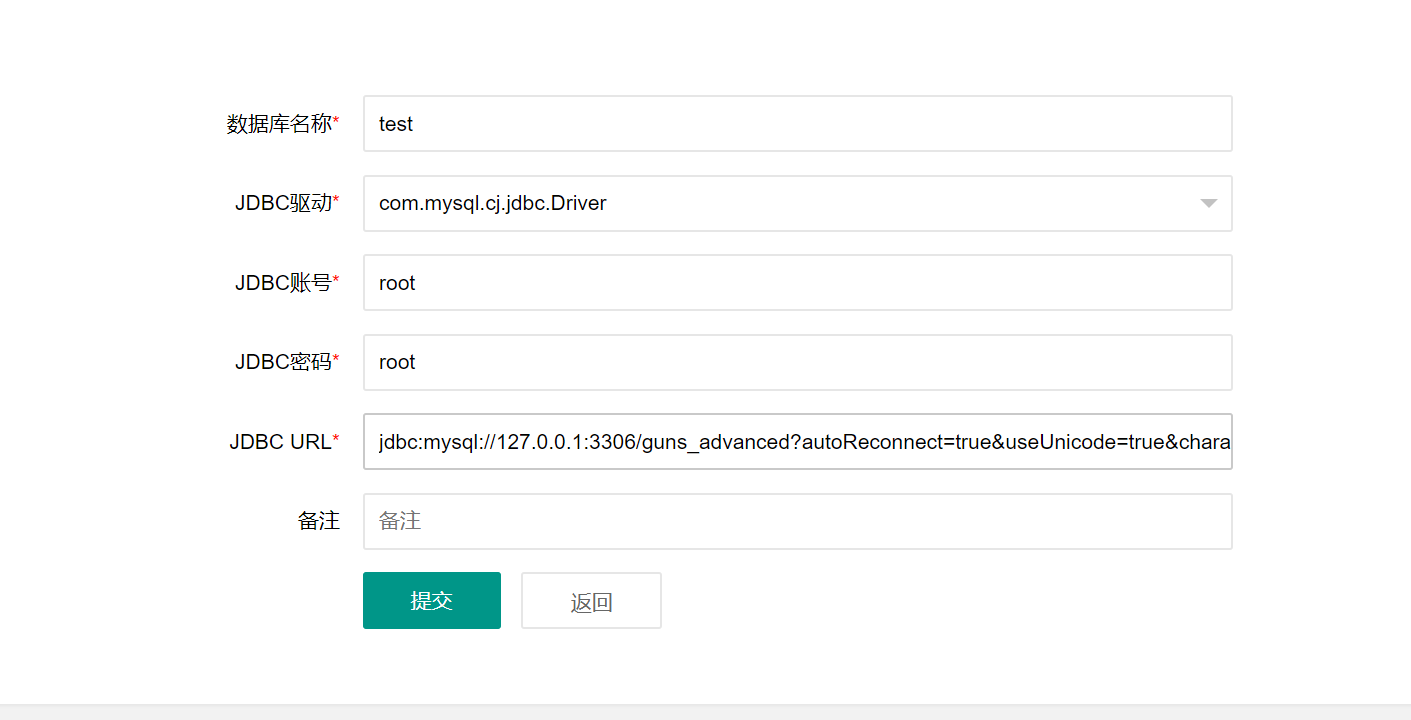
@}

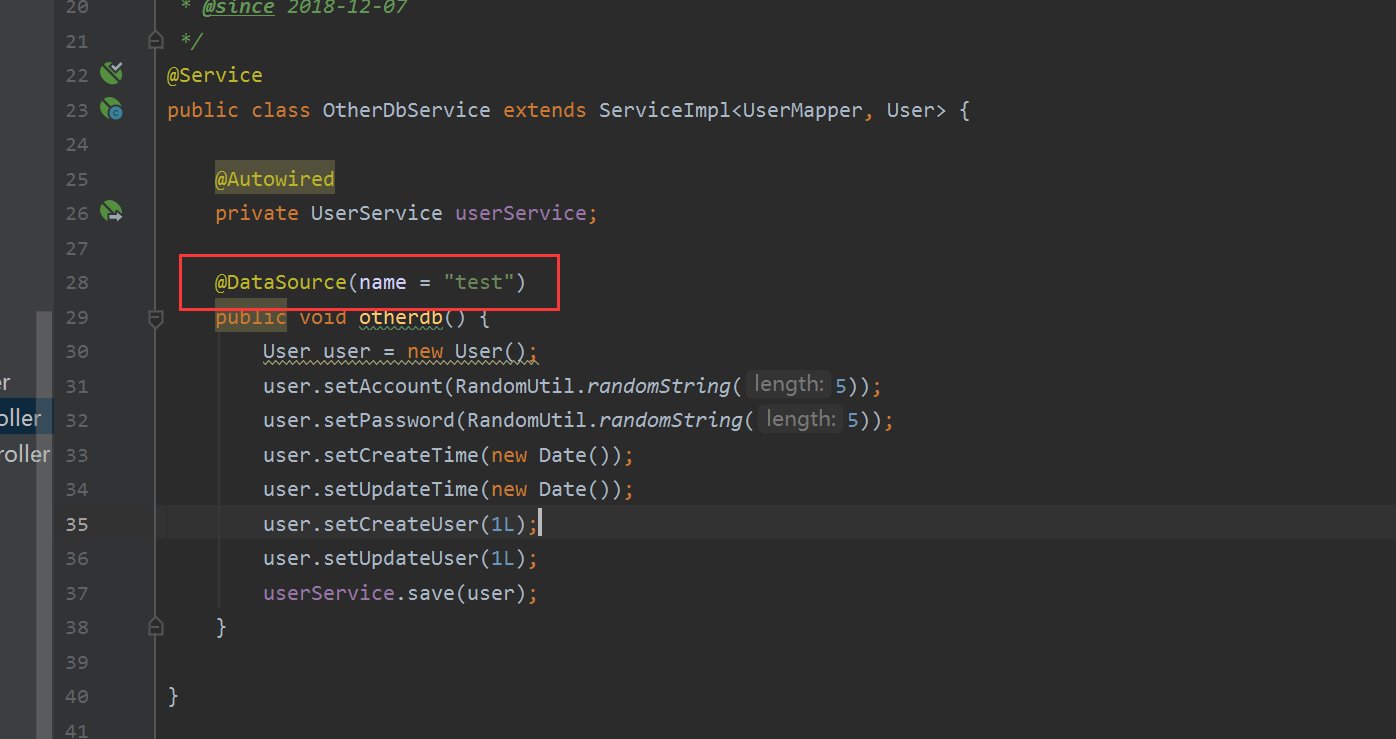
其中shiro.hasPermission()起到了权限检查的作用，如果有该资源对应的权限，则被检查的资源显示，若没有该资源的权限，则按钮不显示

