# 一、新框架简介

基础框架: springboot

数据库连接池: HikariCP

数据库元数据管理(表结构管理): Liquibase, SchemacCrawler

ORM: Hibernate, SpringData JPA, QueryDSL, Spring JdbcTemplate

工作流引擎: Flowable

批处理框架: Spring Batch

计划任务调度框架: Spring, Quartz

页面模板引擎: Thymeleaf

前端 JS 框架: Bootstrap, Smartclient, Echars

文档编写工具: Asciidoc (Asciidoctor)

权限管理框架: Spring Secuirty

WebService 框架: CXF

开发环境: Eclipse, Git, Gradle, Sonatype Nexus, Docker

# 二、环境搭建

## 2.1、开发环境软件

开发工具：eclipse oxygen 3a

构建工具：gradle 4.8.1

开发环境：JDK1.8

代码管理工具：GIT

**注意：**eclipse不要安装插件

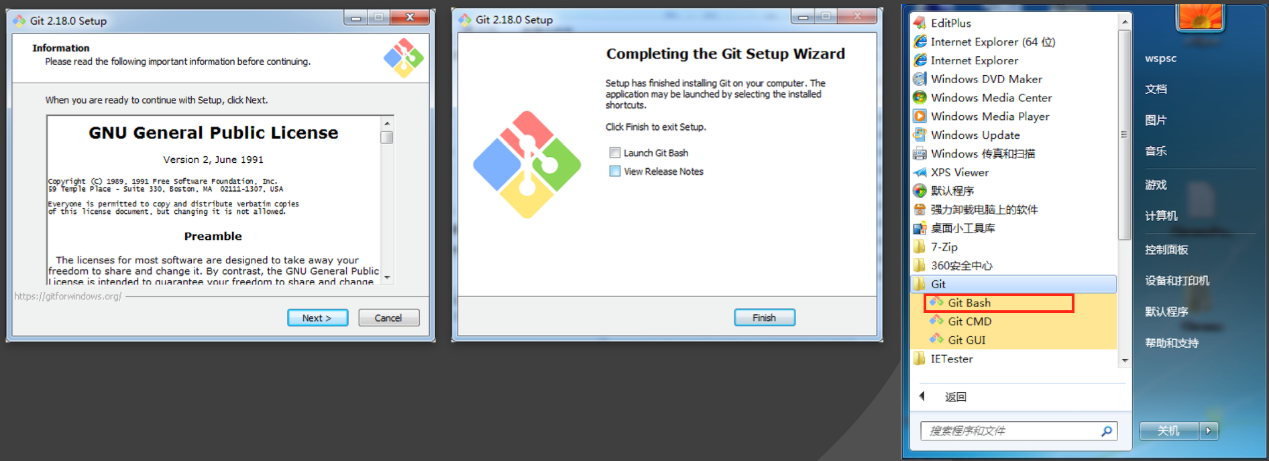
## 2.2、GIT

### 2.2.1、安装 git 源代码版本管理软件客户端

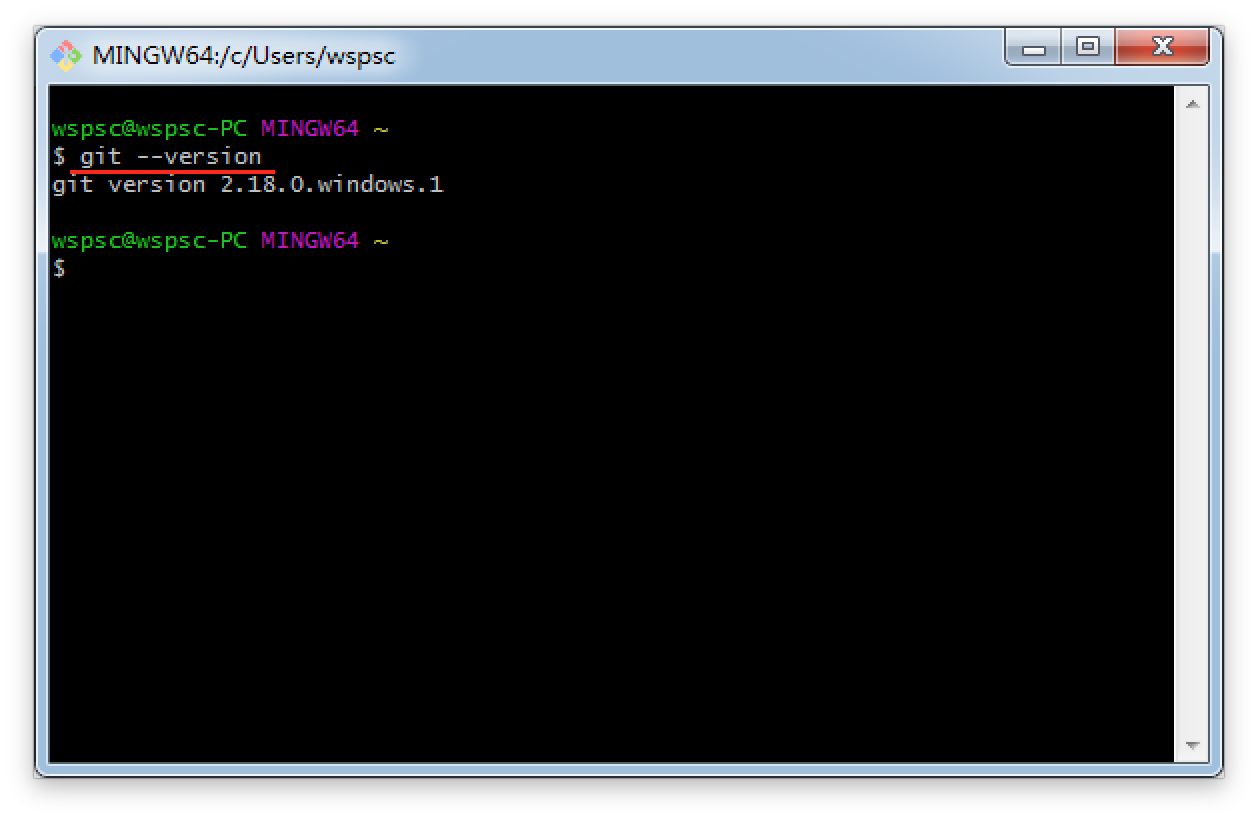
从 https://git-scm.com/download/win 下载 git for windows 软件



