

[IntelliJ Idea 教程]

[涵盖安装、配置、常见问题&技巧、Maven、Git、Tomcat、快捷键、项目配置等]

[作者: Ricky]
[交流群: 244930845]
2017 年 6 月 1 日

目录

入门.....	5
安装步骤.....	5
目录说明.....	8
启动配置.....	9
配置空间.....	10
首次启动.....	11
首次配置.....	15
Appearance&Behavior (外观和行为)	16
KeyMap.....	20
Editor (编辑器)	20
Plugins (插件)	47
Version Control (版本控制)	47
Build Execution Deployment (构建执行部署)	48
Languages&Frameworks.....	52
Tools.....	52
第一次启动后.....	53
调出面板和按钮组.....	54
面板说明.....	54
项目配置.....	55
Project (项目)	55
Modules (模块)	55
Libraries (类库)	56
Facets (特征)	57
Artifacts (打包)	58
SDK (系统开发工具)	58
Global libraries (全局类库)	59
Problems (问题)	60
Maven 专题.....	60
配置.....	61
主配置.....	61
Import 配置.....	61
Ignore Files 配置.....	62
Runner 配置.....	63
RunnerTest.....	63
Repositories 配置.....	64
使用入门.....	65
面板说明.....	65
命令模式.....	68
Tomcat 专题.....	70
安装配置.....	70
启动.....	73
面板说明.....	73

Run with coverage.....	75
Tomcat 集成原理.....	76
Conf (配置)	77
Logs.....	78
Work.....	78
GIT 专题.....	79
安装.....	79
使用.....	79
拉取项目.....	80
更新项目.....	81
提交项目.....	81
面板说明.....	82
Local Changes.....	82
Shelf 面板.....	83
Log 面板.....	83
Console 面板.....	83
History 面板.....	84
项目 git 面板.....	84
仓库选项.....	84
SVN 专题.....	85
配置.....	85
1.1 下载&安装 svn.....	85
1.2 配置.....	85
检出项目.....	86
面板说明.....	92
工具栏面板.....	92
VersionControl (版本控制)	92
常用操作.....	95
加入版本控制.....	95
提交远程.....	96
更新项目.....	98
冲突解决.....	99
SSM 搭建.....	100
Maven 项目.....	101
Jar 包.....	105
添加 Spring 支持.....	106
数据库和 mybatis.....	107
其他 jar.....	108
配置文件.....	109
Web.xml.....	109
Spring-service.xml.....	110
Spring-mvc.xml.....	112
项目目录结构.....	113
说明.....	114

Demo 地址:	114
常用技巧&问题	114
创建自定义快捷列表	114
Tomcat 部署失败	115
情况 1	115
热部署注意事项	116
修改 JAVA 编译版本	116
UTF8BOM 格式转 utf8	117
插件安装(本地)	118
JSP 实时编译问题	119
搜索功能失效	120
自动导入依赖	120
提示不区分大小写	120
剪贴板数量设置	121
生成 JAVADOC	121
设置 Spring 支持	123
启动时不自动打开项目	123
取消注释检查	124
全屏设置	124
本地历史	125
搜索	125
所有文件	125
项目文件	126
取消重复代码提示	126
设置字符集	127
项目字符集	127
单个文件	128
Idea 优化配置	128
启动参数优化	129
插件优化	130
运行优化	130
Git 证书失效	130
取消更新	131
快捷键	131
Ctrl	131
Alt	133
Shift	133
Ctrl + Alt	134
Ctrl + Shift	134
Alt + Shift	136
Ctrl + Shift + Alt	136
其他	136

入门

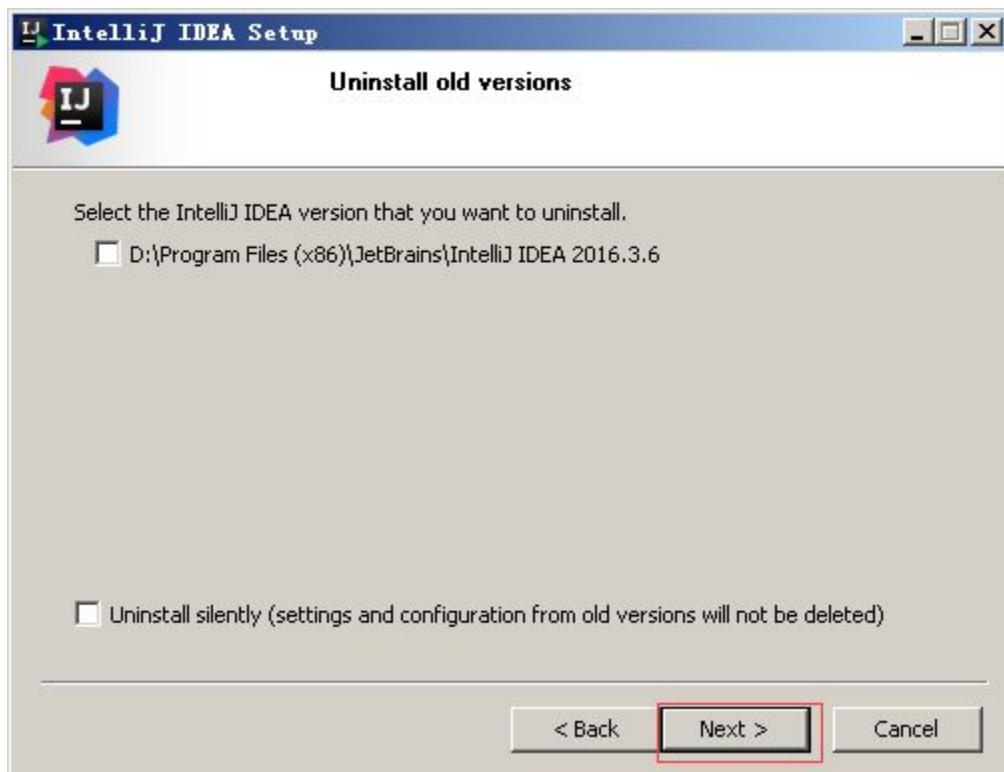
这里只做简单的入门，以及基础配置。

安装步骤

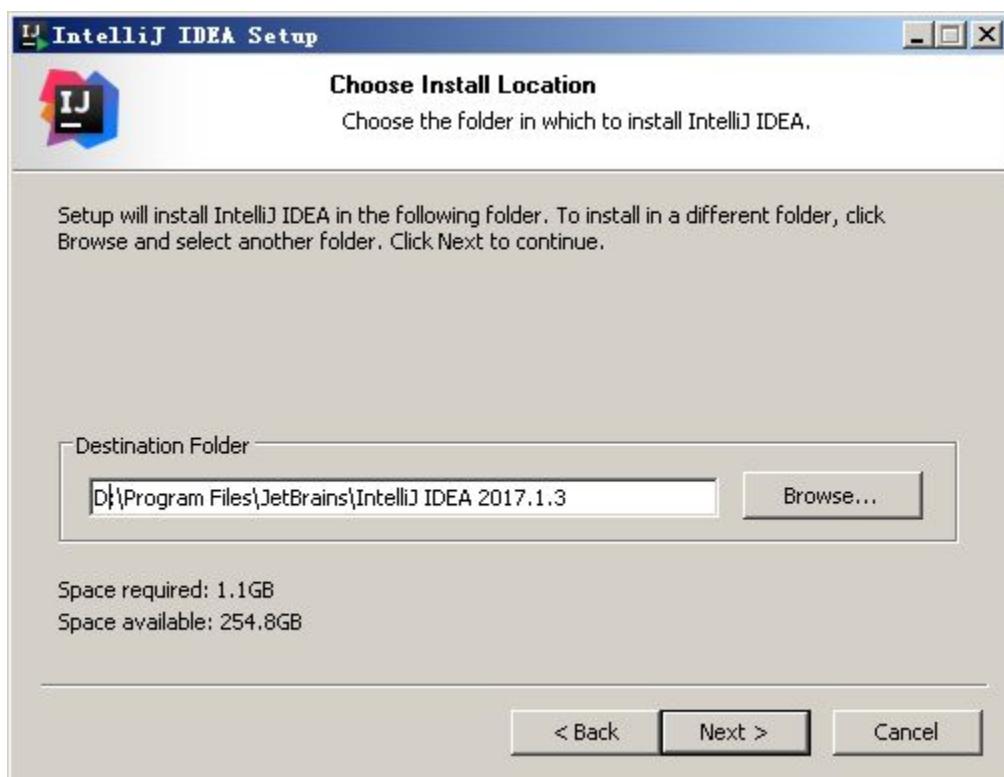
运行安装包，出现以下界面



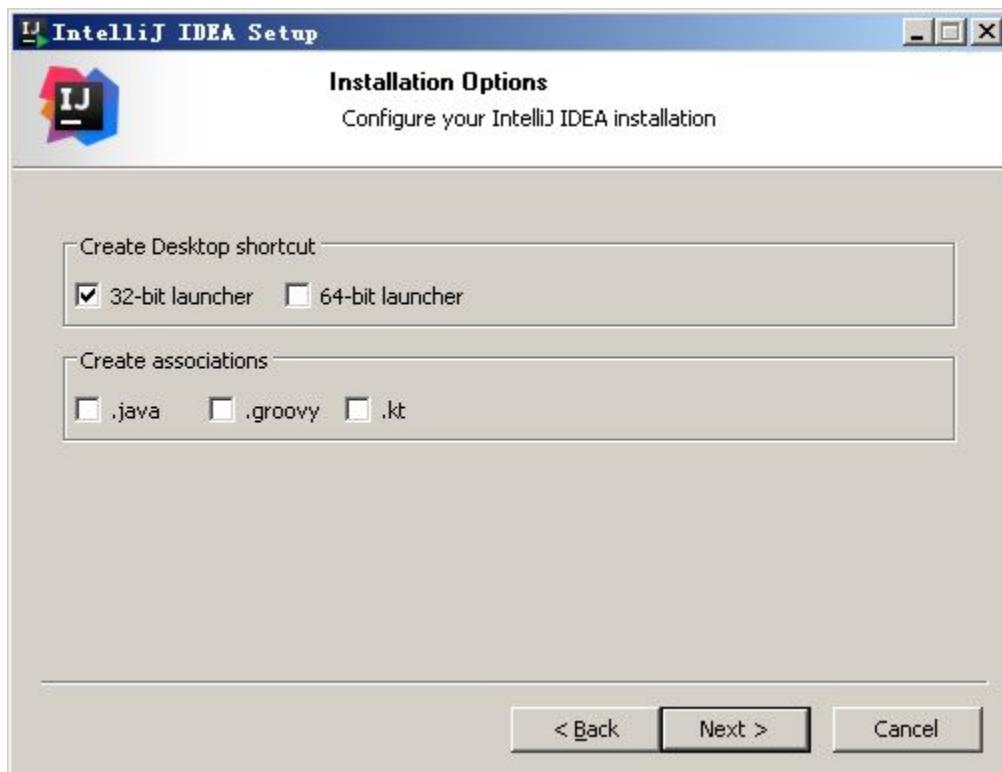
点击下一步，不卸载旧版本



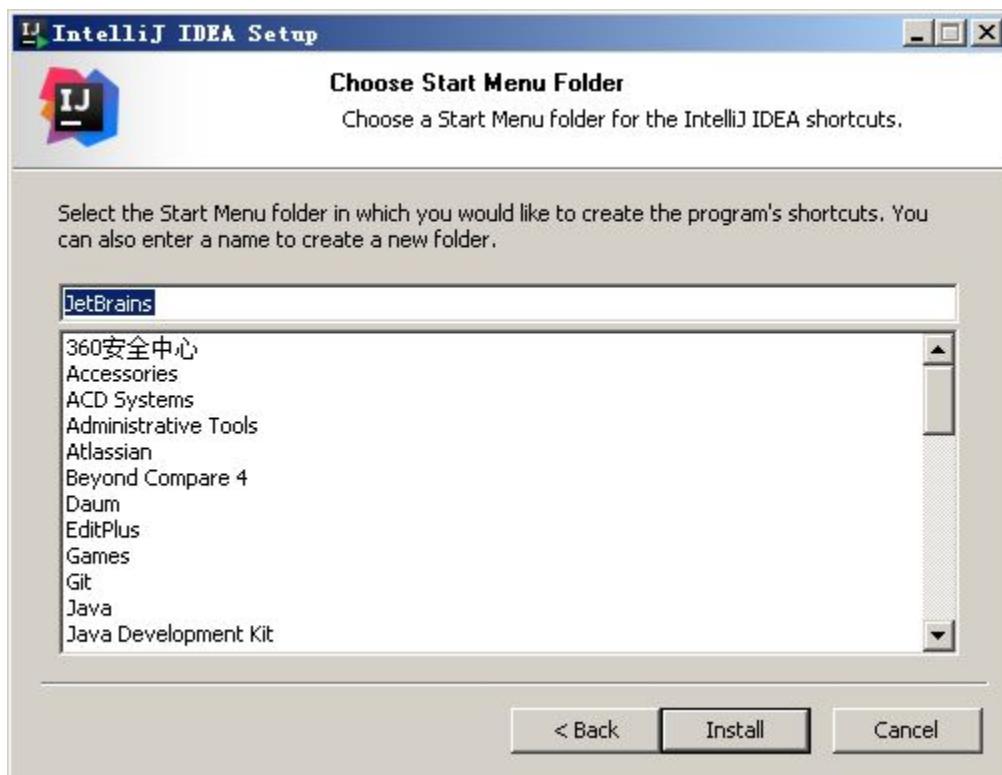
下一步，选择安装目录



下一步，选择桌面快捷和关联文件打开方式



下一步，配置开始菜单目录



开始安装，等待安装完成。



完成



目录说明

bin	2017/5/18 13:37	文件夹	
help	2017/5/18 13:40	文件夹	
jre64	2017/5/18 13:37	文件夹	
lib	2017/5/18 13:36	文件夹	
license	2017/5/18 13:36	文件夹	
plugins	2017/5/18 13:36	文件夹	
redist	2017/5/18 13:36	文件夹	
build.txt	2017/5/13 0:49	TXT 文件	1 KB
Install-Windows-zip.txt	2017/5/13 0:49	TXT 文件	3 KB
ipr.reg	2017/5/13 0:49	注册表项	1 KB

Bin: 容器，执行文件和启动参数等。

Help: 快捷键文档和其他帮助文档

Jre64:64 位 java 运行环境

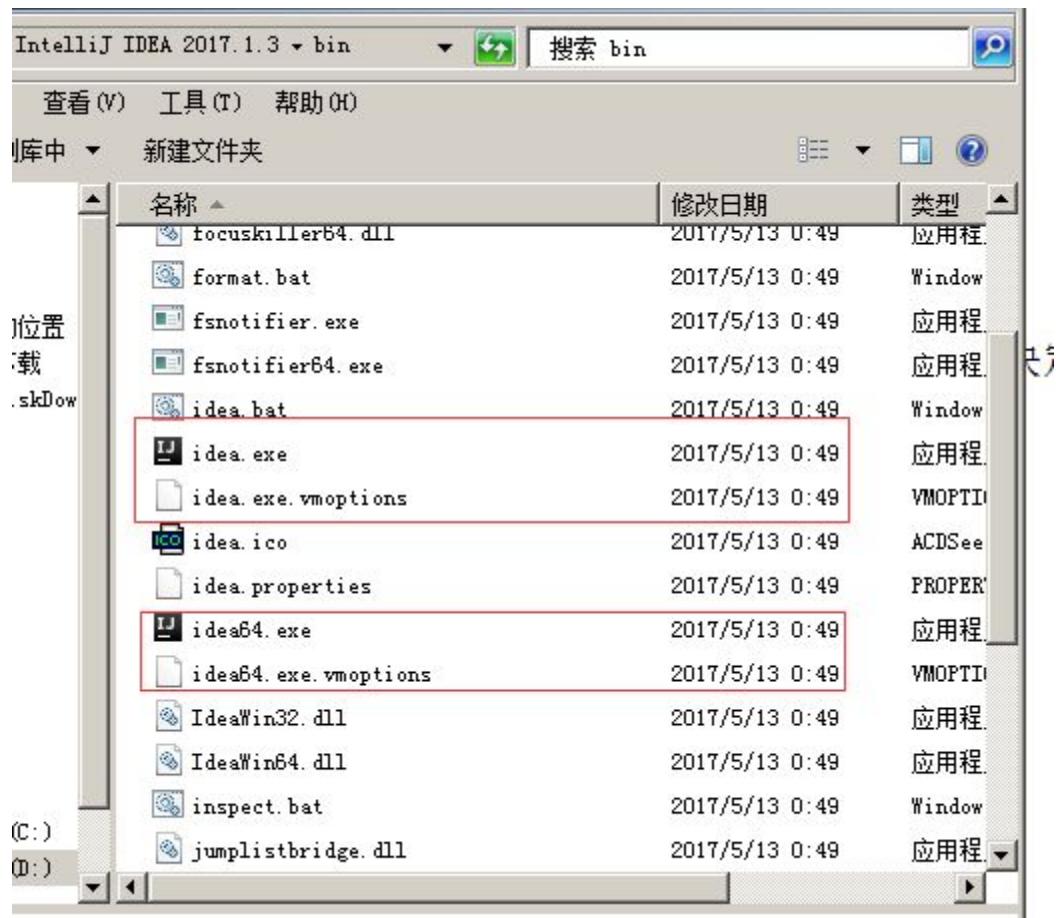
Lib: idea 依赖的类库

License: 各插件许可

Plugin: 插件

启动配置

Idea 启动有 32 位和 64 位之分，具体根据运行环境决定



这里以 idea64 为例进行说明

可以根据机器情况配置 vm 参数

```
D:\Program Files\JetBrains\IntelliJ IDEA 2017.1.3\bin\idea.exe.vmoptions - Sub
文件 (F) 编辑 (E) 选项 (O) 查找 (I) 查看 (V) 转到 (G) 工具 (T) 项目 (P) 首选项 (W) 帮助 (H)

idea教 * generat * 工作密 * 工作任 * config.p * idea.exe * 瑞钱宝 * 

1 -server
2 -Xms128m
3 -Xmx512m
4 -XX:ReservedCodeCacheSize=240m
5 -XX:+UseConcMarkSweepGC
6 -XX:SoftRefLRUPolicyMSPerMB=50
7 -ea
8 -Dsun.io.useCanonCaches=false
9 -Djava.net.preferIPv4Stack=true
10 -XX:+HeapDumpOnOutOfMemoryError
11 -XX:-OmitStackTraceInFastThrow
12
```

配置空间

配置这个的目的是方便进行迁移，即在新的环境中不用在手动配置相关配置(比如主题，maven，jdk等)

```

1 # Use ${idea.home.path} macro to specify location relative to IDE installation
2 # Use ${xxx} where xxx is any Java property (including defined in previous line)
3 # Note for Windows users: please make sure you're using forward slashes (e.g. c:
4
5 #-
6 # Uncomment this option if you want to customize path to IDE config folder. Mak
7 #-
8 # idea.config.path=${user.home}/.IntelliJIdea/config
9 bgtidea=d:/bgtidea 用户空间
10 idea.config.path=${bgtidea}/.IntelliJIdea/config config目录
11 #-
12 # Uncomment this option if you want to customize path to IDE system folder. Mak
13 #-
14 # idea.system.path=${user.home}/.IntelliJIdea/system
15 idea.system.path=${bgtidea}/.IntelliJIdea/system 系统目录
16
17 #-
18 # Uncomment this option if you want to customize path to user installed plugins
19 #-
20 # idea.plugins.path=${idea.config.path}/plugins
21 idea.plugins.path=${idea.config.path}/plugins 插件
22 #-
23 # Uncomment this option if you want to customize path to IDE logs folder. Make
24 #-
25 # idea.log.path=${idea.system.path}/log
26 idea.log.path=${idea.system.path}/log 日志
27 #-
28 # Maximum file size (kilobytes) IDE should provide code assistance for.
29 # The larger file is the slower its editor works and higher overall system memo
30 # if code assistance is enabled. Remove this property or set to very large numb
31 # code assistance for any files available regardless their size.
32 #-
33 idea.max.intellisense.filesize=2500

```

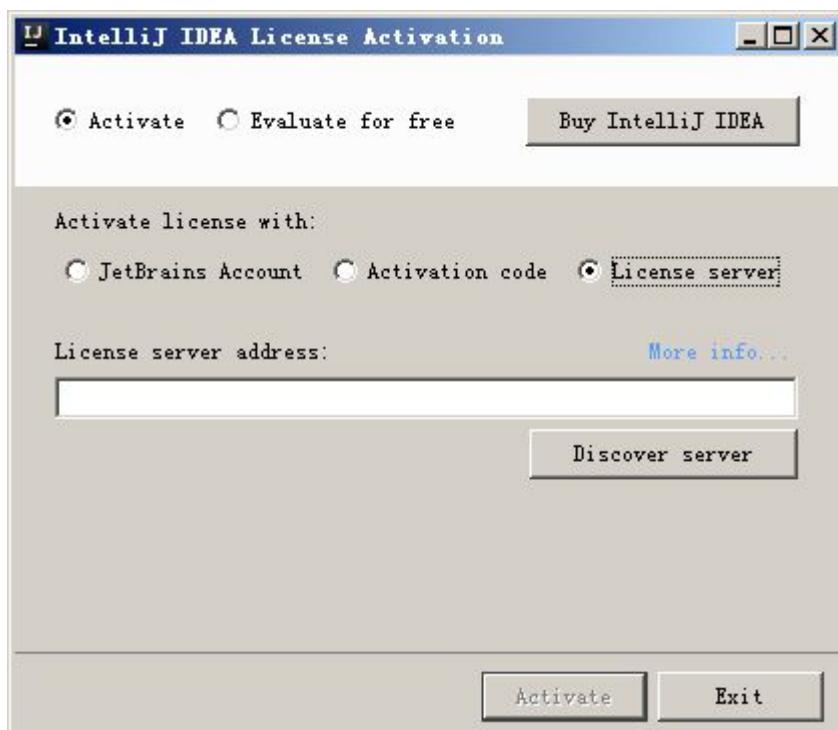
配置后，所有的插件，使用习惯配置，索引，项目部署相关都会在自定义目录中，其他 idea 中配置此目录，即可沿用以前习惯。

首次启动

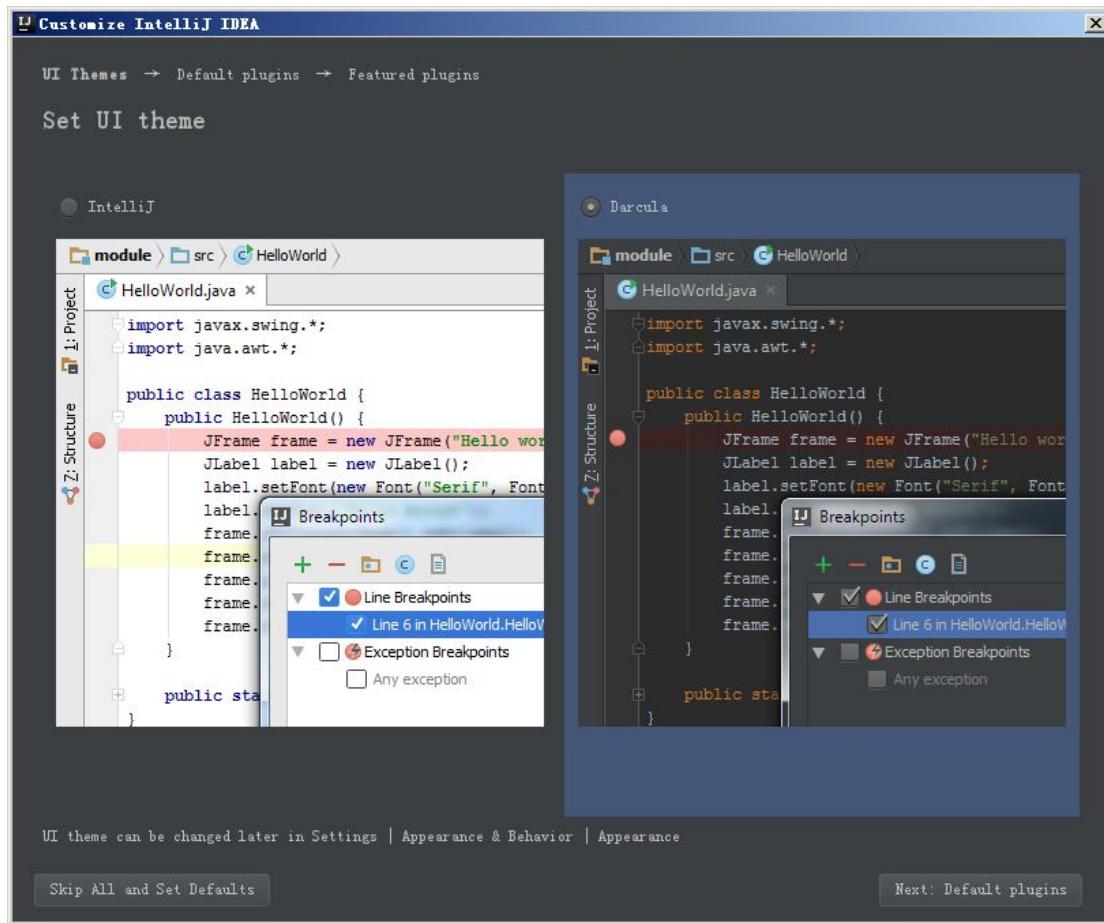
配置 settings，可以导入之前使用过得配置空间，也可以手动再配置



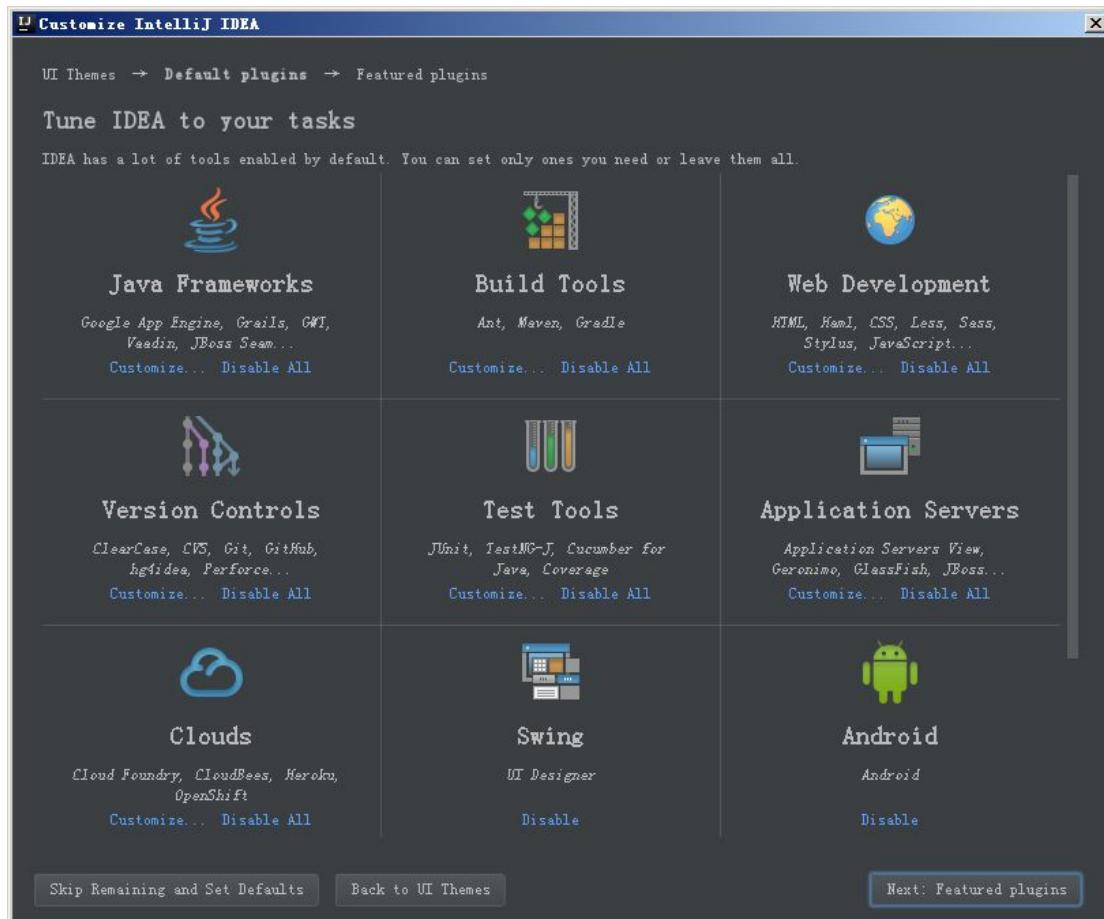
注册软件，选择合适的方式进行注册。

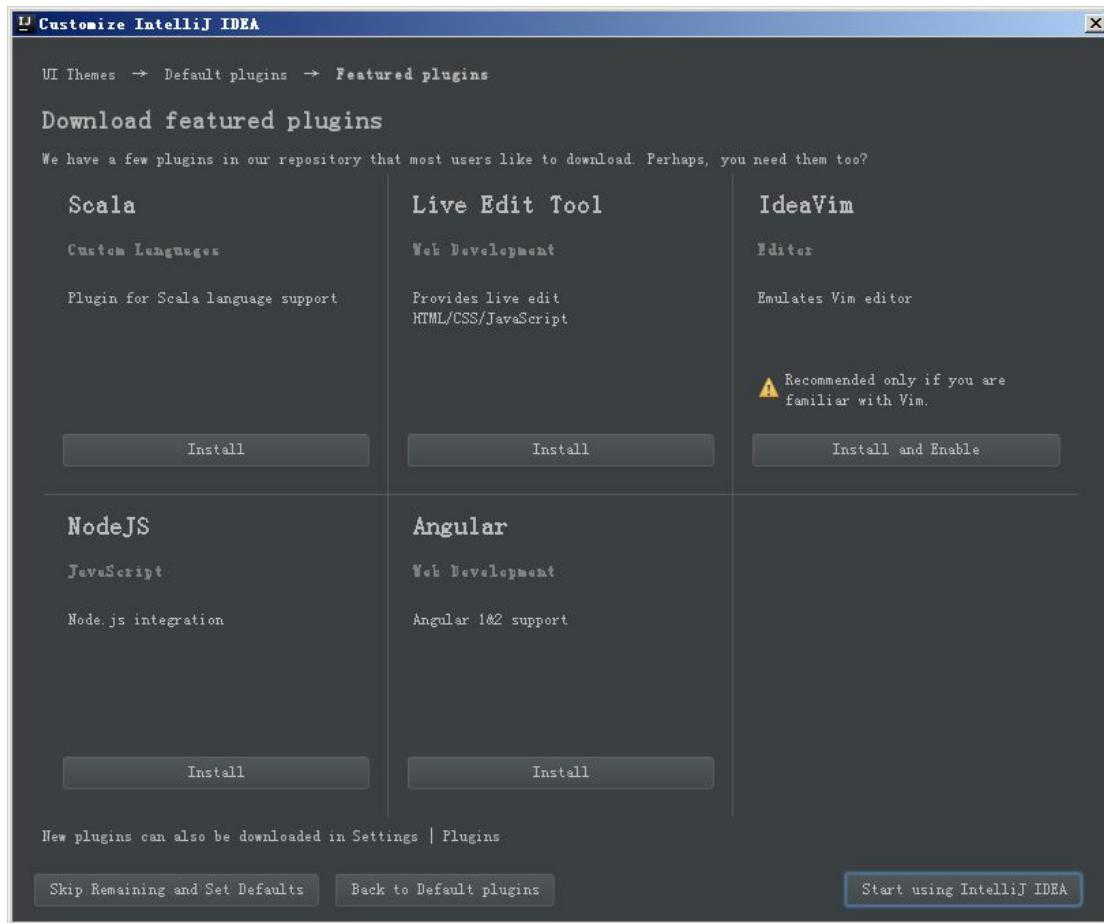


选择主题



选择默认插件，不需要的可以禁止，以优化启动速度。





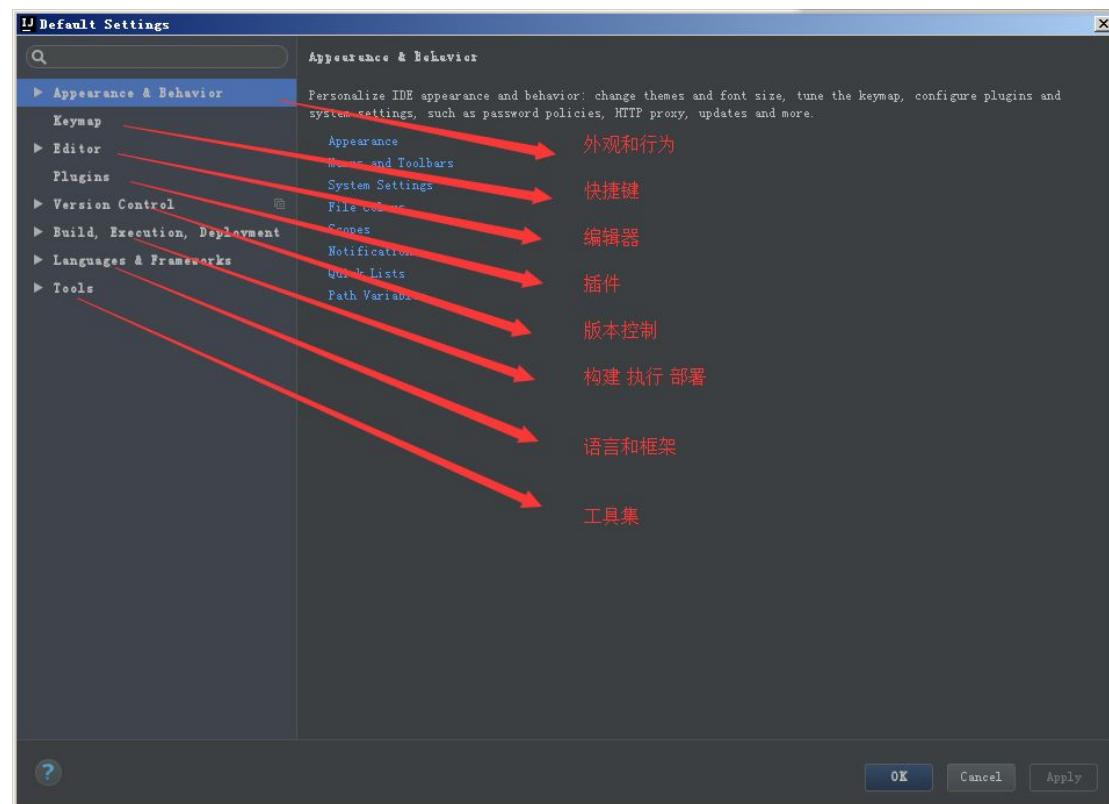
最后弹出欢迎页面。



首次配置

进入配置页面的方式:

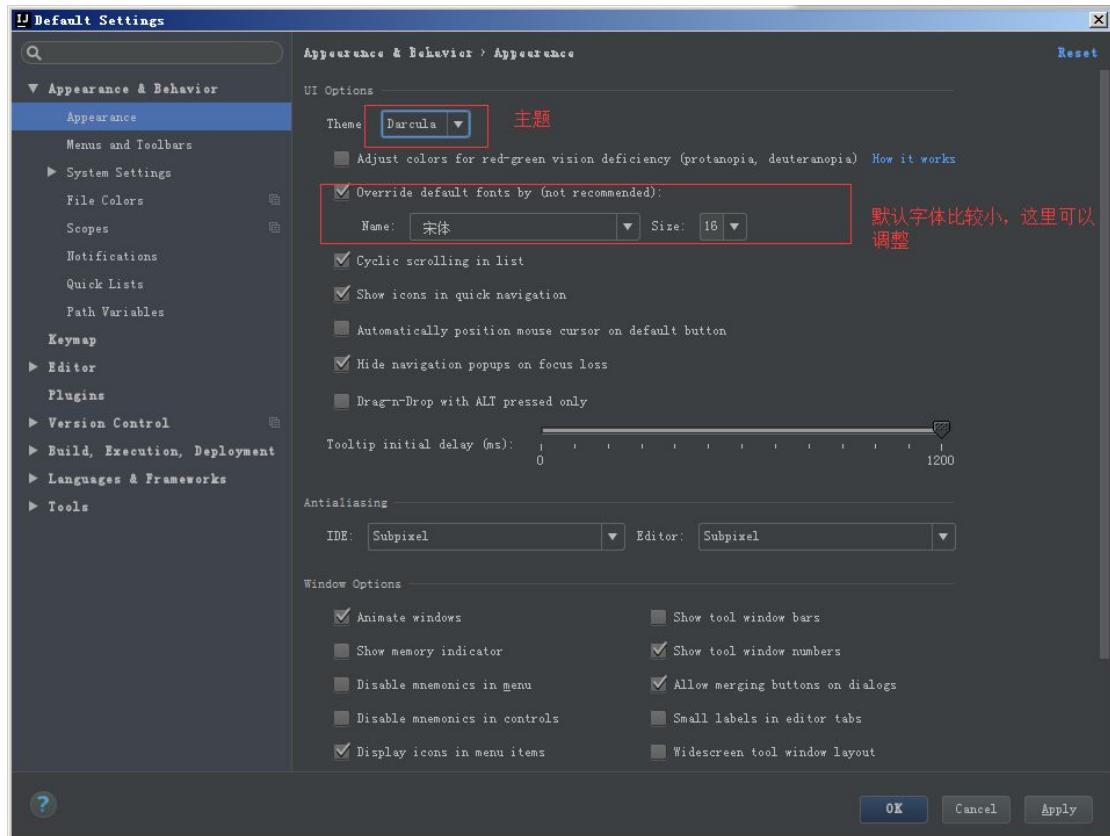
Configuration—>settings



Appearance&Behavior (外观和行为)