[2.4 多数据源的使用](https://www.stylefeng.cn/doc/guns/#/dev/?id=_24-%e5%a4%9a%e6%95%b0%e6%8d%ae%e6%ba%90%e7%9a%84%e4%bd%bf%e7%94%a8)

新版的企业版使用多数据源非常简单，独创了多数据源容器的概念，即使是配置100个数据源也不在话下。

**第一步，在线新增数据源** 

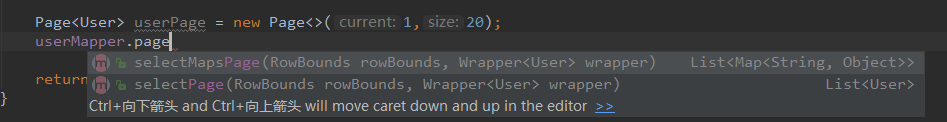
**第二步，配置数据源** 

**第三步，使用数据源** 只需在使用多数据源（非主数据源）上边，加上@DataSource(name = "test")注解即可使用多数据源，其中name的值就是所添加数据源的名称。 

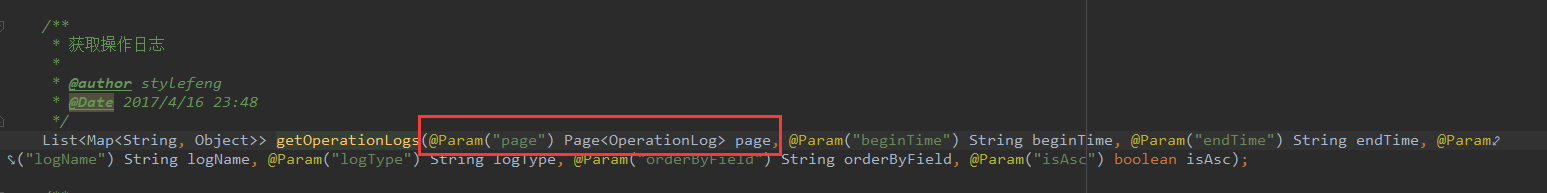
[2.5 如何分页](https://www.stylefeng.cn/doc/guns/#/dev/?id=_25-%e5%a6%82%e4%bd%95%e5%88%86%e9%a1%b5)

Guns的分页是通过[mybatis-plus的分页插件](http://baomidou.oschina.io/mybatis-plus-doc/#/page-plugin)实现的，大体分如下两种情况

[2.5.1 简单查询的分页](https://www.stylefeng.cn/doc/guns/#/dev/?id=_251-%e7%ae%80%e5%8d%95%e6%9f%a5%e8%af%a2%e7%9a%84%e5%88%86%e9%a1%b5)

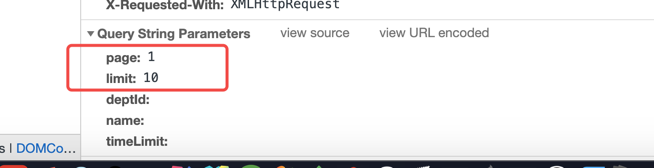
如果查询结果为单表查询，例如查询用户列表，则可以调用mybatis plus的自动生成的mapper中的selectPage()或者selectMapsPage()方法，Page类的构造函数中第一个参数为当前查询第几页，第二个参数为每页的记录数。 

[2.5.2 复杂查询的分页](https://www.stylefeng.cn/doc/guns/#/dev/?id=_252-%e5%a4%8d%e6%9d%82%e6%9f%a5%e8%af%a2%e7%9a%84%e5%88%86%e9%a1%b5)

若查询结果是关联多个表的操作，则需要用到自定义的mapper，此时的分页操作也很简单，只需要给mapper的第一个参数设置为Page对象即可，例如Guns中LogController中的查询操作日志列表，用的就是复杂查询的分页，我们可以看到在mybatis接口的第一个参数中，传递了Page对象，如下  当mybatis执行此方法的时候，会被mybatis-plus的分页插件自动拦截到，并且把分页查询的结果返回到这个Page对象中!

[2.5.3 获取前端表格插件传值](https://www.stylefeng.cn/doc/guns/#/dev/?id=_253-%e8%8e%b7%e5%8f%96%e5%89%8d%e7%ab%af%e8%a1%a8%e6%a0%bc%e6%8f%92%e4%bb%b6%e4%bc%a0%e5%80%bc)

Guns中前端表格用的[Layui Table](https://www.layui.com/doc/modules/table.html)插件，在前端执行查询时，插件会自动往后台传递分页参数，并且默认的格式如下，



Layui Table会传递page(第几页)，limit(每页条数)这两个参数，与之对应，后台封装了一个通用的接受分页参数的类LayuiPageFactory，从而不用每次都来request.getParameter()这样接收参数，如下所示，

public class LayuiPageFactory {

public static Page defaultPage() {

HttpServletRequest request = HttpContext.getRequest();

//每页多少条数据

int limit = Integer.valueOf(request.getParameter("limit"));

//第几页

int page = Integer.valueOf(request.getParameter("page"));

return new Page(page, limit);

}

public static LayuiPageInfo createPageInfo(IPage page) {

LayuiPageInfo result = new LayuiPageInfo();

result.setCount(page.getTotal());

result.setData(page.getRecords());

return result;

}

}

在后台代码中如需接收参数，构建分页Page对象的时候，只需如下这样一调用即可构建分页对象

Page page = LayuiPageFactory.defaultPage();

[2.6 数据范围](https://www.stylefeng.cn/doc/guns/#/dev/?id=_26-%e6%95%b0%e6%8d%ae%e8%8c%83%e5%9b%b4)

[2.6.1 介绍](https://www.stylefeng.cn/doc/guns/#/dev/?id=_261-%e4%bb%8b%e7%bb%8d)

Guns的数据范围是指当前部门的用户可以看到当前部门和子部门的数据，子部门的数据不可以看到上级部门的数据，但超级管理员例外，例如，userA和userB两个用户都有查看用户列表的权限，但是userA在总公司部门，userB在运营部，他们有如下部门关系



那么userA在查看用户列表的时候能看到公司所有人的数据，userB只能看到运营部的数据，这就是数据范围!