启动上述下载的安装程序，开始安装，在安装过程中所有选项均为默认值，直到安装完毕。



启动 git bash,执行 git --version 命令查看是否安装成功



**注意**: 此 git bash 终端不同于 windows 自带的 cmd 控制台, 该终端是 MinGW (inimalist GNU for Windows),其命令非常类似 linux,

如未特殊说明,以下所有命令操作均基于此终端进行。

## 2.3、gradle

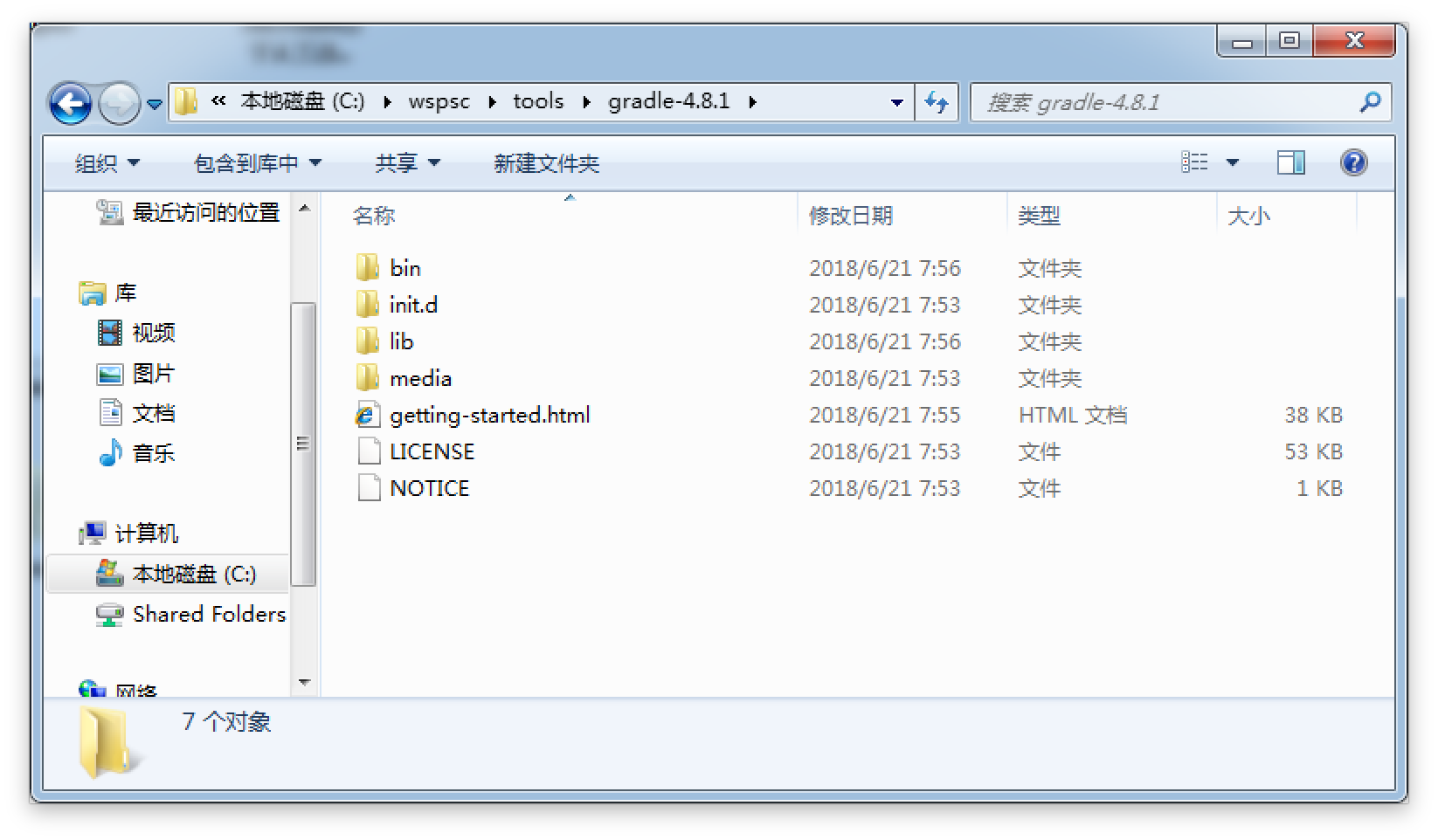
**注意：**千万不要给eclipse安装gradle插件

### 2.3.1、安装 gradle 构建工具

从 https://gradle.org/releases 下载 gradle 软件



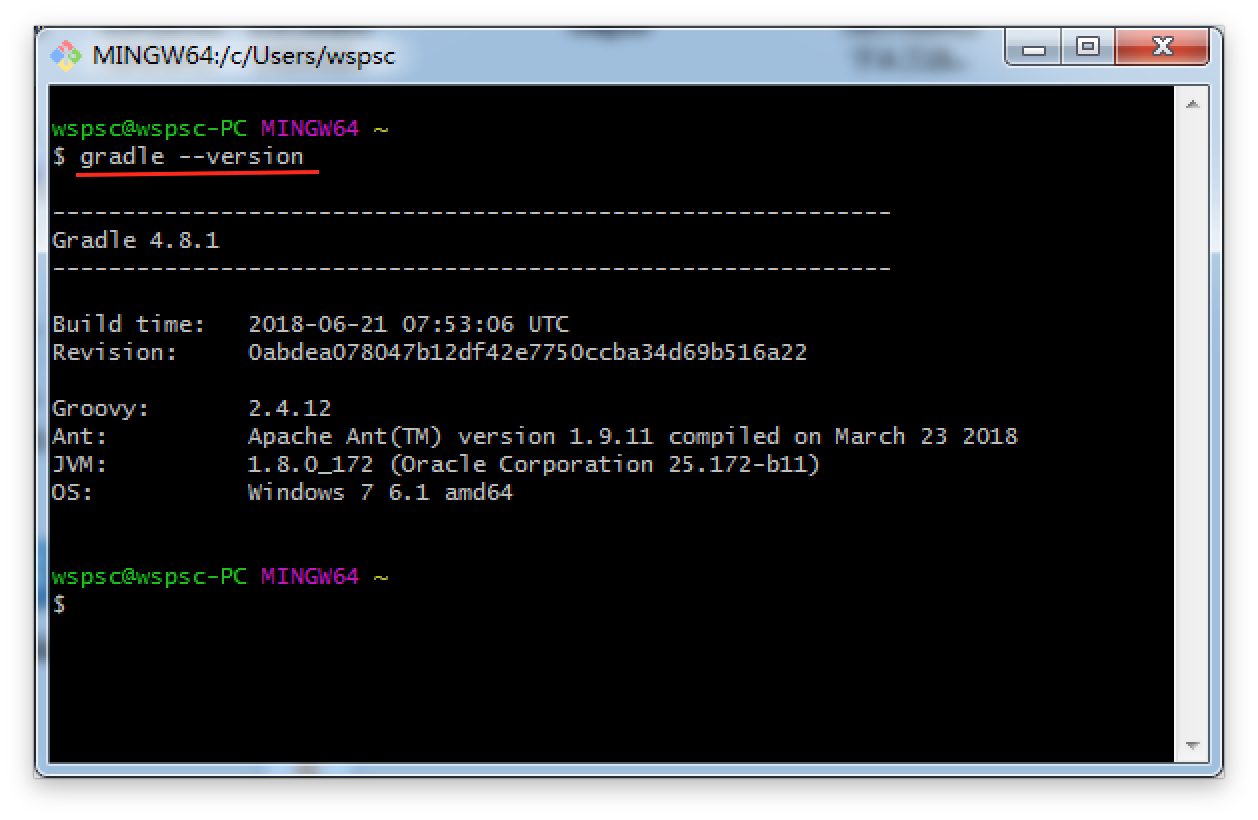
解压缩上面下载的 gradle-4.8.1-bin.zip 文件,例如解压到 c:\wspsc\tools\gradle-4.8.1 目录



添加 PATH 环境变量



测试安装是否成功



### 2.3.2、gradle修改本地库地址

此修改地址可不修改，根据个人喜好。

修改bin目录中的gradle文件，大概在31行。默认情况下是空的，会放在C盘中。需要添加如下参数来修改



DEFAULT\_JVM\_OPTS="-Dgradle.user.home=D:/gradle-4.8.1"

