renren-security开发文档

renren-security是一个轻量级权限管理系统，其核心设计目标是开发迅速、学习简单、轻量级、易扩展等。

具有如下特点

- 轻量级的权限系统，只涉及Spring、Shiro、Mybatis后端框架，降低学习使用成本
- 友好的代码结构及注释，便于阅读及二次开发
- 支持HTML、JSP、Velocity、Freemarker等视图，零技术门槛
- 灵活的权限控制，可控制到页面或按钮，满足绝大部分的权限需求(如需控制到按钮级别，需使用Shiro标签，控制按钮的显示或隐藏)
- 页面交互使用Vue2.x，极大的提高了开发效率
- 完善的代码生成机制，可在线生成entity、xml、dao、service、page、js代码，减少70%以上的开发任务
- 引入quartz定时任务，可动态完成任务的添加、修改、删除、暂停、恢复及日志查看等功能
- 引入路由机制，刷新页面会停留在当前页

如何交流、反馈、参与贡献？

- 演示地址：http://security.renren.io (账号密码：admin/admin)
- 源码地址：http://git.oschina.net/babaio/renren-security
- 编程教程：http://www.renren.io

- 官方QQ群：324780204
- 如需关注项目最新动态，请watch、star项目，同时也是对项目最好的支持
- 技术讨论、二次开发等咨询、问题和建议，请移步到QQ群324780204，我会在第一时间进行解答和回复！

技术选型

- 核心框架：Spring Framework 4.2
- 安全框架：Apache Shiro 1.3
- 视图框架：Spring MVC 4.2
- 持久层框架：MyBatis 3.3
- 定时器：Quartz 2.2
- 数据库连接池：Druid 1.0
- 日志管理：SLF4J 1.7、Log4j
- 页面交互：Vue2.x

环境需求

- JDK1.7+
- MySQL5.5+
- Tomcat7.0+
- Maven3.0+

项目部署

- 通过git下载源码，源码地址：https://git.oschina.net/babaio/renren-security.git
- 创建数据库renren-security，数据库编码为UTF-8
- 执行doc/db.sql文件，初始化数据
- 修改db.properties文件，更新MySQL账号和密码
- Eclipse、IDEA执行【clean package tomcat7:run】命令，即可运行项目
- 项目访问路径：http://localhost
- 非Maven方式启动，则默认访问路径为：http://localhost:8080/renren-security
- 建议使用阿里云Maven仓库

```
<mirror>
      <id>alimaven</id>
      <name>aliyun maven</name>
      <url>http://maven.aliyun.com/nexus/content/groups/public/</url>
      <mirrorOf>central</mirrorOf>
</mirror>
```

定时任务使用

定时任务使用场景还是比较多的，一般项目都会使用到定时任务，如：定时发送短信、邮件、商品定时上架、下架、优惠券过期、未支付订单取消、快递信息更新、数据报表、数据统计、结算等等；本项目使用Quartz2.2.x实现定时任务功能，支持添加、修改、删除、暂停、恢复、集群及日志查看等功能。

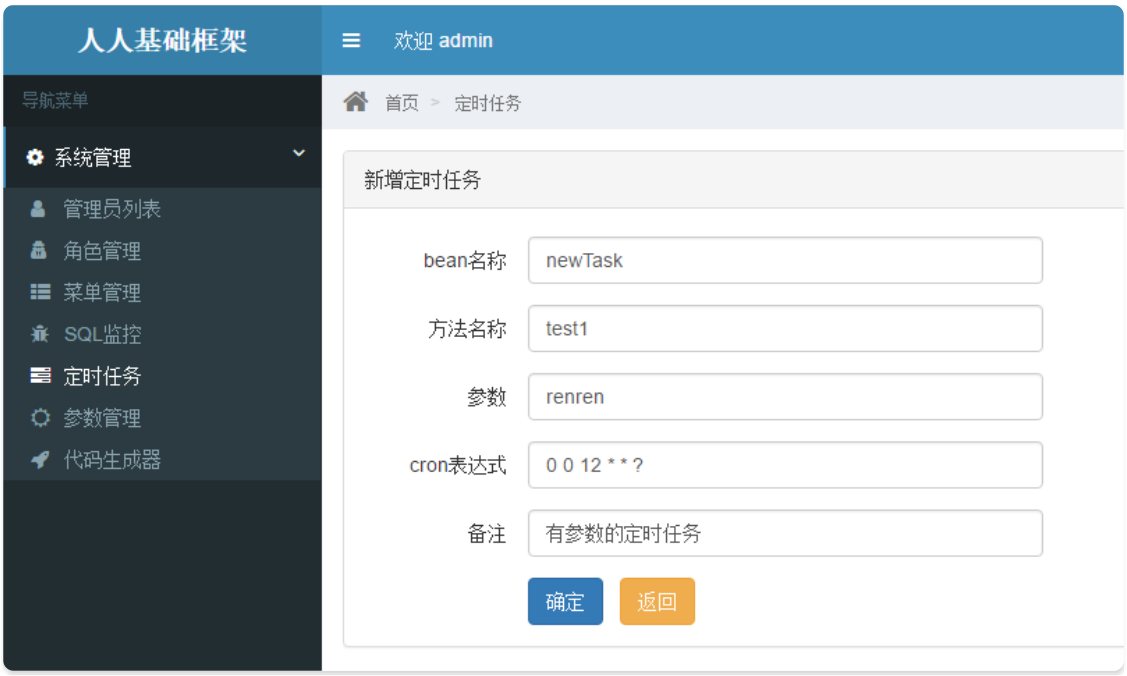
1、新增定时任务，只需创建spring bean即可，如下所示：

```java
/**
 * 新增定时任务
 *
 * bean的名称为【newTask】
 */
@Component("newTask")
public class NewTask {
    private Logger logger = LoggerFactory.getLogger(getClass());

    public void test1(String params){
        logger.info("我是带参数的test1方法，正在被执行，参数为：" + params);
    }

    public void test2(){
        logger.info("我是不带参数的test2方法，正在被执行");
    }
}
```