[2.6.2 如何使用](https://www.stylefeng.cn/doc/guns/#/dev/?id=_262-%e5%a6%82%e4%bd%95%e4%bd%bf%e7%94%a8)

使用时，只需要new一个DataScope，并在构造方法中传递给当前用户用后的部门权限(一般我们用封装好的ShiroKit.getDeptDataScope()方法即可获取到当前用户的部门权限集合)，之后，传递给mybatis的dao方法的第一个参数即可，例子如下

DataScope dataScope = new DataScope(ShiroKit.getDeptDataScope());

List<Map<String, Object>> users = managerDao.selectUsers(dataScope, name, beginTime, endTime， deptid);

**注意:** 在使用过程中，原mybatis的dao方法的查询结果中必须包含deptid字段(默认情况)，若部门id不叫deptid也可也初始化DateScope对象的时候，修改该对象的scopeName属性，改为自定义的部门id字段名即可

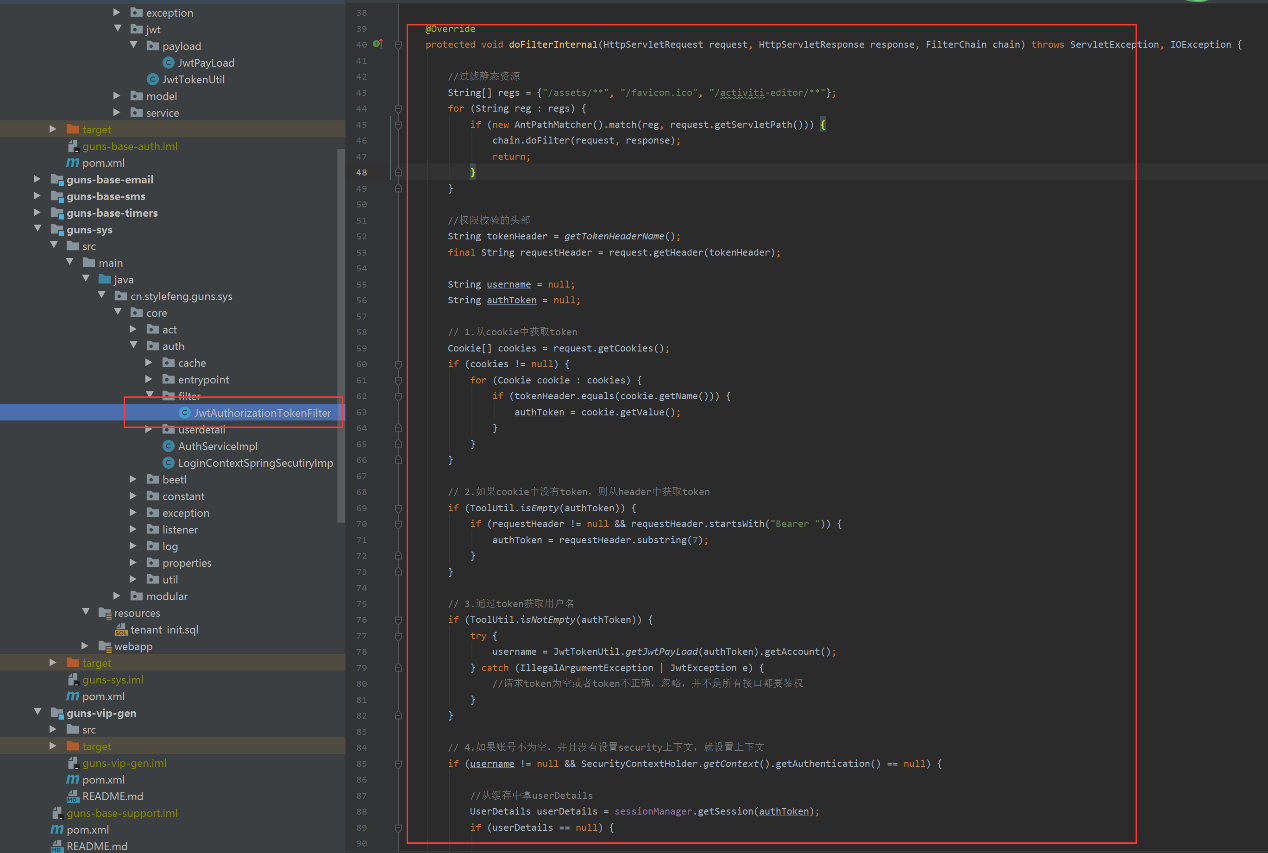
[2.6.3 原理](https://www.stylefeng.cn/doc/guns/#/dev/?id=_263-%e5%8e%9f%e7%90%86)

数据范围的原理是利用了mybatis拦截器，类似于mybatis-plus的分页插件，在原查询结果之上包装了一层select筛选查询，如下

select (原语句字段) from (原语句) where deptid in (DataScope对象中包含的部门id列表)

[2.7 spring security + jwt鉴权](https://www.stylefeng.cn/doc/guns/#/dev/?id=_27-spring-security-jwt%e9%89%b4%e6%9d%83)

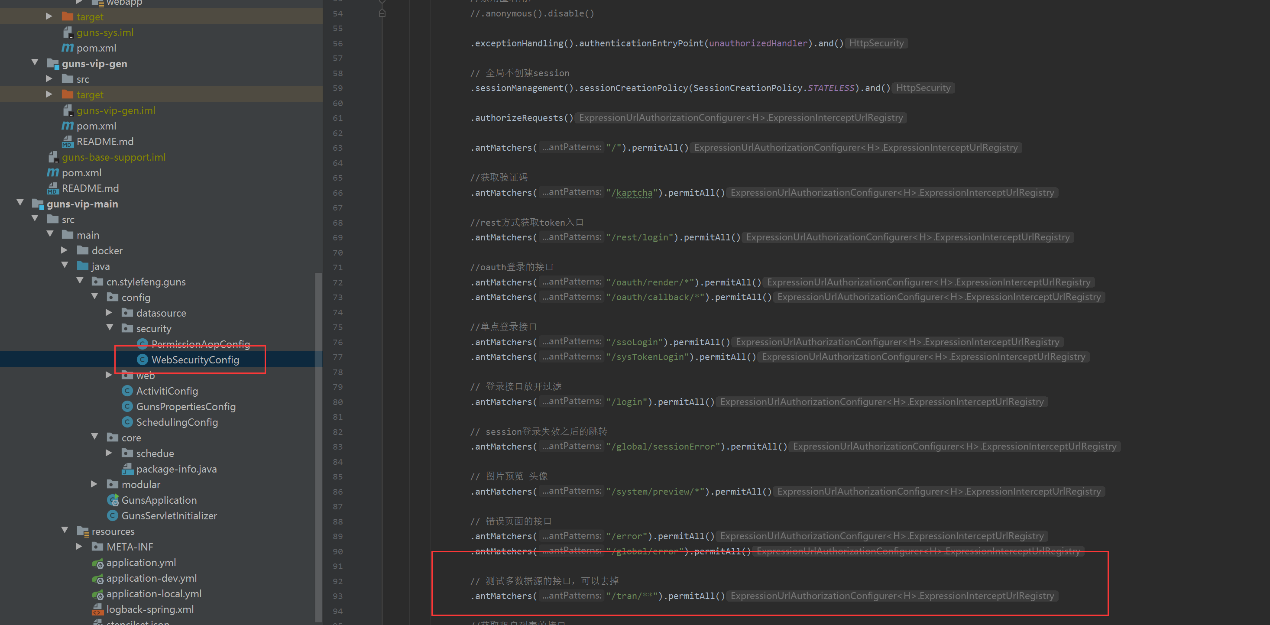
在新版的框架里项目已经整体换成spring security + jwt鉴权，security鉴权相关的配置在WebSecurityConfig类中，系统核心过滤器为JwtAuthorizationTokenFilter，如下：