更改红色框中的路径为gradle的根路径。

## 2.4、搭建开发环境

从SVN上面下载代码，放到eclipse工作区间。

生成 eclipse 项目

# 进入项目目录（根据自己存放的目录进入）

cd /c/wspsc/workspace/application

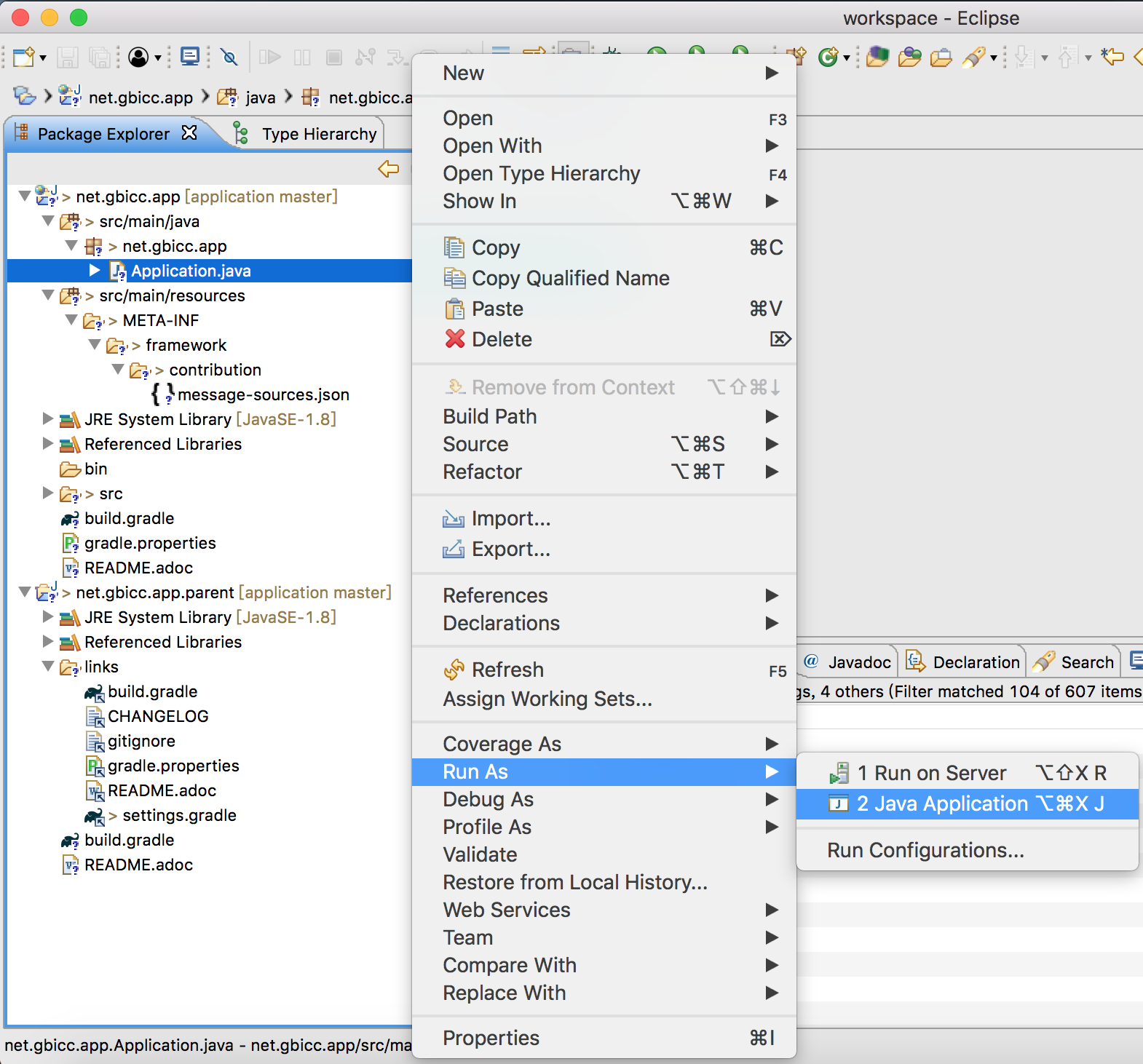
# 生成 eclipse 工程

gradle eclipse

将所有工程导入到 eclipse 中

启动项目启动入口文件：

选中项目 run as—》application



# 三、模块项目的开发

添加模块有俩种方法：

直接在项目根路径下使用命令进行模块的创建，命令如下：

**gradle createModule --name=myProjectName**

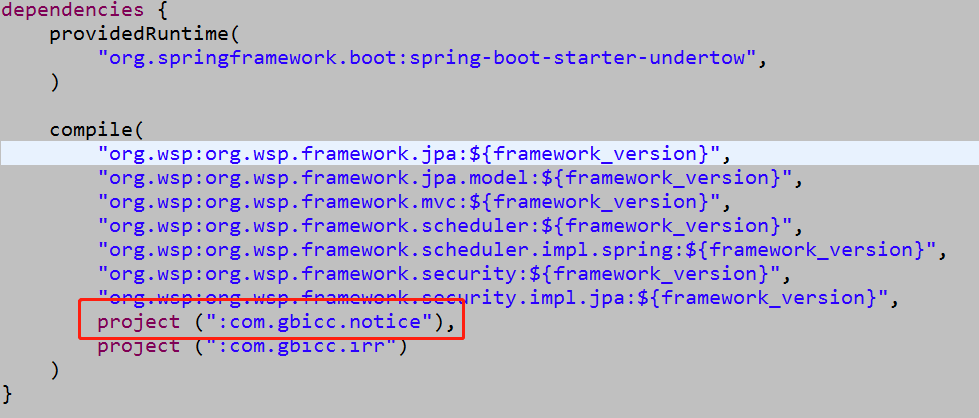
命令创建后，模块的结构就自动生成。并且项目也已经整合好。

**注意：**使用命令构建好子项目后，要在**启动模块下的build.gradle**添加配置。

## 3.1、整合模块

创建好模块之后，需要在俩个配置文件中配置上模块进行整合。

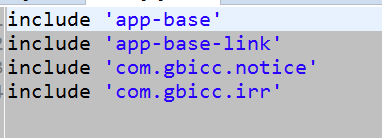
### 3.1.1、配置启动模块下的build.gradle



如上图所示，需要在dependencies 配置项的compile中配置上模块。

**注意：project(“:com.gbicc.notice”)**，括号中的也可以使用单引号(’’)，特别要注意配置的开头有一个冒号（：）。

### 3.1.2、配置项目根路径下的settings.gradle



如图所示，需要直接配置，例如 **include ‘com.gbicc.notice’**

### 3.1.3、构建项目

当配置完俩个配置文件后，需要使用 GIT BASH 在项目的**根路径**下执行

gradle eclipse，对项目进行构建。

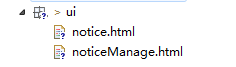
