**JFast开发手册**

# 平台说明

## 平台说明

JFast快速开发平台是一套轻量级的，集业务建模和代码生成于一身的基础框架，按照框架规范，可快速生成一套演示Demo系统，稍微改造即可变成一套可交付实施的软件系统。

## 技术架构

本平台的核心采用JFinal+LayUI+MySQL的组合，轻量高效，扩展性强，适合大多数据业务场景。架构方面，平台采用前后端分离的方式，将所有的业务功能全部以标准接口的方式对外提供。

### JFinal

JFinal 是基于Java 语言的极速 web 开发框架，其核心设计目标是开发迅速、代码量少、学习简单、功能强大、轻量级、易扩展、Restful。在拥有Java语言所有优势的同时再拥有ruby、python等动态语言的开发效率。官方网站：https://jfinal.com/doc

### LayUI

layui，是一款采用自身模块规范编写的前端 UI 框架，遵循原生 HTML/CSS/JS 的书写与组织形式，门槛极低，拿来即用。官方网站：https://www.layui.com/doc/

本项目使用了基于LayUI的第三方封装框架layuimini，它是一套功能完善全功能后台UI框架，结合本项目自身对业务操作层的封装，减少了开发者95%的前端代码量，使用代码生成器的代码也基本不需要调整JS。官方网站：http://layuimini.99php.cn/， 本项目使用的iframe版本，即主页面加iframe框架的组合：http://layuimini.99php.cn/iframe/v2/index.html

### 数据库MySQL

本项目设计阶段使用的是MySQL数据库，可以无缝扩展到其它常见关系型数据库，如Oracle、SQLServer等。系统提供基于Excel的数据表生成工具，要求数据表名和字段均采用下划线的方式进行命名。https://dev.mysql.com/downloads/mysql/

### RBAC基于角色的权限控制

RBAC是主流的权限控制体系，以角色为中介主体进行权限控制：角色分配多个菜单，限制操作范围；用户分配多个角色，进行操作限制。本平台提供精确到操作的简化实现，提出“行为”的概念，即将需要控制的操作URL做为“行为”进行统一管理，后台在监听到有相关请求进，对用户权限进行判定，看是否有对应的权限分配；为方便操作，目前将“行为”绑定在菜单上，角色分配了菜单就有了相应的“行为”权限，不需要再进行二次分配；后期可根据实际需要进行细分，现平台只提供简单的样例，详见SecurityUtil。

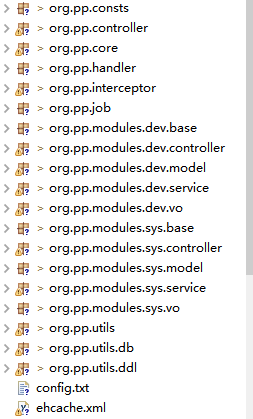
## 适用场景

1. 需求调研过程中，快速生成原型系统
2. 个人开发者快速交付中小型项目
3. 初次接触java项目的团队
4. 其它没有要求技术路线的中小型项目
5. 纯后台接口型项目
6. 自己运营型项目

# 代码说明

## Java代码

### 目录结构



* 常量：consts
* 核心封装:core
* 拦截器：interceptor
* Handler：handler
* 公共功能模块:controller
* 定时任务:job
* 工具类:utils
* 子模块代码包

Modules下为各业务模块（子系统）的子包，每个业务代码包必备base(Model基类),controller（接口控制器）,model（模型）,service（业务层）这四个子包

### 核心类 org.pp.core

1. BaseController 控制器基类
2. BaseService 通用的业务层操作，实现了增删查改的基本逻辑
3. ExcelService导入导出的抽象模板，要在service中实现自己写入和读取的实现
4. AdminController标准控制器，封装了分页查询，保存，更新，删除，全部查询，单条数据查询、导入、导出的标准实现

### 工具类 org.pp.utils

1. ExcelUtil Excel文件读写操作工具
2. CodeUtil 页面代码生成器
3. JFinalUtil Java代码生成器
4. ModelUtil 模型数据操作工具
5. XDataUtil 通用数据操作工具
6. SecurityUtil 安全认证工具类
7. FileUtil 文件操作工具
8. ImportDb.java从Excel中导入生成数据表
9. ImportDict 从Excel中导入生成数据字典

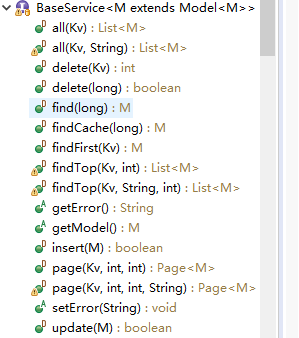
### 公共模块 org.pp.controller|handler|interceptor