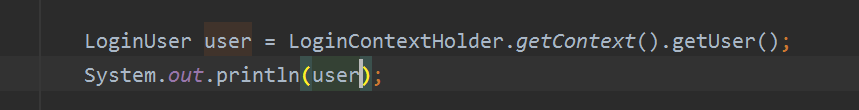
详细了解过滤器的内容对提高框架的认识有非常大的作用。

[2.7.1 放开权限过滤](https://www.stylefeng.cn/doc/guns/#/dev/?id=_271-%e6%94%be%e5%bc%80%e6%9d%83%e9%99%90%e8%bf%87%e6%bb%a4)

若项目中一些接口不需要权限过滤，可在WebSecurityConfig类中进行放开过滤。如下所示



[2.7.2 获取当前登录用户](https://www.stylefeng.cn/doc/guns/#/dev/?id=_272-%e8%8e%b7%e5%8f%96%e5%bd%93%e5%89%8d%e7%99%bb%e5%bd%95%e7%94%a8%e6%88%b7)

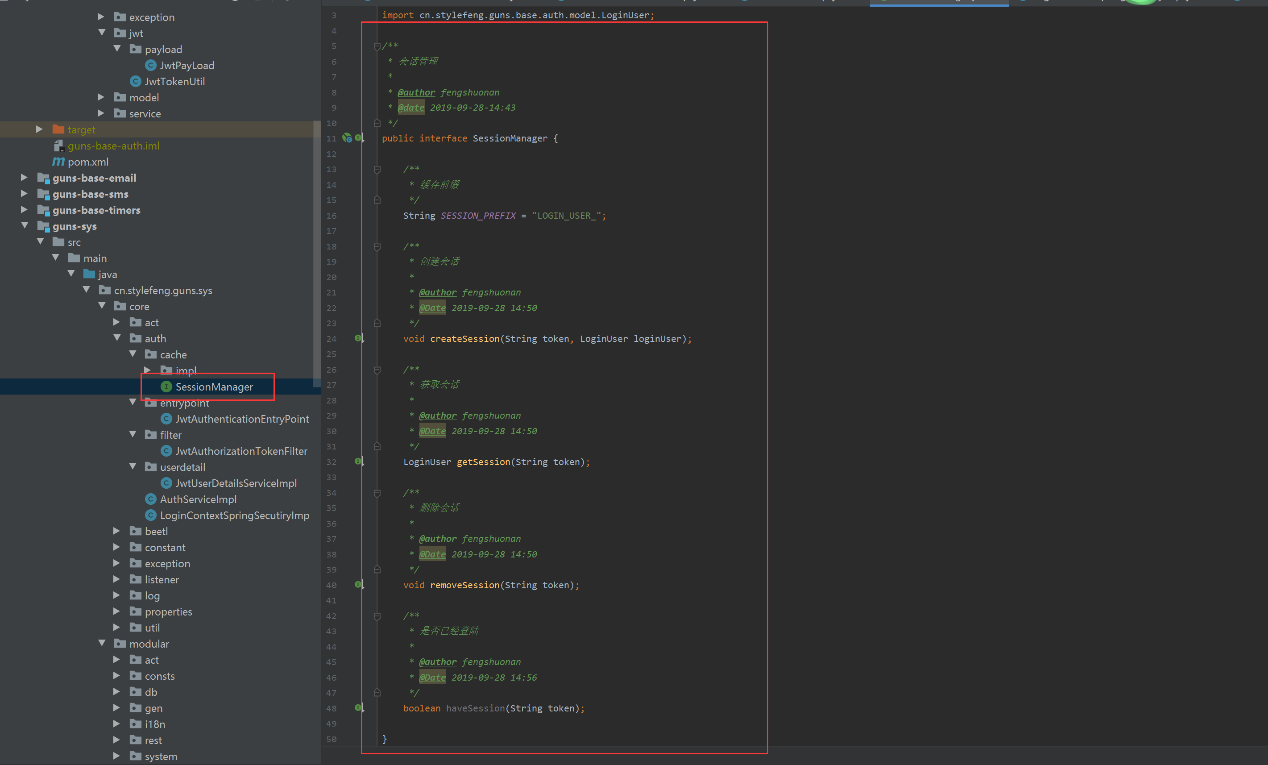
新系统去掉了以前的ShiroKit类，取而代之，用LoginContextHolder.getContext().getUser()获取当前登录用户。 

[2.7.3 会话管理](https://www.stylefeng.cn/doc/guns/#/dev/?id=_273-%e4%bc%9a%e8%af%9d%e7%ae%a1%e7%90%86)

系统虽然用jwt进行鉴权，但并非完全依赖jwt鉴权，如果单采用jwt鉴权则会存在jwt token销毁不及时的问题，当用户退出后，如果token无法销毁，用久的token仍可继续访问系统，会带来隐患。

因此，当用户登录系统后，会缓存当前登录用户的token，如果缓存的token被删除，则该用户无法访问系统接口。

系统管理session的接口如下：



默认采用的是map实现，您也可以根据需要采用redis的实现，并且定义自己的过期时间等策略。

[2.8 日志记录](https://www.stylefeng.cn/doc/guns/#/dev/?id=_28-%e6%97%a5%e5%bf%97%e8%ae%b0%e5%bd%95)

在我们日常开发中，对于某些关键业务，我们通常需要记录该操作的内容，例如修改了什么数据，修改的内容是什么，删除了哪些数据等等，在Guns中有一整套完善的解决方案来完成此项功能

[2.8.1 业务日志](https://www.stylefeng.cn/doc/guns/#/dev/?id=_281-%e4%b8%9a%e5%8a%a1%e6%97%a5%e5%bf%97)

我们通过@BussinessLog注解来记录日志，该注解源码如下，

/\*\*

\* 标记需要做业务日志的方法

\*

\* @author fengshuonan

\* @date 2017-03-31 12:46

\*/

@Inherited

@Retention(RetentionPolicy.RUNTIME)

@Target({ElementType.METHOD})

public @interface BussinessLog {

/\*\*

\* 业务的名称,例如:"修改菜单"

\*/

String value() default "";

/\*\*

\* 被修改的实体的唯一标识,例如:菜单实体的唯一标识为"id"

\*/

String key() default "id";

/\*\*

\* 字典(用于查找key的中文名称和字段的中文名称)

\*/

Class<? extends AbstractDictMap> dict() default SystemDict.class;

}

其中，value为需要记录日志的业务名称，key为修改或删除内容的唯一标识，通过这个唯一标识可以知道具体的修改的哪条记录，删除的哪条记录等等，dict为对修改字段的中文翻译字典，因为程序记录的都是英文的字段名称，这里通过字典，把英文字段和中文名称对应起来，那么日志信息记录到数据库中就可以变为中文的记录