### 3.1.4、引入添加模块后的项目

模块构建完成后，需要导入添加的模块，然后刷新项目。

**注意：**有的时候，刷新项目、导入添加模块的方法会不好用，如果无效，就在eclipse中删除项目，千万不要删除硬盘中的项目，然后再重新导入项目就好了。

## 3.2、界面

在src/main/resources/ui/中创建一个html文件。



### Html文件的结构

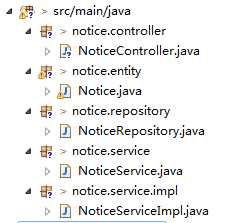


<html th:replace=*"template/smartclient/smartclient.html::html"*>

<title th:text=*"#{application.title}"*></title>

**注意：**上面的html、title标签的写法。

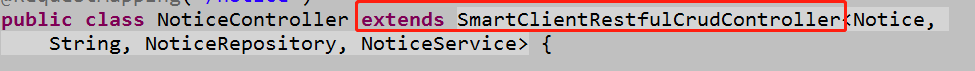
## 3.3、JAVA代码



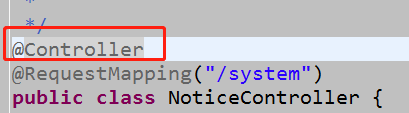
### 3.3.1、controller层

此层的类**必须**extends类SmartClientRestfulCrudController<T,PK,Dao,Service>

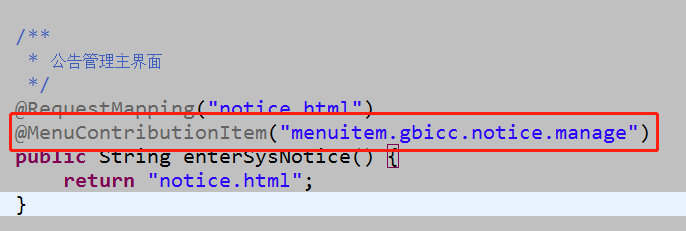
如图：



**注意：**controller类中的注解**必须要使用@controller**。如图：

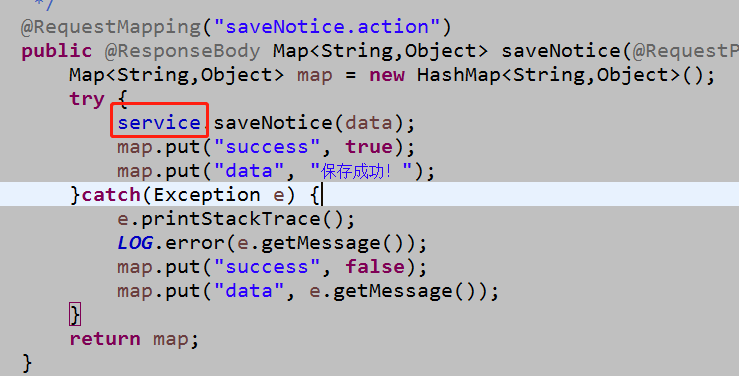


配置菜单跳转URL，如图：

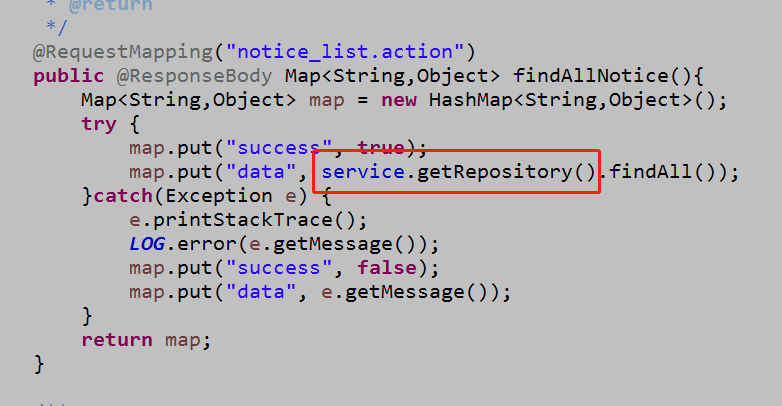


使用注解**@MenuContributionItem**，值为在menu.json配置的菜单**ID**。

Controller中调用相应的service，**无须注入**，只需要写service.。。。就可使用，service调用持久层Respository也是一样的。如图：