1. IndexController 入口控制器
2. PublicController 登录、退出、验证码控制器
3. UploadController 文件上传控制器，需要用户先登录
4. HtmlController 后台页面输出控制器
5. HtmlHandler 处理后台静态页面，主要是表单和列表页面，请求地址必须以.html结尾，该Handler会根据地址找到真实的模板文件并渲染
6. ExceptionInterceptor 全局异常捕捉拦截器，并负责将当前登录用户注入SecutiryUtil，优先从Session中获取用户信息，如果没有，从请求的头信息中寻找xtoken，如果存在，则根据令牌查询用户信息。
7. BackendInterceptor 后台权限认证拦截器：首先判断用户是否登录，如果未登录则要求用户登录：如果已经登录，则判断用户对当前资源是否有访问权限，没有则提示用户

## 后端代码

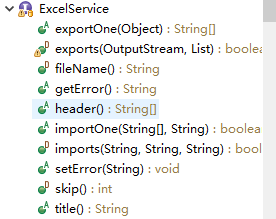
### 基础服务层接口BaseService

* All方法 查询条件，可以指定排序字段
* delete方法 可按主键删除或指定的条件删除
* Find方法 按主键查询一要数据
* FindCache 按主键从缓存中查询一条数据（没有则查数据库），配置cachename为表名
* findFirst 根据指定的条件查询第一条数据
* FindTop 根据指定的条件，返回指定数量的条数，还可以指定排序规则
* Insert保存新数据
* Update 更新数据
* Page分页查询，可指定排序条件
* GetModel 返回当前Service对应的Model，需要实现类提供



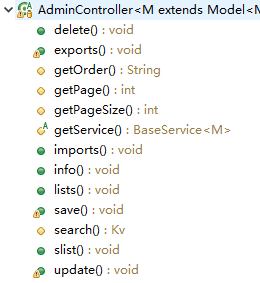
### ExcelService 导入导出Excel标准化封装

* ExportOne 导出一条数据为Excel文件中的一行文本，一般导出只需要实现该方法即可
* Exports 导出数据到的指定的输出流
* FileName 导出的文件名
* Header 导出文件的表头
* importOne Excel文件中的一行文本数据如果入库，一般导入中需要实现该方法即可
* Imports 第一个参数是Excel的路径，第二是参数是遇到错误的处理方法，第三是参数是重复数据的处理方式
* Skip是读取Excel文件是路过的表头行数
* Title是导出的Excel中的工作表名称



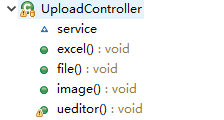
### AdminController 后台控制器基类

* gerOrder 获取当前查询的排序字段，默认是id desc
* getPage 获取当前页码
* GetPageSize 获取分页限制的条数
* GetService 返回当前操作的服务器，需要控制器实现
* Info 返回指定数据的详细信息
* Lists分页查询
* slist查询全部
* Save 保存新数据
* Update 更新数据
* Delete 删除数据，默认是按主键进行删除，多个用，隔开
* Search 生成本业务对应数据支持的查询条件，需要控制器实现
* Exports 按条件导出数据
* Imports 导入数据



### UploadController 文件上传控制器

* Excel 接收上传的excel待导入文件，不保存入库，后期由Excel任务删除
* file是接收任何文件的上传并入库
* image是接收图片的上传，后期可自定义图片处理
* ueditor是接入了UEditor编辑器，支持编辑器内的文件上传



## 前端代码

前端的底层是LayUI，layuimimi做了框架层面的标准封装，提供了整体的页面框架结构：一个主页面，含头部应用导航和左侧菜单导航；一个基本Tab的多标签页面操作区，一个标签页面对应一个iframe。

### Curd封装

Curd（statics/js/lay-module/curd/curd.js）是本平台对常见页面操作流程的封装,主体是生成一个数据表格，并对表格工具栏和数据操作按钮进行了统一的业务封装。标准事件如下：

1. view 打开查看窗口
2. Frame 打开新标签页面
3. Form 打开表单窗口
4. Ajax 发起动态请求
5. Open 在新窗口打开新页面
6. Js 调用js方法，方法名为当前元素的fun属性

**操作控制需要定义url属性（除js事件），指定要操作的目标；系统支持携带选中对象(select)和搜索表单(search)数据做为动态参数，方法就是在事件后边加个扩展，如ajax\_select是会把所有选中行的id追加到url进行操作地址，open\_form则会在要打开的页面地址上追加搜索表单中的数据。**