以UserDict为例，

/\*\*

\* 用户的字典

\*

\* @author fengshuonan

\* @date 2017-05-06 15:01

\*/

public class UserDict extends AbstractDictMap {

@Override

public void init() {

put("userId","账号");

put("avatar","头像");

put("account","账号");

put("name","名字");

put("birthday","生日");

put("sex","性别");

put("email","电子邮件");

put("phone","电话");

put("roleid","角色名称");

put("deptid","部门名称");

put("roleIds","角色名称集合");

}

@Override

protected void initBeWrapped() {

putFieldWrapperMethodName("sex","getSexName");

putFieldWrapperMethodName("deptid","getDeptName");

putFieldWrapperMethodName("roleid","getSingleRoleName");

putFieldWrapperMethodName("userId","getUserAccountById");

putFieldWrapperMethodName("roleIds","getRoleName");

}

}

翻译字典类中包含两个方法init()和initBeWrapped()，其中init()为存放英文字段和中文字段的匹配，initBeWrapped()操作的是把某些字段的数字值翻译为中文直观名称的过程，例如当修改用户信息时，用户修改了一个人性别信息(数据库中1是男，2是女)，由1变为了2，程序记录的是数据库中1变为2，但是这句话给业务人员看到他是不知道1和2是什么东西的，所以这里做了一个值的包装，把1包装为对应的中文名称男，2包装为对应的中文名称女，这样，记录到数据库中，信息就变为了，xxx用户操作了修改用户功能，值由男变为了女. 在initBeWrapped()方法中putFieldWrapperMethodName()这个方法的第一参数是被包装的字段名，第二个参数是ConstantFactory中的方法名，因为默认会调用ConstantFactory来包装值属性

下面介绍业务日志记录的具体步骤:

* 1.在需要被记录日志的接口上添加@BussinessLog注解，并根据需要填写三个属性(value，key，dict)
* 2.若是添加或者修改业务，往往需要去编写Dict字典类
* 3.若是修改业务，例如修改用户信息，因为点击更新用户的时候不会提交修改之前的数据，所以在更新用户信息之前需要保存一下用户的旧的信息才可以记录用户修改的内容，这个缓存用户临时信息的地方一般添加在跳转到用户详情接口，用LogObjectHolder.me().set(user);这行代码来缓存用户的旧的信息，具体用法可以参考UserMgrController类中的userEdit()和edit()

[2.8.2 异常日志](https://www.stylefeng.cn/doc/guns/#/dev/?id=_282-%e5%bc%82%e5%b8%b8%e6%97%a5%e5%bf%97)

由于Guns有统一的异常拦截器，一般程序的报错，不管是业务异常还是未知的RuntimeException都会拦截并记录到数据库，若是您有自己的异常日志需要记录到数据库或者日志文件，推荐如下做法

1. 如果记录到数据库，调用Guns的日志记录工具类，如下

LogManager.me().executeLog();

该方法为异步记录日志的方法，executeLog()方法中需要传递一个TimerTask对象，TimerTask对象可以用LogTaskFactory类创建，在LogTaskFactory类中，有5个方法，可以分别记录不用的日志，有登录日志，退出日志，业务日志，异常日志等等，可以自行选择调用

1. 若需要记录日志到文件中，可以采用slf4j的org.slf4j.Logger类记录，具体方法如下

//首先在类中初始化

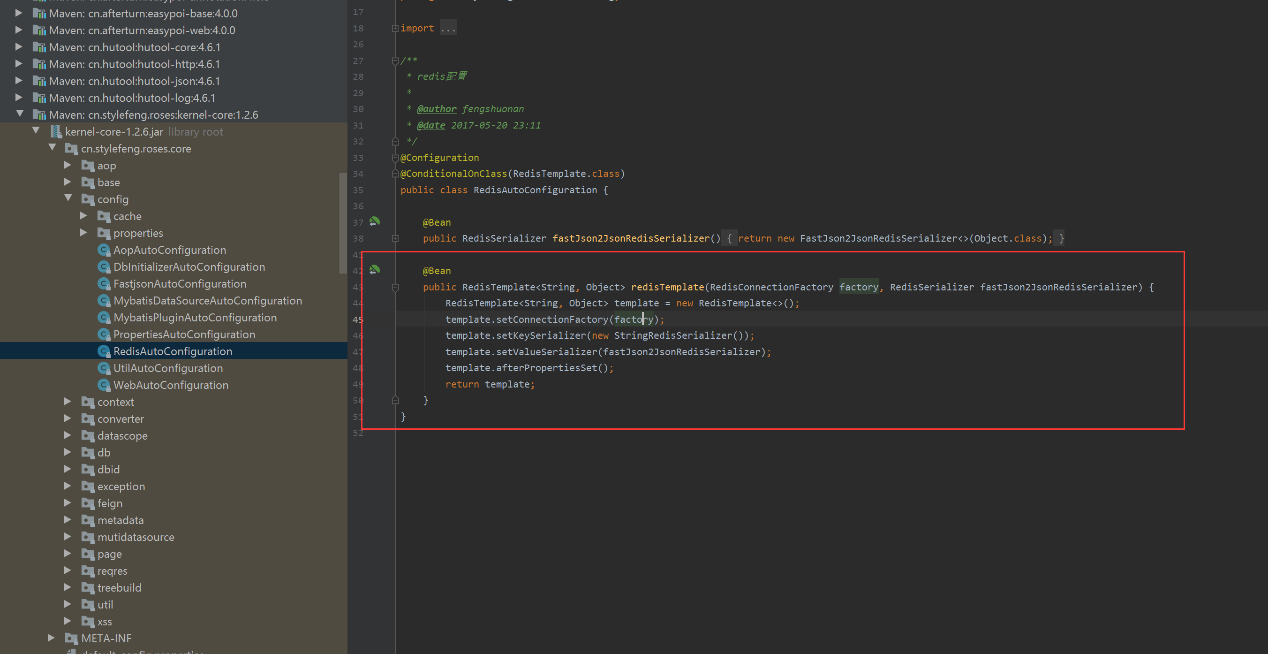
private Logger log = LoggerFactory.getLogger(this.getClass());

//再在方法中调用

log.error("业务异常:", e);

[2.9 如何使用缓存](https://www.stylefeng.cn/doc/guns/#/dev/?id=_29-%e5%a6%82%e4%bd%95%e4%bd%bf%e7%94%a8%e7%bc%93%e5%ad%98)