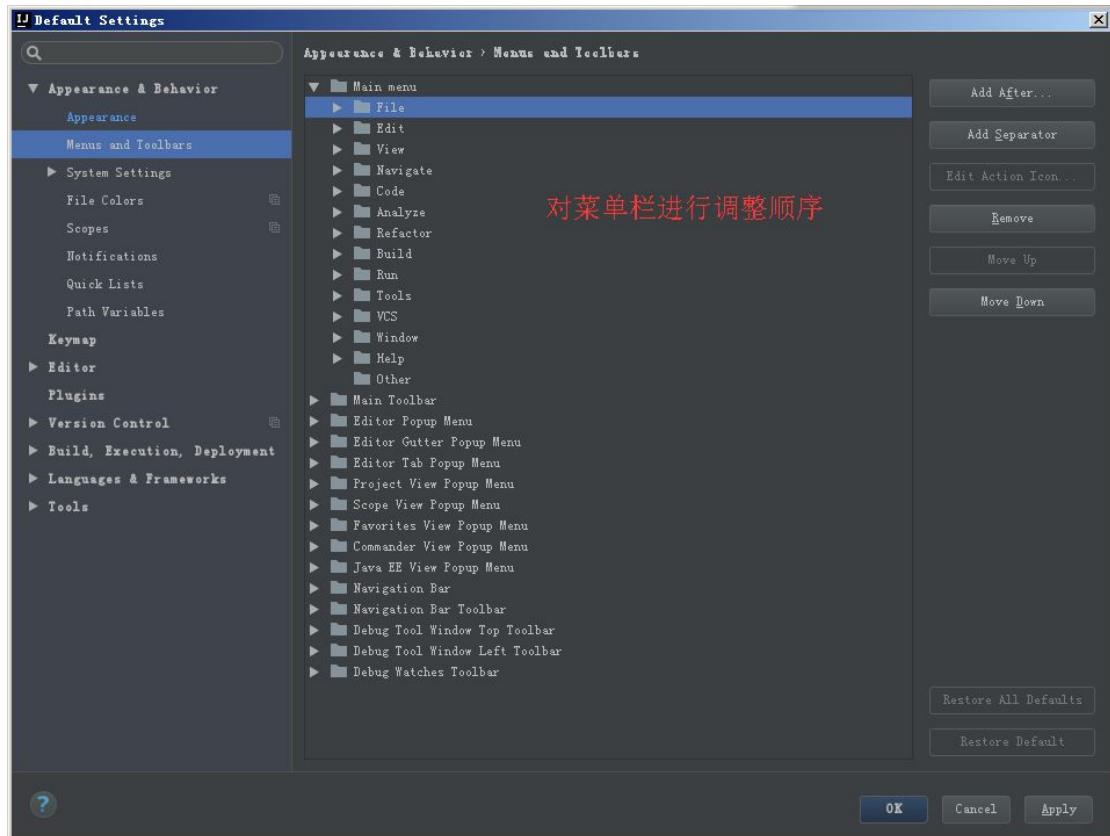
Appearance (外观)

配置主题、字体、字号、工具类以及其他视图工具。



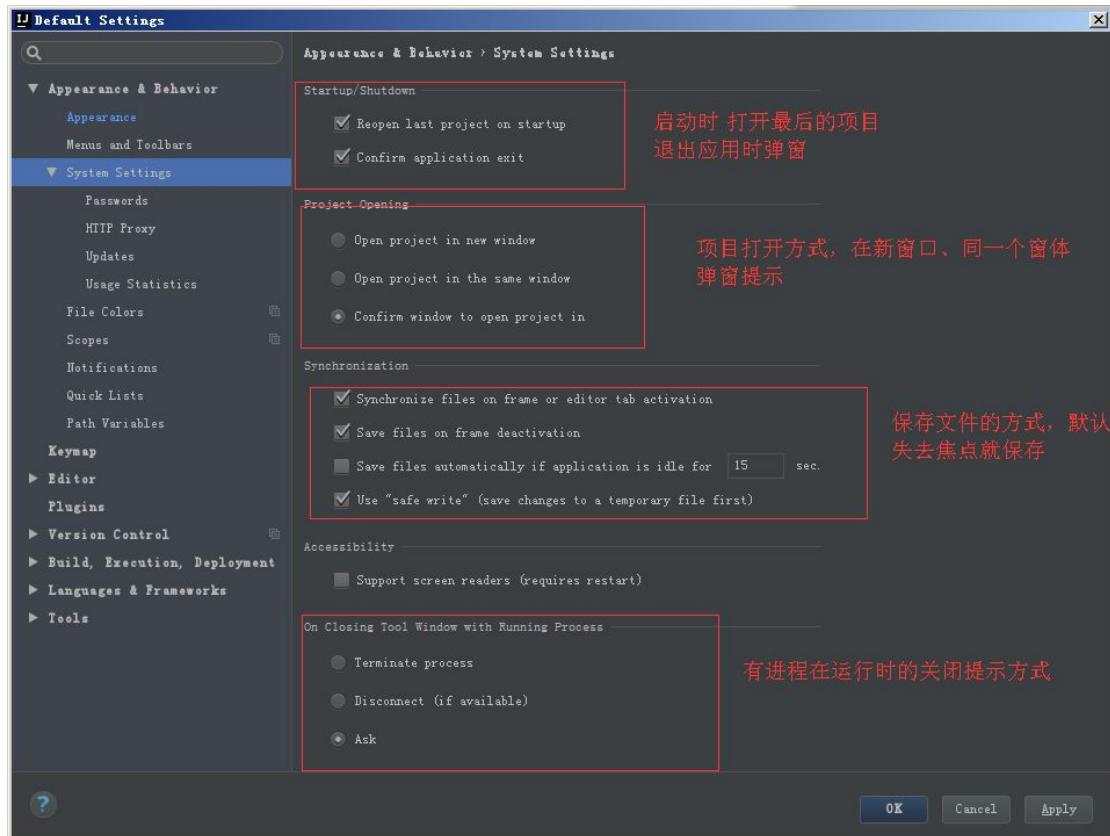
Menus and Toolbars (菜单和工具栏)

可以对菜单进行增删改



System Setting (系统设置)

配置启动时是否打开项目，项目打开方式，保持文件方式，退出时的方式等。



File Colors

文件颜色，保持默认即可

Scope

作用域，保持默认即可

Notifications

通知，在启动的时候会进行的通知，比如 Spring 通知等。保持默认即可。

Quick Lists

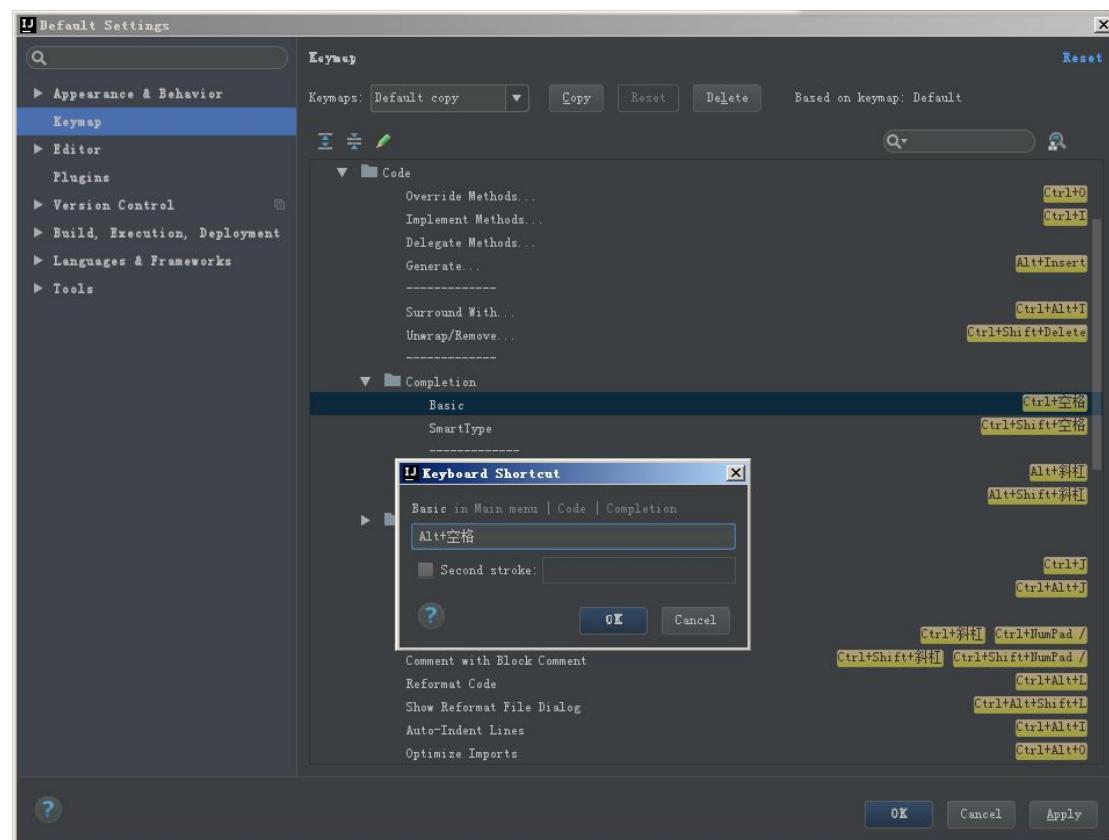
快捷列表，自定义快捷操作列表，保持默认即可。

Path variable

可用路径配置，保持默认即可。

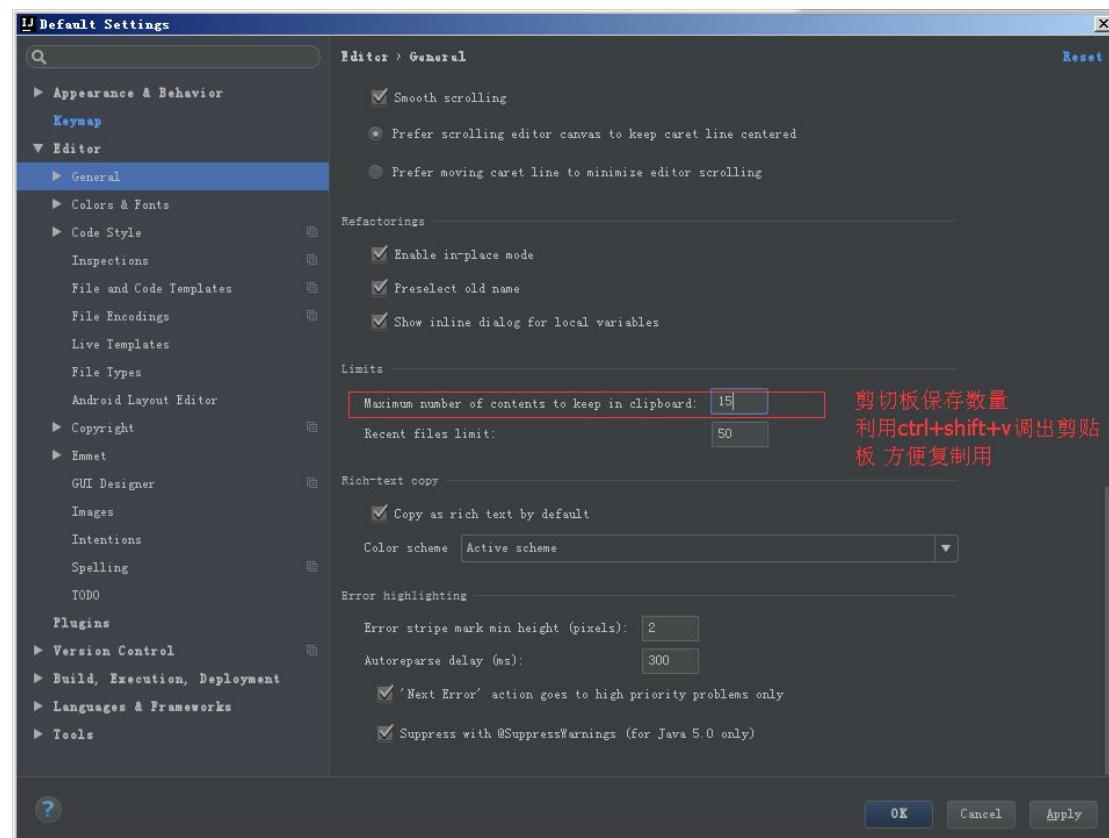
KeyMap

快捷键配置，配置快捷键主题，快捷键修改等。

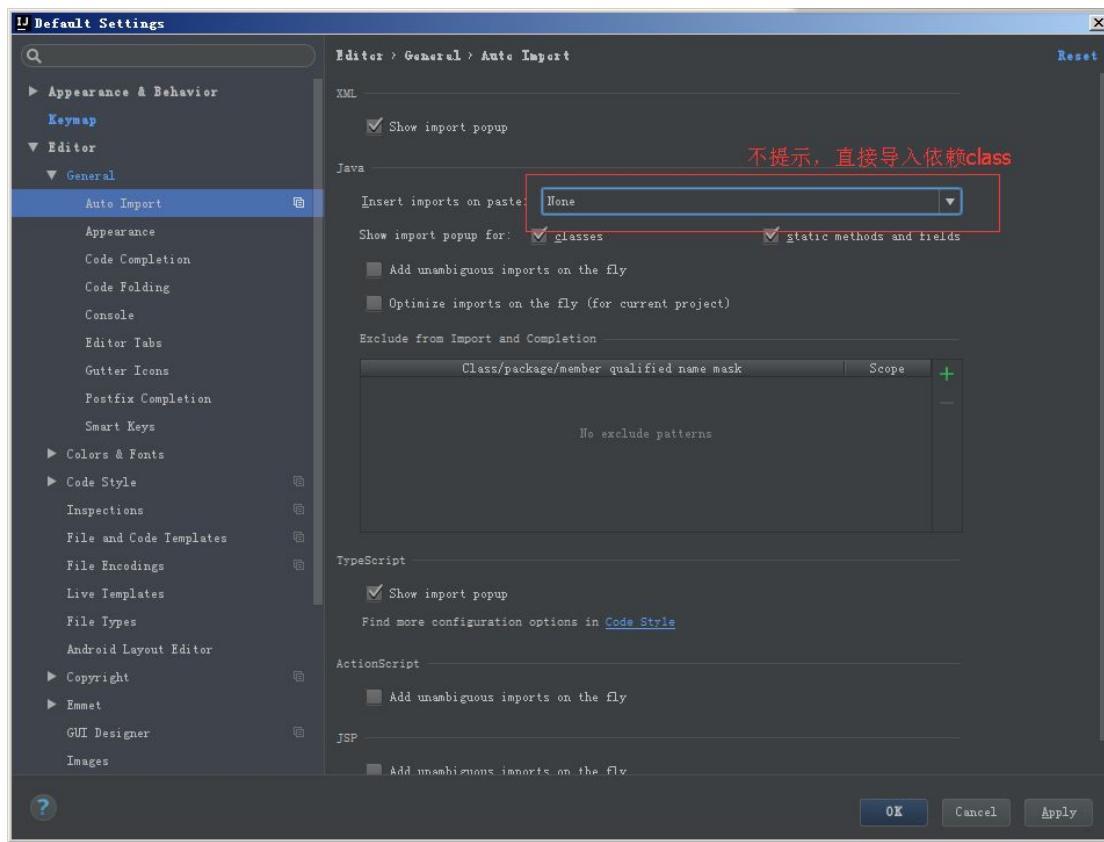


Editor (编辑器)

General (通常)

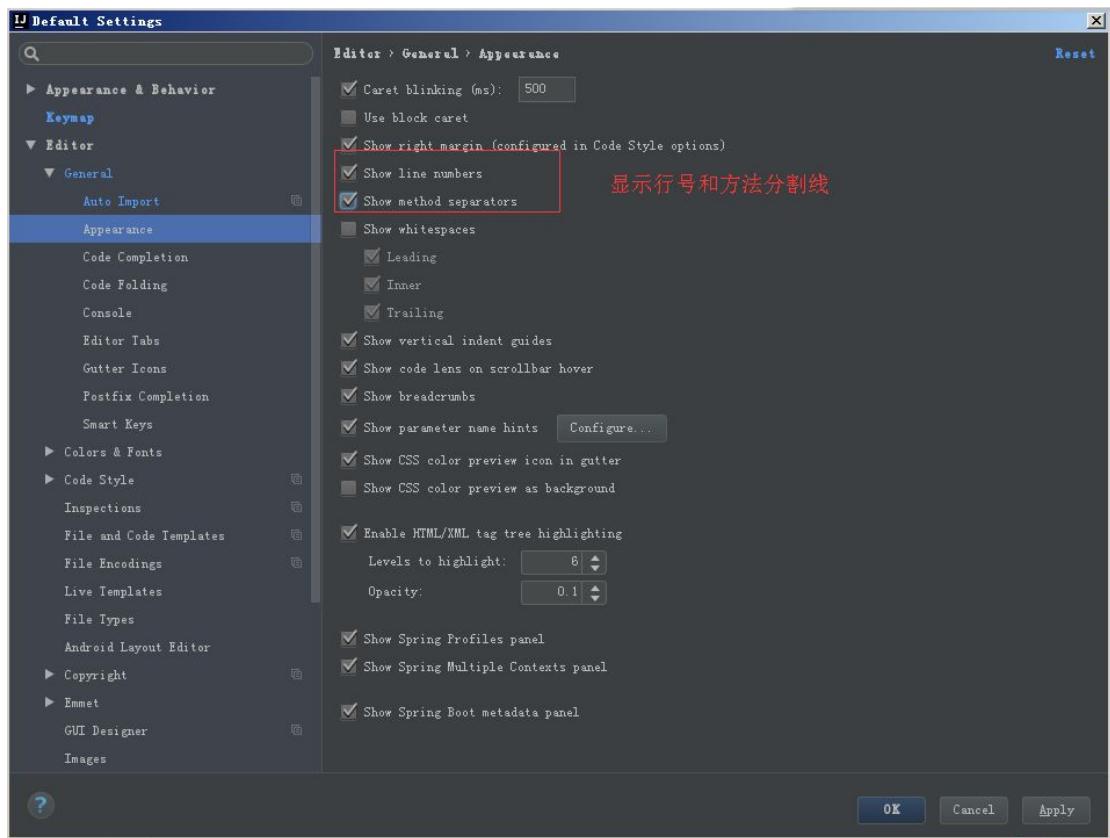


Autoimporting (自动导入配置)



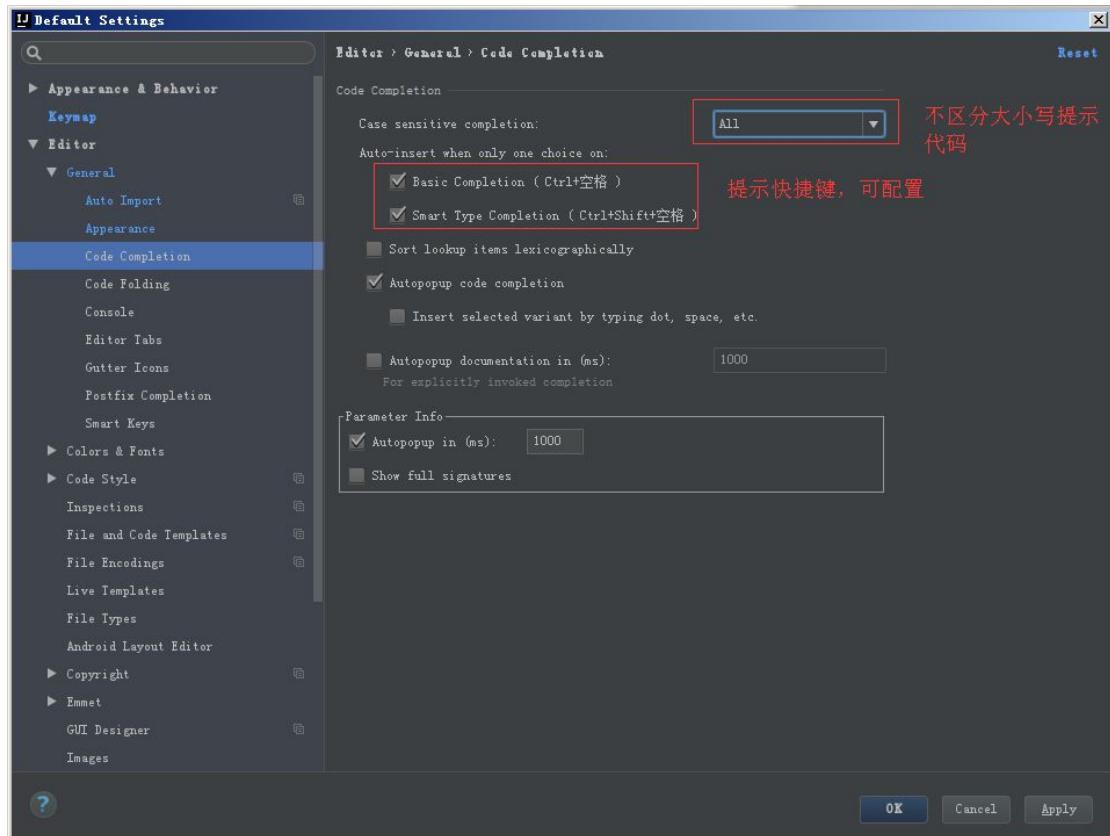
Appearance (外观)

配置编辑器显示视图，比如行号，分割线等。

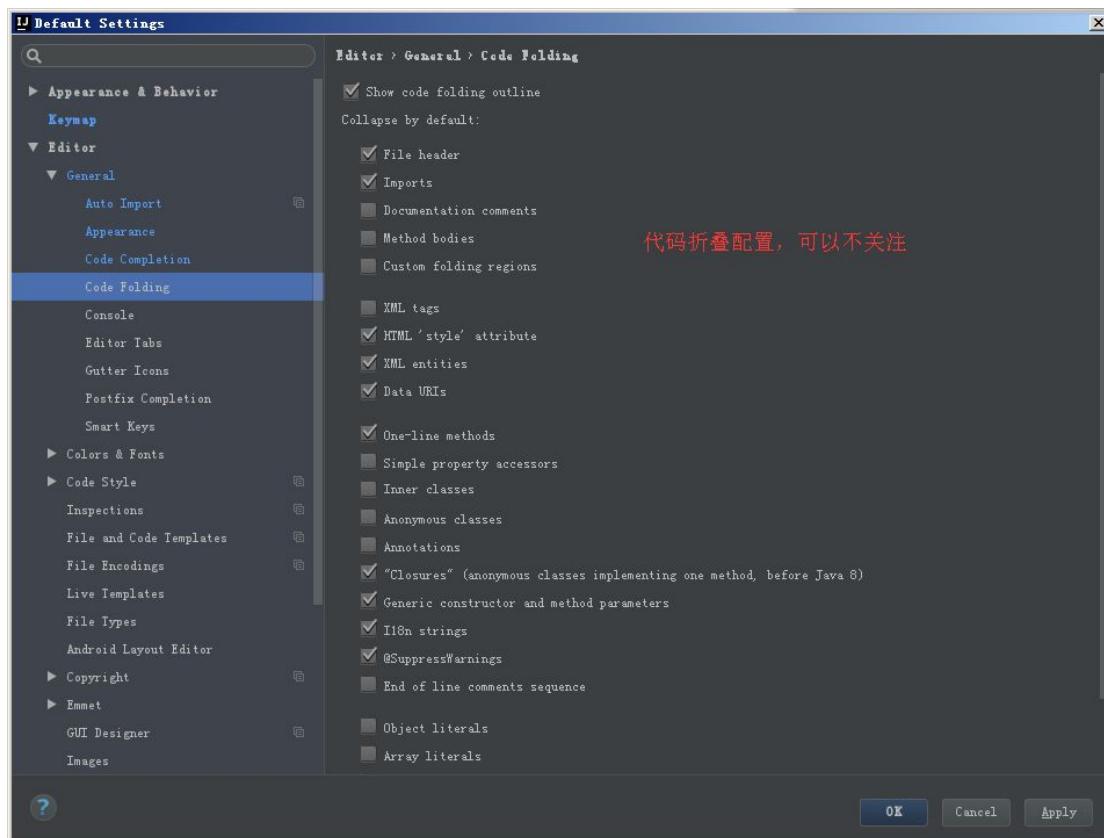


Code Complete (代码提示)

代码提示配置，配合快捷键自动提示代码。

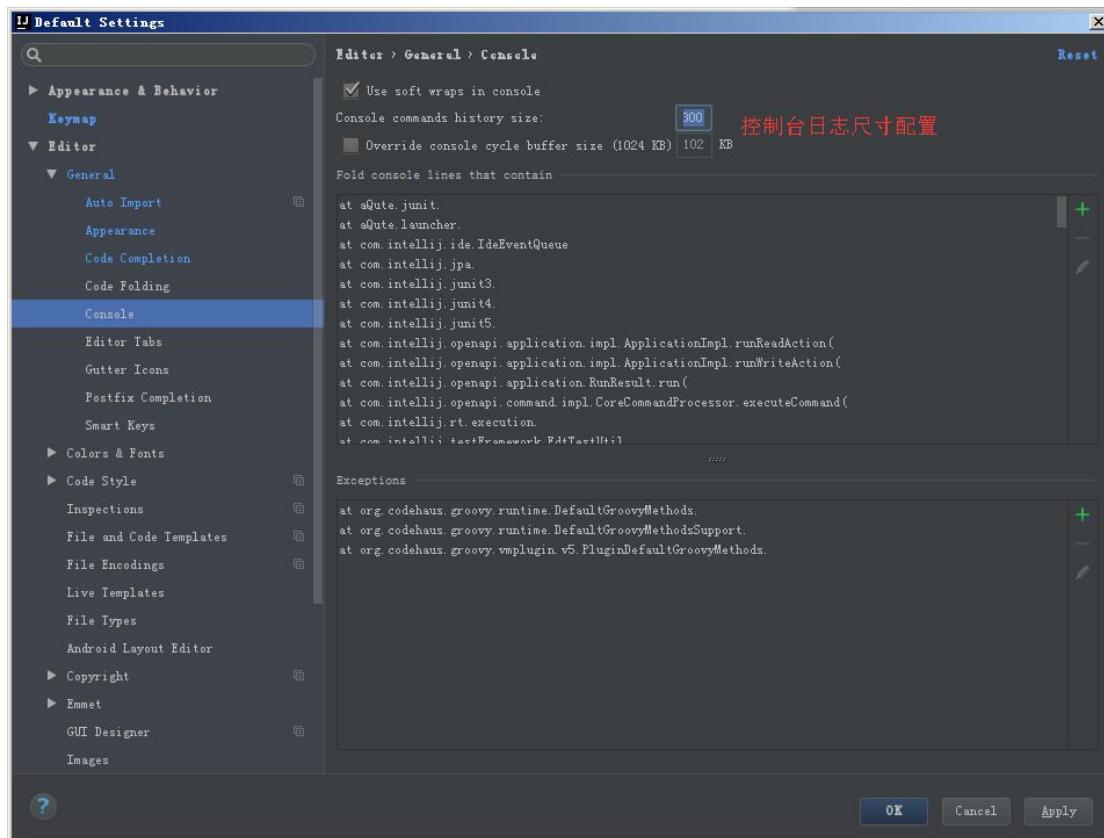


Code Folding (代码折叠)



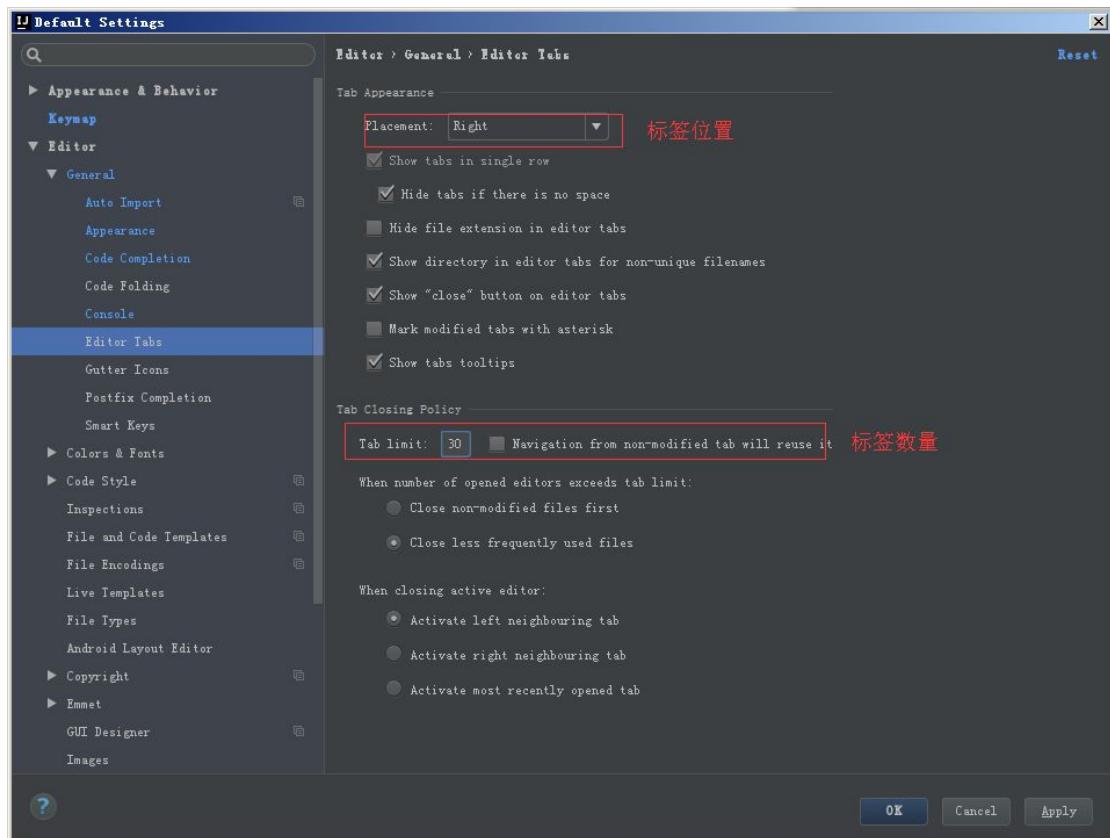
Console (控制台)

控制台配置



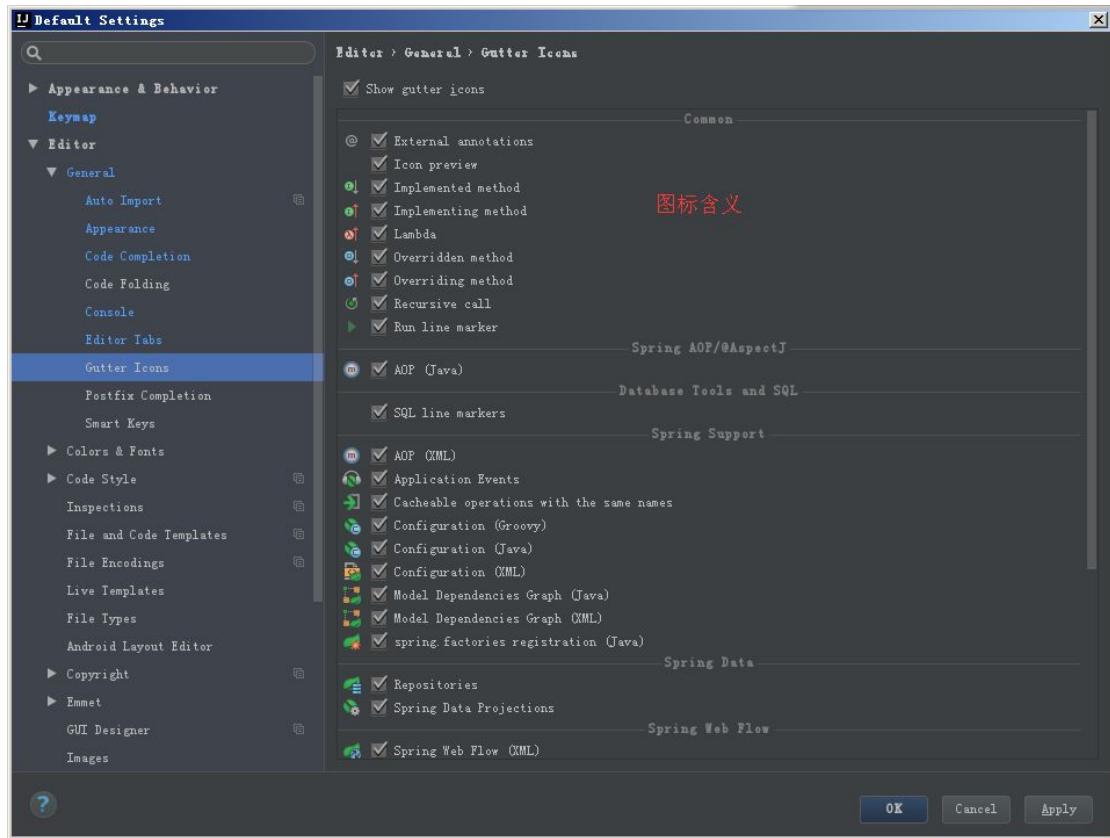
Editor tab (编辑器标签)

配置编辑器标签显示位置，显示方式等。



Gutter icons (图标库)

配置图标，也可以在此处查看图标的含义。



Postfix completion (后缀完善)

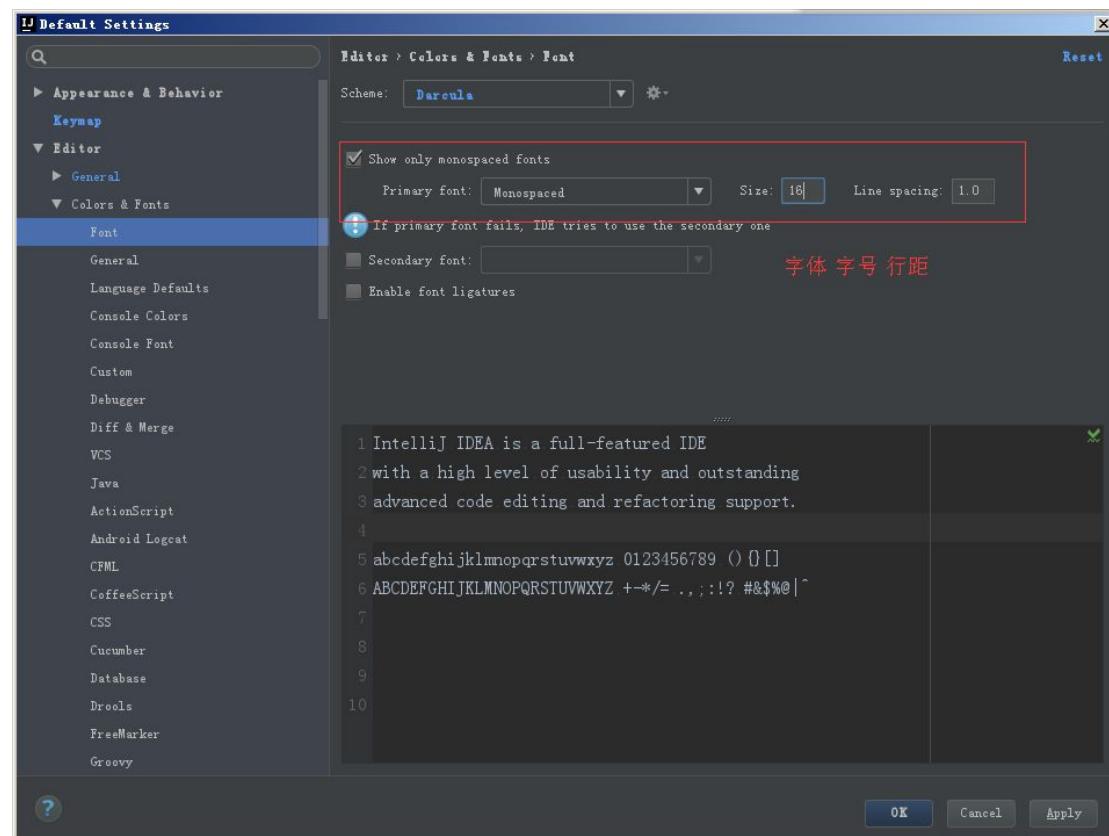
保持默认即可。

Smart keys (敏捷开发)

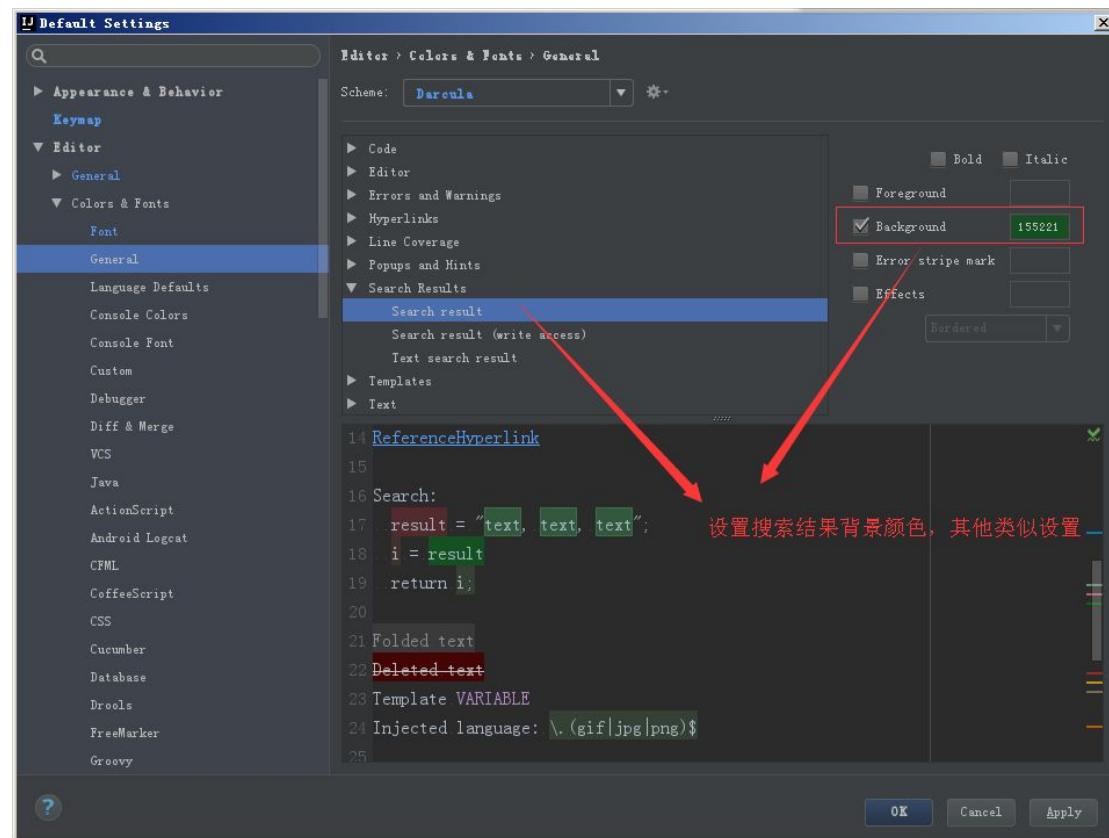
比如 html 标签自动补全, {}补全等, 保持默认即可。

Color&Fonts (颜色与字体)