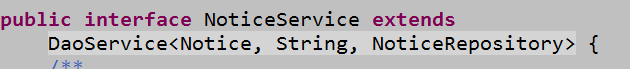


也可以通过service调用持久层，如图：



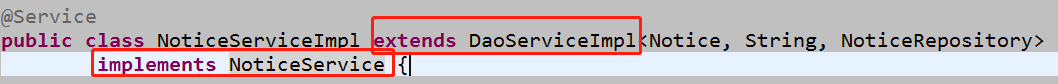
**注意：**持久层中基本的增删改查已经实现，无须再写。

### 3.3.2、service接口层

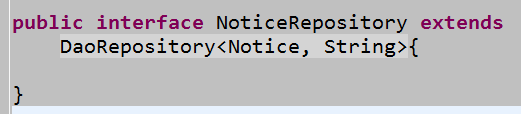
需要继承DaoService<T,PK, Respository >，如图： 

在此类可以写未实现的方法。

### 3.3.3、service实现层

此类需要继承DaoServiceImpl<T,PK,Respository>和实现相应的service接口类。如图：

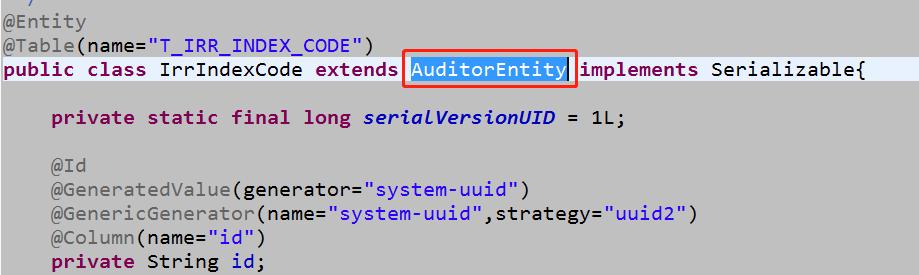
### 3.3.4、respository接口层

此处需要继承DaoRepository<T,PK>,如图：

此处可以写一些Dao的未实现类

### 3.3.5、entity实体层

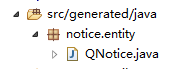
此处需要继承AuditorEntity并实现Serializable，如图：



**注意：实体属性名称不要用\_（下划线）,因为后面此符号会有特殊意义。**

**UUID2为36位的，UUID为32位的。**

### 3.3.6、构建查询对象QEntity

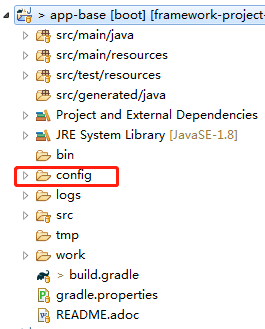
当上述的JAVA文件结构已经全部完毕，需要使用GIT BASH在项目根路径下执行**gradle querydsl** 来自动创建实体entity相对应的QEntity。创建后的目录结构为：src/generated/java…如图：

**注意：此步骤非常关键，一定要执行命令，创建此对象**

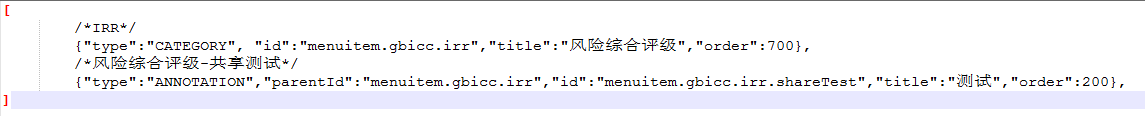
## 3.4、配置菜单

本框架中的菜单是文件、代码配置的，无需在系统中单独配置。

**注意**：此配置有一个BUG，要**删除config文件夹下的.installed**，配置完后需要重新安装，重新安装的时候注意**选择第三个**，删掉数据。



### 3.4.1、配置menu.json文件



type：分为CATEGORY、ANNOTATION。CATEGORY为根菜单，ANNOTATION为controller注解的类型菜单，具有页面内容的菜单。

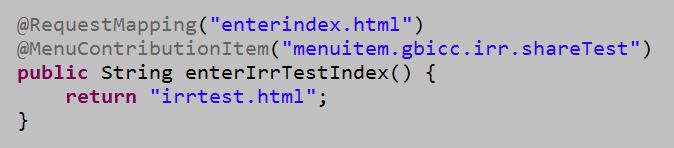
Id：菜单链接唯一的字符串。

title：菜单的名称（也可以使用i18n国际编码）

order：排序

parentId：父级菜单ID

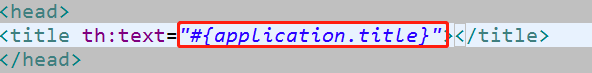
### 3.4.2、controller进入界面方法



@MenuContributionItem：配置menu.json中的type为ANNOTATION的id。

## 3.5、前端页面

### 3.5.1、titile设置



整个项目的title设置是在全局的国际化里面配置的。

找到启动类module下的

**src\main\resources\启动module\i18n\**

messages\_zh\_CN.properties：中文



messages.properties：英文



### 3.5.2、首页面设置

只需要修改application.properties中的security.http.formLogin.loginPage参数即可，比如：



# 四、spring-data-jpa

## 4.1、JPA简洁

JPA是Java Persistence API的简称，中文名Java持久层API，是JDK 5.0注解或XML描述对象－关系表的映射关系，并将运行期的实体对象持久化到数据库中。 [1]

Sun引入新的JPA ORM规范出于两个原因：其一，简化现有Java EE和Java SE应用开发工作；其二，Sun希望整合ORM技术，实现天下归一。

## 4.2、spring-data-jpa简介

SpringData JPA 是Spring提供的持久层的解决方案

SpringData JPA其实就是Spring对JPA操作的封装(entiyManager)

SpringData JPA 是对JPA操作的封装 ,如果要使用SpringDataJPA有两个条件:

1、pring整合JPA

2、需要有一个JPA的实现框架,例如Hibernate

spring-data-jpa会根据方法的名字来自动生成sql语句，我们只需要按照方法定义的规则即可。

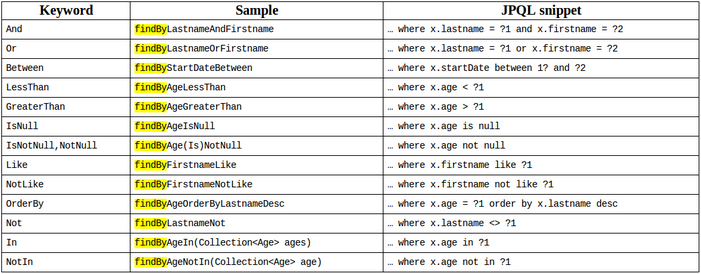
## 4.3、单表查询

repository层接口类，在对单表进行查询的时候。方法名需要符合的结构为：

单字段查询：findBy+表实体属性名（首写字母大写）。如：findByName

多字段查询：findBy+表实体属性名+And+表实体属性名+…。(注意首字母大写)