因为guns的kernel-core模块内置了一个缓存配置，所以如需使用redis无需太多的配置，内置自动配置如下：



1.引入pom

增加如下配置到pom中

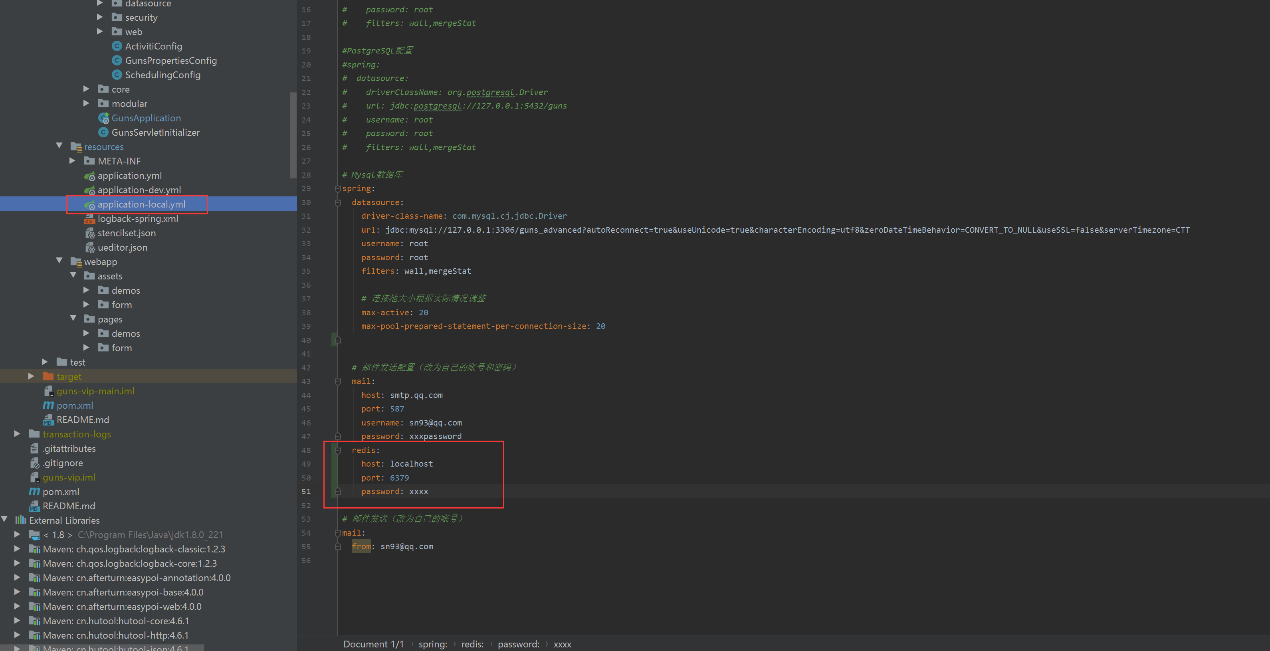
<!--缓存-->

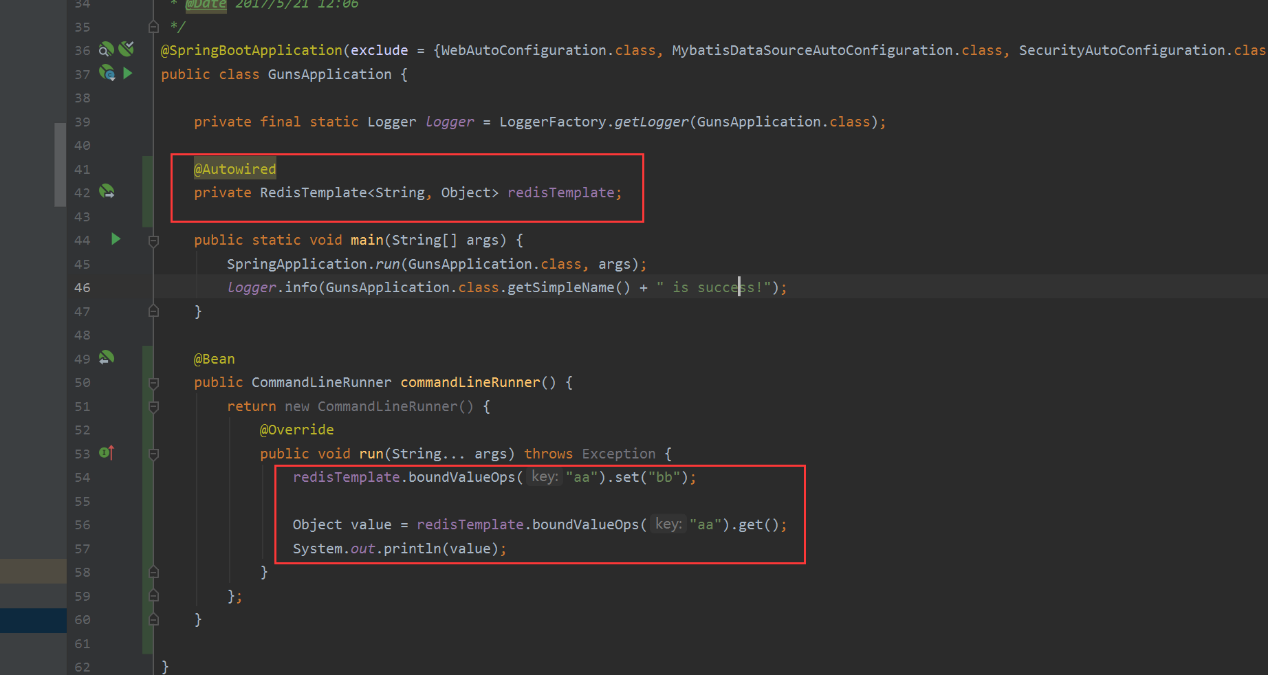
<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-redis</artifactId>

</dependency>

2.在yml中配置redis的连接 

3.在需要用缓存的类中(一般为service)注入RedisTemplate，并编写缓存操作代码： 

[2.10 使用枚举](https://www.stylefeng.cn/doc/guns/#/dev/?id=_210-%e4%bd%bf%e7%94%a8%e6%9e%9a%e4%b8%be)

在Guns中，枚举一般分两类，一种是状态枚举，一种是异常枚举，状态枚举的作用是枚举状态，列出状态的所有值，例如

/\*\*

\* 菜单的状态

\*

\* @author fengshuonan

\* @Date 2017年1月22日 下午12:14:59

\*/

public enum MenuStatus {

ENABLE(1, "启用"),

DISABLE(0, "禁用");

int code;

String message;

MenuStatus(int code, String message) {

this.code = code;

this.message = message;

}

...

}

异常枚举的作用是枚举所有出现的业务异常，例如，

/\*\*

\* 所有业务异常的枚举

\*

\* @author fengshuonan

\* @date 2016年11月12日 下午5:04:51

\*/

public enum BizExceptionEnum implements ServiceExceptionEnum{

/\*\*

\* 错误的请求

\*/

SESSION\_TIMEOUT(400, "会话超时"),

SERVER\_ERROR(500, "服务器异常");

BizExceptionEnum(int code, String message) {

this.code = code;

this.message = message;

}

private Integer code;

private String message;

...