### 列表页面

列表页面分普通列表和树形列表，布局使用list.html，只需要定义各列的基本属性和接口信息即可；列表页面加载filter.html做为搜索条件；列表页面里可以根据curd的规则添加操作按钮，可实现大部分的业务需要，如果不满足也可以自己写事件进行处理。

### 表单页面

新增和编辑采用相同的form页面，布局使用add.html或者edit.html，在这个页面里可以使用layui的各种组件，平台增加了动态数据和二级联动的select组件扩展。在编辑页面，会调用info接口获取数据并填充到表单中，如果页面复杂，需要自定义编码。

### 报表页面

图表页面是代码生成器生成的报表样例，核心是使用echartHelper扩展，调用平台标准数据接口，生成图表或简单列表；其它地方也可以调用组件生成报表；报表页面的名称就是报表代码。

## 代码生成器

系统中的代码分三步走：

1. 进行整体设计，按照Excel模板形成数据库设计文件，使用ImportDb导入数据库，生成表结构的字段信息，在导入的过程中会生成一定的数据验证规则，如果不够或者根设计习惯有偏差的，可以后期在后台调整规则或者先调整导入代码。
2. 使用JFinalUtil按模块生成业务代码或重启开发环境，进入后台“模块管理”，生成模块代码,并自动生成模型、字段、验证等信息;核心是JFinalUtil.genModule方法；在org.pp.cord.Config中加入\_MappingKit和路由信息Routes
3. 重启开发环境，进入后台“模型管理”，选择模型，生成列表和表单页面，并根据需求进行调整；在这之前 ，还可以调整字段信息；验证信息可以随时调整；模型发生变化时，先调整数据结构，然后在本功能中刷新表结构或手工增加字段，然后重新生成代码即可（重新生成代码需要删除之前的代码，少量情况下也可手工调整页面代码）。

## 权限体系详解

### 精于资源（URL请求）的最小权限控制

简单来说，系统中的权限控制可以精确到每一个具体的URL请求，在系统中称之为“行为”，即控制器中的一个方法；如果需要对数据进行权限控制，需要在自己的业务代码中去实现，不在本权限体系的考虑范围内。

要想通过本系统的权限体系管控一个权限，首先要把该权限添加到系统“行为”库中，这个在生成模型的代码时可以勾选生成默认的全部行为，也可以后期在控制器中添加方法后手工添加到“行为”库中。不在行为库中的权限（URL请求）是不受管控的，默认对外开放。如果是有BackendInterceptor拦截，则会要求用户登录，否则对外完全开放，任何人都可以访问。

权限控制的核心代码在SecurityUtil中，在初始化时，会将全部的权限加载进来做为键，然后把每个权限已经授权的角色做一个Set形成值，在系统中一起做为一个Map<String,Set<Integer>>，后期可以很方便的调用,通过url找到可以访问该行为的角色列表，然后再判断当前用户所拥有的角色是否在角色列表中，如果在就可以访问，不在就不能访问。

### 权限分配的流程和思路

权限和用户并不是直接绑定的，中间还有角色和菜单。为了方便权限分配的操作，首先菜单和权限绑定，这是从普通的后台应用操作层面出发的，你能看到菜单就应该拥有该菜单对应页面里的全部操作权限。然后角色分配菜单，自然就关联上了菜单里的资源。最终用户分配角色，形成最终的权限体系。

默认情况下，一个业务模型的功能包括一个列表页面、一个添加页面、一个编辑页面：

列表页面是入口，该页面是一个静态页面（HtmlHandler控制），目前没有加入权限体系（也可以在HtmlHandler中添加访问控制代码，前提是把页面访问地址加入到行为库），只要登录了就可以访问，但是页面里要展示数据就要访问数据接口，如分页查询接口，删除数据接口，也包括导入、导出功能。这些默认都是自动进入行为库受保护的，所以如果没有分配菜单权限，即使通过直接输入页面地址能看到页面也不能看到数据，打开的是带基础框架的空页面；而分配了菜单权限，就自动拥有了该菜单关联的行为操作，就可以访问相就的接口，取得数据。如果在页面里需要定义标准操作之外自己的业务功能，就需要把这个业务功能的行为添加到行为库，并在菜单管理中进行关联。

### 权限使用示例-控制静态页面

上边讲到“可以在HtmlHandler中添加访问控制代码，前提是把页面访问地址加入到行为库”，那么具体如何操作呢？我们以“用户管理”列表页面为例，它的URL为/sys/user/list.html：