如：findByNameAndPassword



## 4.4、多表查询

多表查询时repository层接口类中表的关联要使用**\_(下划线)**，这就是在实体类中不要对属性名使用下划线的原因。

# 五、系统管理

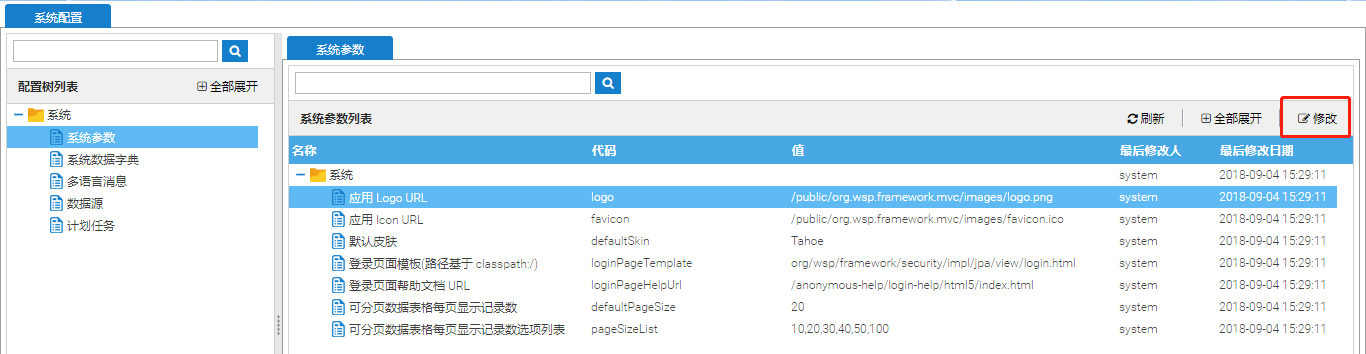
## 5.1、系统配置

系统配置中的配置树列表，在系统刚开始初始化的时候，是任何人都看不到的。需要用初始化时的管理员账号（admin）登录配置给一个用户（例如：admin），才可以看到。这样做为了系统配置更加稳定，安全。

