# 开发文档

**快速了解**

**简介**

RuoYi 是一个 Java EE 企业级快速开发平台，基于经典技术组合（Spring Boot、Apache Shiro、MyBatis、Thymeleaf、Bootstrap、Hplus），内置模块如：部门管理、角色用户、菜单及按钮授权、数据权限、系统参数、日志管理、通知公告等。在线定时任务配置；支持集群，支持多数据源。

**主要特性**

* + - 完全响应式布局（支持电脑、平板、手机等所有主流设备）
    - 强大的一键生成功能（包括控制器、模型、视图、菜单等）
    - 支持多数据源，简单配置即可实现切换。
    - 支持按钮及数据权限，可自定义部门数据权限。
    - 对常用JavaScript插件进行二次封装，使JavaScript代码变得简洁，更加易维护
    - 完善的XSS防范及脚本过滤，彻底杜绝XSS攻击
    - Maven多项目依赖，模块及插件分项目，尽量松耦合，方便模块升级、增减模块。
    - 国际化支持，服务端及客户端支持
    - 完善的日志记录体系简单注解即可实现

**技术选型**

1. **系统环境**

* Java EE 8
* Servlet 3.0
* Apache Maven 3

1. **主框架**

* Spring Boot 2.0
* Spring Framework 5.0
* Apache Shiro 1.4

1. **持久层**

* Apache MyBatis 3.4
* Hibernate Validation 6.0
* Alibaba Druid 1.1

1. **视图层**

* Bootstrap 3.3
* Hplus 4.1
* Thymeleaf 3.0

**内置功能**

* 用户管理：用户是系统操作者，该功能主要完成系统用户配。
* 部门管理：配置系统组织机构（公司、部门、小组），树结构展现支持数据权限。
* 岗位管理：配置系统用户所属担任职务。
* 菜单管理：配置系统菜单，操作权限，按钮权限标识等。
* 角色管理：角色菜单权限分配、设置角色按机构进行数据范围权限划分。
* 字典管理：对系统中经常使用的一些较为固定的数据进行维护。
* 参数管理：对系统动态配置常用参数。
* 通知公告：系统通知公告信息发布维护。
* 操作日志：系统正常操作日志记录和查询；系统异常信息日志记录和查询。
* 登陆日志：系统登录日志记录查询包含登录异常。
* 在线用户：当前系统中活跃用户状态监控。
* 定时任务：在线（添加、修改、删除)任务调度包含执行结果日志
* 代码生成：前后端代码的生成（java、html、xml、sql)支持CRUD下载 。
* 系统接口：根据业务代码自动生成相关的api接口文档。
* 服务监控：监视当前系统CPU、内存、磁盘、堆栈等相关信息。
* 在线构建器：拖动表单元素生成相应的HTML代码。
* 连接池监视：监视当期系统数据库连接池状态，可进行分析SQL找出系统性能瓶颈。

**更新日志**

**环境部署**

**准备工作**

>= 1.8 (推荐1.8版本)

Mysql >= 5.5.0 (推荐5.7版本)

Maven >= 3.0

**运行系统**

**必要配置**

1. **修改数据库连接**

编辑resources目录下的application-druid.yml  
url: 服务器地址  
username: 账号  
password: 密码

1. **开发环境配置**

编辑resources目录下的application.yml  
port: 端口  
context-path: 部署路径

**部署系统**

bin/package.bat 在项目的目录下执行

然后会在项目下生成 target文件夹包含 war 或jar （多模块生成在ruoyi-admin）

1、jar部署方式

使用命令行执行：java –jar RuoYi.jar 或者执行脚本：bin/run.bat

2、war部署方式

pom.xml packaging修改为war 放入tomcat服务器webapps

SpringBoot去除内嵌tomcat

****

**常见问题**

如果使用Mac 需要修改application.yml文件路径profile

如果使用Linux 提示表不存在，设置大小写敏感配置在/etch/my.cnf 添加lower\_case\_table\_names=1，重启MYSQL服务

如果提示当前权限不足，无法写入文件请检查profile是否可读可写，或者无法访问此目录。

**项目介绍**

**文件结构**

com.ruoyi

├── common // 工具类

│ └── annotation // 自定义注解

│ └── config // 全局配置

│ └── constant // 通用常量

│ └── core // 核心控制

│ └── enums // 通用枚举

│ └── exception // 通用异常

│ └── json // JSON数据处理

│ └── utils // 通用类处理

│ └── xss // XSS过滤处理

├── framework // 框架核心

│ └── aspectj // 注解实现

│ └── config // 系统配置

│ └── datasource // 数据权限

│ └── manager // 异步处理

│ └── shiro // 权限控制

│ └── util // 通用工具

│ └── web // 前端控制

├── ruoyi-generator // 代码生成（可移除）

├── ruoyi-quartz // 定时任务（可移除）

├── ruoyi-system // 系统代码

├── ruoyi-admin // 后台服务

├── ruoyi-xxxxxx // 其他模块

**配置文件**

通用配置 application.yml



数据源配置 application-druid.yml



**核心技术**

**Spring Boot框架**

1、介绍

Spring Boot是一款开箱即用框架，提供各种默认配置来简化项目配置。让我们的Spring应用变的更轻量化、更快的入门。 在主程序执行main函数就可以运行。你也可以打包你的应用为jar并通过使用java -jar来运行你的Web应用。它遵循"约定优先于配置"的原则， 使用SpringBoot只需很少的配置，大部分的时候直接使用默认的配置即可。可以与Spring Cloud的微服务无缝结合。