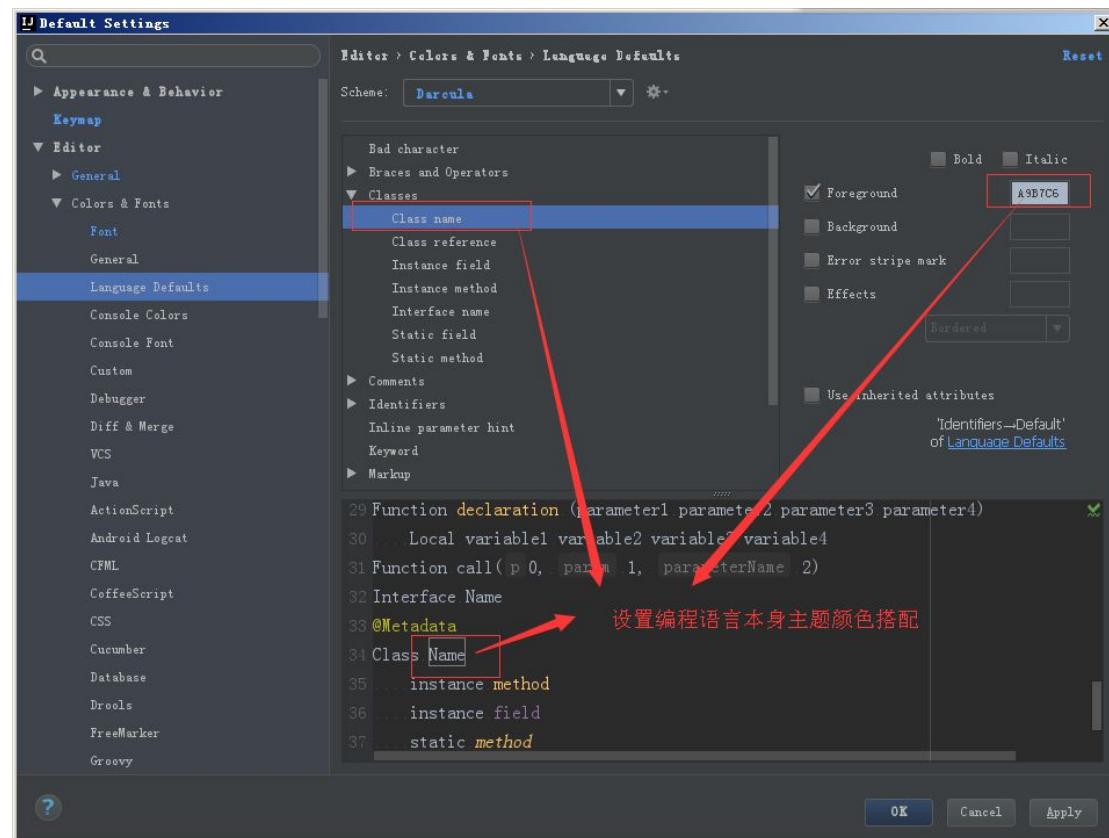
Font (字体)



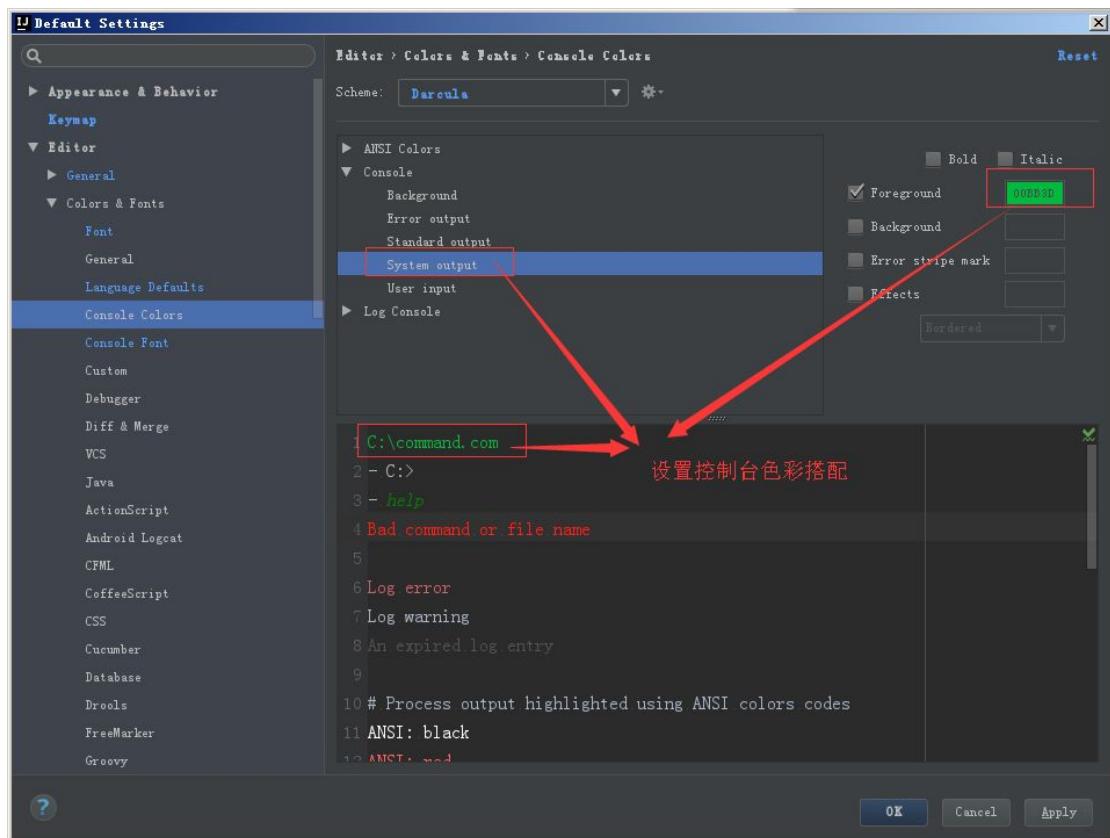
General (通用)



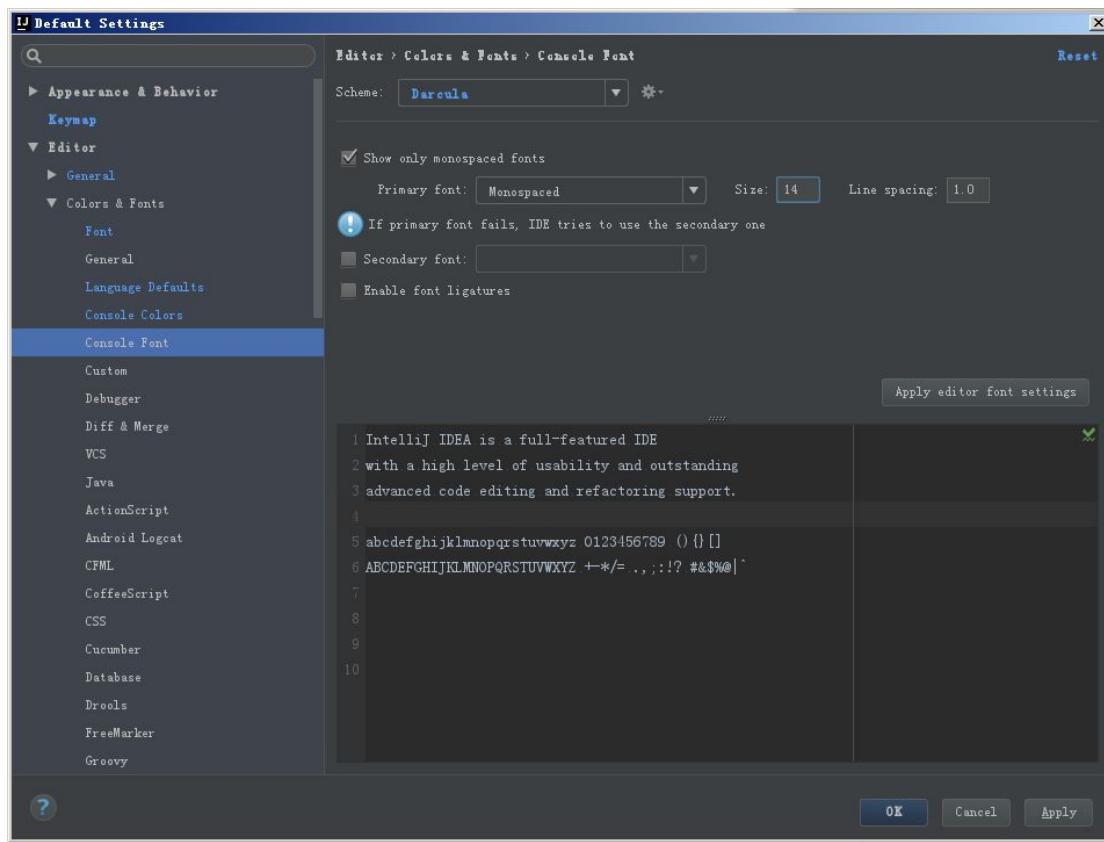
Language Defaults (语言默认配置)



Console Colors (控制台色彩)



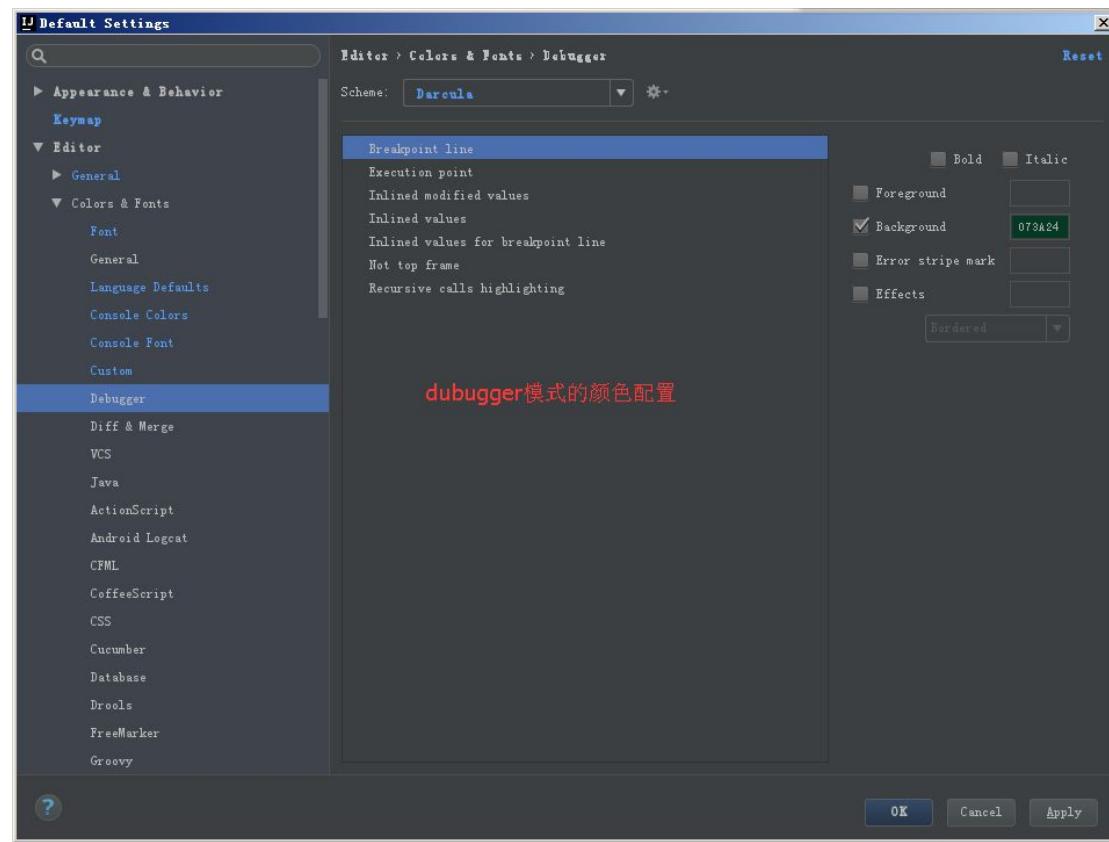
ConsoleFont (控制台字体)



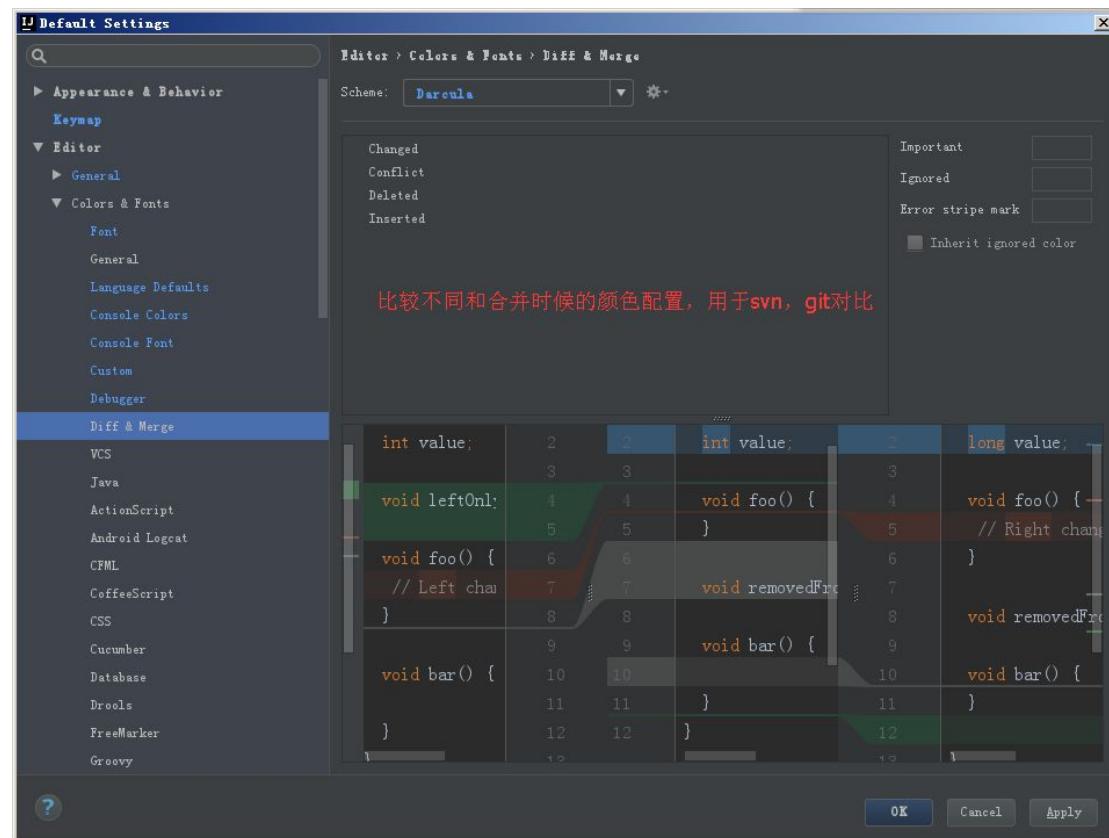
Custom (用户)

定制的习惯配置。

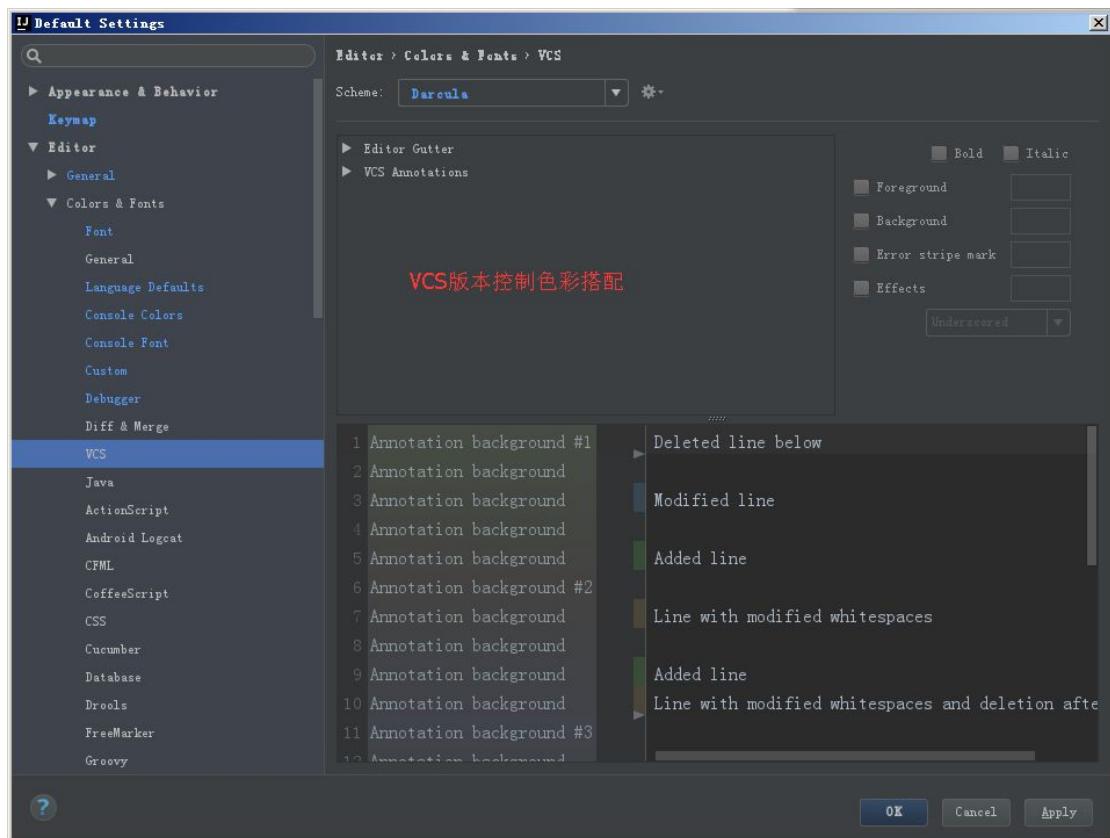
Debugger(断点)



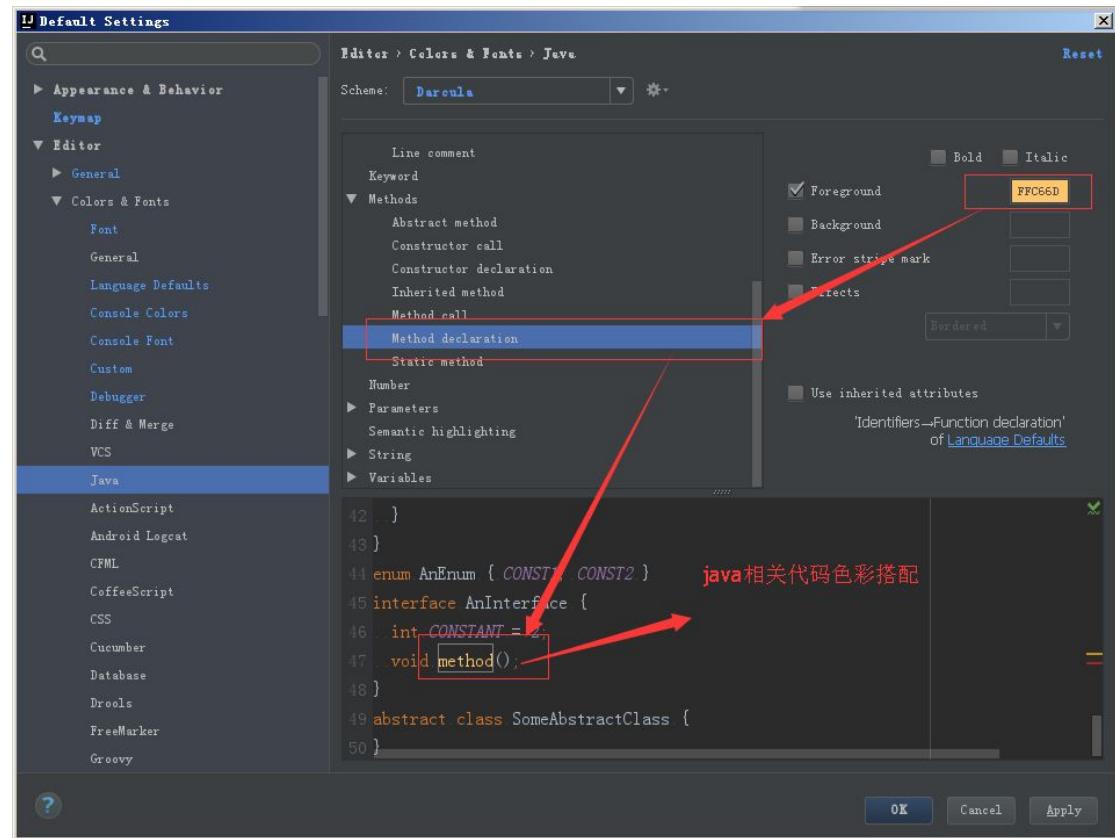
Diff&merge (比较合并)



Vcs (版本控制系统)



Java



ActionScript

As 脚本配置， 默认即可

Android Logcat

安卓日志配置，用的话可以配置

CFML

ColdFusion Markup Language， 默认配置即可

CoffeeScript

支持的一种脚本语言， 默认配置即可

Css

Css 配置， 默认即可

Cucumber

Cucumber 是一个能够理解用普通语言 描述的测试用例的支持行为驱动开发（BDD）的自动化测试工具，用 Ruby 编写，支持 Java 和 .Net 等多种开发语言。

Database

数据库色彩配置， 默认即可

Drools

Java 规则引擎色彩搭配， 默认即可

FreeMarker

模板语言色彩搭配， 默认即可

Groovy

Groovy 是一种基于 JVM（Java 虚拟机）的敏捷开发语言， 默认即可

Haml

Haml 是一种用来描述任何 XHTML web document 的标记语言， 默认即可

Html

超文本标记语言， 默认即可

JavaScript

浏览器脚本语言， 默认即可

Jpa/hibernate QL

数据库 sql 相关组件， 默认即可

JSON

一种数据传输格式， 默认即可

JSP

JavaEE 的页面技术， 默认即可

Kotlin

Kotlin 是一个基于 JVM 的新的编程语言， 默认即可

Less

敏捷开发 css 的一种语言， 默认即可

Markdown

Markdown 是一种可以使用普通文本编辑器编写的标记语言， 默认即可

OGNL

是一种功能强大的表达式语言， 默认即可

OSGI Manifest

与上边类似， 默认即可

Properties

特性文件， 默认即可

RegExp

正则表达式， 默认即可

Sass/SCSS

前端敏捷开发语言， 默认即可

Spring EL

Spring 正则， 默认即可

SQL

Sql 语言， 默认即可

Stylus

是一款 CSS 的预处理器， 默认即可

Table Diff

用于比较两个非收敛的表中的数据， 默认即可

Tapestry

Tapestry 是一个开源的基于 servlet 的应用程序框架,它使用组件对象模型来创建动态的,交互的 web 应用。默认即可

TypeScript

TypeScript 是一种由微软开发的自由和开源的编程语言。默认即可

Velocity

Velocity 是一个基于 java 的模板引擎(template engine)。默认即可

XML

可扩展标记语言， 默认即可

Xpath

XPath 即为 XML 路径语言，它是一种用来确定 XML（标准通用标记语言的子集）文档中某部分位置的语言。默认即可

XSLT

是一种对 XML（标准通用标记语言的子集）文档进行转化的语言， 默认即可

YAML

ML 是一个可读性高的用来表达资料序列的格式。默认即可

Spy-JS

一款前端框架， 默认即可

File Status

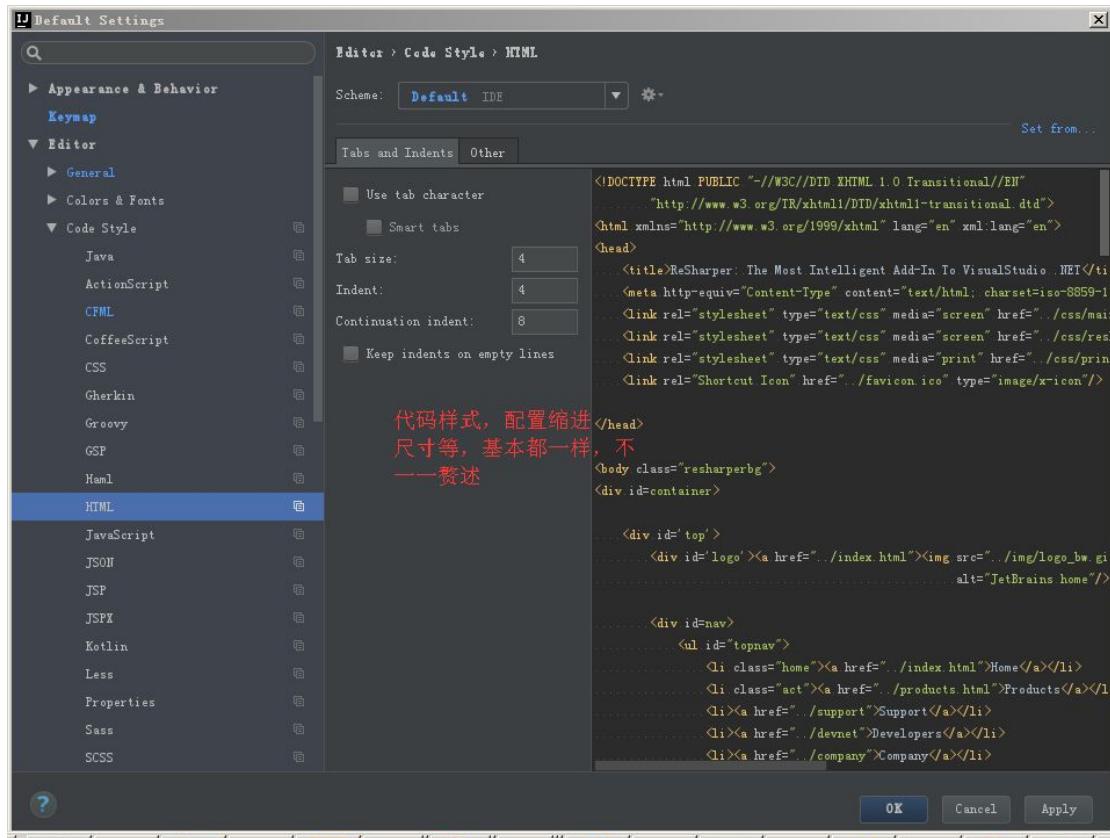
文件状态， 默认即可

By Scope

根据作用域来定， 默认即可

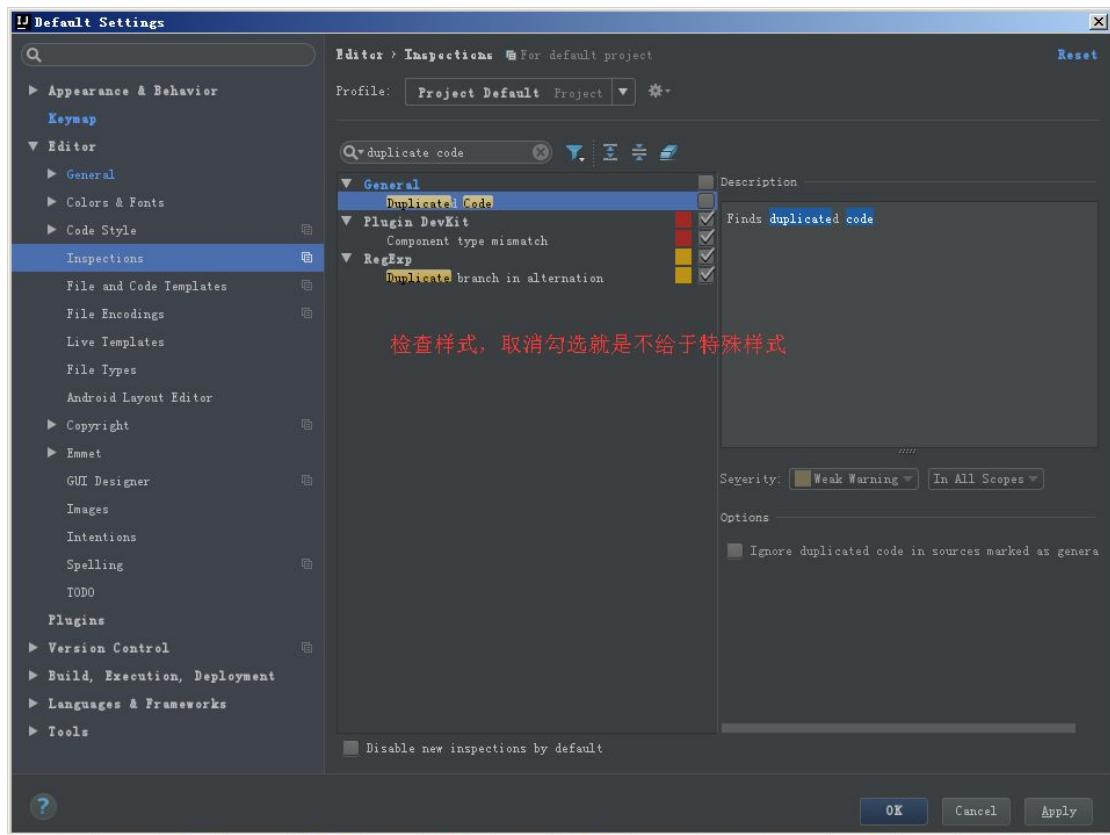
CodeStyle (代码样式)

idea 支持语言（不一一赘述）的代码样式配置，包括缩进，尺寸配置。



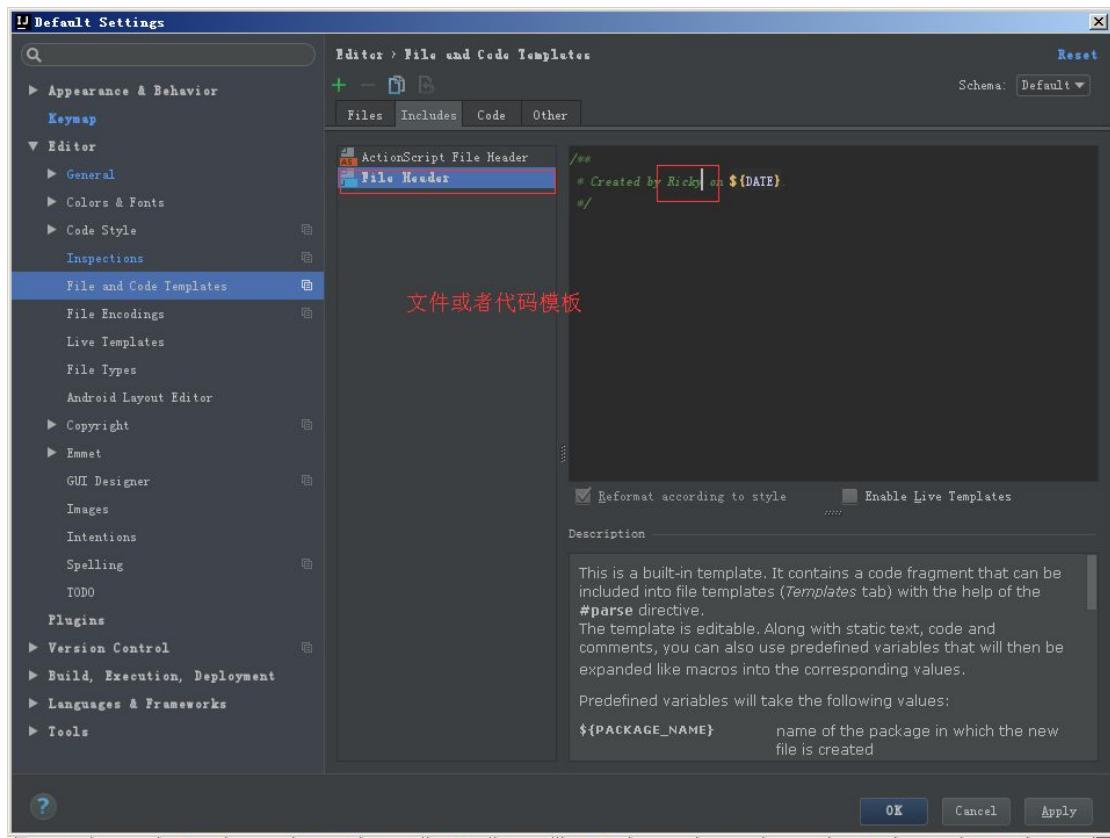
Inspections (检查)

代码审查级别，一般有 error、warn 等，会爆红和警告波浪线等。如果想取消检查，可以在此处配置。



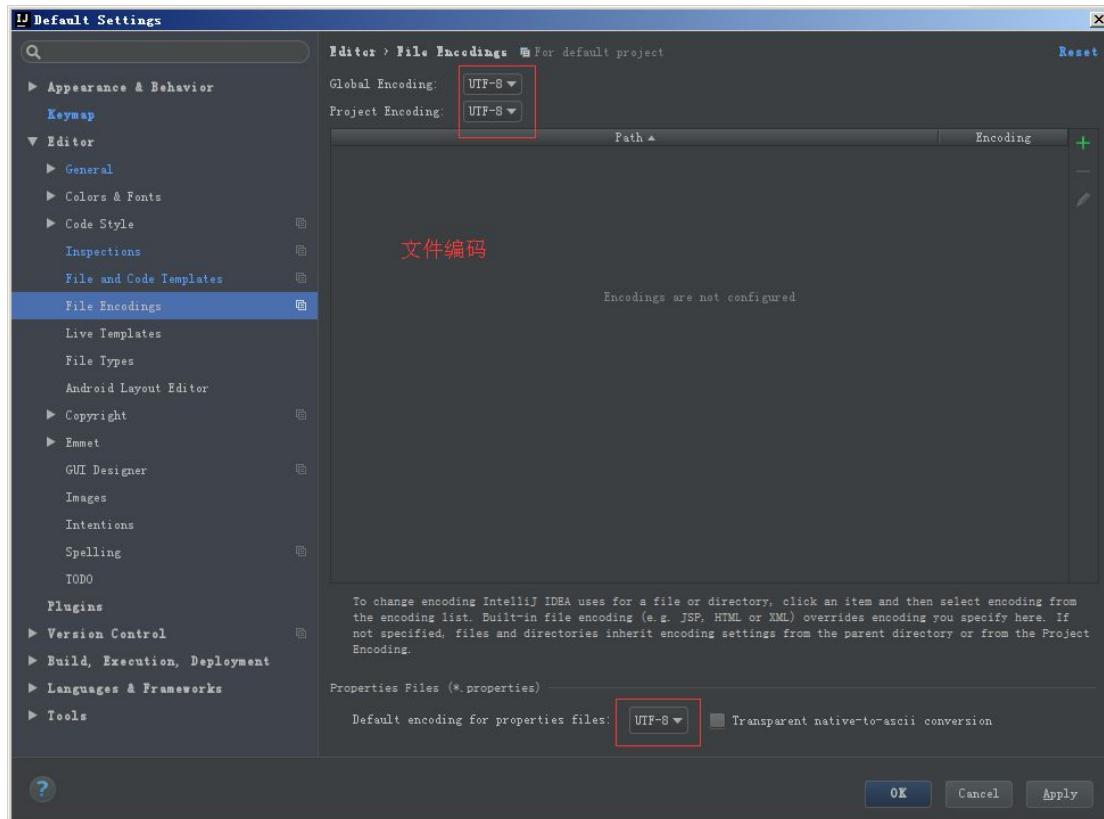
File &Code Template (文件和代码模板)

文件和代码模板，可以在此处配置修改。

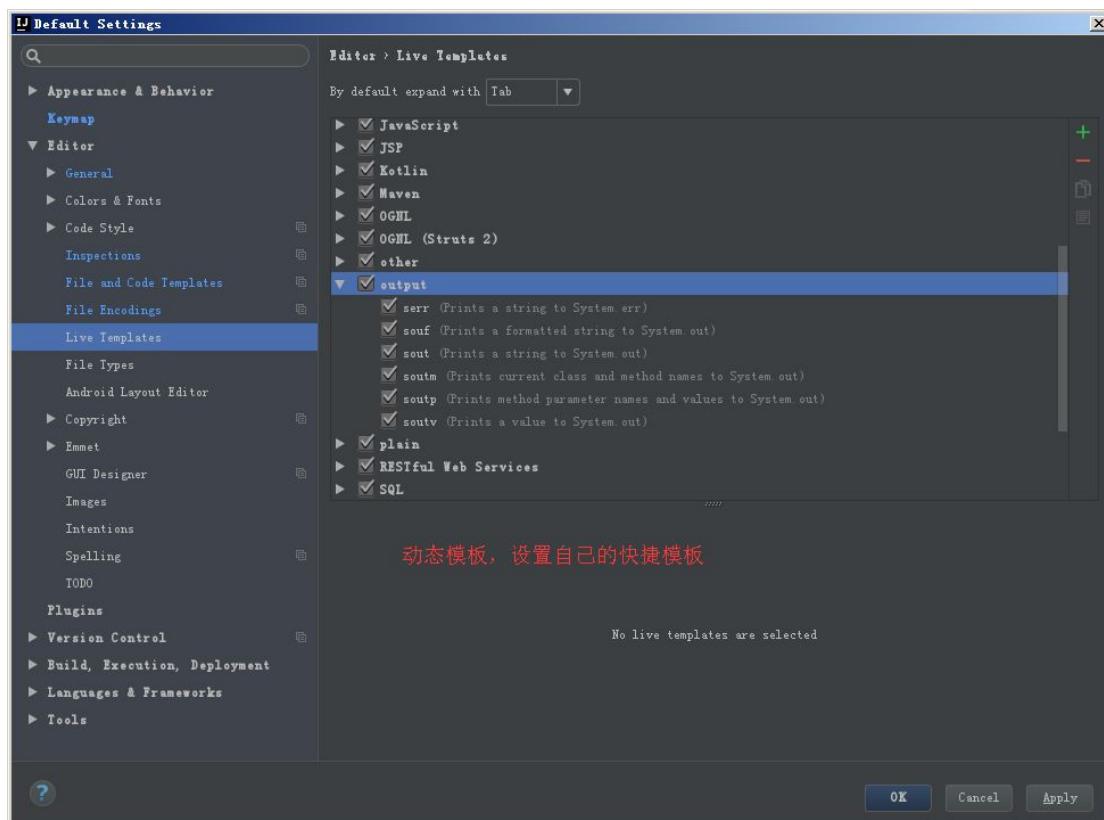


File Encoding (文件编码)

这里配置文件和项目的编码,也可以在 native 和 ascii 进行转换(\XXFO 这种转中文, properties 文件)

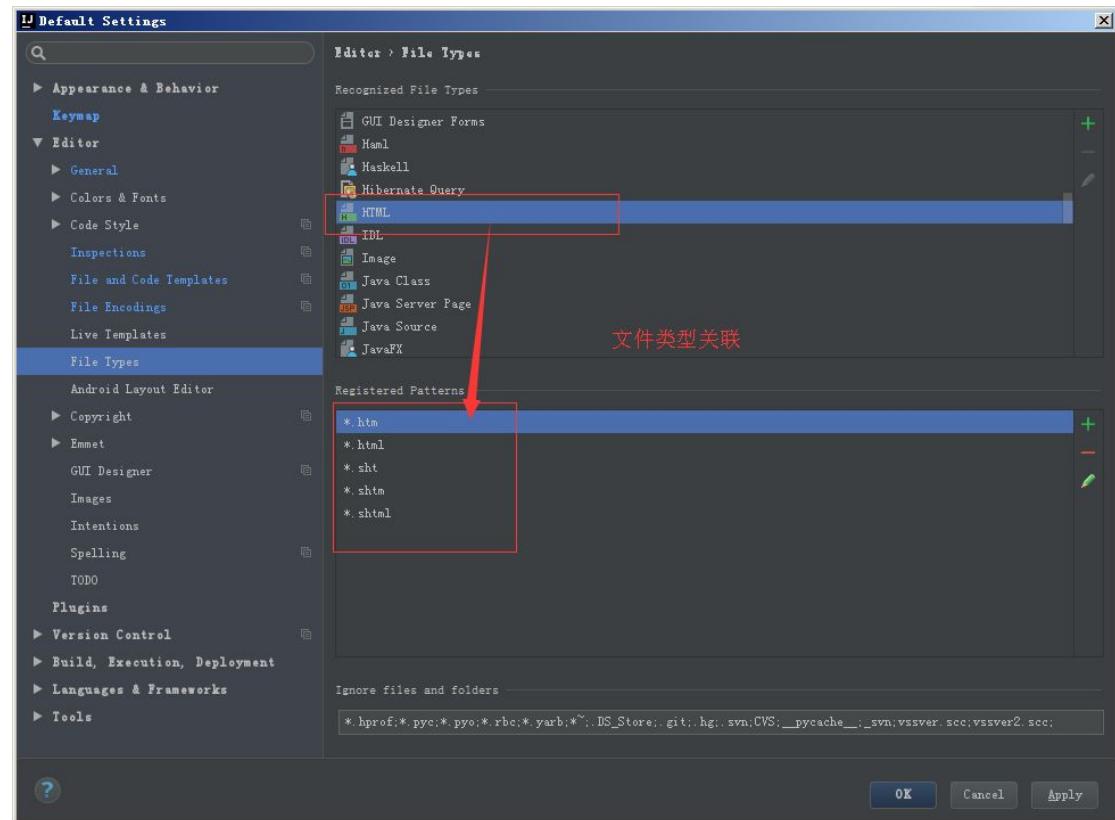


Live Template (实时模板)