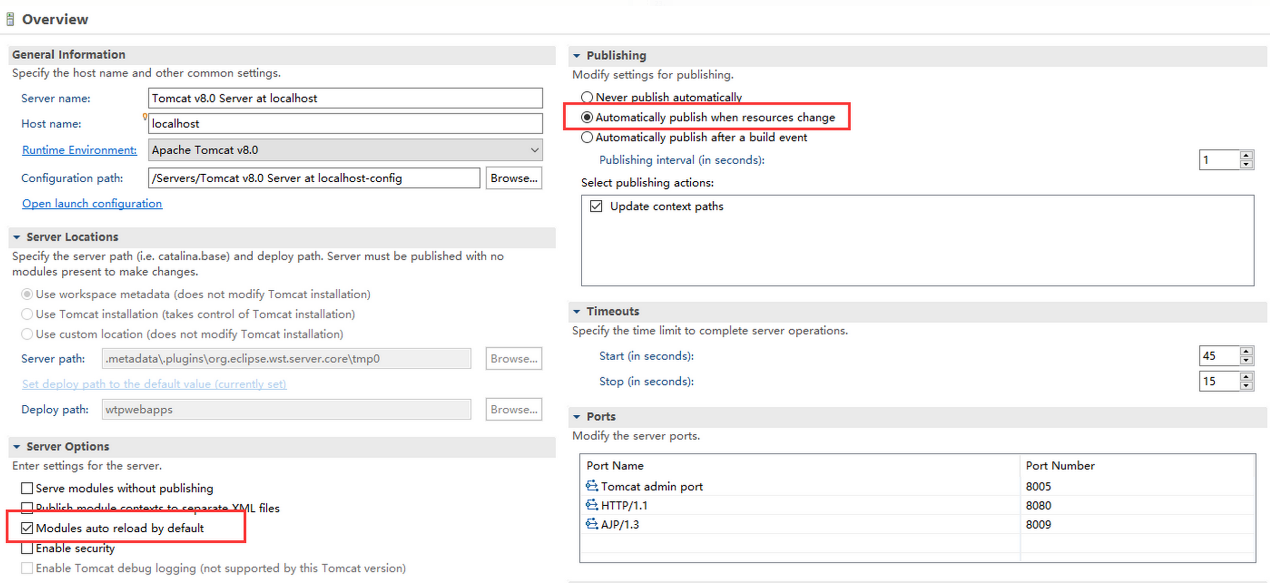
}

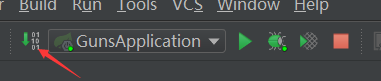
使用枚举可以方便维护一些状态的值和管理所有的业务异常，所以在有状态或者新的业务异常的时候推荐写到枚举里

[2.11 spring boot热部署](https://www.stylefeng.cn/doc/guns/#/dev/?id=_211-spring-boot%e7%83%ad%e9%83%a8%e7%bd%b2)

**热部署的两种情况(适用于main方法启动)**

[2.11.1 重新加载html](https://www.stylefeng.cn/doc/guns/#/dev/?id=_2111-%e9%87%8d%e6%96%b0%e5%8a%a0%e8%bd%bdhtml)

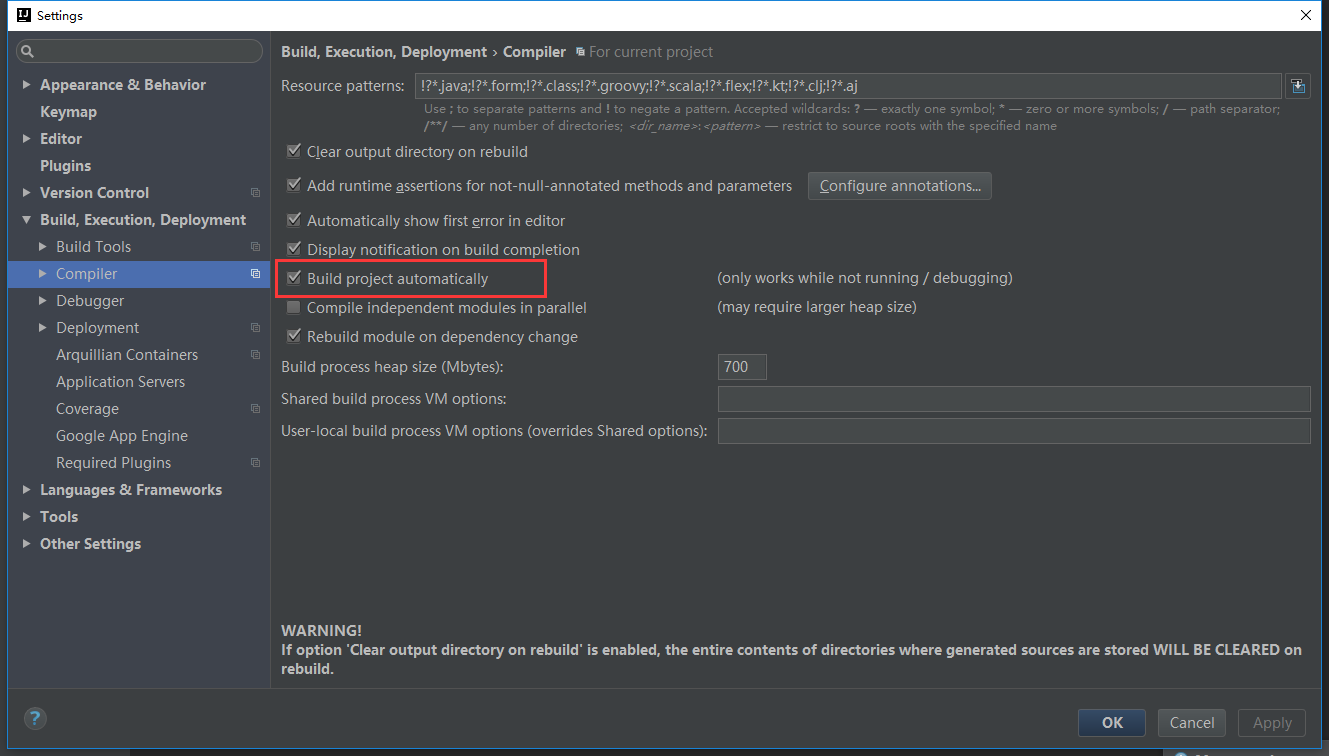
如果是eclipse修改html保存后可以自动替换，如果不能请检查server配置 

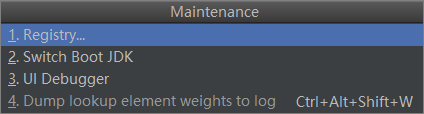
如果是IDEA，可以修改html之后点击这个按钮，或者按快捷键CTRL+F9，即可更新 