Spring Boot2.0 环境要求必须是jdk8或以上版本，Tomcat8或以上版本

2、优点

* 使编码变得简单： 推荐使用注解。
* 使配置变得简单： 自动配置、快速构建项目、快速集成新技术能力 没有冗余代码生成和XML配置的要求
* 使部署变得简单： 内嵌Tomcat、Jetty、Undertow等web容器，无需以war包形式部署
* 使监控变得简单： 自带项目监控

**Shiro 安全控制**

1、介绍  
Apache Shiro是Java的一个安全框架。Shiro可以帮助我们完成：认证、授权、加密、会话管理、与Web集成、缓存等。其不仅可以用在 JavaSE环境，也可以用在 JavaEE 环境。

2、优点

* 易于理解的 Java Security API
* 简单的身份认证，支持多种数据源
* 对角色的简单的授权，支持细粒度的授权
* 不跟任何的框架或者容器捆绑，可以独立运行

3、特性  
Authentication身份认证/登录，验证用户是不是拥有相应的身份  
Authorization授权，即验证权限，验证某个已认证的用户是否拥有某个权限，即判断用户是否能做事情 SessionManagement会话管理，即用户登录后就是一次会话，在没有退出之前，它的所有信息都在会话中  
Cryptography加密，保护数据的安全性，如密码加密存储到数据库，而不是明文存储  
Caching缓存，比如用户登录后，其用户信息，拥有的角色/权限不必每次去查，提高效率  
ConcurrencyShiro支持多线程应用的并发验证，即如在一个线程中开启另一个线程，能把权限自动传播过去  
Testing提供测试支持  
RunAs允许一个用户假装为另一个用户（如果他们允许）的身份进行访问  
RememberMe记住我，这是非常常见的功能，即一次登录后，下次再来的话不用登录了

4、架构  
Subject主体，代表了当前的“用户”，这个用户不一定是一个具体的人，与当前应用交互的任何东西都是Subject，如网络爬虫， 机器人等；即一个抽象概念；所有Subject都绑定到SercurityManager，与Subject的所有交互都会委托给SecurityManager；可以把Subject认为是一个门面；SecurityManager才是实际的执行者  
SecurityManage安全管理器；即所有与安全有关的操作都会与SecurityManager交互；且它管理着所有Subject； 可以看出它是Shiro的核心，它负责与后边介绍的其他组件进行交互  
Realm域，Shiro从Realm获取安全数据（如用户，角色，权限），就是说SecurityManager要验证用户身份， 那么它需要从Realm获取相应的用户进行比较以确定用户身份是否合法；也需要从Realm得到用户相应的角色/权限进行验证用户是否能进行操作；可以有1个或多个Realm，我们一般在应用中都需要实现自己的Realm  
SessionManager如果写过Servlet就应该知道Session的概念，Session需要有人去管理它的生命周期，这个组件就是SessionManager  
SessionDAO大家都用过，数据库访问对象，用于会话的CRUD，比如我们想把Session保存到数据库，那么可以实现自己的SessionDAO，也可以写入缓存，以提高性能  
CacheManager缓存控制器，来管理如用户，角色，权限等的缓存的；因为这些数据基本上很少去改变，放到缓存中后可以提高访问的性能

应用代码通过Subject来进行认证和授权，而Subject又委托给SecurityManager； 我们需要给Shrio的SecurityManager注入Realm，从而让SecurityManager能得到合法的用户及其权限进行判断，Shiro不提供维护用户/权限，而是通过Realm让开发人员自己注入。

Shiro不会去维护用户，维护权限；这些需要自己去设计/提供；然后通过响应的接口注入给Shiro即可。

**Thymeleaf模板**

1、介绍

Thymeleaf是一个用于Web和独立Java环境的模板引擎，能够处理HTML、XML、JavaScript、CSS甚至纯文本。能轻易的与Spring MVC等Web框架进行集成作为Web应用的模板引擎。 与其它模板引擎（比如FreeMaker）相比，Thymeleaf最大的特点是能够直接在浏览器中打开并正确显示模板页面，而不需要启动整个Web应用（更加方便前后端分离，比如方便类似VUE前端设计页面），抛弃JSP吧。 Thymeleaf 3.0是一个完全彻底重构的模板引擎，极大的减少内存占用和提升性能和并发性，避免v2.1版因大量的输出标记的集合产生的资源占用。 Thymeleaf 3.0放弃了大多数面向DOM的处理机制，变成了一个基于事件的模板处理器，它通过处理模板标记或文本并立即生成其输出，甚至在新事件之前响应模板解析器/缓存事件。Thymeleaf是Spring Boot官方的推荐使用模板。