File Type (文件类型)

文件图标以及关联方式



Android Layout Editor

安卓布局， 默认即可

Copyright (版权)

版权，更 Scope 相关， 默认即可

Emmet (前端语法)

Emmet 语法

GUI Designer

用户图像界面设计， 默认即可

Images

图片配置， 默认即可

Intentions

意图， 打算， 默认即可

Spelling

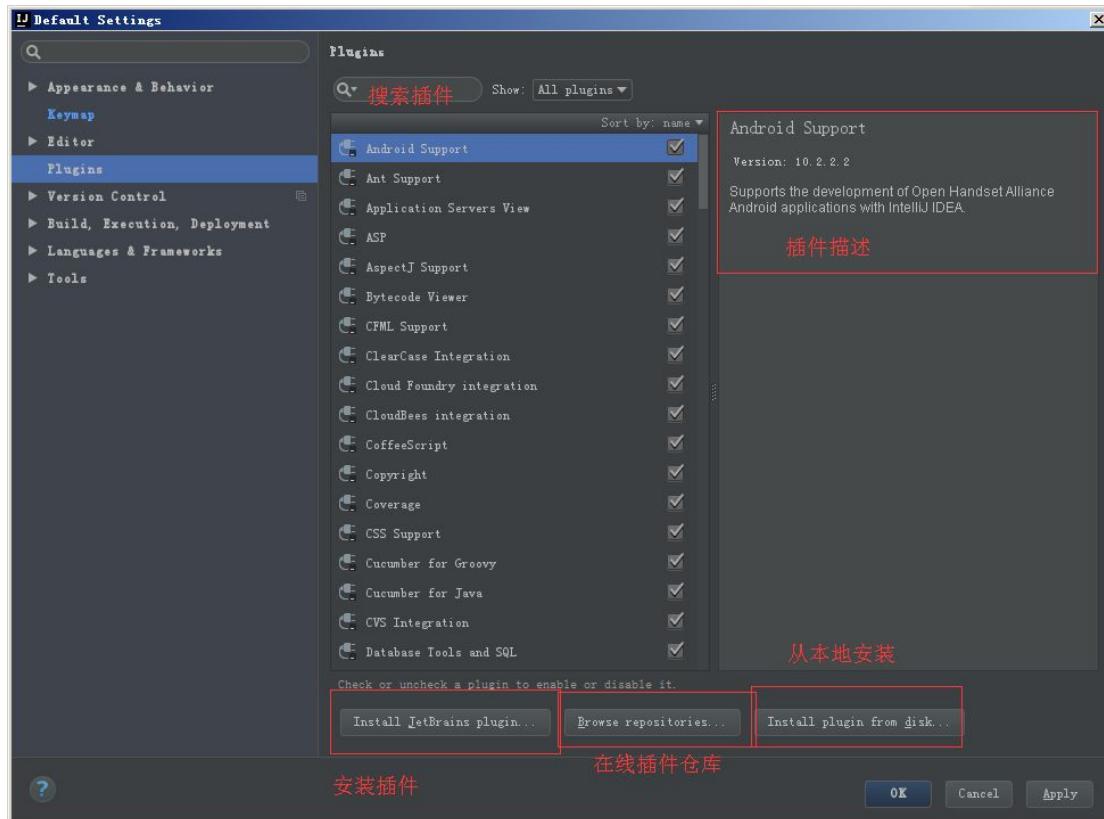
拼写， 默认即可

TODO

待办事项， 默认即可

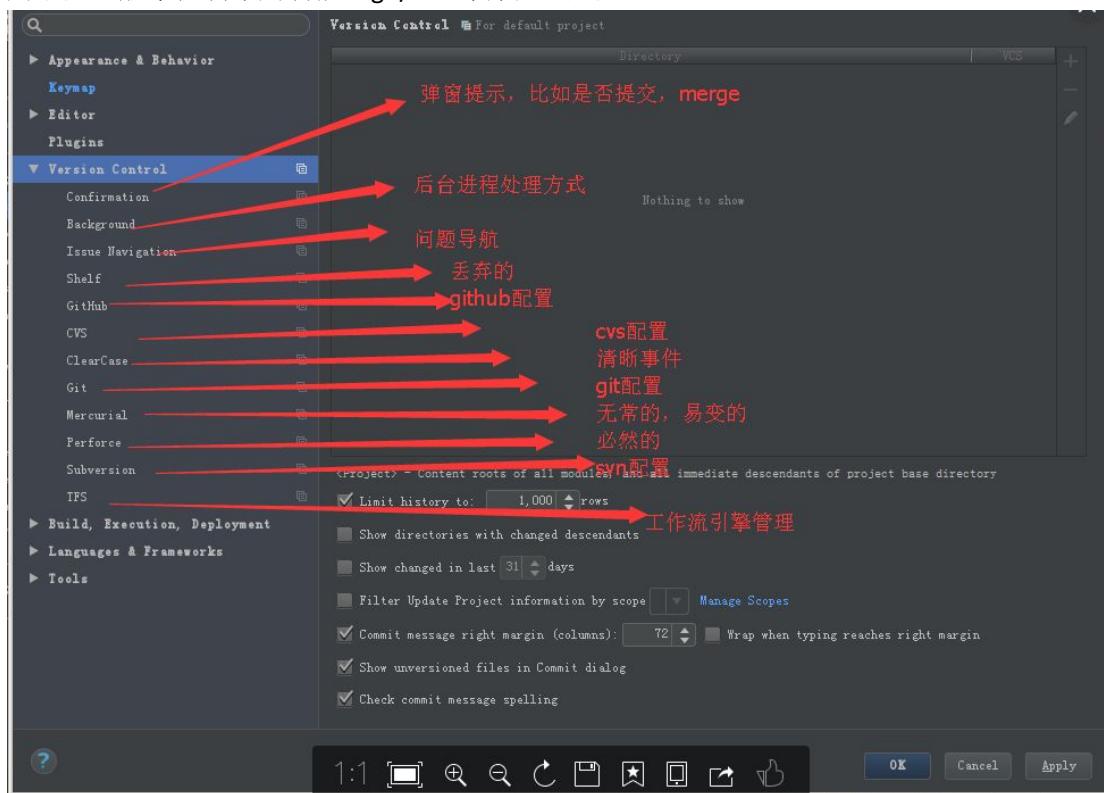
Plugins (插件)

用于取消插件启动， 安装在线和本地插件等。



Version Control (版本控制)

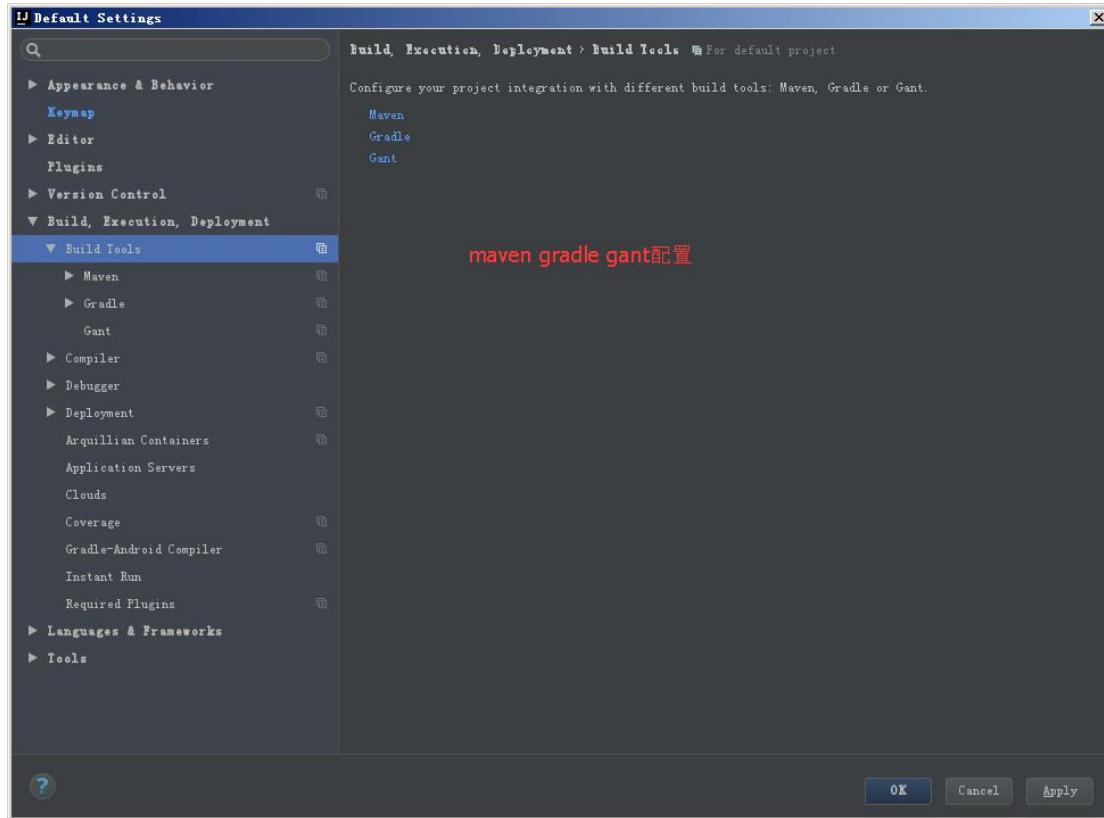
用于配置版本控制常用功能，git/svn 等都在此处配置。



Build Execution Deployment (构建执行部署)

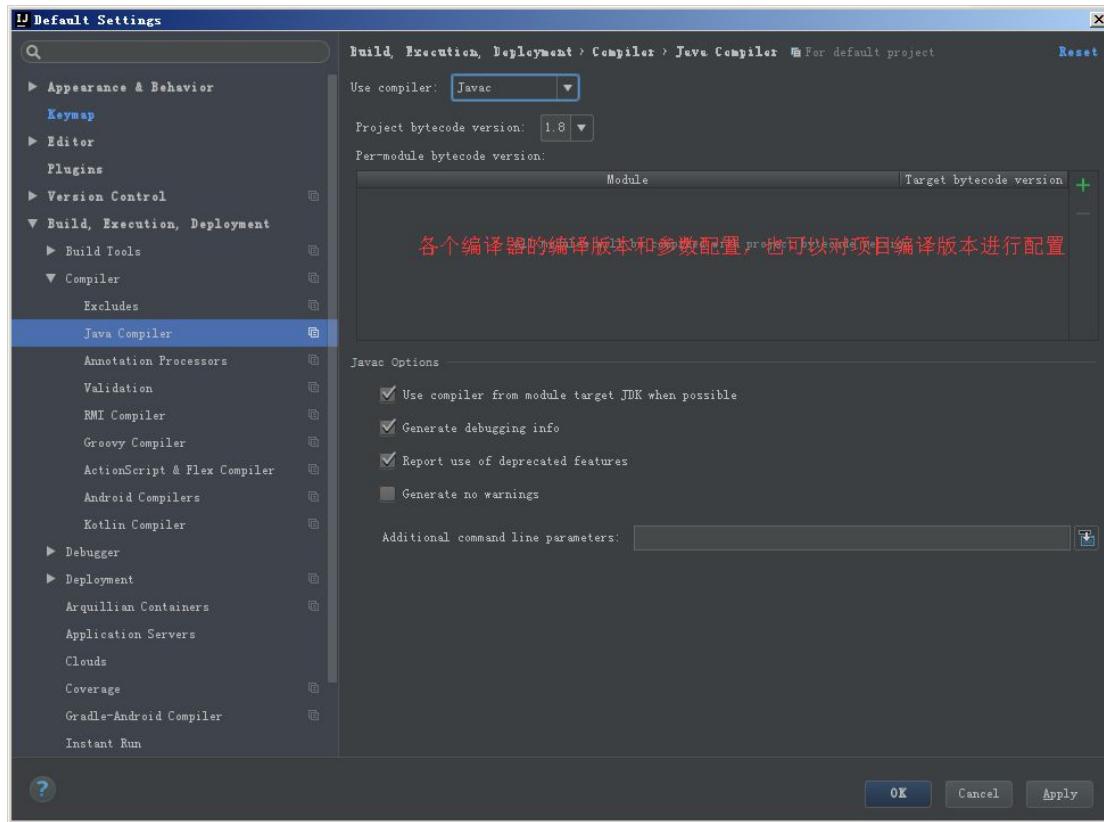
Build Tools (构建工具)

构建工具，内置对 maven, gradle 和 ant 的支持。



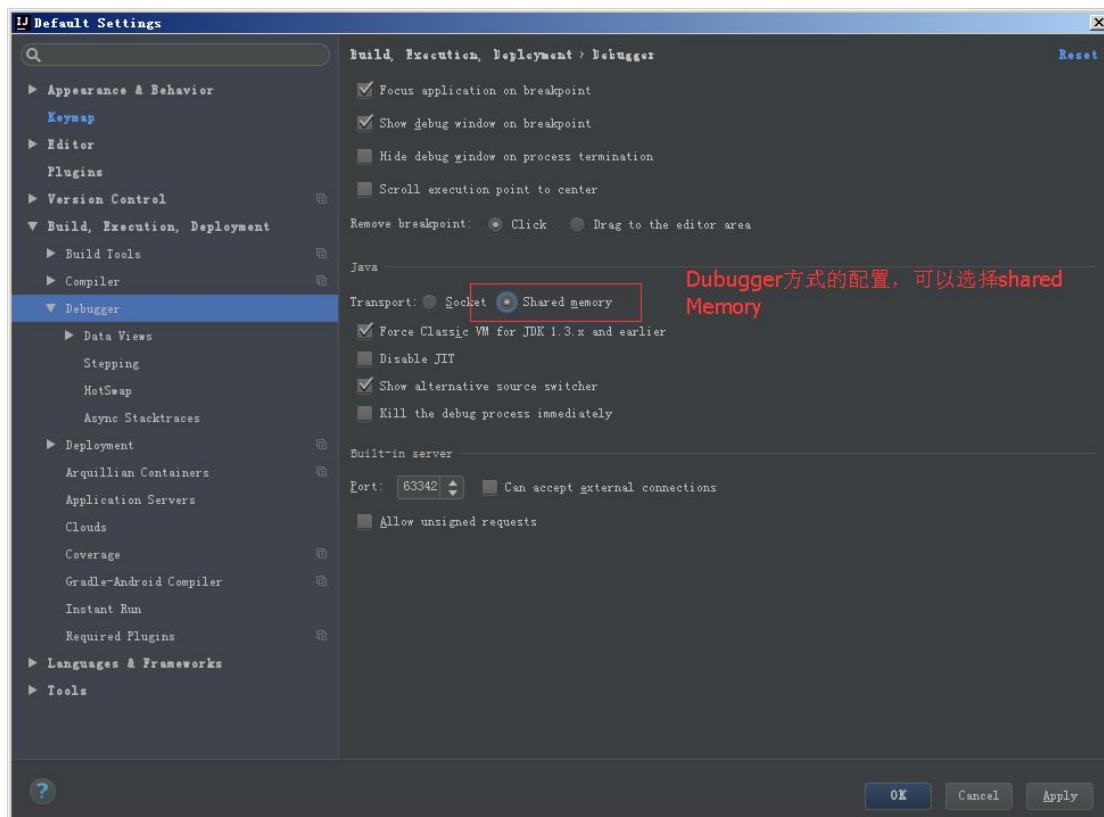
Compiler (编译)

这里可以对编译级别进行选择，包括项目（Module）的编译级别。



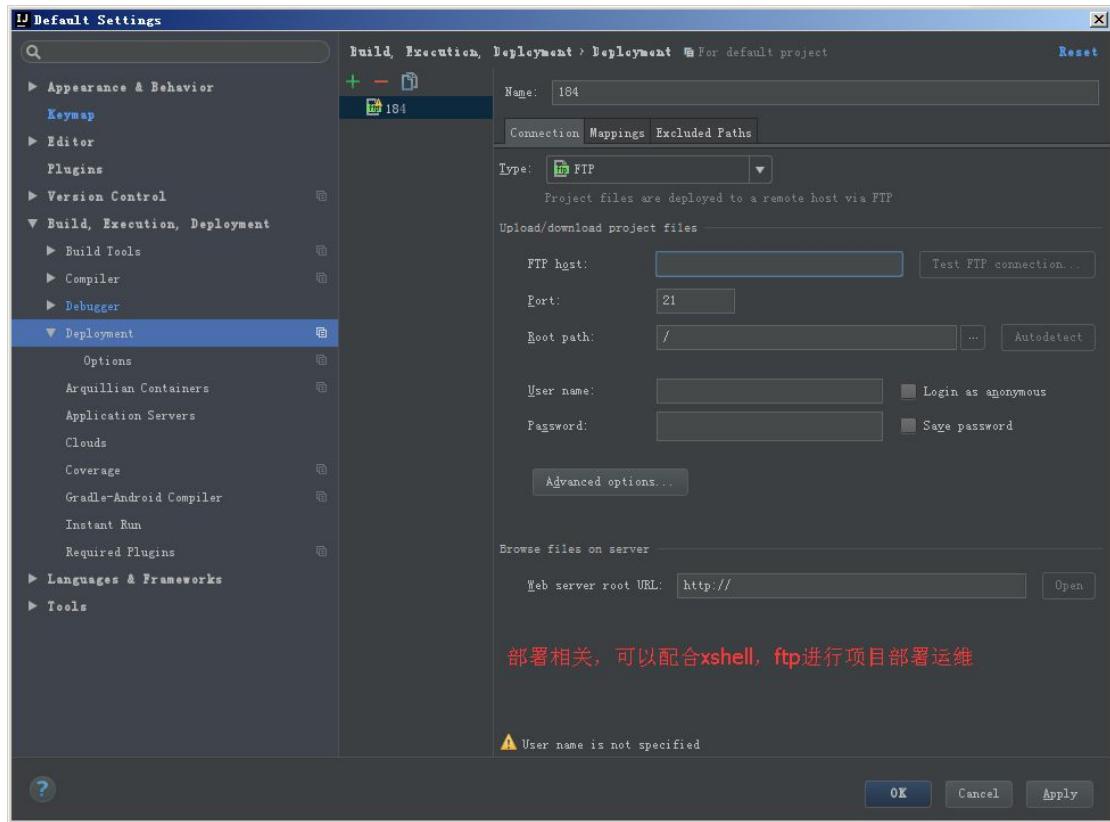
Dubugger (调试)

调试模式的配置，热部署也是基于此。



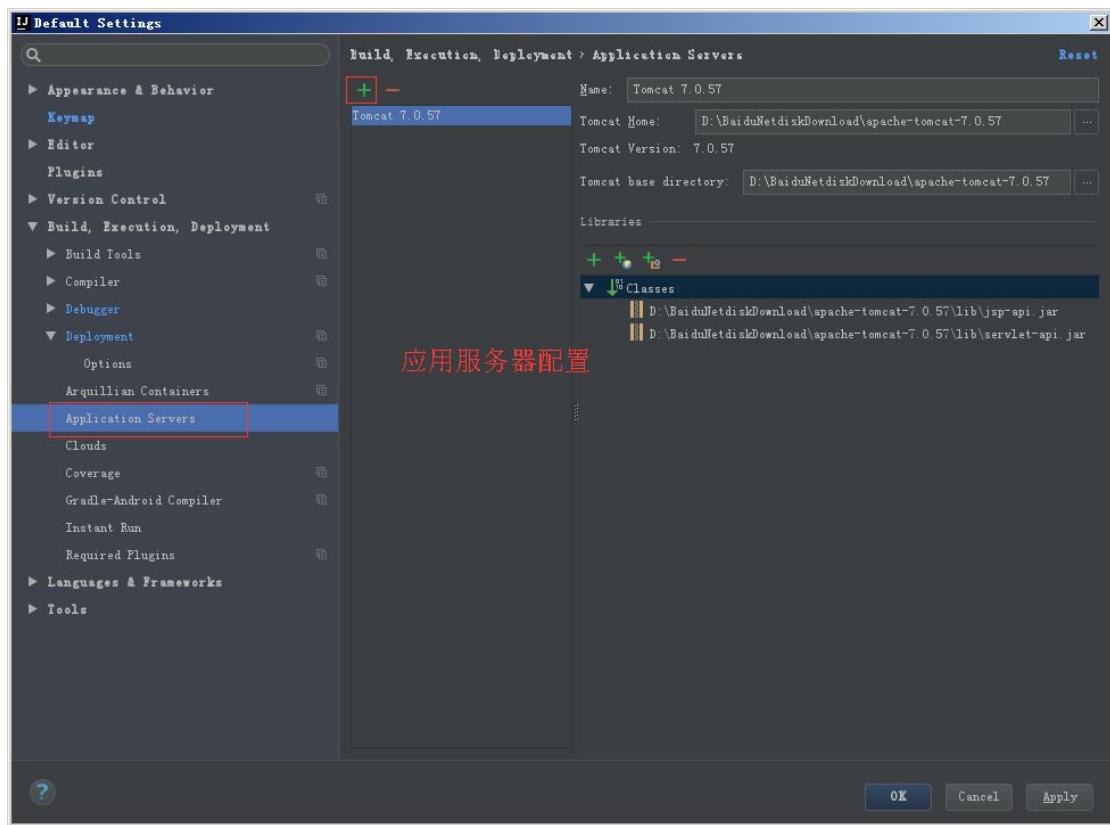
DeployMent (部署)

配合 ssh 等方便部署使用。



ApplicationServer (应用服务器)

这里可以配置 tomcat、jetty、jboss 等服务器。

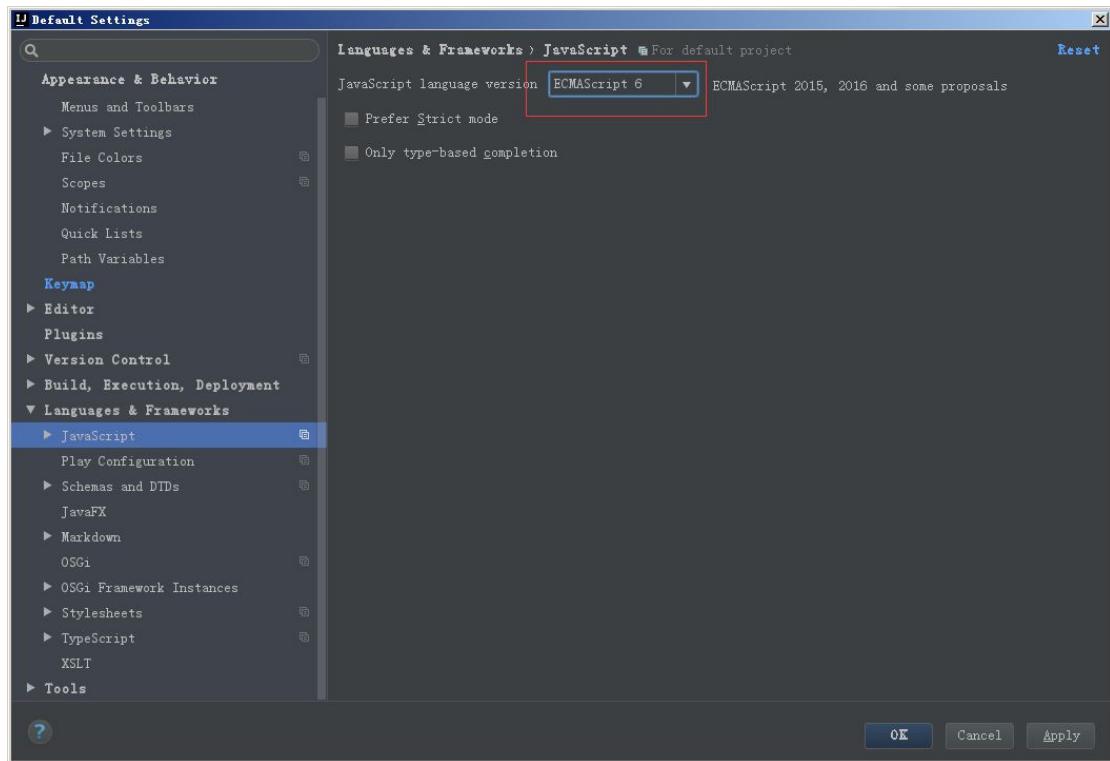


其他

其他默认即可

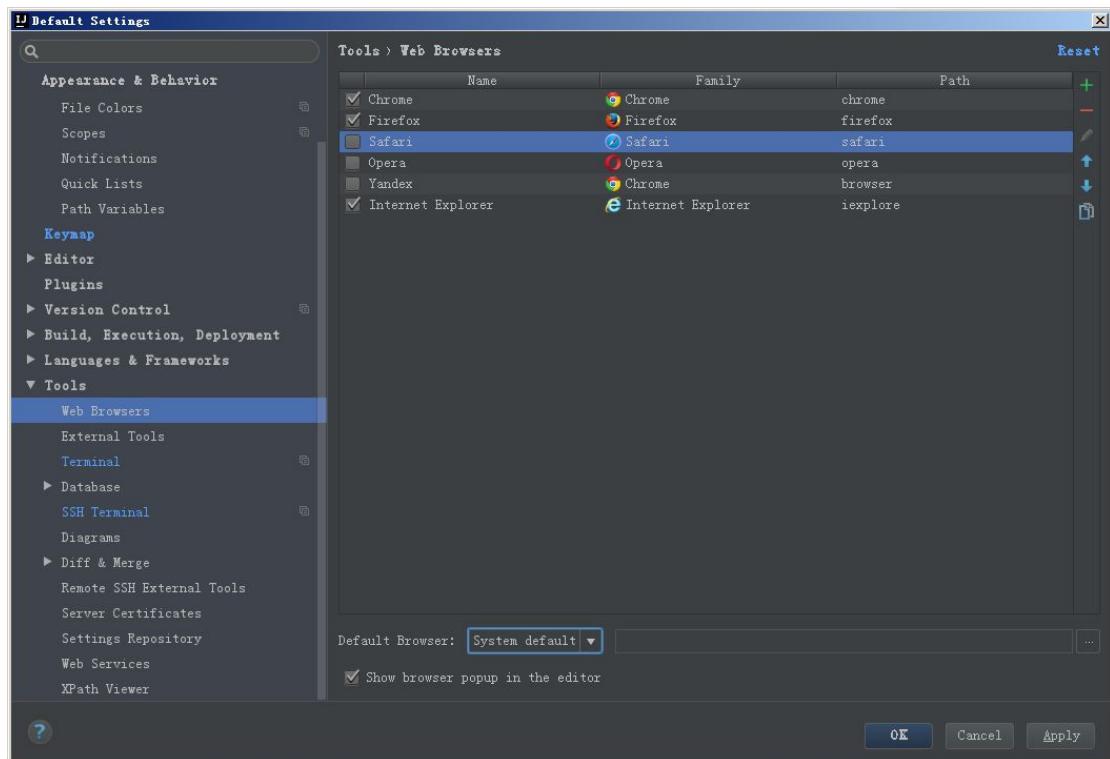
Languages&Frameworks

此模块默认配置即可，功能是配置语言编译版本，比如下面



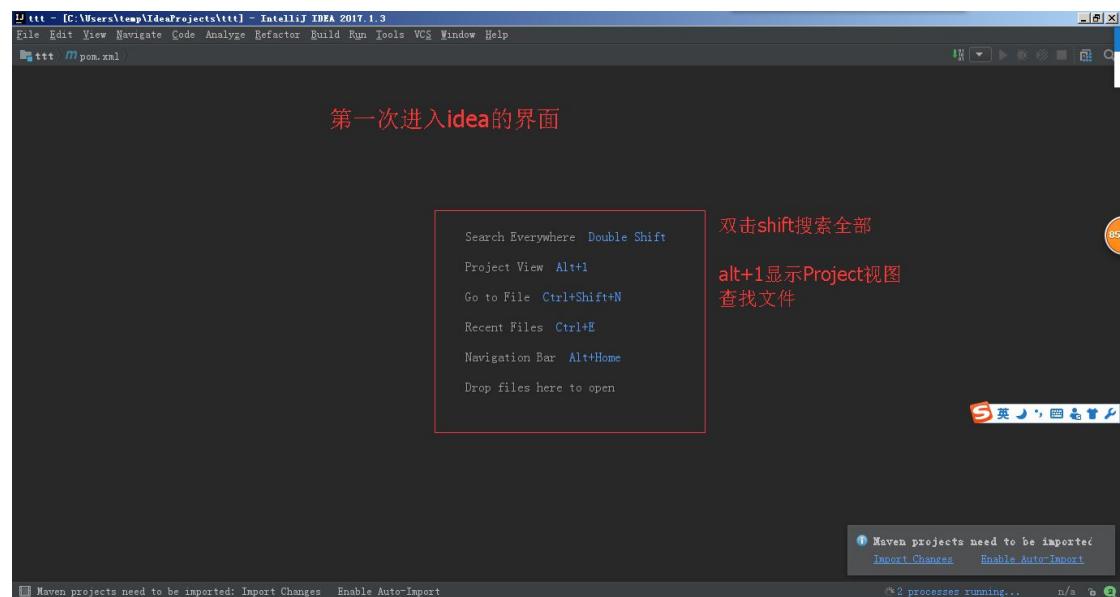
Tools

常用工具， 默认即可

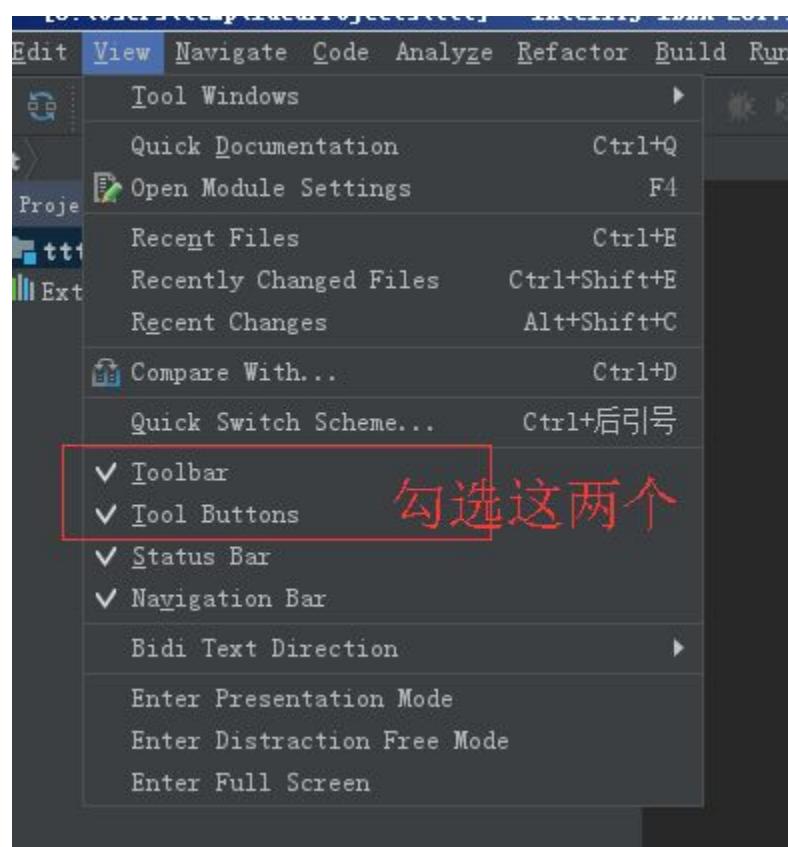


第一次启动后

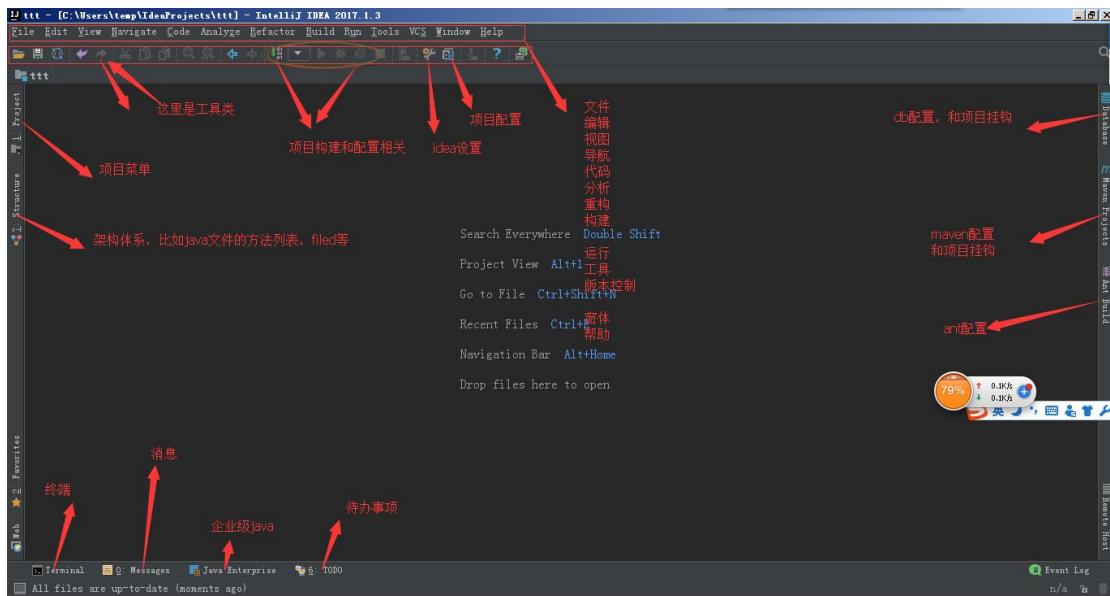
首次进入项目后的一些面板说明。



调出面板和按钮组



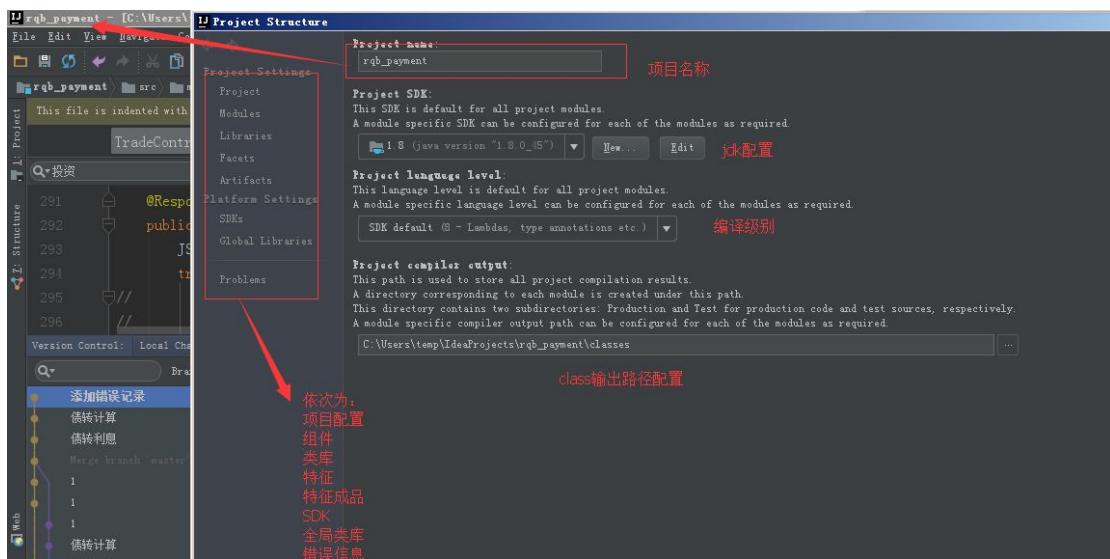
面板说明



项目配置

Project (项目)