* 首先要把这个地址添加到行为库

Graphical user interface, application

Description automatically generated

* 修改HtmlHandler，增加控制判断

List<Module> list = ModuleService.*dao*.findAll();

**for**(Module m : list) {

**if**(target.startsWith("/"+m.getCode().toLowerCase()+"/") && target.endsWith(".html")) {

**if**(SecurityUtil.*checkAccess*(target)) {

request.setAttribute("\_target", target);

next.handle("/html/", request, response, isHandled);

**return**;

}**else** {

//自定义错误处理页面

next.handle("/noauth", request, response, isHandled);

**return**;

}

}

}

### 权限使用示例-第三方应用（含移动端）接入

第三方应用接入一般采用全接口的方式，即开放指定的接口给他们用，以便获取或者操作系统中的数据。这里的接口有些是需要安全认证的，有些是不需要安全认证的，就是完全开放不受任何限制的给所有人都可以调用。

对于需要安全认证的接口，就像我们后台调用一样，用户首先登录，然后去调用即可，后台要判断用户的权限是否可以访问某些接口（行为）。但是接口调用一个是无状态保持的，所以需要通过令牌的机制，登录时调用/token接口，返回的数据有4个字段：用户ID，账号account,昵称nickname和令牌token,令牌的默认有效期是24小时，这里可以在代码中修改或者放到配置中，如果令牌过期就需要重新登录以换取新的令牌。调用接口时需要在头信息中增加xtoken参数，值为登录时获取令牌，这样后台就可以取得已经登录的用户信息。

不需要安全认证的接口，需要在接口的代码层面做一些处理：首先是清除权限控制的拦截器Clear(BackendInterceptor.class)，这个代码可以放在开放接口的控制器上或者接口方法上，如果是控制器上，则整个控制器开放，如果是某个接口上，则只有这个接口开放。对于系统默认的接口如lists等，则需要在自己的控制器上覆盖Override默认的接口，方法体可以只有一行：super.lists();然后将拦截器Clear即可。

@Clear(BackendInterceptor.class)

@Override

public void lists(){

super.lists();

}

**前端调用可参考statics/page/api.html,直接用浏览器打开，不通过http请求。**

# 使用手册

## 安装部署

### 开发工具

首先安装JDK1.8以上和eclipse较新的版本，workspace编码设置为utf-8

### 数据库

安装MySQL5.5以上版本，支持InnoDB引擎，导入jfast.sql脚本。如果是第一次安装，建议使用mysql的免安装压缩包；安装一个mysql客户端，建议使用navicat。

### 启动项目

启动eclipse,导入现有工程即可。也可先在.project文件中修改项目名称。找到src目录下的config.txt按实际情况修改；然后找到org.pp.core.Config类，右键Debug As Java Application即可，默认启动端口为8000。

### 发布项目

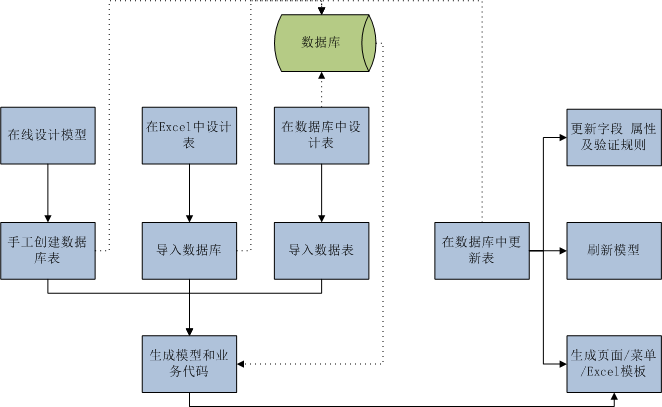
项目的输入文件都在WebRoot中，如果需要将项目发布到Tomcat中运行，只需要将WebRoot目录里的内容全部复制到tomcat下的项目目录，并修改web.xml即可。

## 设计思路

### 模块化设计

采用模块化的设计，每个模块相当于一个子应用，每个模块的数据库表采用相同的前缀，并与其它模块不同。每个模块下有多个模型，每个模型对应一个业务。

### 流程架构



## 模型设计

### 模型

每个模型对应一张数据表，一个模型可以与多个其它模型有关联关系。模型有三种创建方式：

1. 批量设计，根据excel数据库模板设计模块里的全部表，然后通过org.pp.util.ddl.ImportDb将设计好的表导入到数据库中，这时会自动生成模型、字段和验证信息。 一般在系统开发前期使用，大部分模型和数据表通过该功能进入系统。
2. 在线设计，在“模型管理”中“添加”模型，在“字段管理”中“添加”其字段，在“数据验证”中“添加”验证规则即可。一般设计数据模型或小字段的极简模型，不建议大规模使用。
3. 临时设计，直接在数据中设计数据表，然后通过“模型管理”中“导入表”自动生成模型和字段信息。一般用于复制其它系统的功能模块时使用，但还是建议通过方法1进行大量模型的导入。