2、优点

国际化支持非常简单

语法简单，功能强大。内置大量常用功能，使用非常方便

可以很好的和Spring集成

静态html嵌入标签属性，浏览器可以直接打开模板文件，便于前后端联调

Spring Boot 官方推荐，用户群广

**后台手册**

**分页实现**

前端基于Bootstrap的轻量级表格插件 BootstrapTable  
后端分页组件使用Mybatis分页插件 PageHelper

分页的实现流程：

1. 前端调用封装好的方法$.table.init，传入后台url。



2、自定义查询条件参数（特殊情况提前设置查询条件下使用）



3、后台实现查询逻辑，调用startPage()方法即可自动完成服务端分页

****

常见坑点1：selectPostById莫名其妙的分页。例如下面这段代码



原因分析：这种情况下由于user存在null的情况，就会导致PageHelper生产了一个分页参数，但是没有被消费，这个参数就会一直保留在这个线程上。 当这个线程再次被使用时，就可能导致不该分页的方法去消费这个分页参数，这就产生了莫名其妙的分页。上面这个代码，应该写成下面这个样子才能保证安全:



常见坑点2：添加了startPage方法。也没有正常分页。例如下面这段代码



原因分析：只对该语句以后的第一个查询（Select）语句得到的数据进行分页。  
上面这个代码，应该写成下面这个样子才能正常分页:



注意：如果改为其他数据库需修改配置application.yml helperDialect=你的数据库

**导入导出**

在实际开发中经常需要使用导入导出功能来加快数据的操作。在项目中可以使用注解来完成此项功能。 在需要被导入导出的实体类属性添加@Excel注解，目前支持参数如下：

| 参数 | 类型 | 默认值 | 描述 |
| --- | --- | --- | --- |
| name | String | 空 | 导出到Excel中的名字 |
| dateFormat | String | 空 | 日期格式, 如: yyyy-MM-dd |
| readConverterExp | String | 空 | 读取内容转表达式 (如: 0=男,1=女,2=未知) |
| height | String | 14 | 导出时在excel中每个列的高度 单位为字符 |
| width | String | 16 | 导出时在excel中每个列的宽 单位为字符 |
| suffix | String | 空 | 文字后缀,如% 90 变成90% |
| defaultValue | String | 空 | 当值为空时,字段的默认值 |
| prompt | String | 空 | 提示信息 |
| combo | String | Null | 设置只能选择不能输入的列内容 |
| targetAttr | String | 空 | 另一个类中的属性名称,支持多级获取,以小数点隔开 |
| type | Enum | Type.ALL | 字段类型（0：导出导入；1：仅导出；2：仅导入） |

导出实现流程

1. 前端调用封装好的方法$.table.init，传入后台exportUrl



1. 添加导出按钮事件



1. 在实体变量上添加@Excel注解



1. 在Controller添加导出方法



导入实现流程

1. 前端调用封装好的方法$.table.init，传入后台exportUrl



1. 添加导入按钮事件



1. 添加导入前端代码，form默认id为importForm，也可以指定importFile(id)



1. 在实体变量上添加@Excel注解，默认为导出导入，也可以单独设置仅导入Type.IMPORT



1. 在Controller添加导入方法，updateSupport属性为是否存在则覆盖（可选）



**上传下载**

首先创建一张上传文件的表，例如：



上传的实现流程：

1. 代码生成sys\_file\_info表相关代码并覆盖至对应目录。
2. 参考示例代码。



1. 在FileInfoController添加对应上传方法。



1. 上传文件成功后需要预览可以对该属性格式化处理



如需对文件格式控制，设置application.yml中的multipart属性:



注意：如果只是单纯的上传一张图片没有其他参数可以使用通用方法 /common/upload  
请求处理方法 com.ruoyi.web.controller.common.CommonController

**下载实现流程**

1. 参考示例代码

****

1. 在Controller添加对应上传方法



**事务管理**

**异常处理**

**系统日志**

**数据权限**

**多数据源**

**代码生成**

**定时任务**

**系统接口**

**国际化支持**

**前端手册**

**前端组件**

**通用方法**

**表格使用**

**弹层使用**

**权限使用**

**字典使用**

**参数使用**

**组件文档**

**Bootstrap-datetimepicker**

**Laydate**

**Layer**

**Bootstrap-select**

**Select2**

**Jasny-bootstrap**

**Bootstrap-fileinput**

**Bootstrap-duallistbox**

**Jquery-validate**

**Bootstrap-suggest**

**Bootstrap-typeahead**

**项目扩展**

**常见问题**