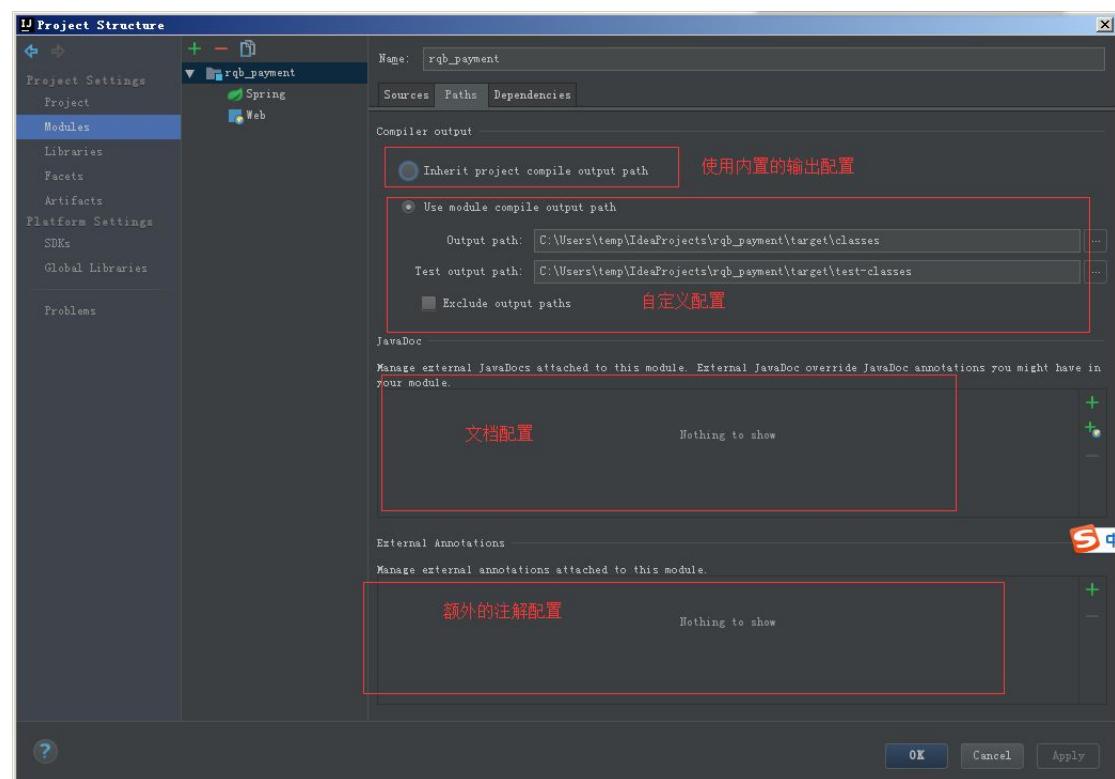
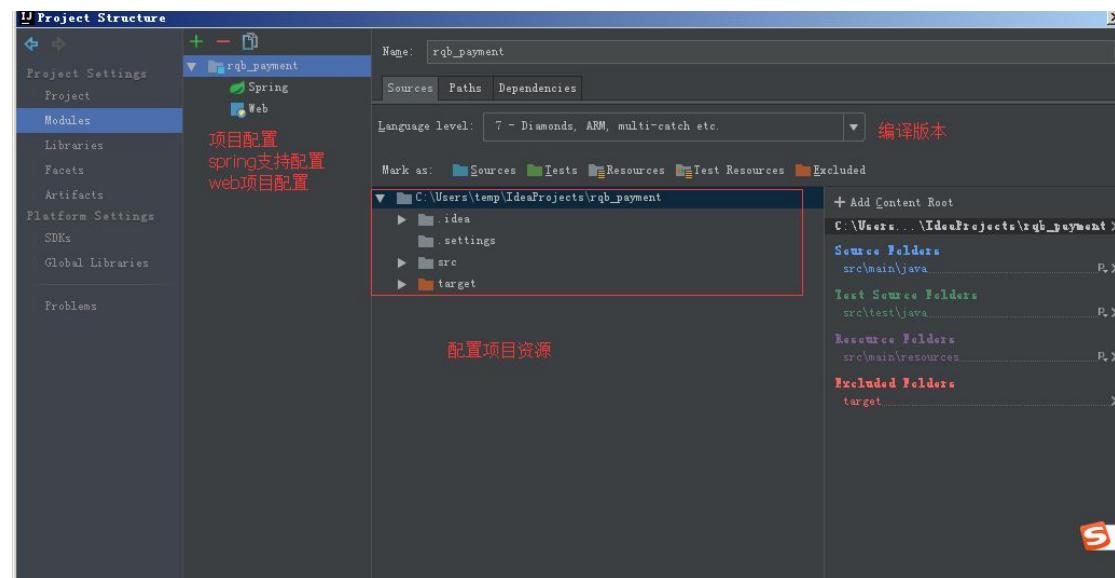
配置项目名, jdk, class 目录等

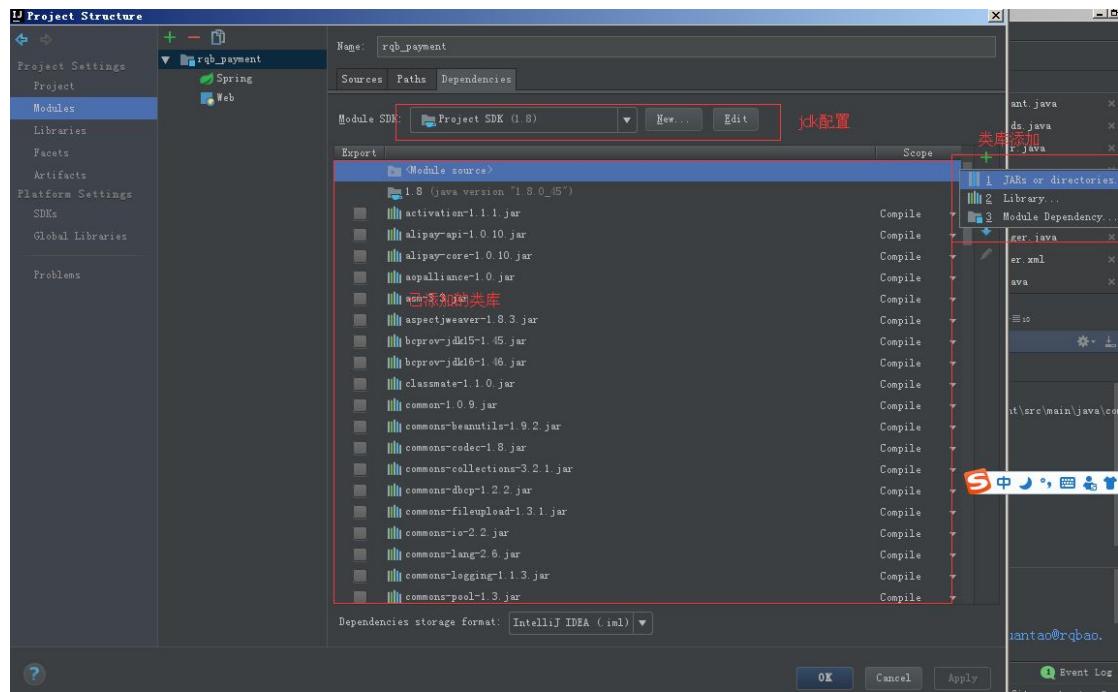


Modules (模块)

项目模块，有整个项目模块的配置和 web、Spring 级别的。

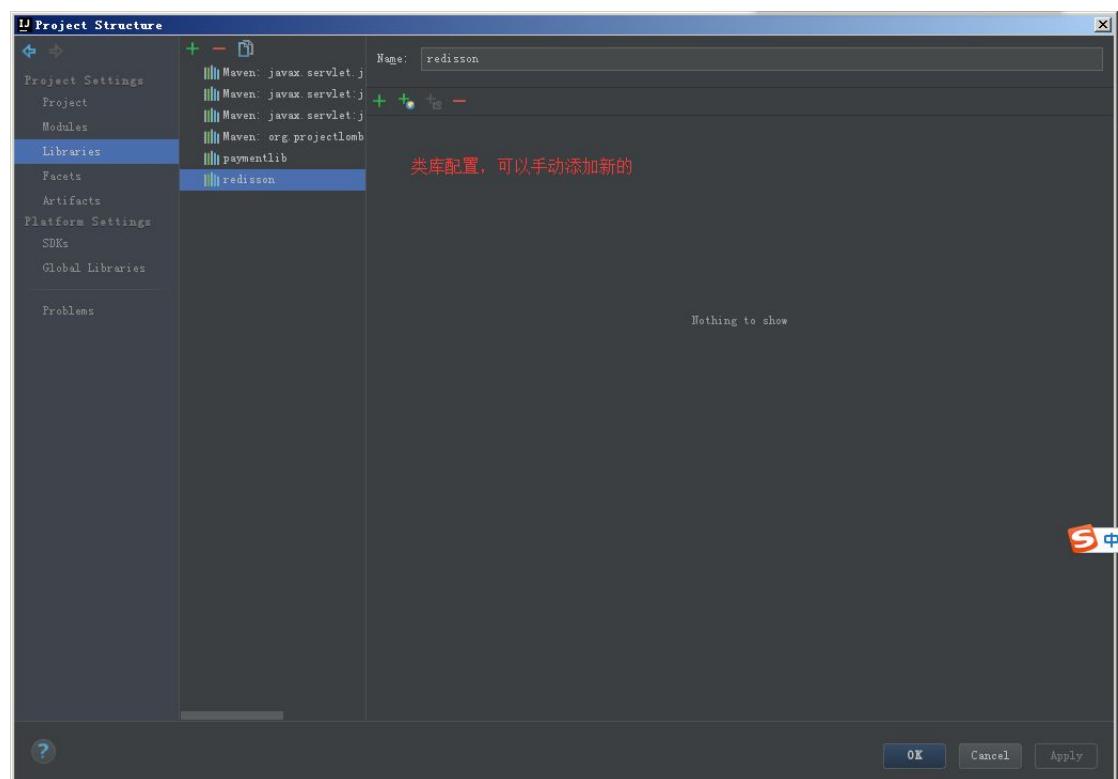
Web 级别是基础，可以配置 web.xml, web 目录，以及创建 artifact (第一次项目的时候需要配置此处)。





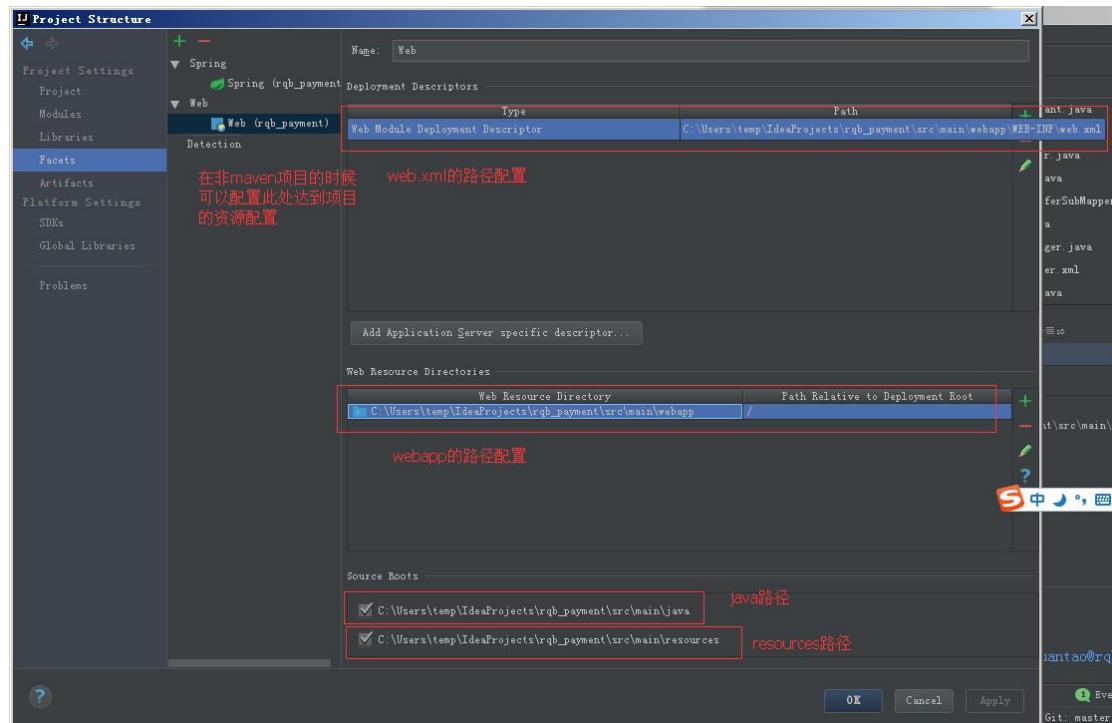
Libraries (类库)

类库，项目依赖的类库



Facets (特征)

表示这个 module 有什么特征，比如 Web, Spring 和 Hibernate 等；



Artifacts (打包)

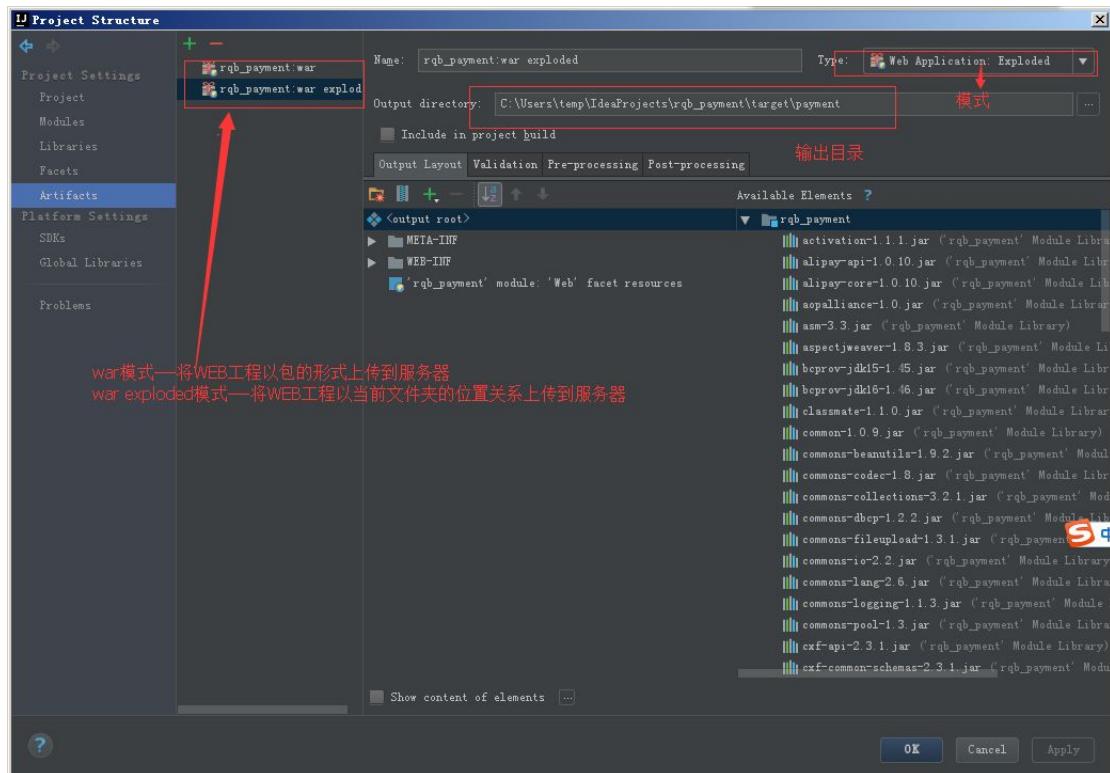
Artifact 是 maven 中的一个概念，表示某个 module 要如何打包，例如 war exploded、war、jar、ear 等等这种打包形式；

一个 module 有了 Artifacts 就可以部署到应用服务器中了！

在给项目配置 Artifacts 的时候有好几个 type 的选项，exploded 是什么意思：

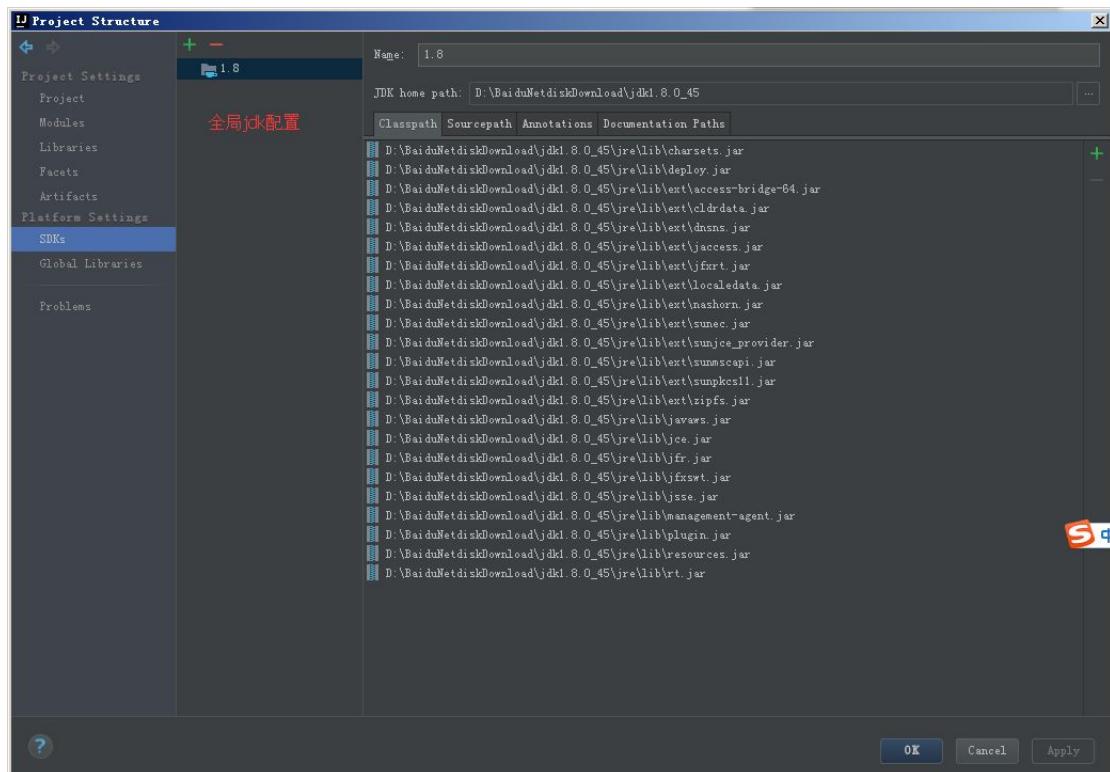
explode 在这里你可以理解为展开，不压缩的意思。也就是 war、jar 等产出物没压缩前的目录结构。建议在开发的时候使用这种模式，便于修改了文件的效果立刻显现出来。

默认情况下，IDEA 的 Modules 和 Artifacts 的 output 目录 已经设置好了，不需要更改，打成 war 包 的时候会自动在 WEB-INF 目录 下生产 classes 目录，然后把编译后的文件放进去。



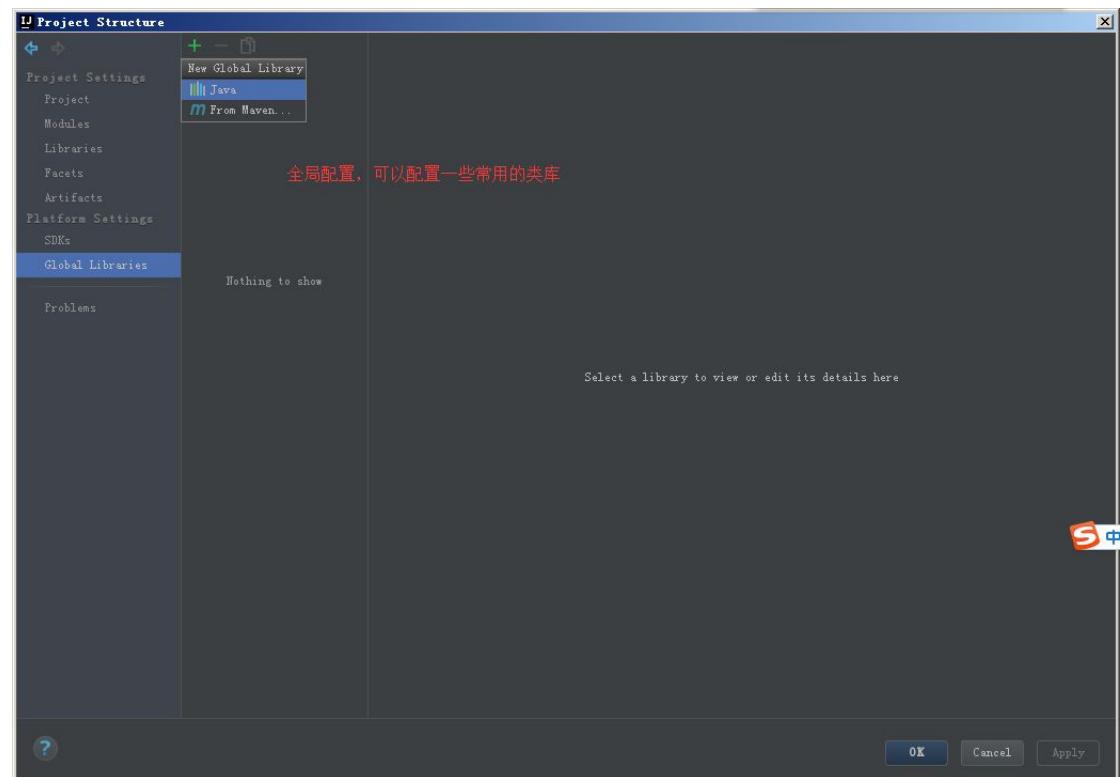
SDK (系统开发工具)

全局 SDK 配置



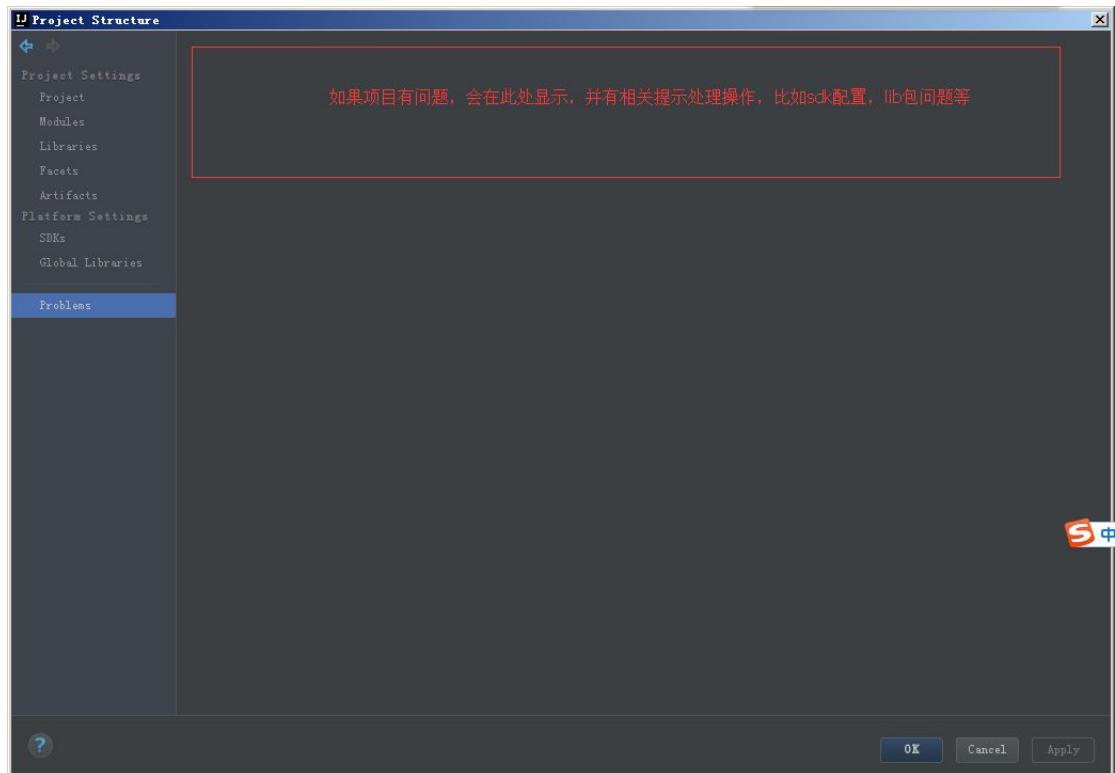
Global libraries (全局类库)

全局类库



Problems (问题)

问题，在项目异常的时候很有用，可以根据提示进行项目修复（**FIXED**）。



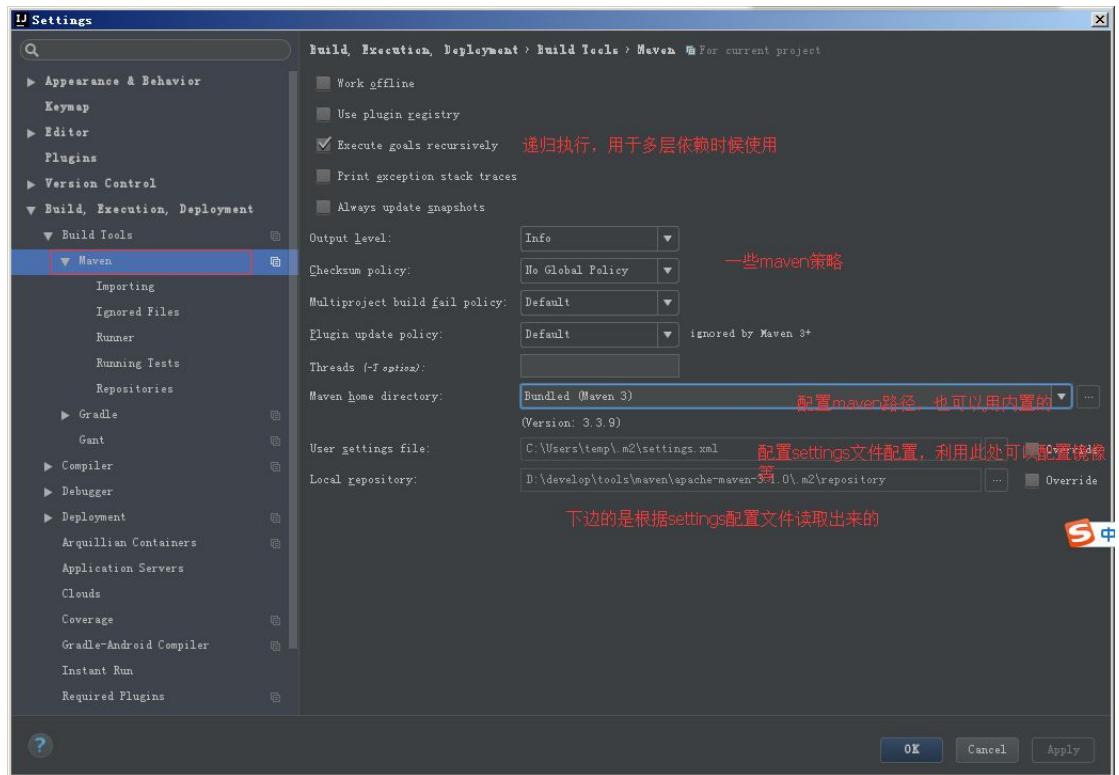
Maven 专题

如果项目是 maven 项目，在没被 maven 管控的时候，右键项目中的 pom 文件，选择 add as maven Project，即可加入 maven 管理