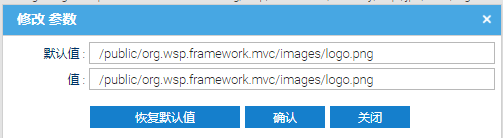
权限分配之后，在系统配置中左边的树行菜单如下：



### 5.1.1、系统参数



系统中只有修改系统参数的功能，选中一条记录，点击右上角的“修改”即可。

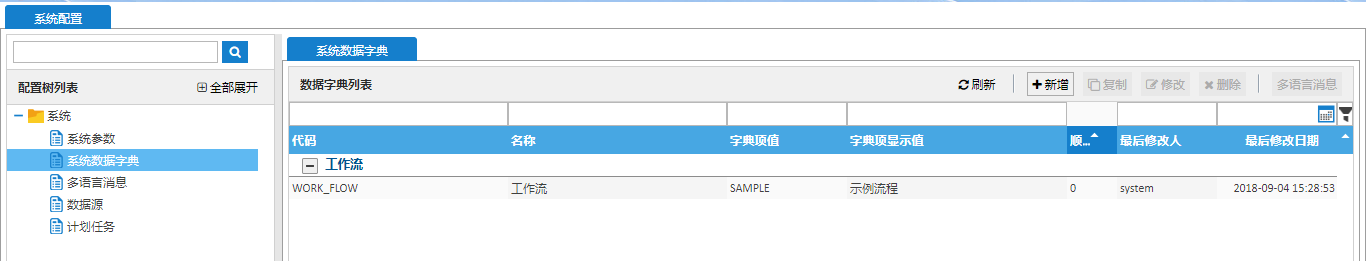


系统参数在修改的界面，可以恢复默认值。

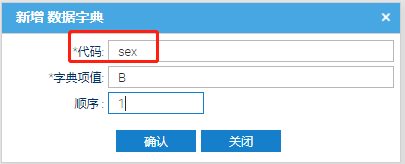
### 5.1.2、系统数据字典

对于系统中需要设置的一些数据字典，在此页面可以进行管理。当前系统的数据字典是**自动分类**的，只要按照相应的规则建立就好。

数据字典主界面如下：

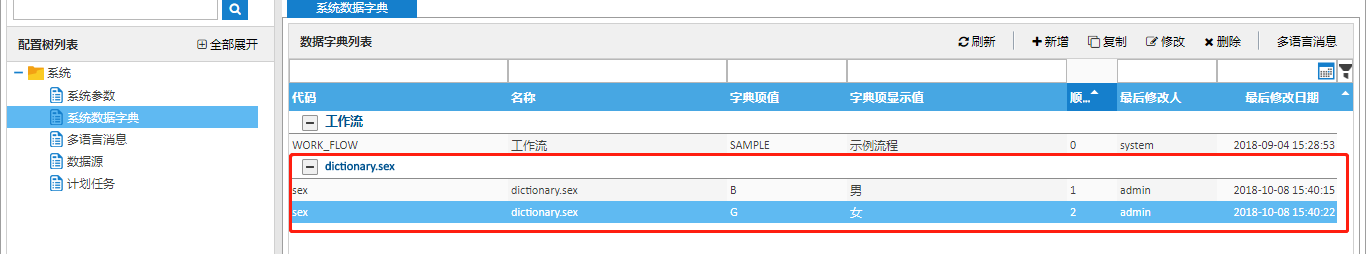


点击右上角新增按钮，进行数据字典的新增



**注意：**新增数据字典的时候，如果是同一组的话，“代码”必须一致，这样才会被自动划分为一组。

新增一组数据字典后，如下：



此数据字典中已经自动的显示字典项显示值已经是中文了，如果配置一个不能显示中文的，需要设置“多语言消息”。

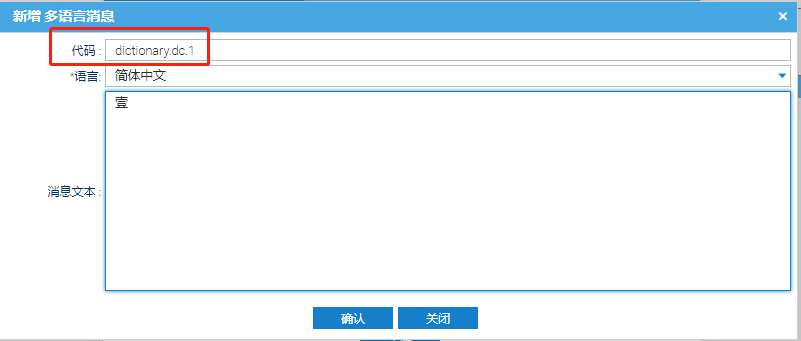
比如，下面新增的这个就没有显示出中文，需要设置多语言信息。