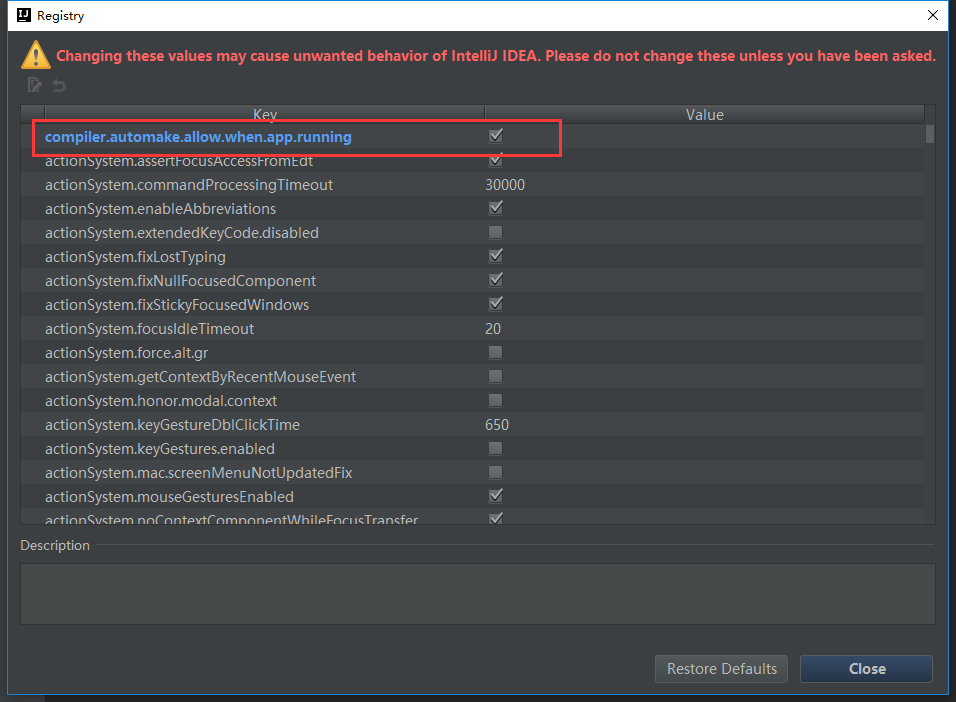
[2.11.2 重新加载java类](https://www.stylefeng.cn/doc/guns/#/dev/?id=_2112-%e9%87%8d%e6%96%b0%e5%8a%a0%e8%bd%bdjava%e7%b1%bb)

如果是eclipse，在application.yml中找到配置spring.devtools.restart.enabled改为true即可

如果是在IDEA中:  
**第一步** 请先修改spring.devtools.restart.enabled=true

**第二步** 如下idea配置，打上对勾 

**第三步** 按下 Shift+Ctrl+Alt+/，选择Registry 

进去之后，找到如下图所示的选项，打勾 

[2.12 发送邮件](https://www.stylefeng.cn/doc/guns/#/dev/?id=_212-%e5%8f%91%e9%80%81%e9%82%ae%e4%bb%b6)

**发送邮件属于Guns企业版功能，依赖于guns-base-email模块。**

1.引入如下依赖

<!-- 邮件发送 -->

<dependency>

<groupId>cn.stylefeng</groupId>

<artifactId>guns-base-email</artifactId>

<version>1.0.0</version>

</dependency>

2.配置相关邮件的配置，在yml中

spring:

mail:

host: smtp.qq.com

port: 587

username: xxx@xx.com

password: xxxpassword

mail:

from: xxx@xx.com

3.使用如下代码发送邮件

SendMailParam sendMailParam = new SendMailParam();

String content = "邮件内容测试！";

sendMailParam.setTo("xxx@xx.com");

sendMailParam.setTitle(content);

sendMailParam.setContent(content);

mailServiceApi.sendMail(sendMailParam);

[2.13 发送短信](https://www.stylefeng.cn/doc/guns/#/dev/?id=_213-%e5%8f%91%e9%80%81%e7%9f%ad%e4%bf%a1)

**发送短信属于Guns企业版功能，依赖于guns-base-sms模块。**

1.引入如下依赖

<!-- 短信发送 -->

<dependency>

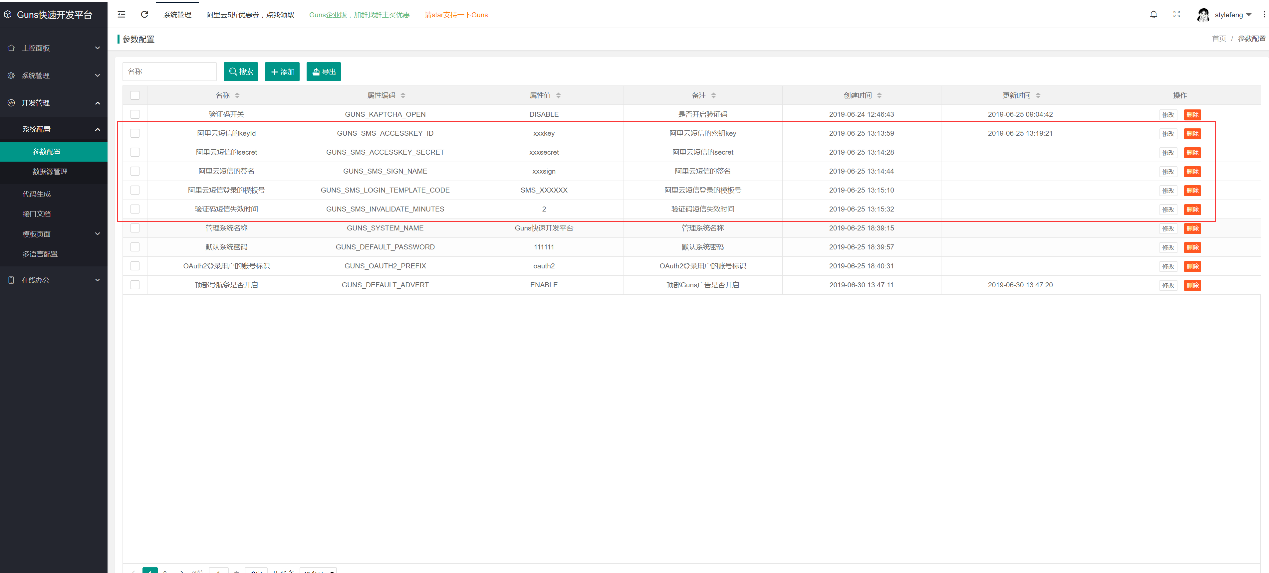
<groupId>cn.stylefeng</groupId>

<artifactId>guns-base-sms</artifactId>

<version>1.0.0</version>

</dependency>

2.在线配置相关的短信配置



3.使用如下代码发送邮件

SendMessageParam sendMessageParam = new SendMessageParam();

sendMessageParam.setPhoneNumbers("18201000000");

//模板号

sendMessageParam.setTemplateCode("SMS\_XXXXXXXXXXX");

//模板里的参数

HashMap<String, Object> params = new HashMap<>();

params.put("xxxx", heartCheck.getServiceName());

sendMessageParam.setParams(params);

smsServiceApi.sendShortMessage(sendMessageParam);