### 字段

导入模型的时候可自动关联生成字段信息，导入后可对字段进行调整，主要是数据（库）类型和字段类型（业务），还有长度及默认值，其中默认值可支持取系统参数，如now表示当前日期，session.user.id表示当前用户ID,rand:16表示16位随机数。这些在org.pp.modules.dev.model.Field中定义，后期也可按业务需要进行扩展。还可以指定该字段是否可以为空值。

### 验证规则

验证规则有多种类型，如长度、唯一性、数字范围、枚举和正则等，需要配合规则参数一起实现，具体参照org.pp.modules.dev.model.Validation。在config.txt中定义了一些基于正则表达式的验证规则，后期也可以调整和扩展。

## 代码生成

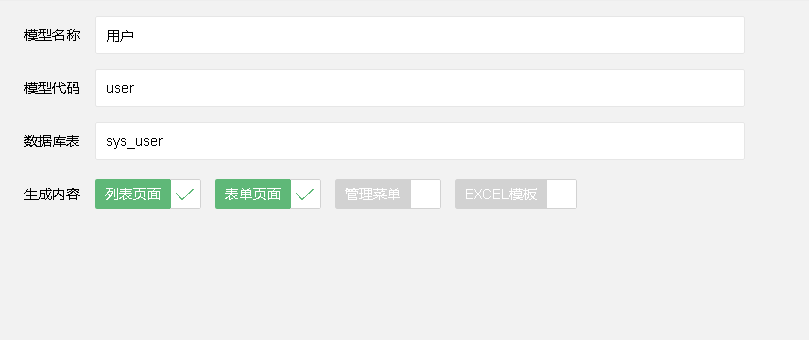
### 业务代码

业务代码，主要是Model,Service,Controller及相关的Route,ActiveRecordMapping。系统提供工具org.pp.utils.JFinalUtil，可自动批量生成：

1. 按模块批量生成genModule，需要传入参数为对应的业务模块数据的前缀，如sys,将会为库中所有sys\_开头的表创建模型，并生成相应代码，如果不需要，可以后期删除
2. 生成指定的表genTable，需要输入参数为数据库表全称，系统会自动生成Model,Service,Controller，但是Route,ActiveRecordMapping相关信息需要手工添加

### 页面代码

页面代码没有自动生成是考虑到自动生成的模型字段可能根实际需求有偏差，需要调整，所以需要人工触发一下。“模型管理”中“生成代码”，选择要生成的内容点击“确定”即可。生成的页面代码可根据需求调整。（view/模块代码/模型代码小写/）



### 业务变动

如果数据字段发生了变动，可以将模型和代码删除掉重新生成，也可以通过增量修改的方式进行：

* 修改数据库表结构
* “模型管理”中“刷新表结构”，更新字段信息，人工核对，删除不需要的字段信息
* 人工修改BaseModel
* 修改页面代码或删除页面代码，重新生成

## 统计报表设计

图形报表基于百度ECharts开发，默认可以呈现折线图、柱状图、饼图和仪表盘四种图形，支持标准属性；查询列表基于LayUI表格控件。只需要按标准创建SQL查询，形成数据集，再设计报表属性即可。

### 数据集

数据集就是一个标准的SQL查询，可以当做统计报表的数据源，也可以直接做为数据的查询监测手段，数据集的核心是一条查询SQL，支持JFinal的SQL模板：

select count(u.id) value,d.title title

from sys\_user u

left join sys\_department d on u.depart\_id=d.id

group by depart\_id

#for(x : cond)

#(for.first ? "WHERE": "AND") #(x.key) #para(x.value)

#end

### 报表管理

设计报表时，设置代码、选择数据集和报表类型即可，创建完成后可生成代码文件（veiw/dev/chart/code.html）,后期可对页面进行修改，包括添加查询条件，增加ECharts参数等，调用报表页面地址为/dev/chart/view?id=?或/dev/chart/view?code=?，数据集需要的数据如下：

* 普通折线图：X轴=title，Y轴=value
* 多条拆线图：X轴=title，Y轴=value，分组=category
* 普通柱状图：X轴=title，Y轴=value
* 多条柱状图：X轴=title，Y轴=value，分组=category
* 饼图:块名=title，值=value
* 仪表盘：显示名称=title，值=value