配置

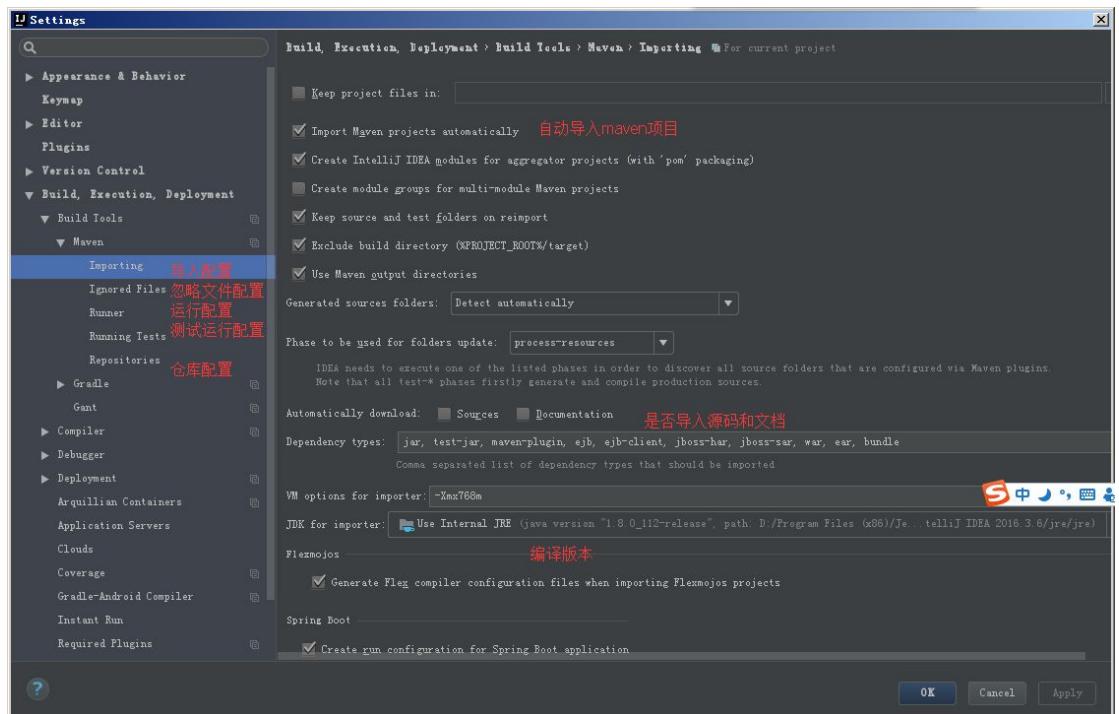
主配置

配置 maven 插件，以及工作方式。



Import 配置

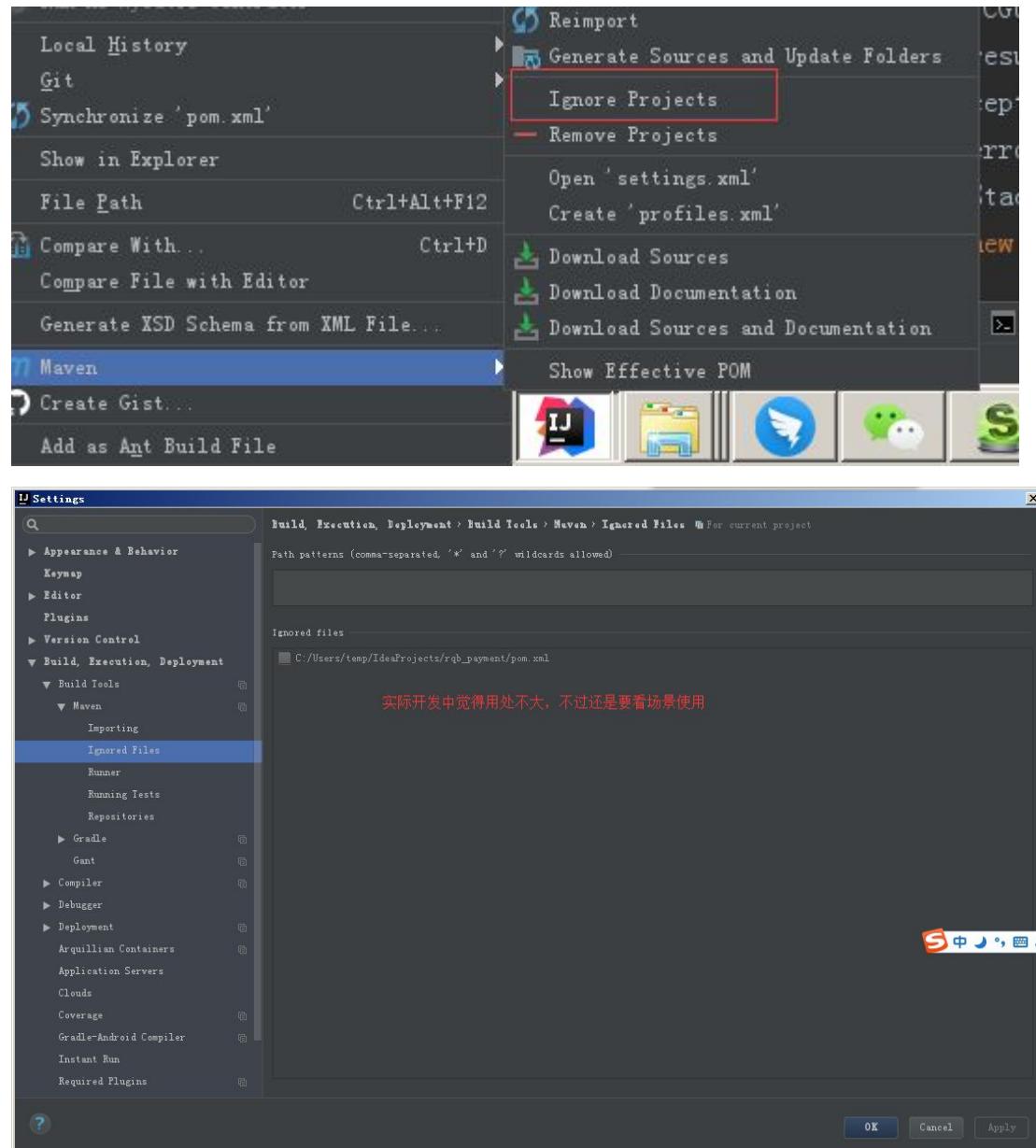
配置 maven 导入方式



Ignore Files 配置

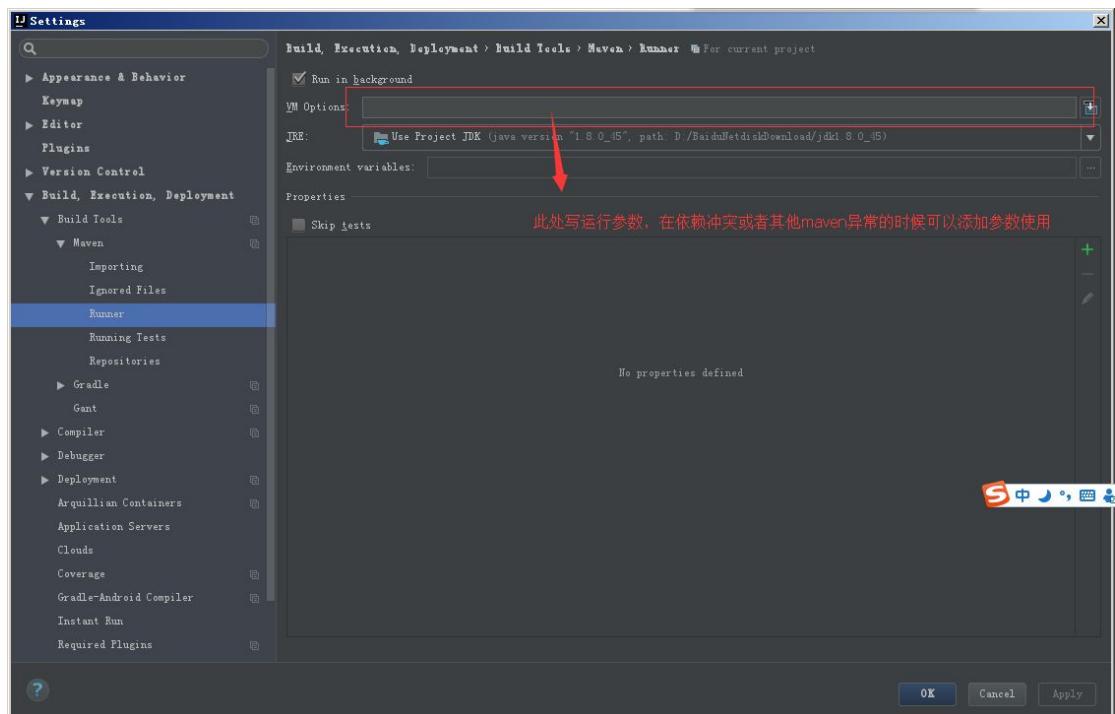
配置忽略文件。个人觉得用处不大。

通常与下面配合使用



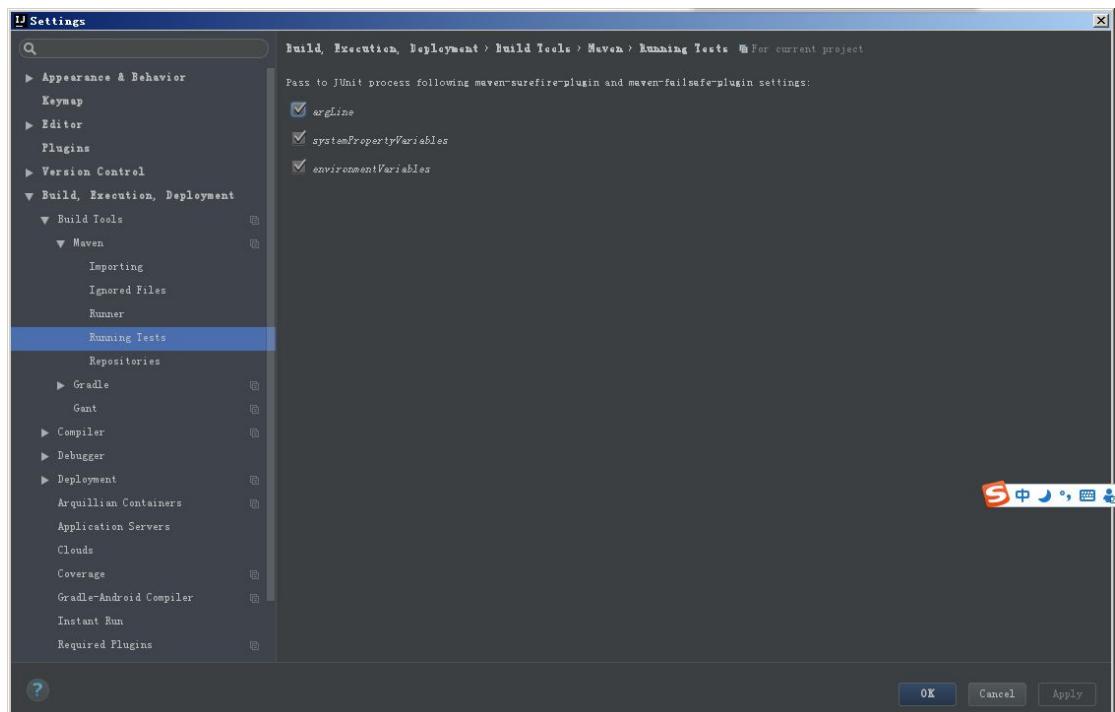
Runner 配置

运行配置，个人只在包冲突的时候用过，一般默认的即可。



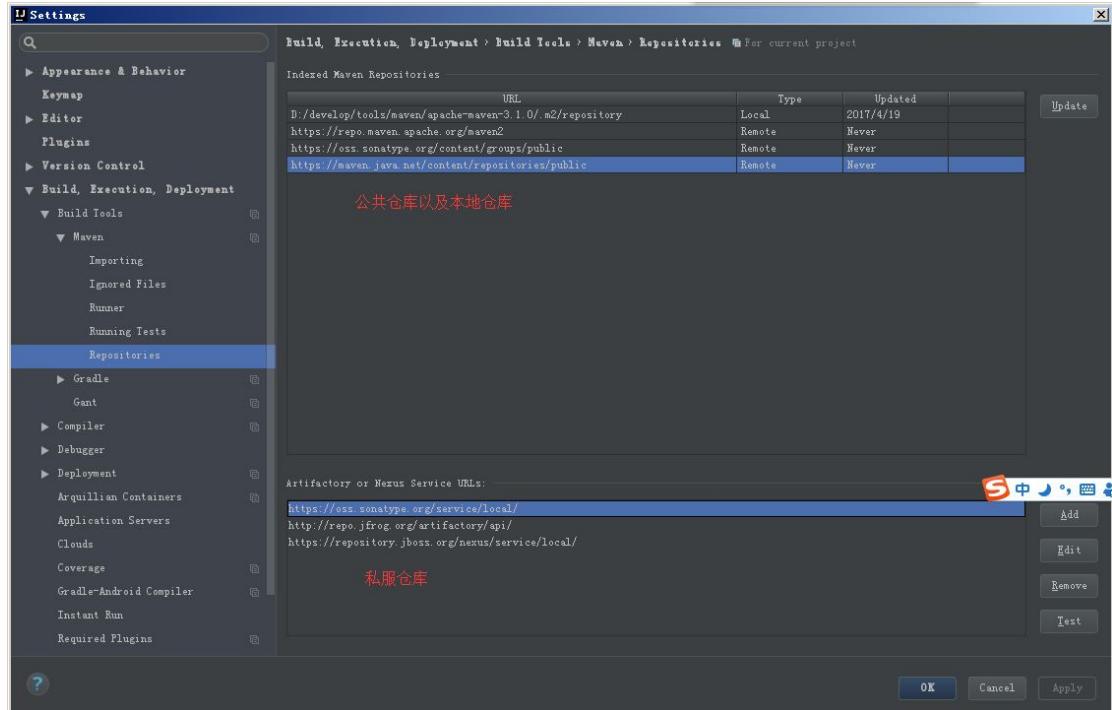
RunnerTest

保持默认即可。



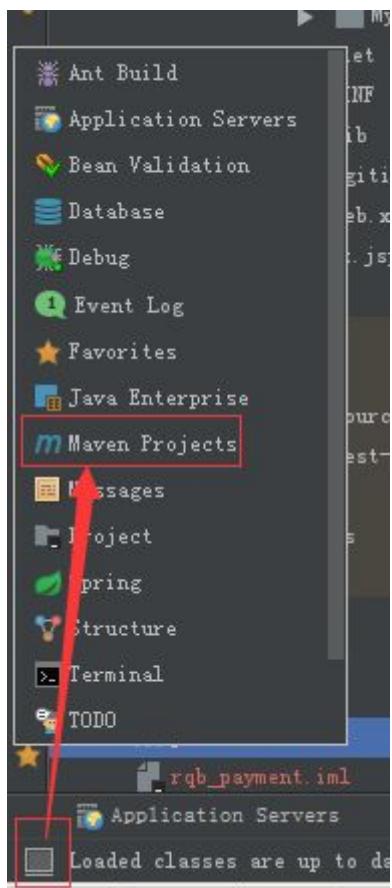
Repositories 配置

仓库的配置，以 settings.xml 优先。

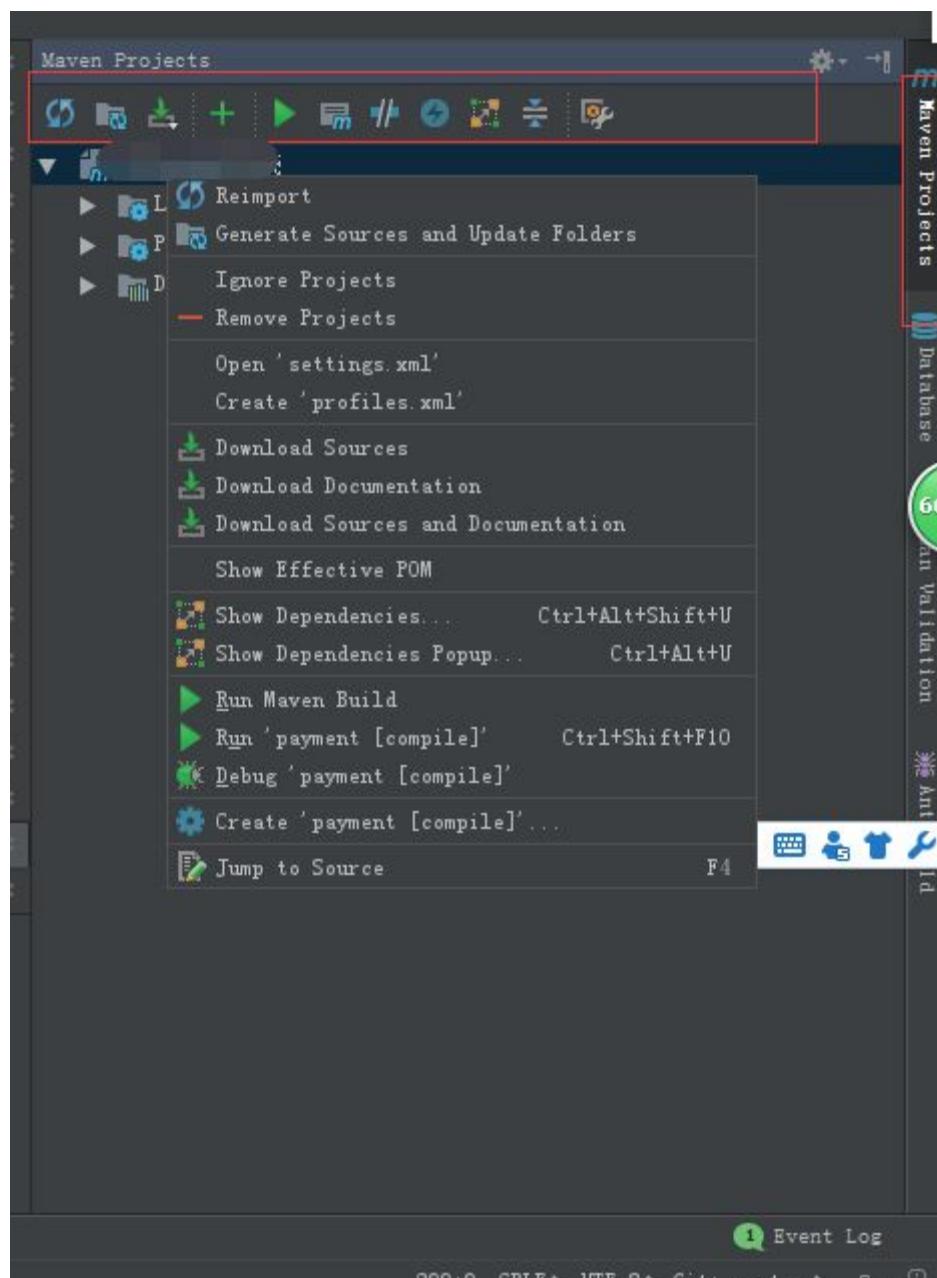


使用入门

通过如下方式调出面板



面板说明



上边一栏，依次为：



重新导入 maven 项目



对项目进行更新



下载文档和源码



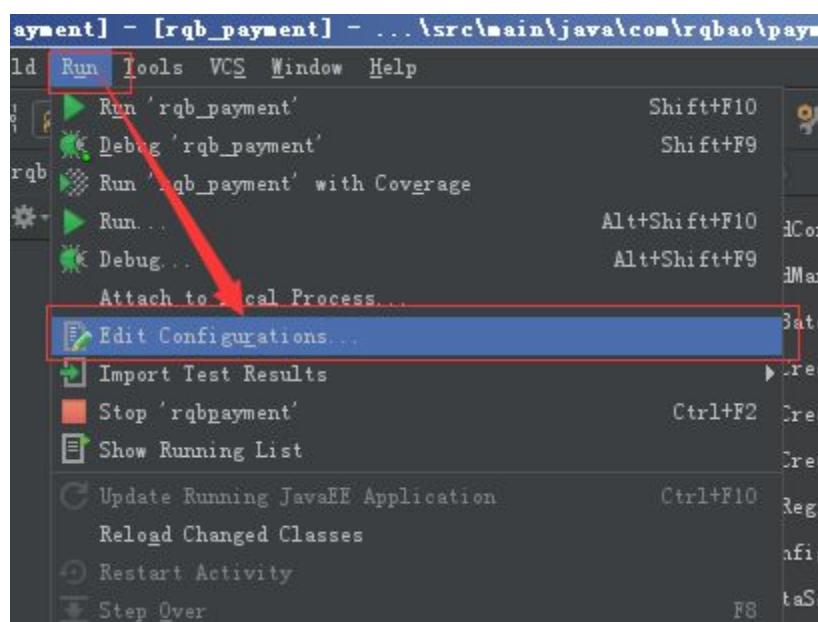
添加 maven 项目

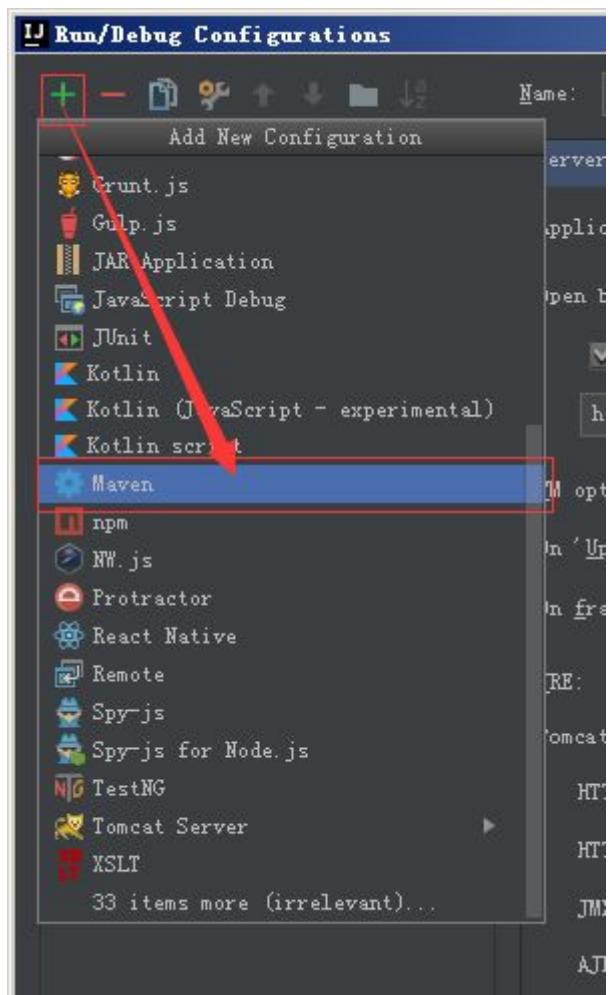


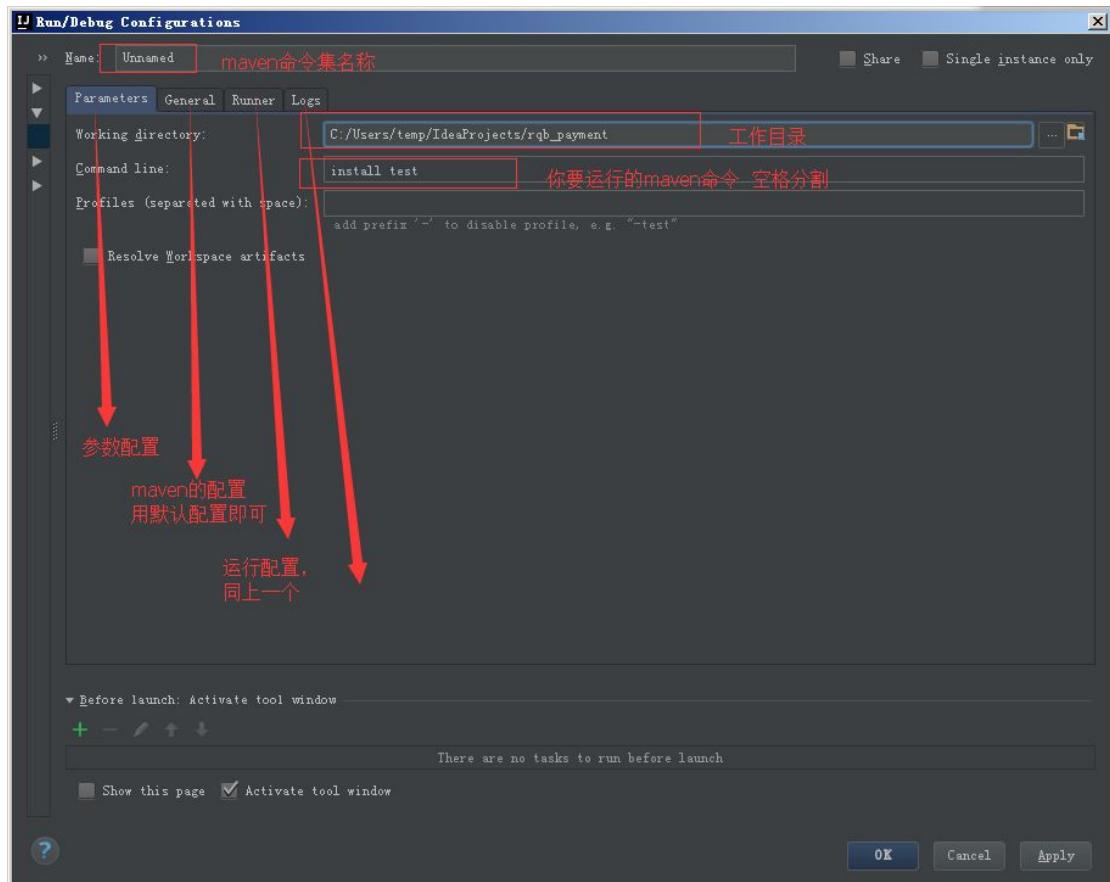
右键菜单的操作，基本类似

命令模式

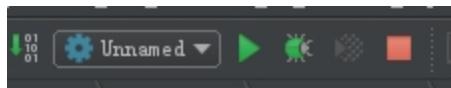
配置



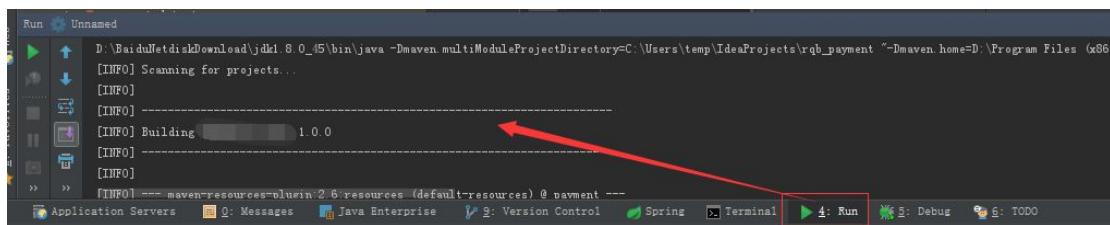




点击 ok 之后



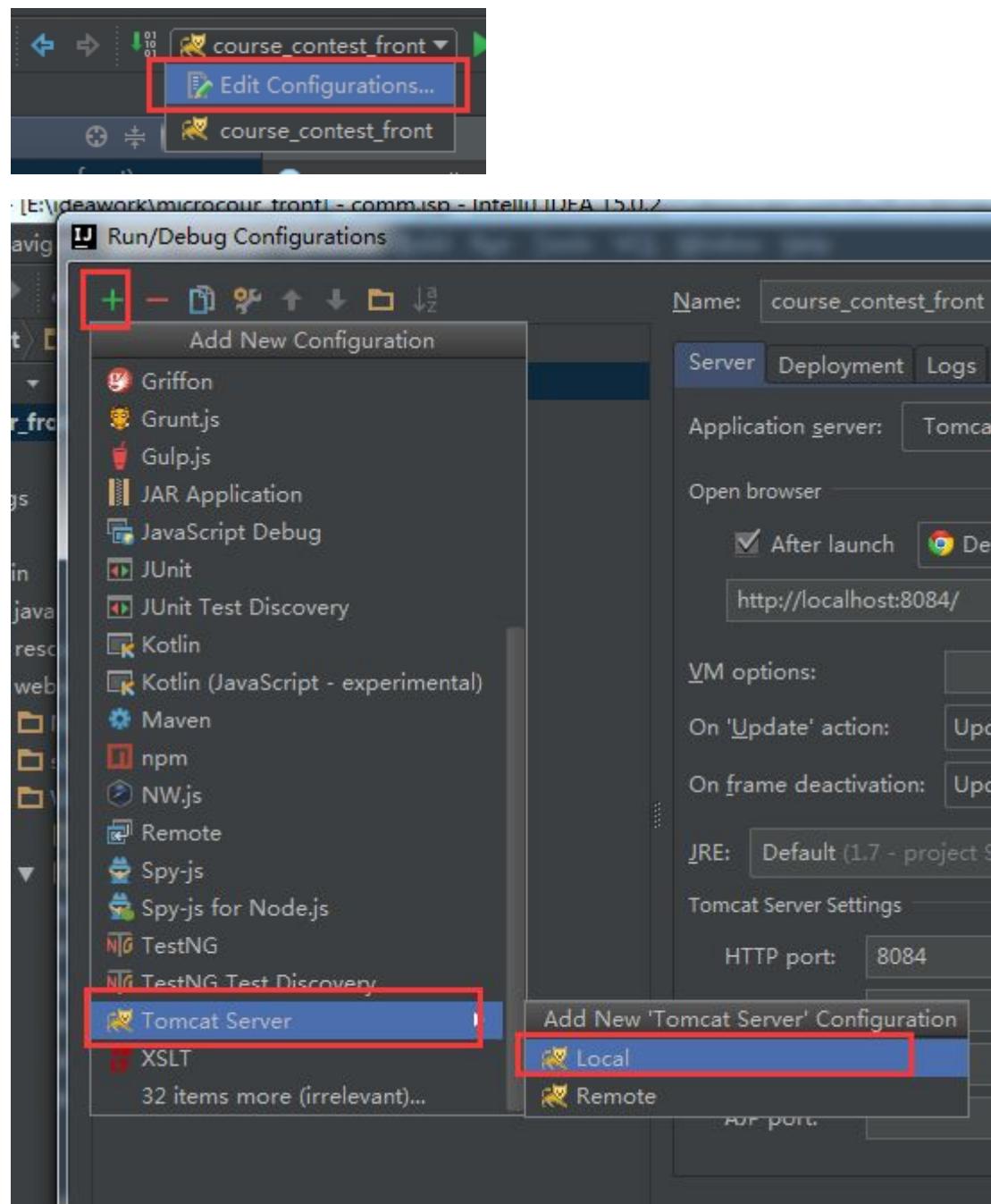
Run

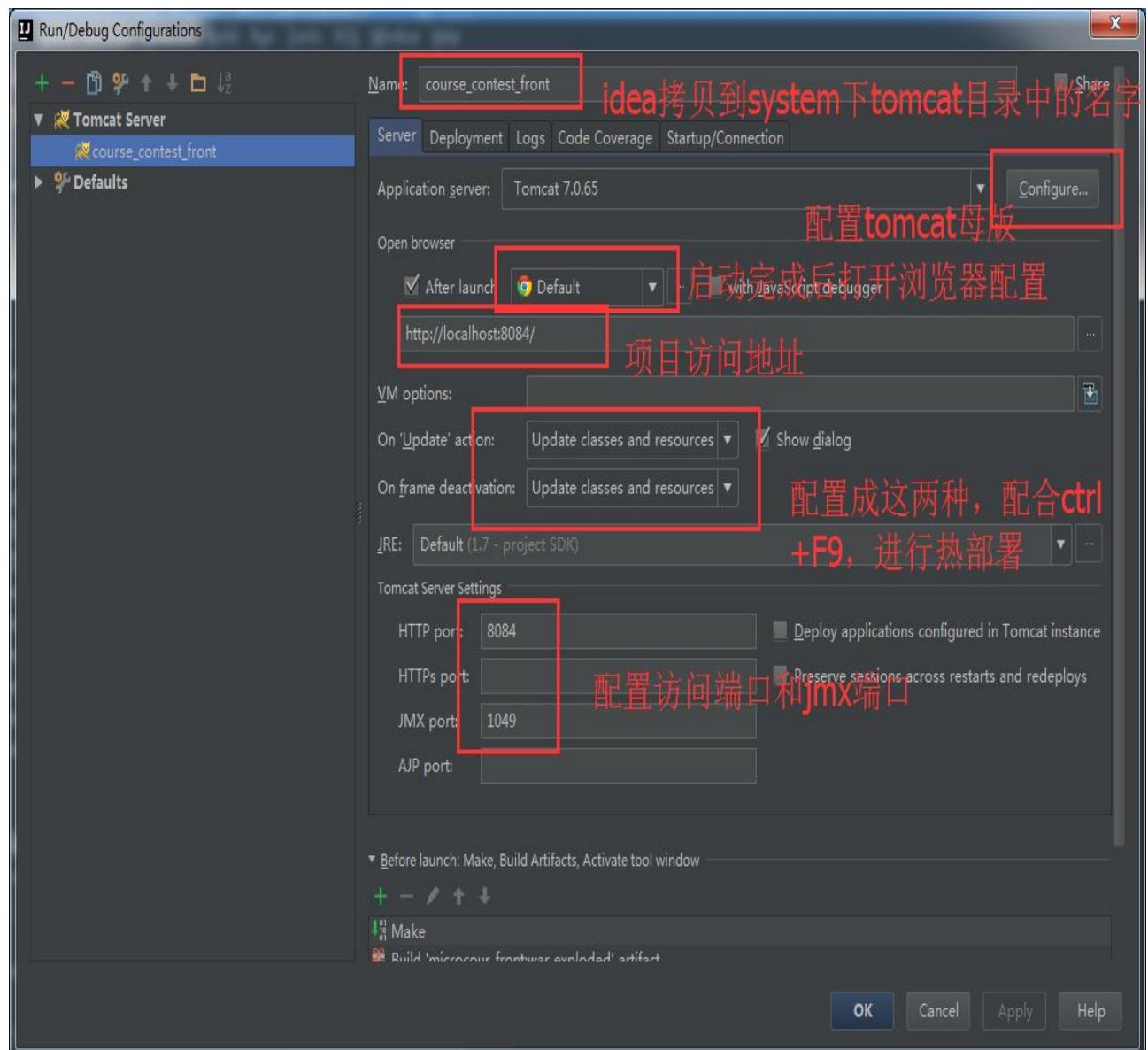


Tomcat 专题

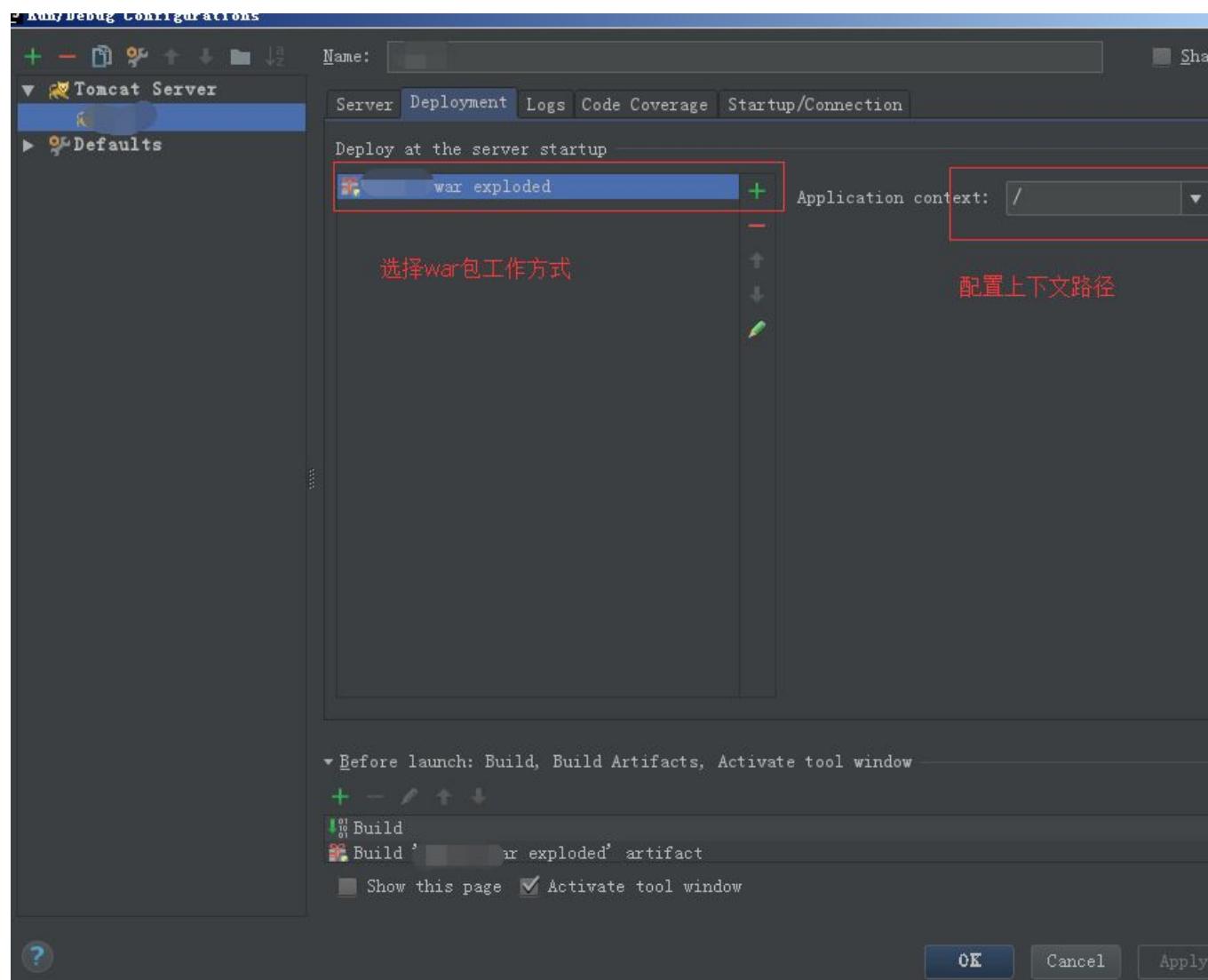
这里只做一个简单的入门指南，如有不对之处，还望指正。

安装配置



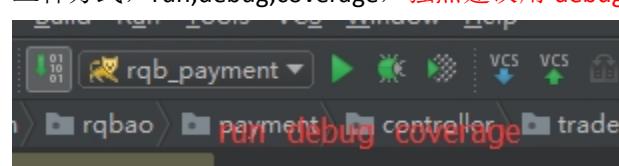


配置 DeployMent

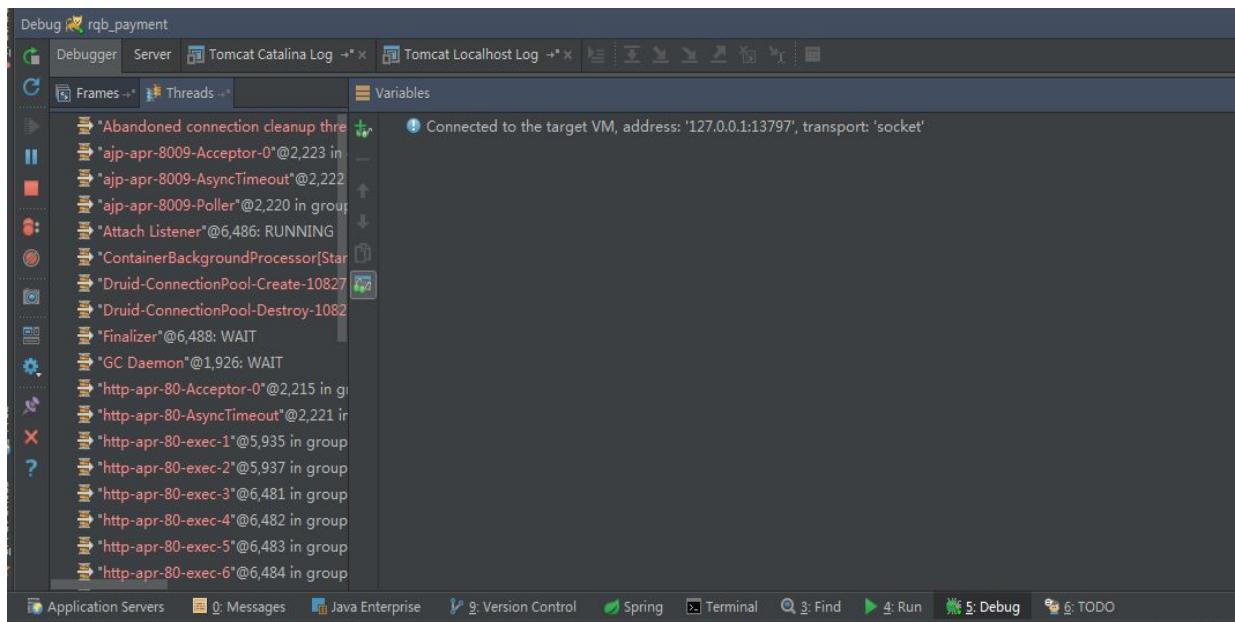


启动

三种方式, run, debug, coverage, 强烈建议用 debugger 模式启动



面板说明



上面一排

Debugger: debug 模式的时候显示方法，调用关系，参数值等，

Server: 打印日志控制台

tomcatCatalinaLog: catalina 日志，程序跑不起

来的时候可以查看此项

Localhostlog : 本地日志

并排的一堆按钮是调试的时候进入跳出方法按钮: F8 下一步,F7 进入

方法, F9 下一个断点

左侧竖列

依次为:

Run: 重新启动应用

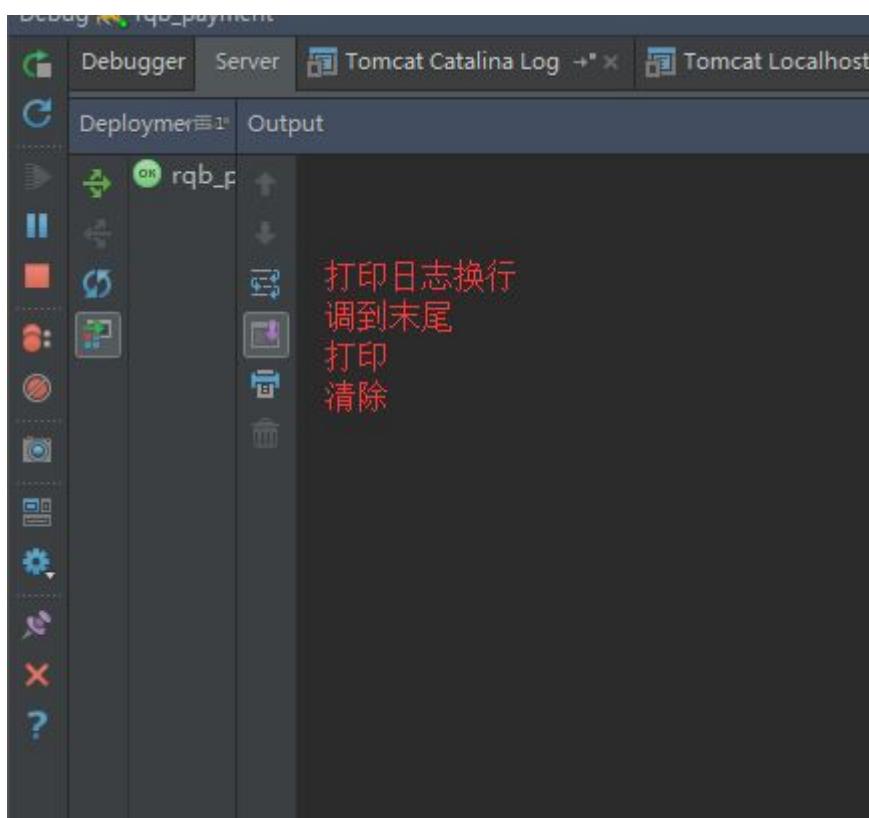
Update application: 更新应用

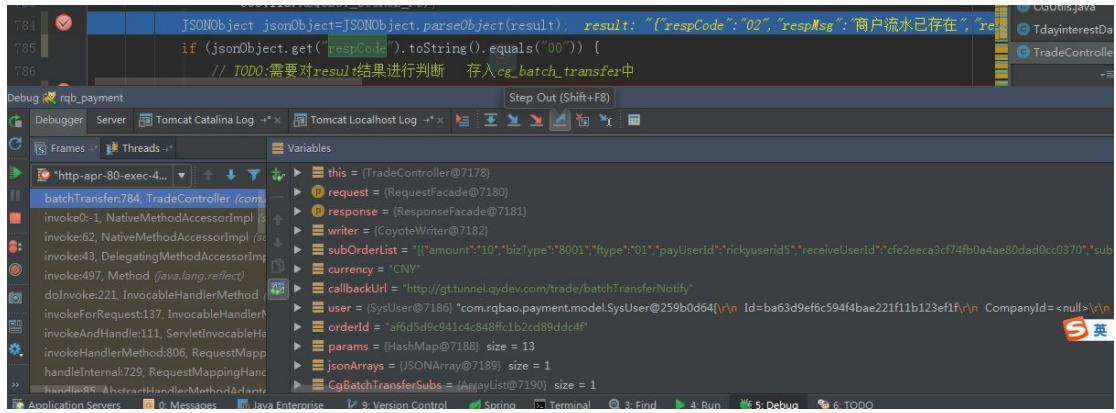
Resume application: 恢复应用

Pause application: 暂停应用

Stop: 停止应用

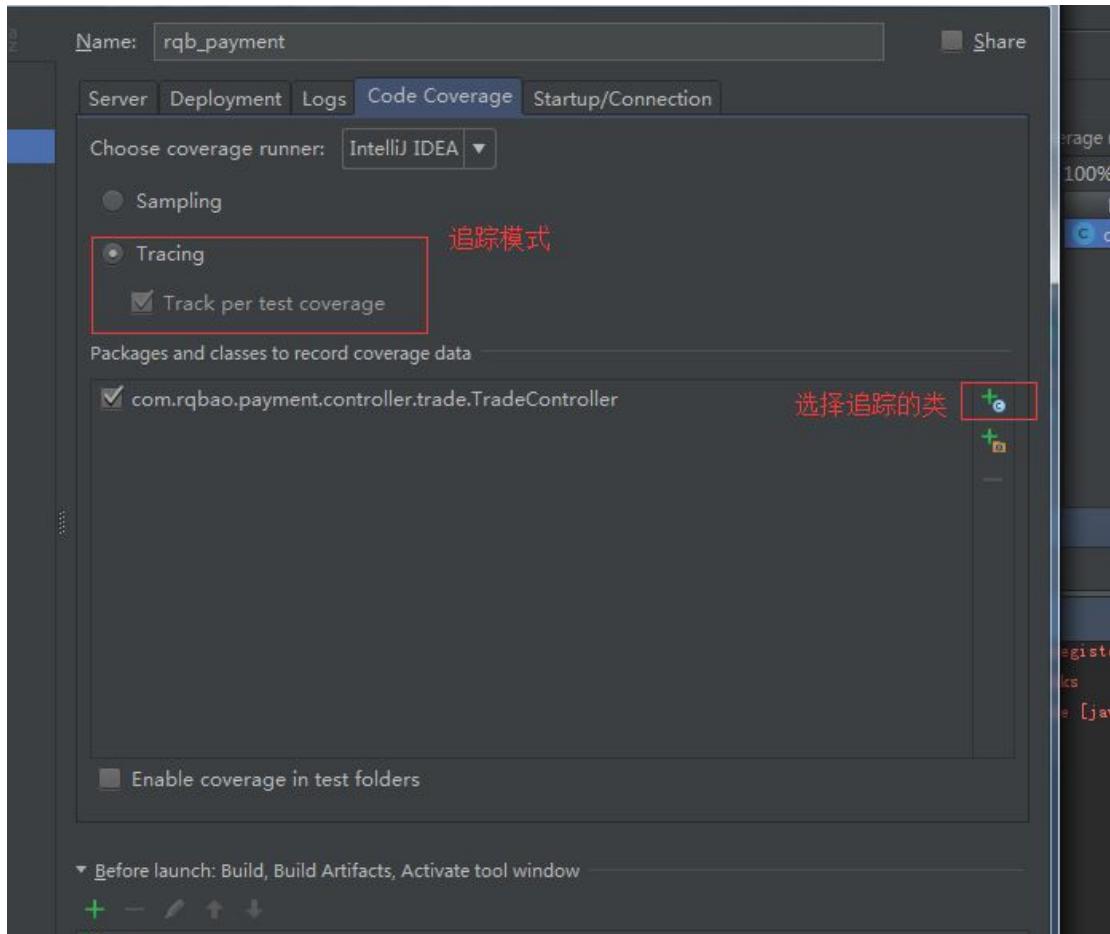
-  **View breadkPoints:** 查看断点
-  **Mute breadkpoints:** 禁用断点
-  **Get Thread Dump** 获取线程堆
-  **Restore layout:** 重置布局
-  **Settings:** 设置
-  **Pin tab:** 固定面板
-  **Close:** 关闭
-  **Help:** 帮助



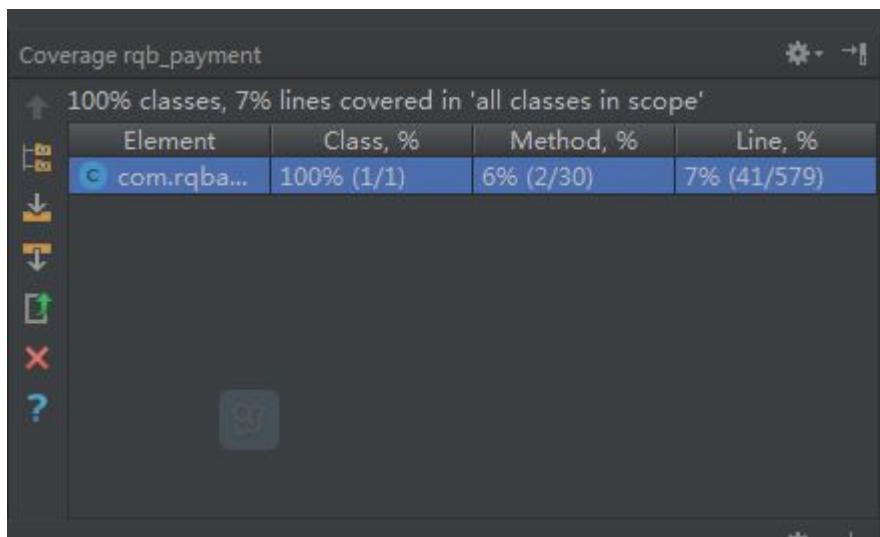


Run with coverage

覆盖模式运行。用于统计方法使用情况。



终止程序后会显示调用比例



Tomcat 集成原理

用户为项目配置了 tomcat 后，idea 会拷贝一份配置到系统目录中，如下



Conf (配置)

此处的端口和项目都和前边配置的一致。

本地磁盘 (D:) \ bgtidea \ .IntelliJIdea \ system \ tomcat \ Unnamed_SSMDEMO_3 \ conf \

V) 工具 (T) 帮助 (H)

新建文件夹

名称	修改日期	类型	大小
Catalina	2017/5/22 15:05	文件夹	
catalina.policy	2014/11/3 11:43	POLICY 文件	13 KB
catalina.properties	2017/5/22 15:05	PROPERTIES 文件	7 KB
catalina.properties.0	2014/11/3 11:43	0 文件	7 KB
context.xml	2017/4/20 12:20	XML 文档	2 KB
logging.properties	2014/11/3 11:43	PROPERTIES 文件	4 KB
server.xml	2017/5/22 15:05	XML 文档	7 KB
server.xml.0	2017/5/22 14:24	0 文件	7 KB
server.xml.1	2017/5/22 15:05	1 文件	7 KB
tomcat-users.xml	2017/4/20 12:20	XML 文档	2 KB
web.xml	2017/5/22 15:05	XML 文档	148 KB
web.xml.0	2017/4/20 12:20	0 文件	160 KB

Tomcat 之所以可以 root 启动和热部署，原因在如下

计算机 \ 本地磁盘 (D:) \ bgtidea \ .IntelliJIdea \ system \ tomcat \ Unnamed_SSMDEMO_3 \ conf \ Catalina \ localhost

V) 工具 (T) 帮助 (H)

新建文件夹

名称	修改日期	类型	大小
ROOT.xml	2017/5/22 15:06	XML 文档	1 KB

D:\bgtidea\ .IntelliJIdea\system\tomcat\Unnamed_SSMDEMO_3\conf\Catalina\localhost\ROOT.xml - Sublime Text