2、在管理后台，添加定时任务，如下图所示：



完成上面2步，新的定时任务就添加完成了，每天12点都会执行一次

代码生成器使用

- 代码生成器是根据表结构，自动生成相应的代码，需先在MySQL中建好表结构，再使用代码生成器
- 代码生成器是通过velocity模板实现的，依赖velocity所需jar包，可在线生成entity、xml、dao、service、page、js代码的zip压缩文件

- 修改包名、作者、作者邮箱，需在`generator.properties`中配置
- 如需去掉表`tb_user`前缀`tb_`，可配置`tablePrefix`项，如：`tablePrefix=tb_`，则生成的实体类为`UserEntity`，否则生成`TbUserEntity`
- 可根据自己的需求，自行修改模板，模板代码位置：【resources\template】
- 模板数据封装在如下所示map里，其中`tableEntity`对象为`TableEntity.java`实例，`config`为`generator.properties`配置文件数据

```
//模板数据
Map<String, Object> map = new HashMap<>();
map.put("tableName", tableEntity.getTableName());
map.put("comments", tableEntity.getComments());
map.put("pk", tableEntity.getPk());//数据库主键，没有则为第一个字段
map.put("className", tableEntity.getClassName());
map.put("classname", tableEntity.getClassname());
map.put("pathName", tableEntity.getClassname().toLowerCase());
map.put("columns", tableEntity.getColumns());
map.put("package", config.getString("package"));
map.put("author", config.getString("author"));
map.put("email", config.getString("email"));
map.put("datetime", DateUtils.format(new Date(), DateUtils.DATE_TIME_PATTERN));
```

- MySQL数据类型与Java数据类型转换，在`generator.properties`中配置，如有些类型的转换关系不存在，则需在`generator.properties`中添加，如：`bigint=Long`
- 生成的html、js代码，需要修改html代码里的js路径，避免404错误
- 添加相应的菜单即可【权限标识一定要添加，不然没权限访问接口】

多视图使用

- 默认是支持JSP、Velocity、Freemarker视图
- 文件后缀以jsp结尾【如：test.jsp】，则会使用JSP视图
- 文件后缀以html结尾【如：test.html】，则会使用Velocity视图
- 文件后缀以ftl结尾【如：test.ftl】，则会使用Freemarker视图
- 本系统是通过iframe方式嵌入页面的，可以使用任何前端框架，如：ExtJS等等，也非常方便与现有的系统集成

异常处理

系统做了异常统一处理，无需try catch，只需往外抛就行，异常处理如下：

```
@Component
public class RRExceptionHandler implements HandlerExceptionResolver {
    private Logger logger = LoggerFactory.getLogger(getClass());

    @Override
    public ModelAndView resolveException(HttpServletRequest request,
            HttpServletResponse response, Object handler, Exception ex) {
        R r = new R();
        try {
            response.setContentType("application/json;charset=utf-8");
            response.setCharacterEncoding("utf-8");

            if (ex instanceof RRException) {
                r.put("code", ((RRException) ex).getCode());
                r.put("msg", ((RRException) ex).getMessage());
            }else if(ex instanceof DuplicateKeyException){
                r = R.error("数据库中已存在该记录");
            }else if(ex instanceof AuthorizationException){
                r = R.error("没有权限，请联系管理员授权");
            }else{
                r = R.error();
            }

            //记录异常日志
            logger.error(ex.getMessage(), ex);

            String json = JSON.toJSONString(r);
            response.getWriter().print(json);
        } catch (Exception e) {
            logger.error("RRExceptionHandler 异常处理失败", e);
        }
        return new ModelAndView();
    }
}
```

项目打赏


微信扫一扫 支付