选中一条记录，点击右上角的“多语言消息”，弹出如下对话框



点击新增，弹出对话框，并填写

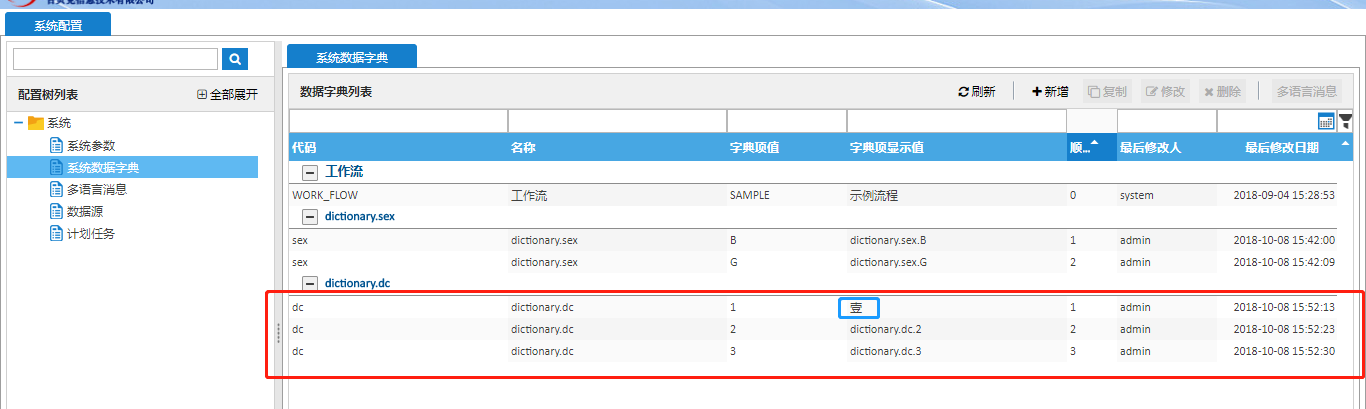


**注意**：代码这一项必须与“字典项显示值”显示的代码一致。

添加后选中添加的多语言消息，点击“立即生效”



数据字典的“字典项显示值”就变成你选择的多语言消息的文字了。如图：



新框架字典在前端使用下拉列表的例子

<select class="form-control select2" id="process" name="process">

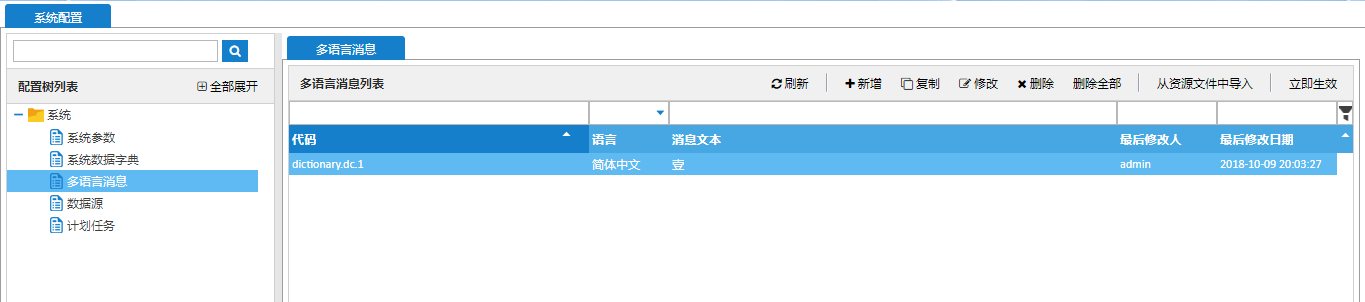
<option value=""> -- 请选择 -- </option>

<option th:each="option: ${@frSystemDictionaryService.getDictionaryMap('WORK\_FLOW',#locale)}" th:value="${option.key}" th:text="${option.value}"></option>

</select>

### 5.1.3、多语言消息

多语言消息是对当前系统的多语言消息进行管理。



例如刚才添加的一个多语言消息，在此界面就可以看到。

对多语言消息进行的管理操作，如右上角所示。