文件 (F) 编辑 (E) 选项 (O) 查找 (T) 查看 (V) 转到 (G) 工具 (T) 项目 (P) 首选项 (Q) 帮助 (H)

idea.prop ✘ idea.exe.w ✘ 消息队列 ✘ ROOT.xml ✘ 工作密码 ✘ 工作任务 ✘ 放款业务 ✘ 瑞钱宝标 ✘ p2p

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <Context path="" docBase=
C:\Users\temp\IdeaProjects\SSMDEMO\target\SSMDEMO" />
3
4

```

Logs

可以查看项目日志，在控制台日志无法查询到的时候，可以来到此目录查看



Work

存放编译的 class 和 jsp

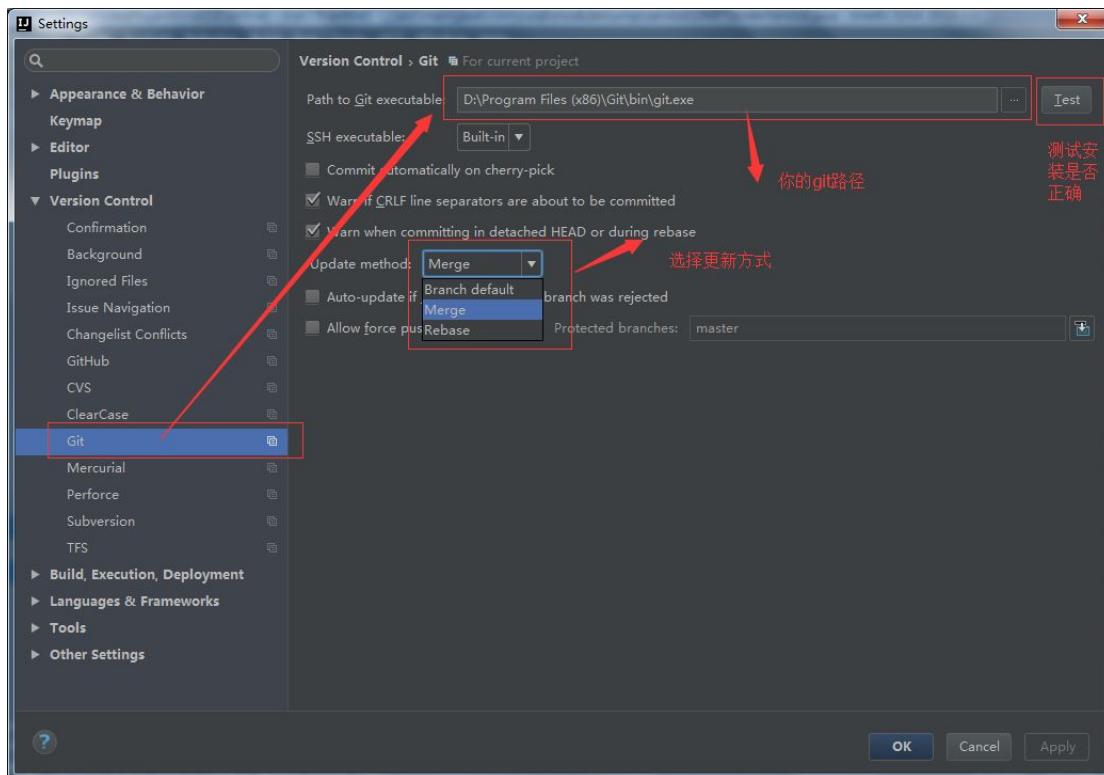


GIT 专题

这里只做一个简单的入门指南，如有不对之处，还望指正。

安装

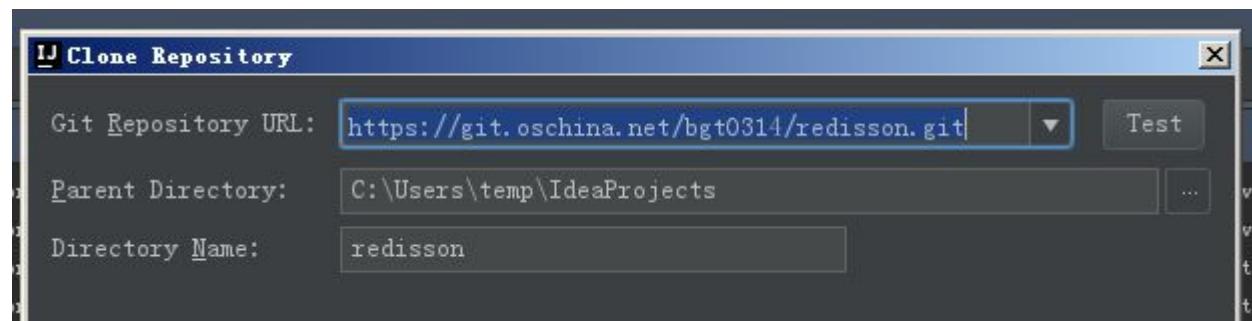
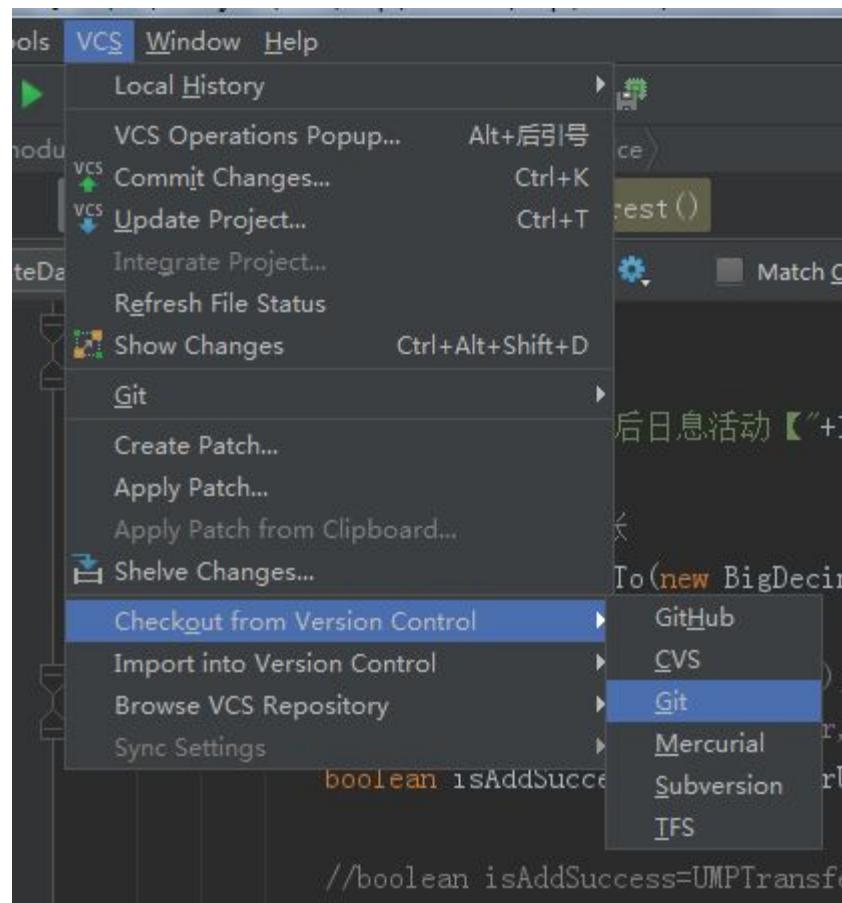
选择 VersionControl 下的 Git，依据下图进行 git 配置



使用

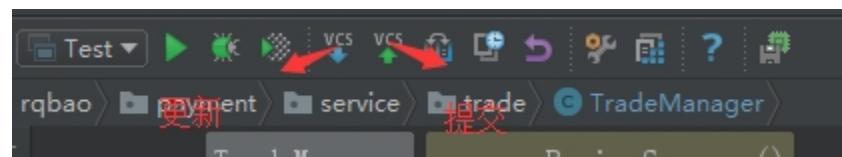
拉取项目

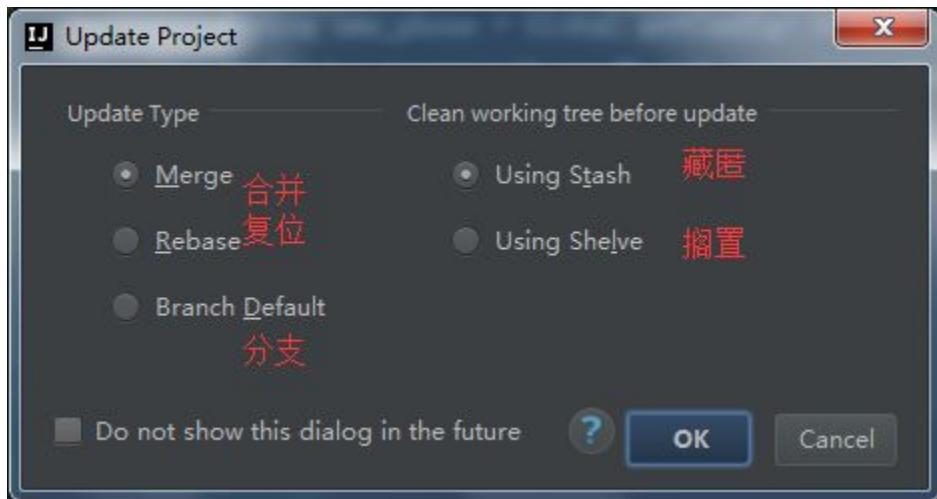
从 VCS 菜单选择 checkout from version control-->git



更新项目

点击下面按钮，或者 **ctrl+t** 快捷键
如果本地有分支，一般是从本地，没有的话则是从远程





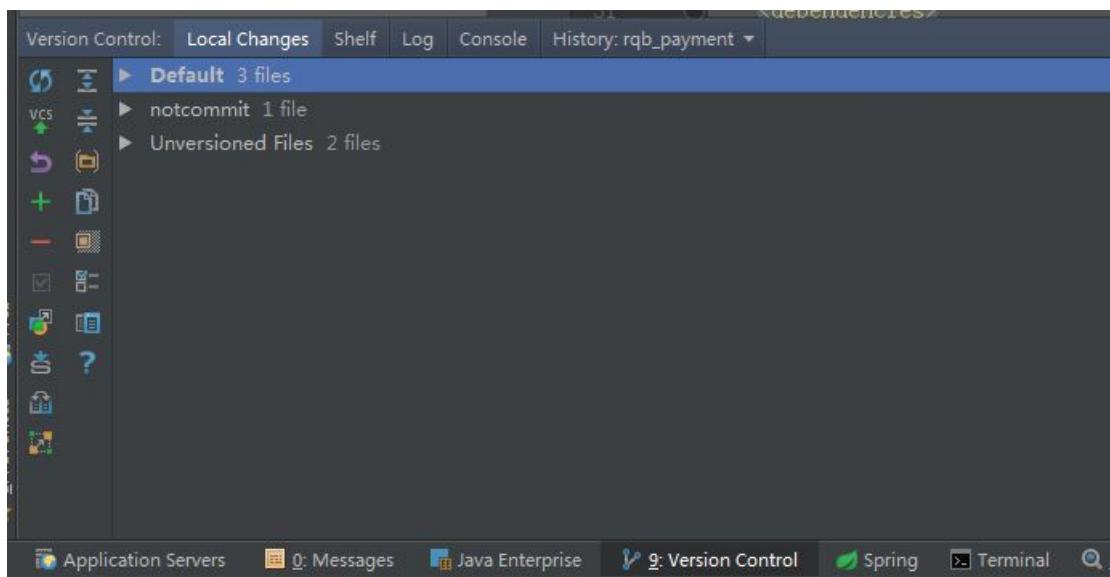
提交项目

Ctrl+k 即是提交，（注意 ctrl+k 一般是提交到本地仓库，ctrl+shift+k 是提交远程）

面板说明

Alt+9 跳转到 versionControl 面板

Local Changes

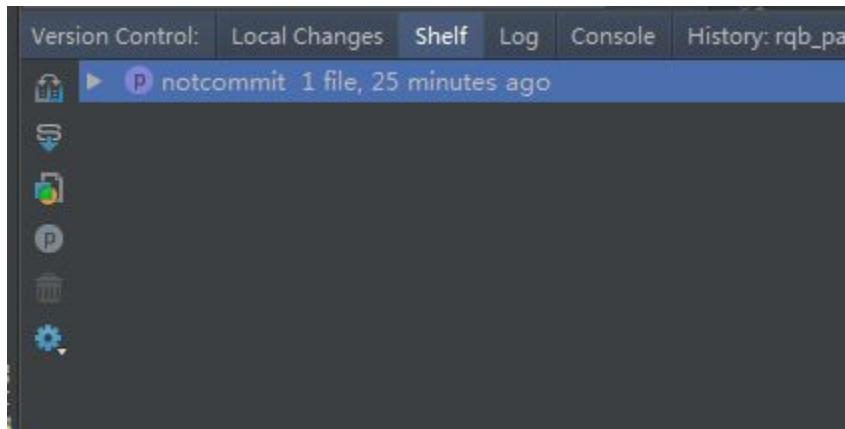


左侧菜单自上至下依次为:

刷新	展开
提交	折叠
还原	复制

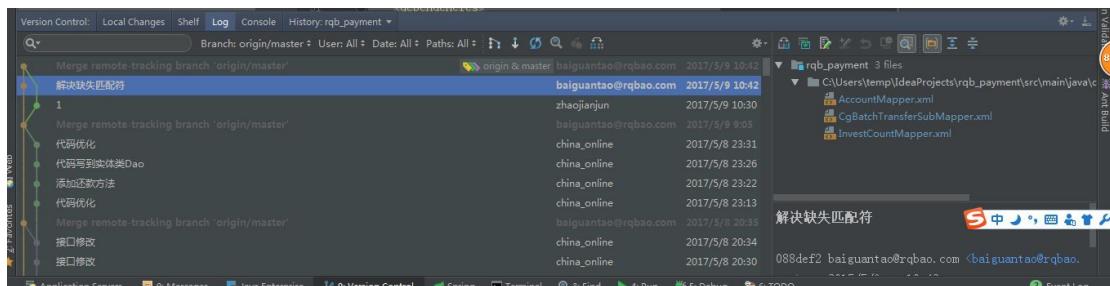
新建版本列表	显示忽略文件
删除版本列表	设置忽略文件规则
设置列表活动态	预览不同
移动列表	帮助
搁置列表更改内容	
展示不同	
展示变动	

Shelf 面板



搁置操作后的面板，可用于还原搁置的操作

Log 面板

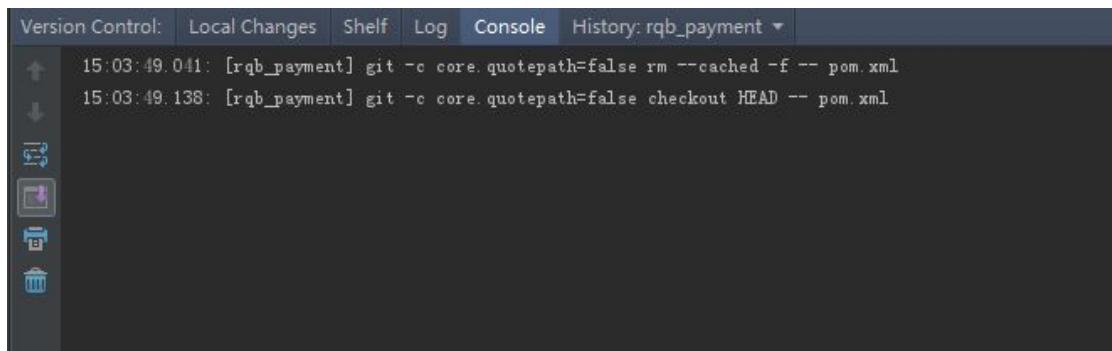


自上至下---》自左到右

搜索(提交消息)、分支筛选、用户筛选、时间筛选、路径、排序开关、显示长优势(边缘)、刷新、进入分支等、pick、高亮 pick

提交消息、用户信息、提交时间、提交明细

Console 面板



打印操作的 git 命令

History 面板

The screenshot shows the IntelliJ IDEA interface with the 'History' tab selected in the top navigation bar. The title bar indicates the project is 'History: rqb_payment'. The history window displays a list of commits:

Version	Date	Author	Commit Message
0b7e2fa	Today 10:42	baiguantao@rqbao.com	Merge remote-tracking branch 'origin/master'
463f9fc	Today 10:30	zhaojianjun	1
088def2	Today 10:42	baiguantao@rqbao.com	解决缺失匹配符
2d6d03f	Today 9:05	baiguantao@rqbao.com	Merge remote-tracking branch 'origin/master'
7916e4c	Yesterday 23:31	china_online	代码优化
c498e6e	Yesterday 23:26	china_online	代码写到实体类Dao
2d1266a	Yesterday 23:22	china_online	添加还款方法

Below the history table, there is a note: 'Merge remote-tracking branch 'origin/master''. At the bottom of the window, there are tabs for Application Servers, Messages, Java Enterprise, Version Control, Spring, Terminal, Find, Run, Debug, TODO, and Event Log.

版本号 时间 作者 提交消息

提交消息完整内容

项目 git 面板

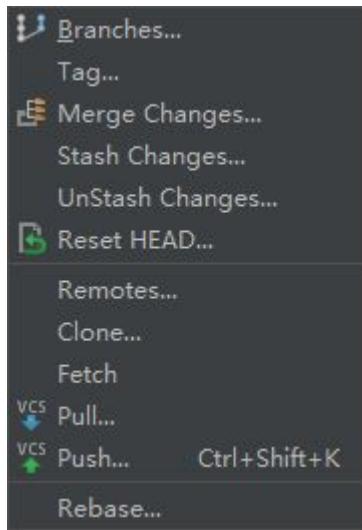
The screenshot shows the IntelliJ IDEA interface with the 'Git' tab selected in the top navigation bar. The title bar indicates the project is 'Synchronize 'rb_payment''. The right side of the screen shows a context menu for a selected commit:

- Commit Directory... 提交目录
- Add 加入版本 Ctrl+Alt+A
- Annotate 注释
- Show Current Revision
- Compare with the Same Repository Version
- Compare with Latest Repository Version 对比
- Compare with...
- Compare with Branch...
- Show History 显示历史
- Show History for Selection
- Revert... 还原 Ctrl+Alt+Z
- Repository 仓库选项

The left side of the screen shows a list of git operations:

- Show Image Thumbnails
- Reformat Code
- Optimize Imports
- Remove Module
- Build Module 'rb_payment'
- Rebuild Module 'rb_payment'
- Run 'All Tests'
- Debug 'All Tests'
- Run 'All Tests' with Coverage
- Create 'All Tests'...
- Run As Mybatis Generator
- Local History
- Git
- Synchronize 'rb_payment'

仓库选项



分支、标签、合并、隐藏变动、不隐藏变动、重置 head、远程地址、克隆、获取、拉取、推送、复位

SVN 专题

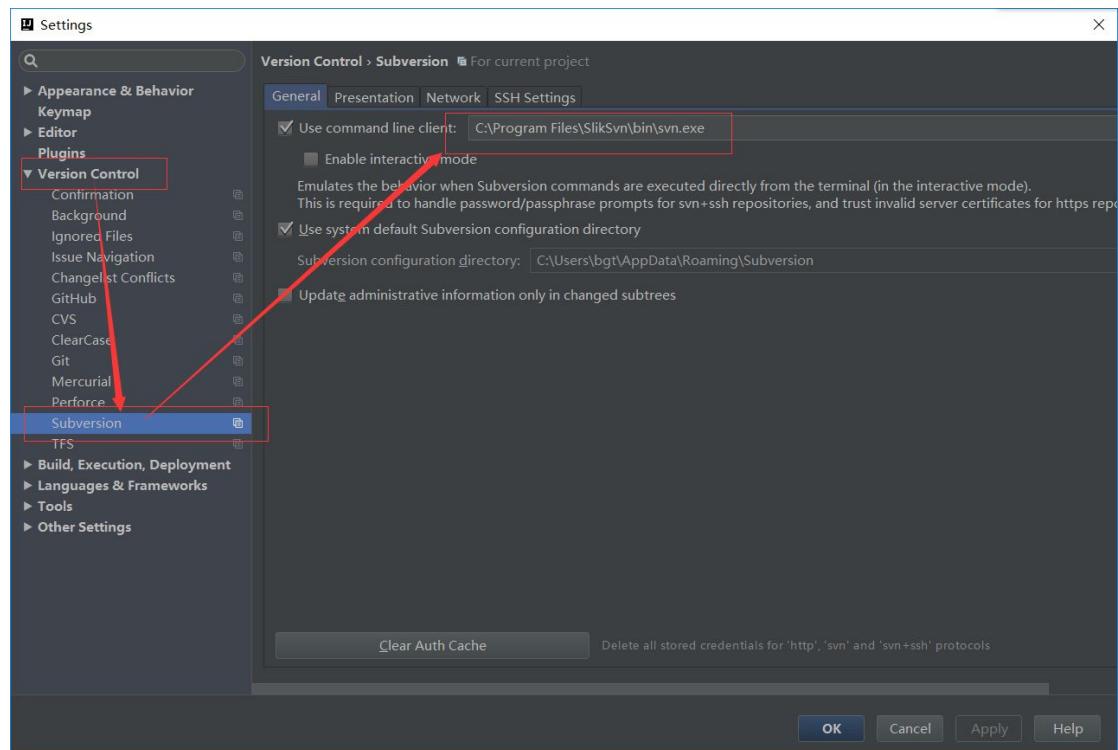
这里只做一个简单的入门指南，如有不对之处，还望指正。

配置

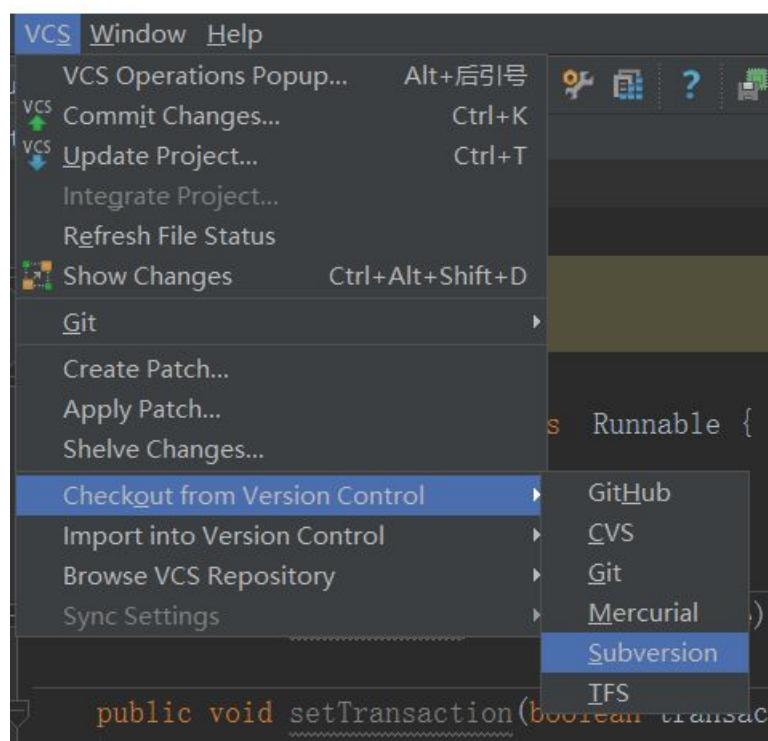
1.1 下载&安装 svn

地址: <https://sliksvn.com/download/>

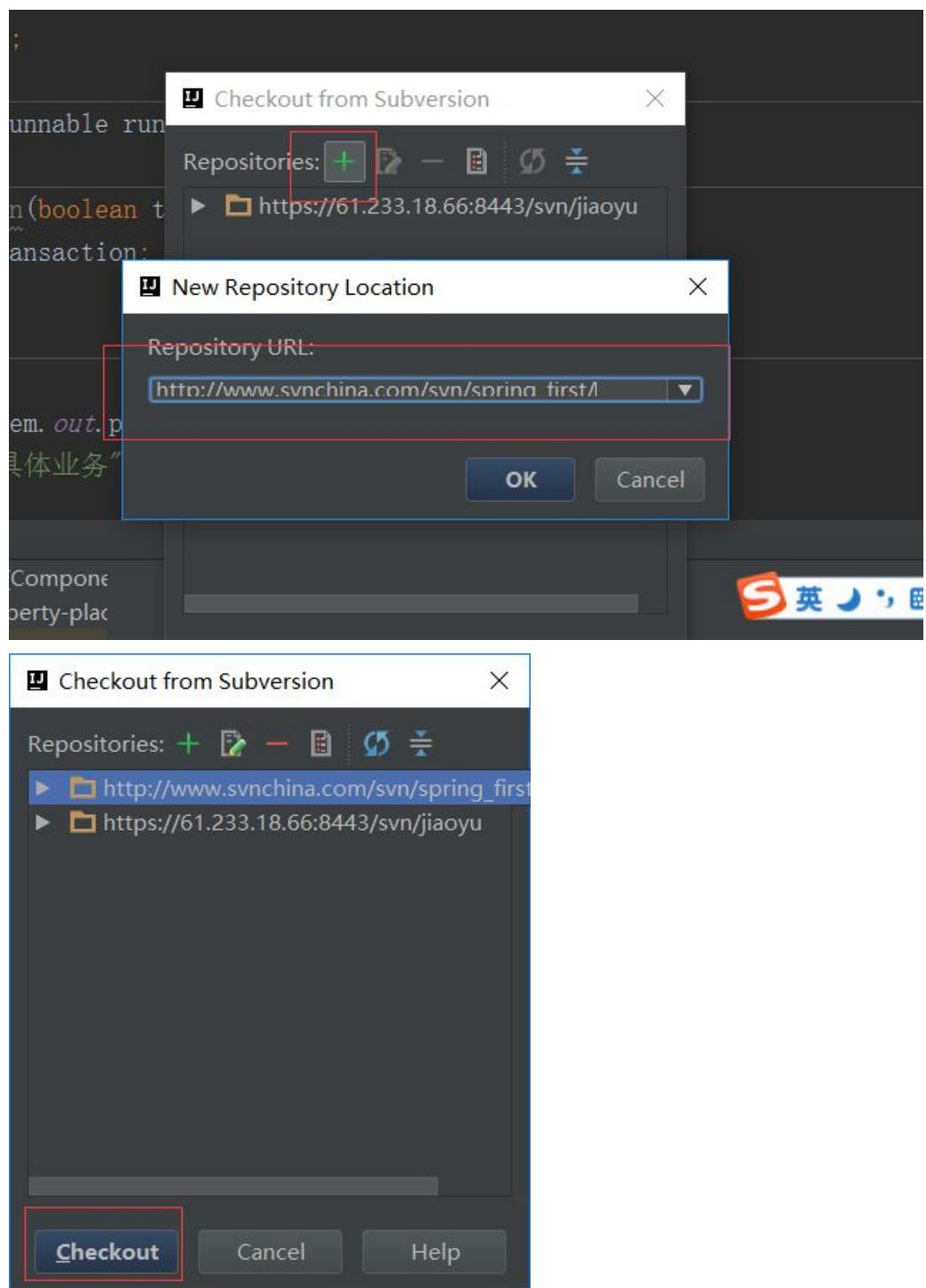
1.2 配置



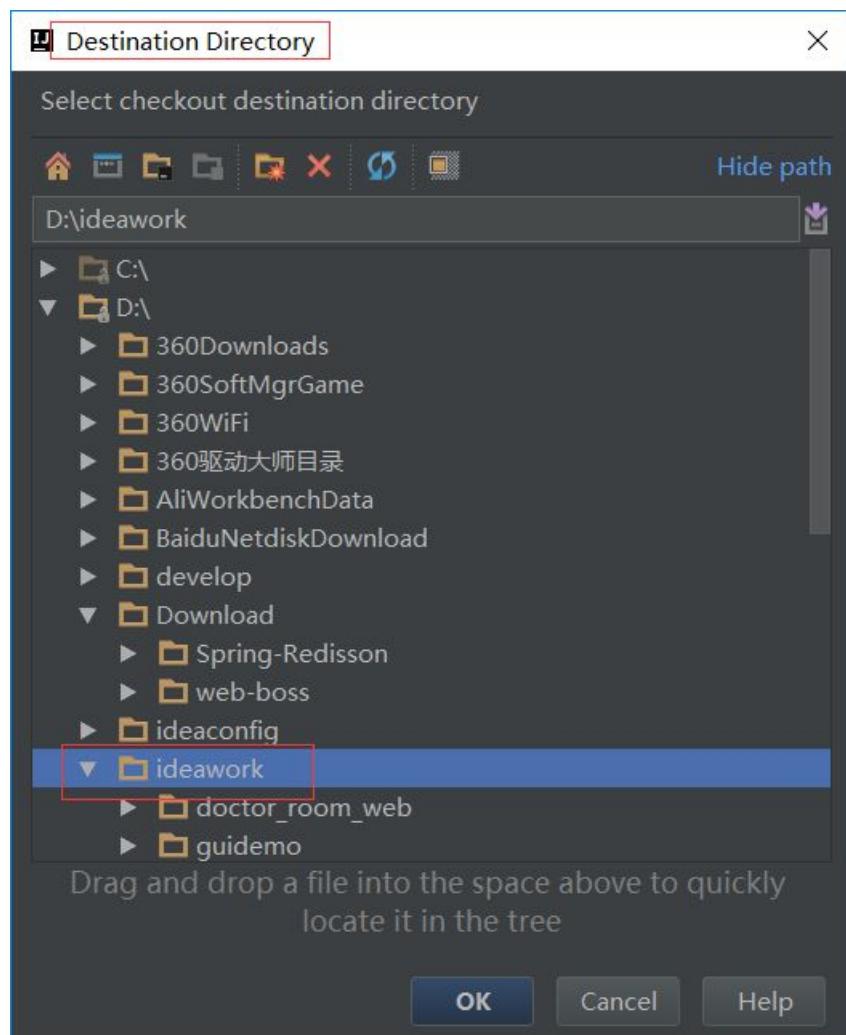
检出项目



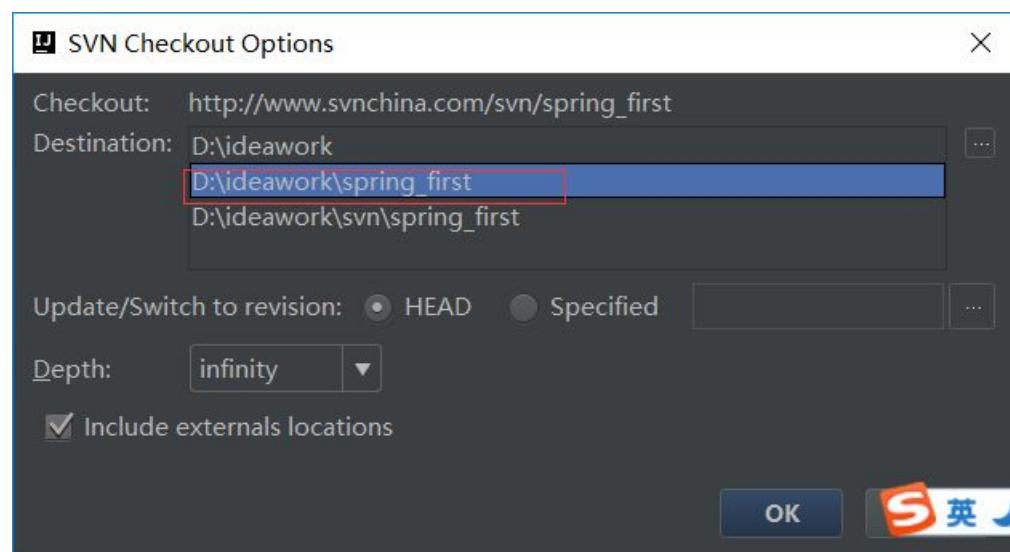
配置 svn 地址



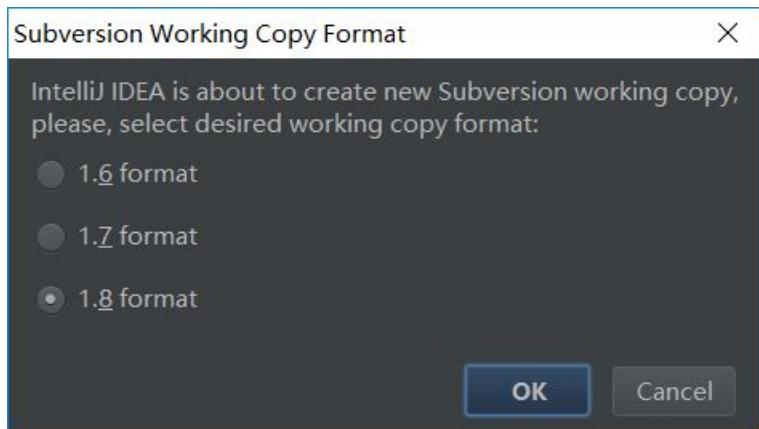
配置工作目录



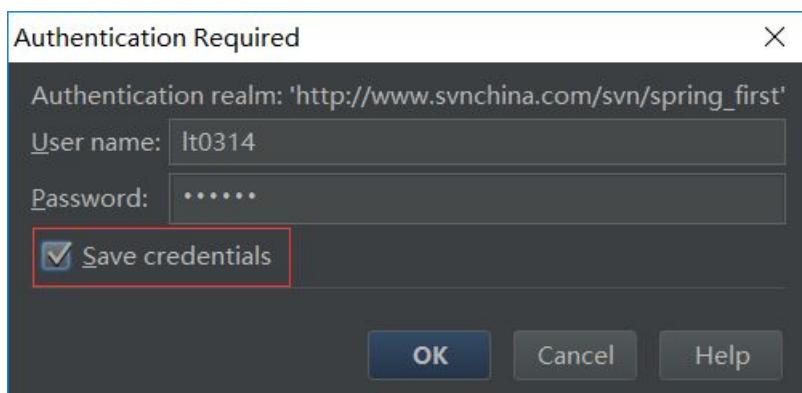
配置项目名称



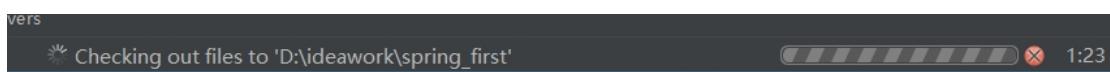
选择 format 版本



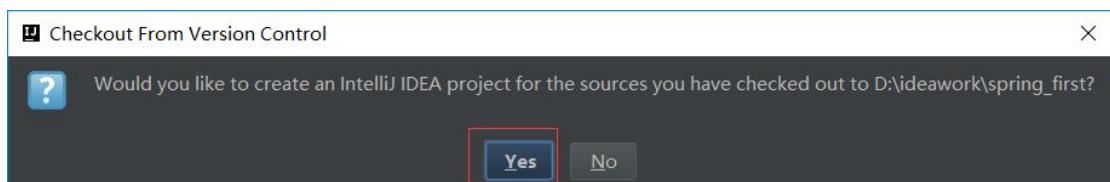
开始检出，验证账号密码



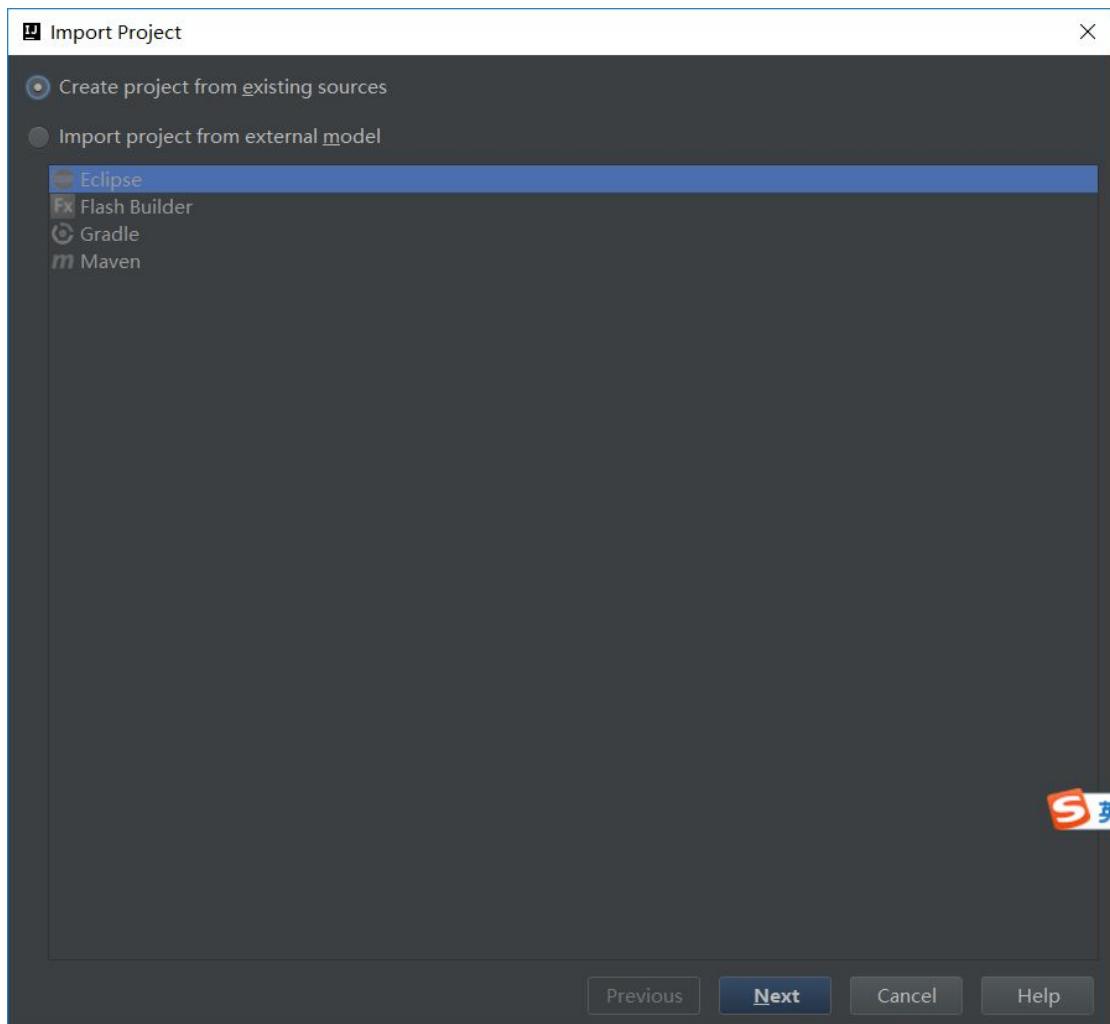
检出：



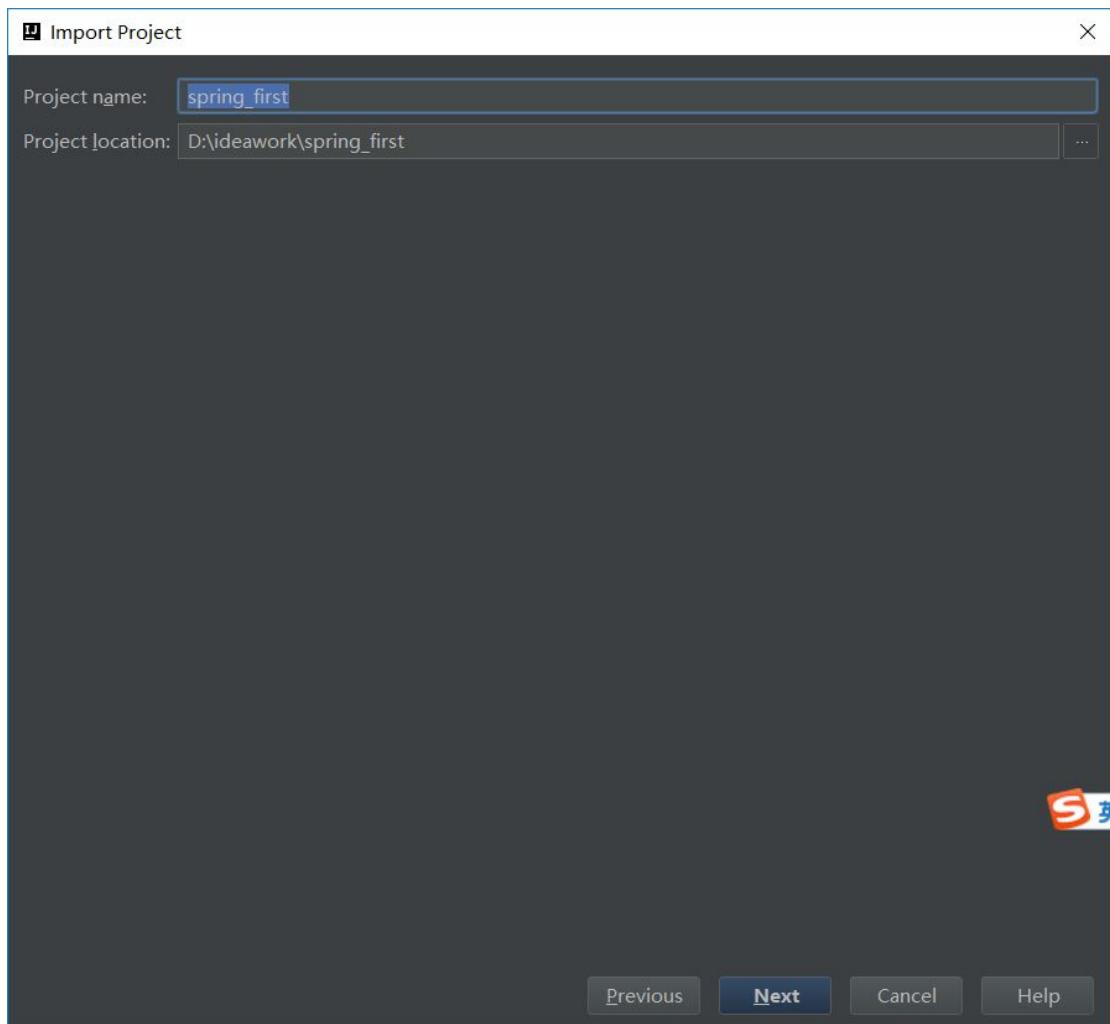
检出完成



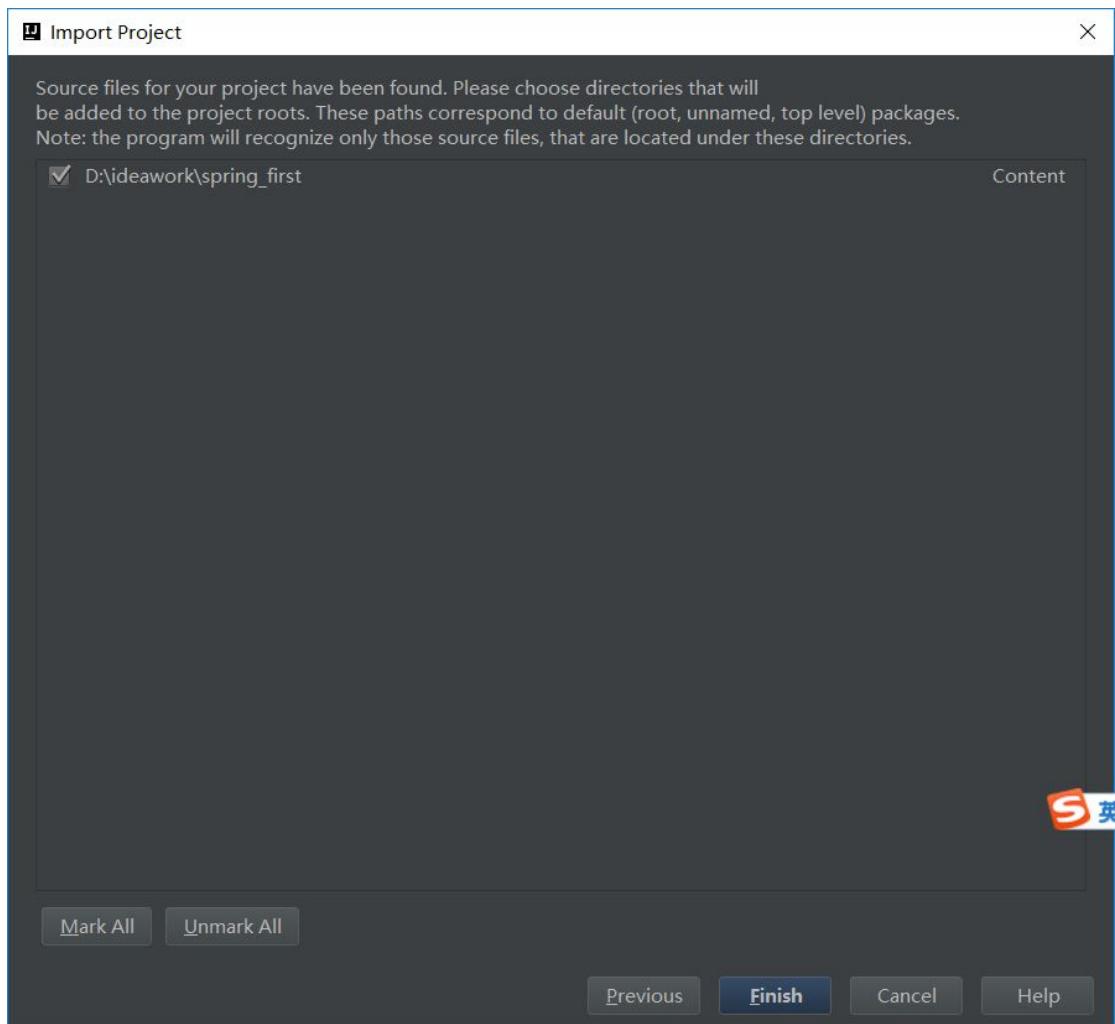
导入配置



Next



Next



Finish

完成项目检出全部工作。

面板说明

常用面板说明。

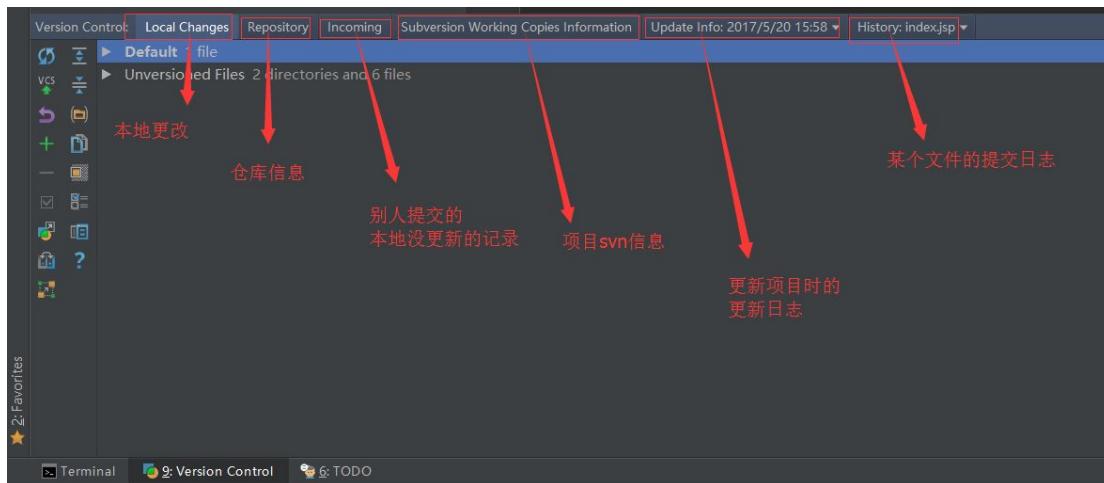
工具栏面板



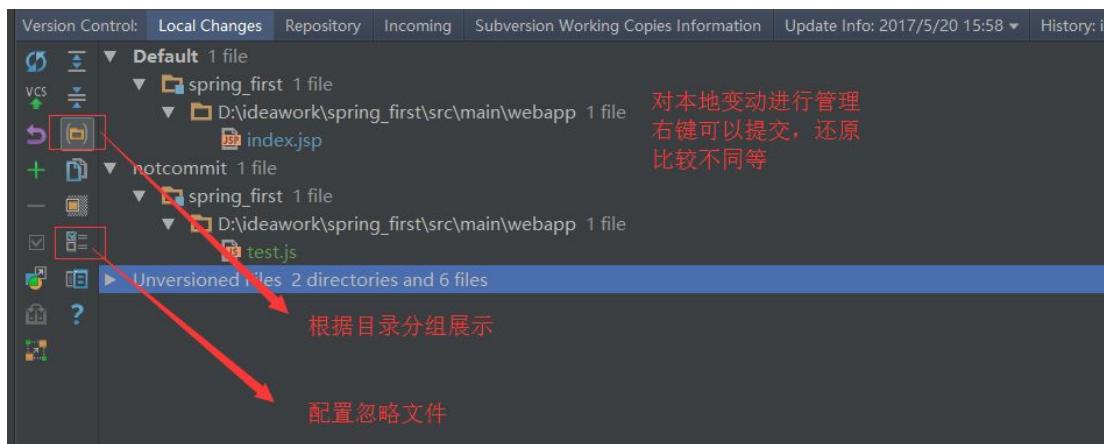
更新、提交、对比、显示历史、还原

VersionControl (版本控制)

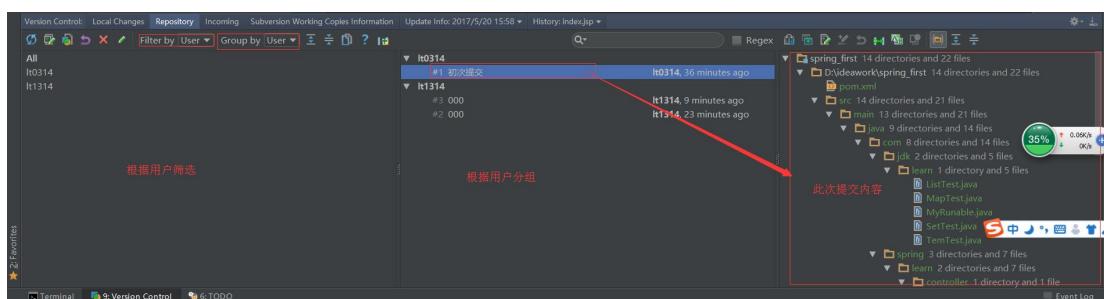
总览，版本控制中都包含的有哪几种面板，以及含义。



LocalChanges 本地更改



Repository (仓库)



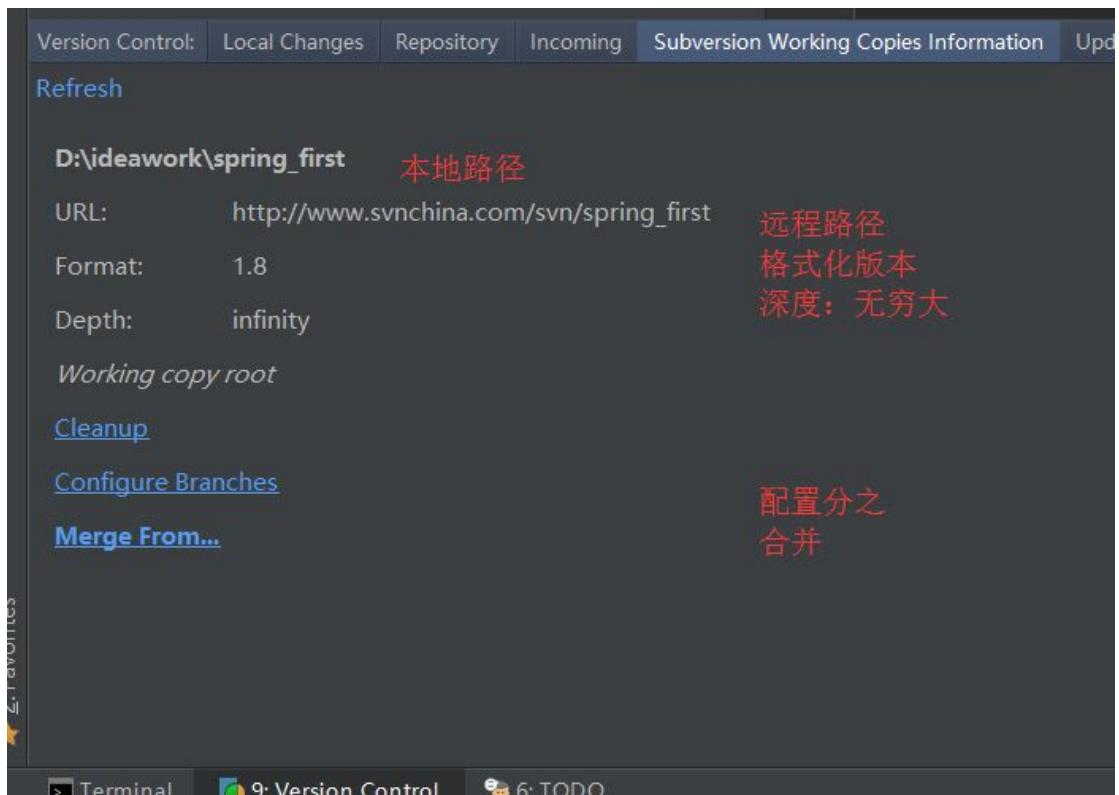
Incoming(即将到来)

本地仓库没有的，别人提交到仓库的代码。



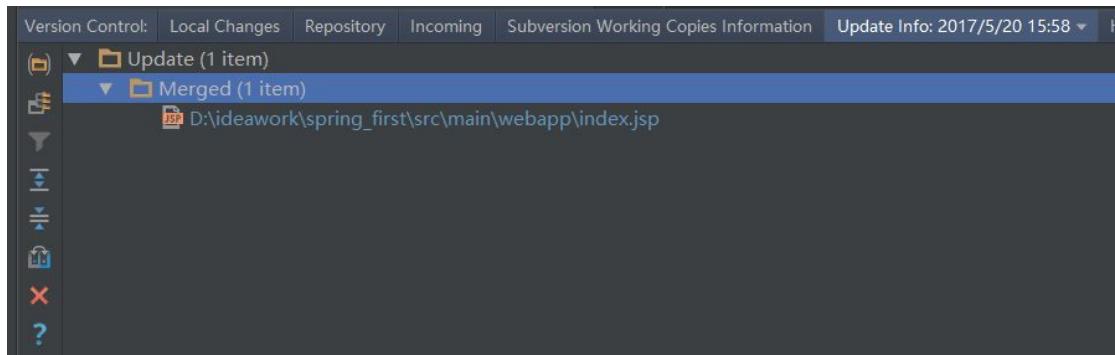
Subversion Working Copies Information (svn 工作信息)

Svn 的配置信息，路径，格式化等。



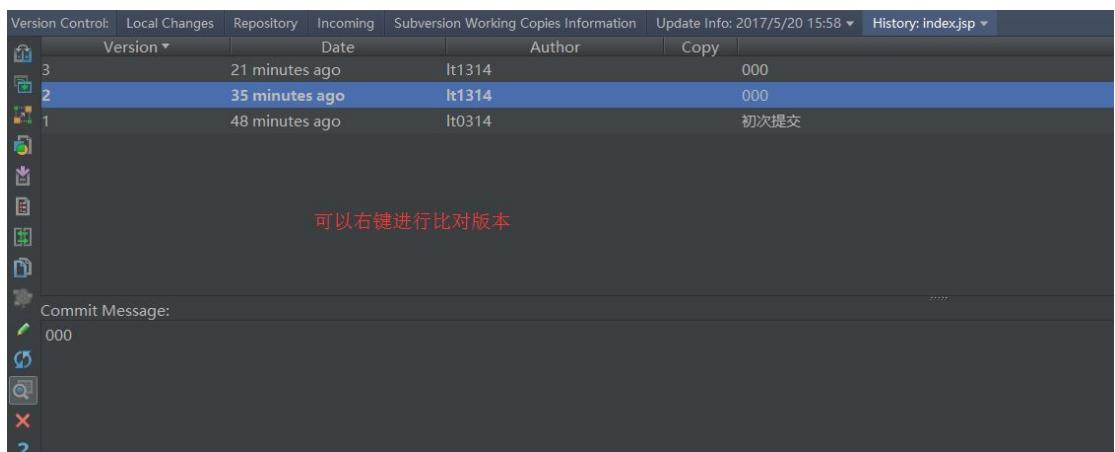
UpdateInfo(更新日志)

在更新操作的时候，会弹出给用户查阅添加删除更改合并的文件都有那些。



History

历史记录，可用于版本还原，比对等。



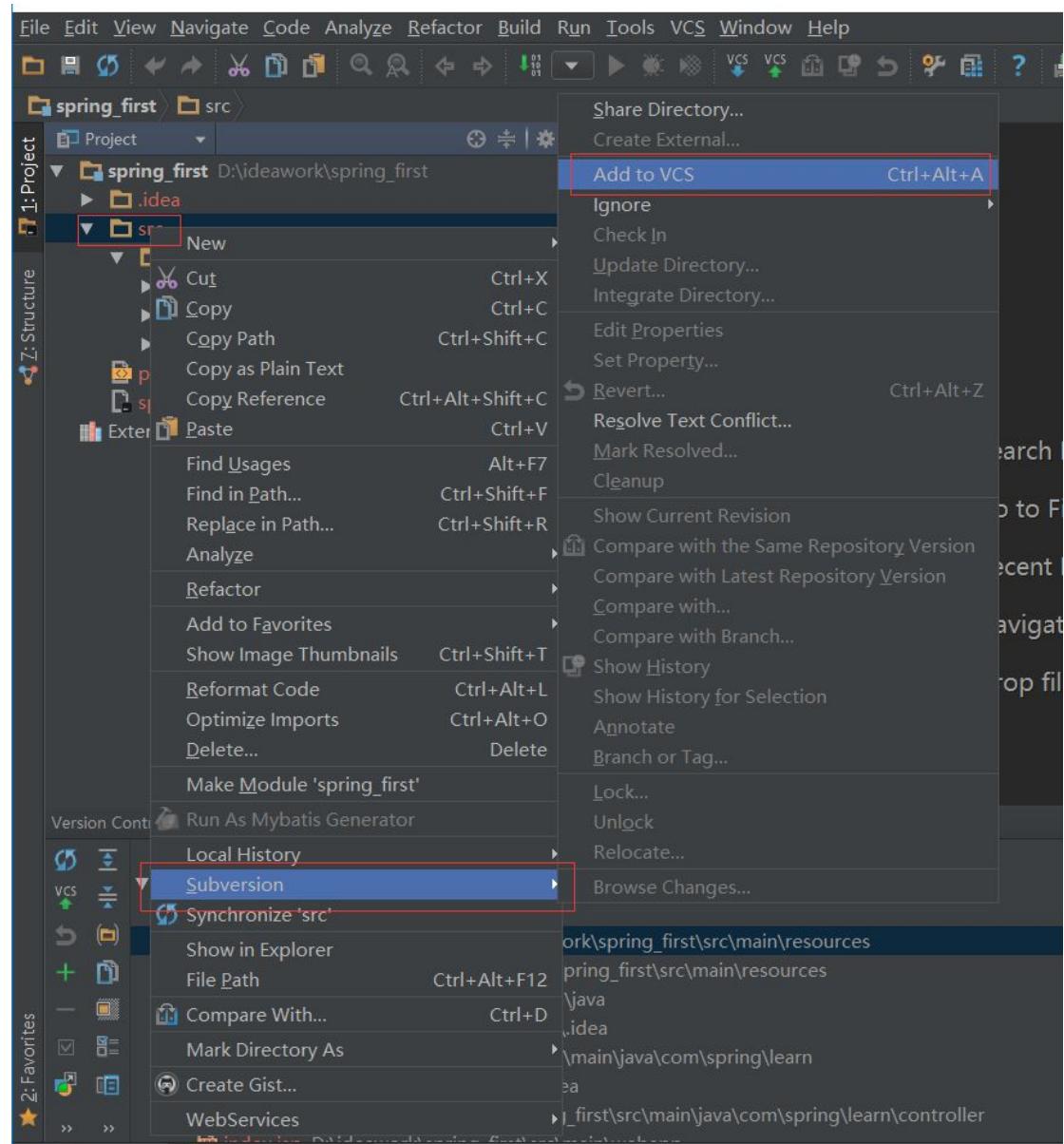
常用操作

红色是未被版本控制 pom.xml, 绿色是新加入版本 main

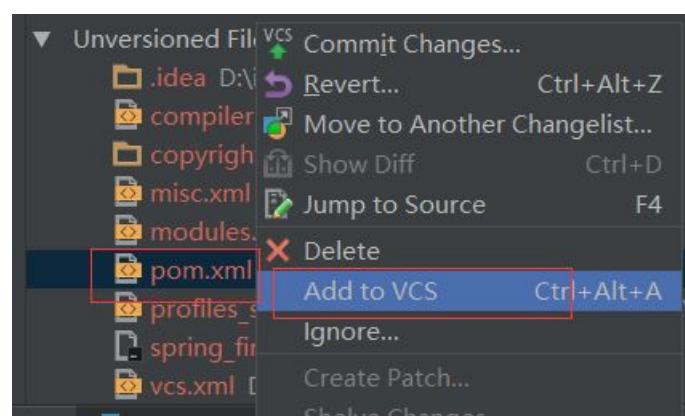
蓝色是修改过的 index.jsp

加入版本控制

第一种：

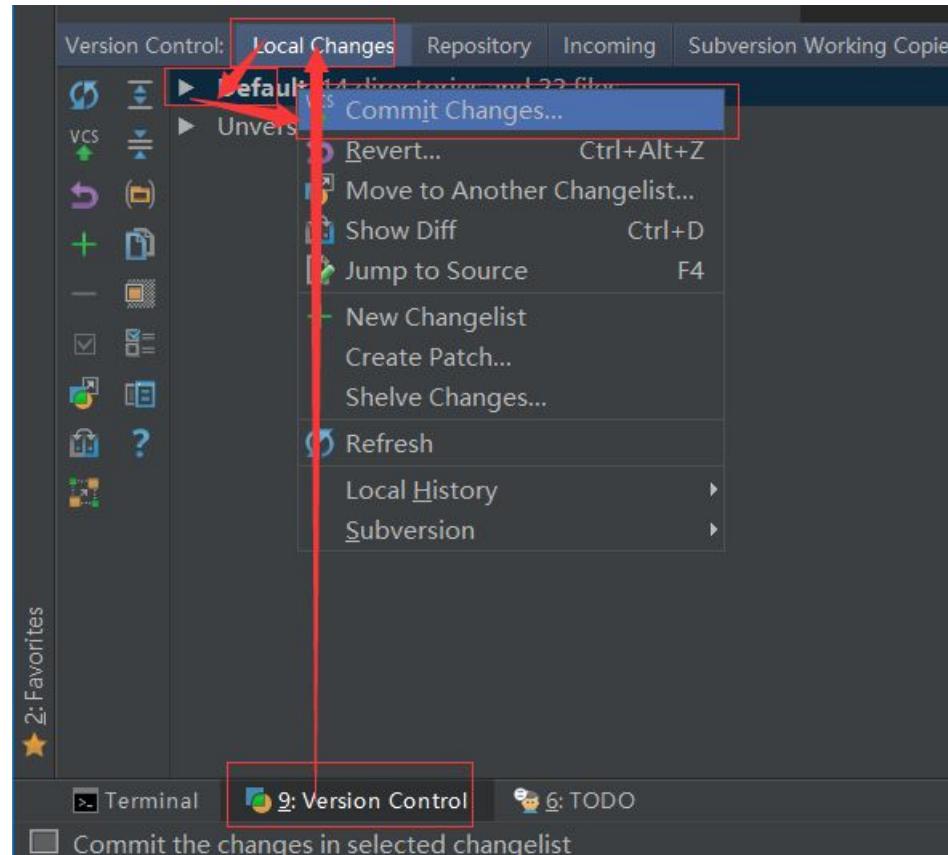


第二种

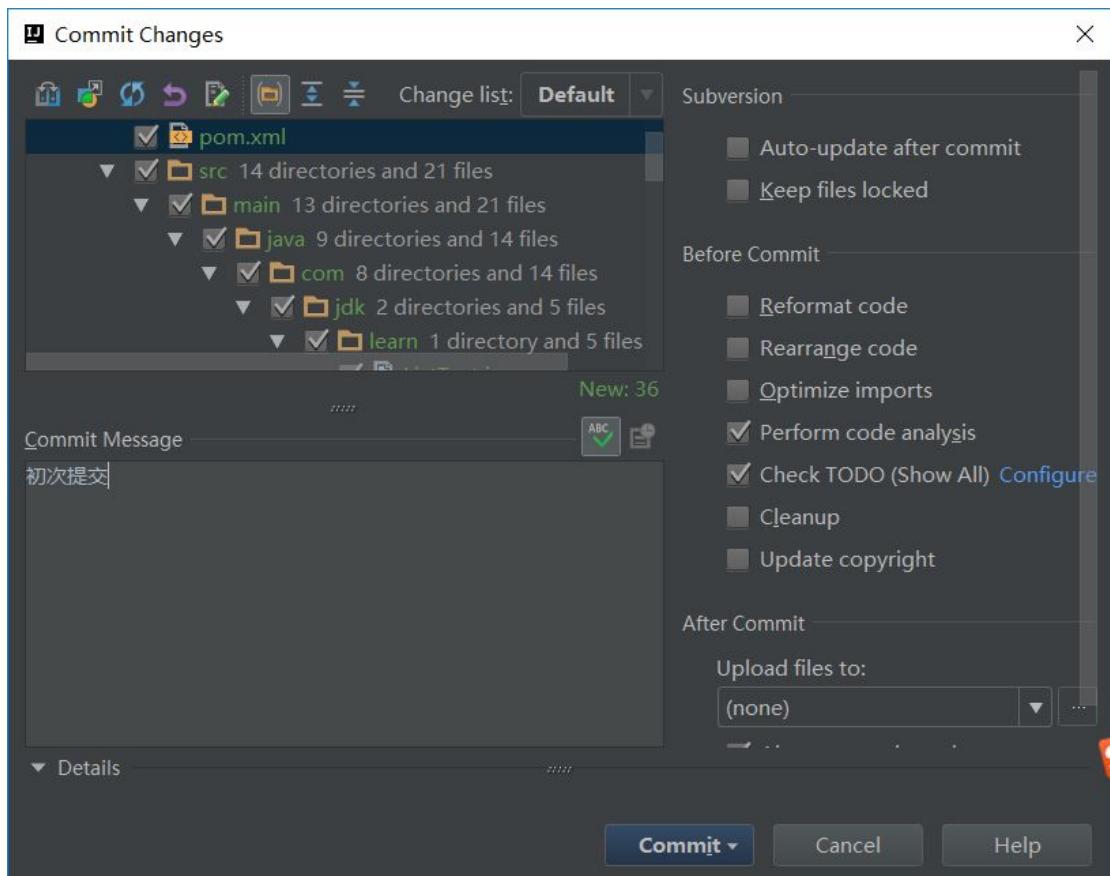


提交远程

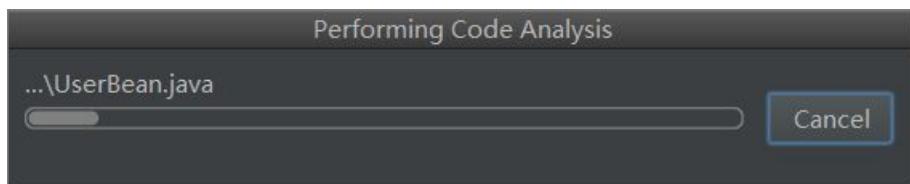
在 local changes 面板中, 选择 default



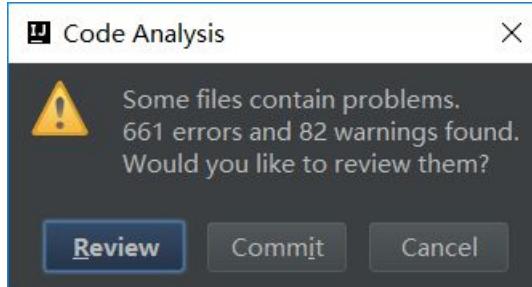
提交选项



点击 commit



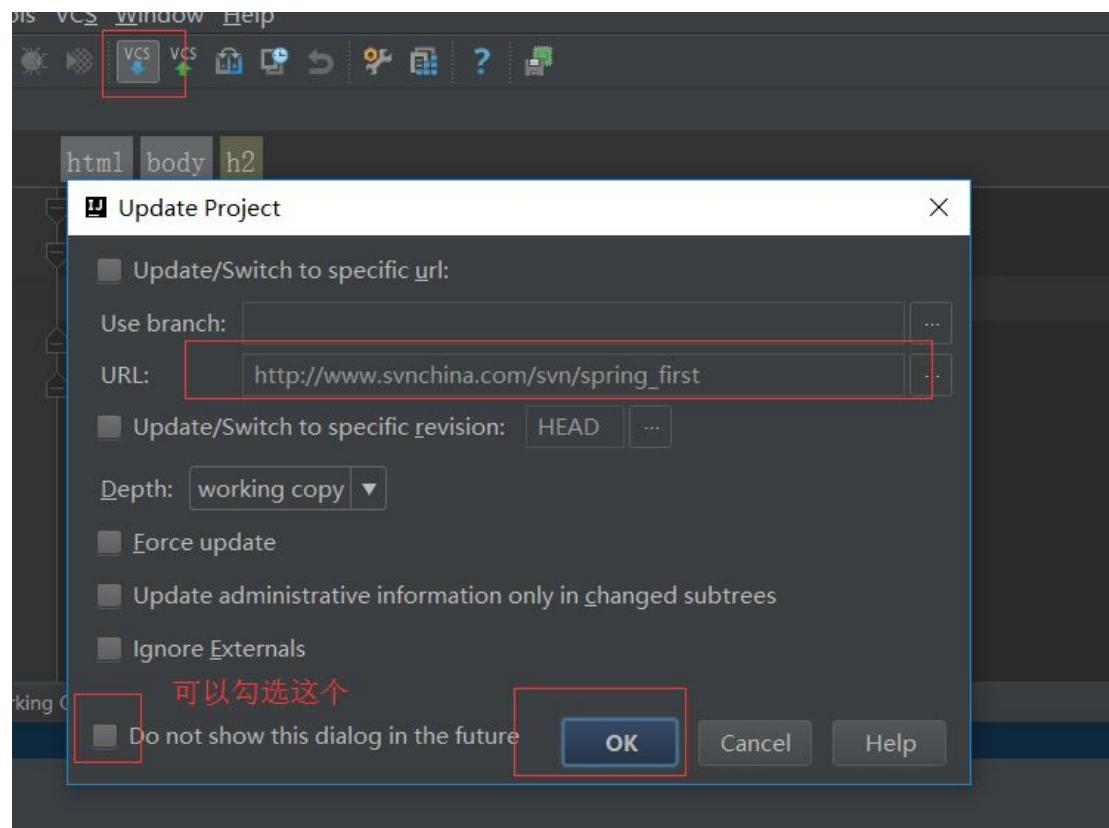
再次 commit



提交成功



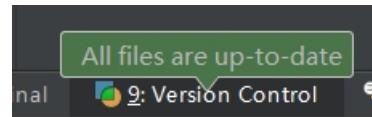
更新项目



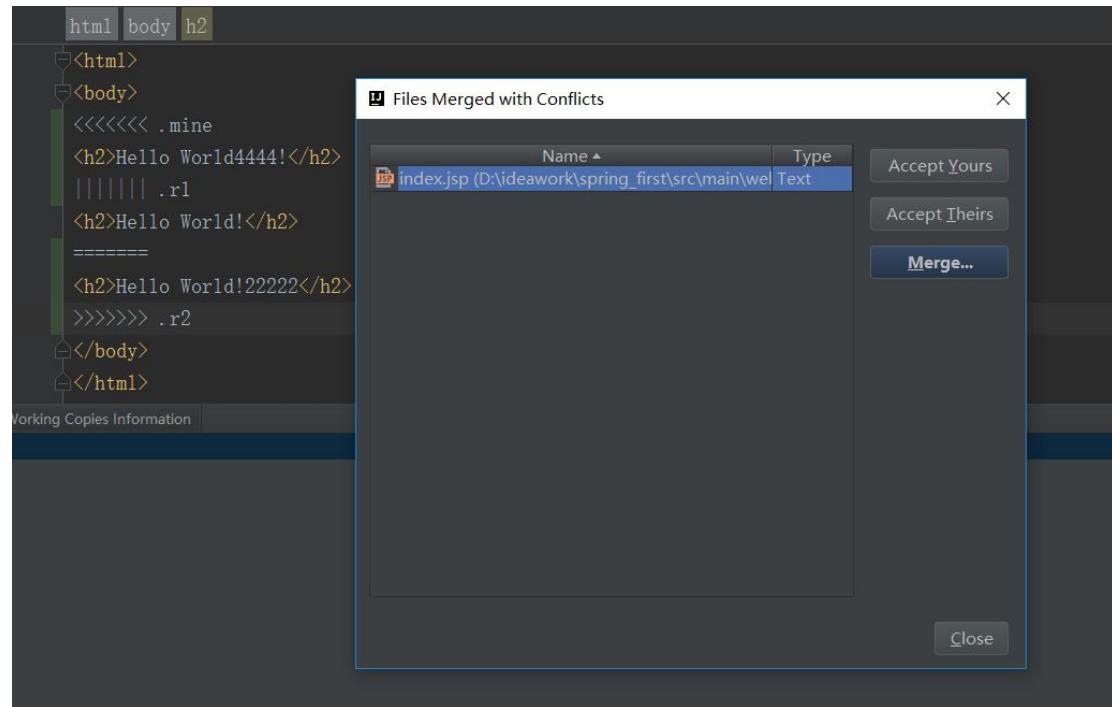
更新进度



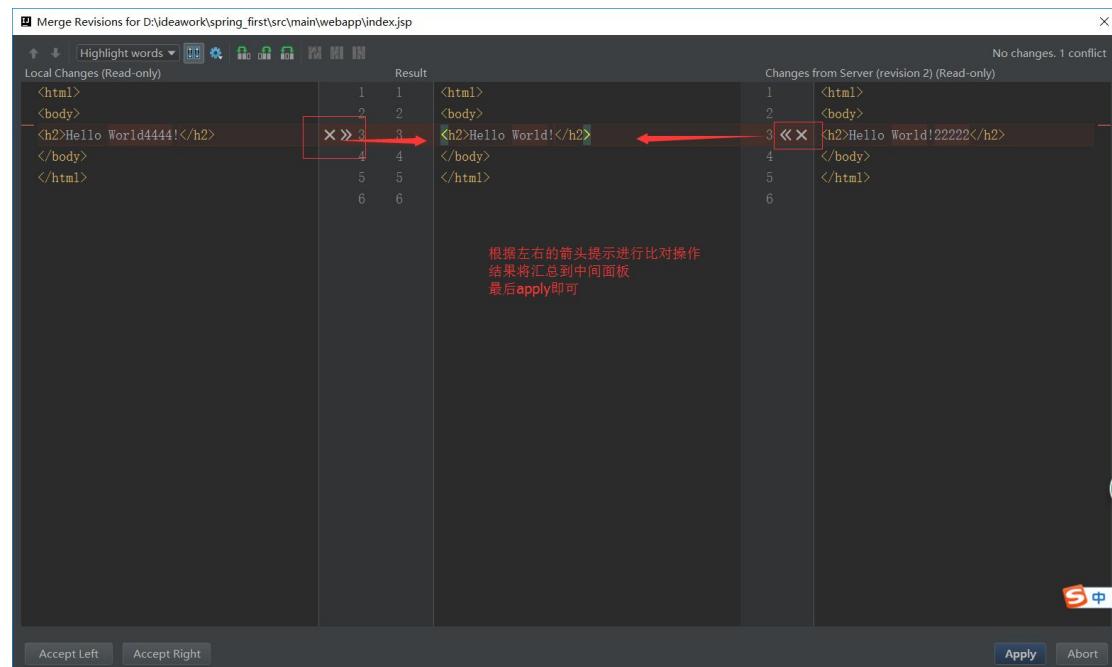
更新结果



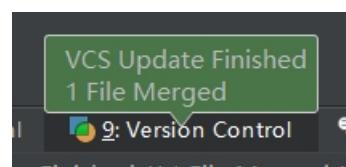
冲突解决



冲突合并



处理结果

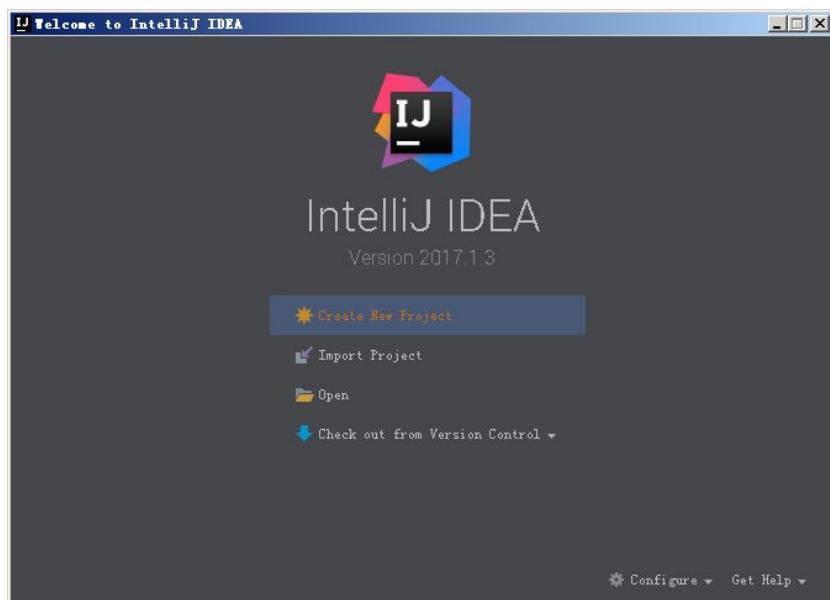


SSM 搭建

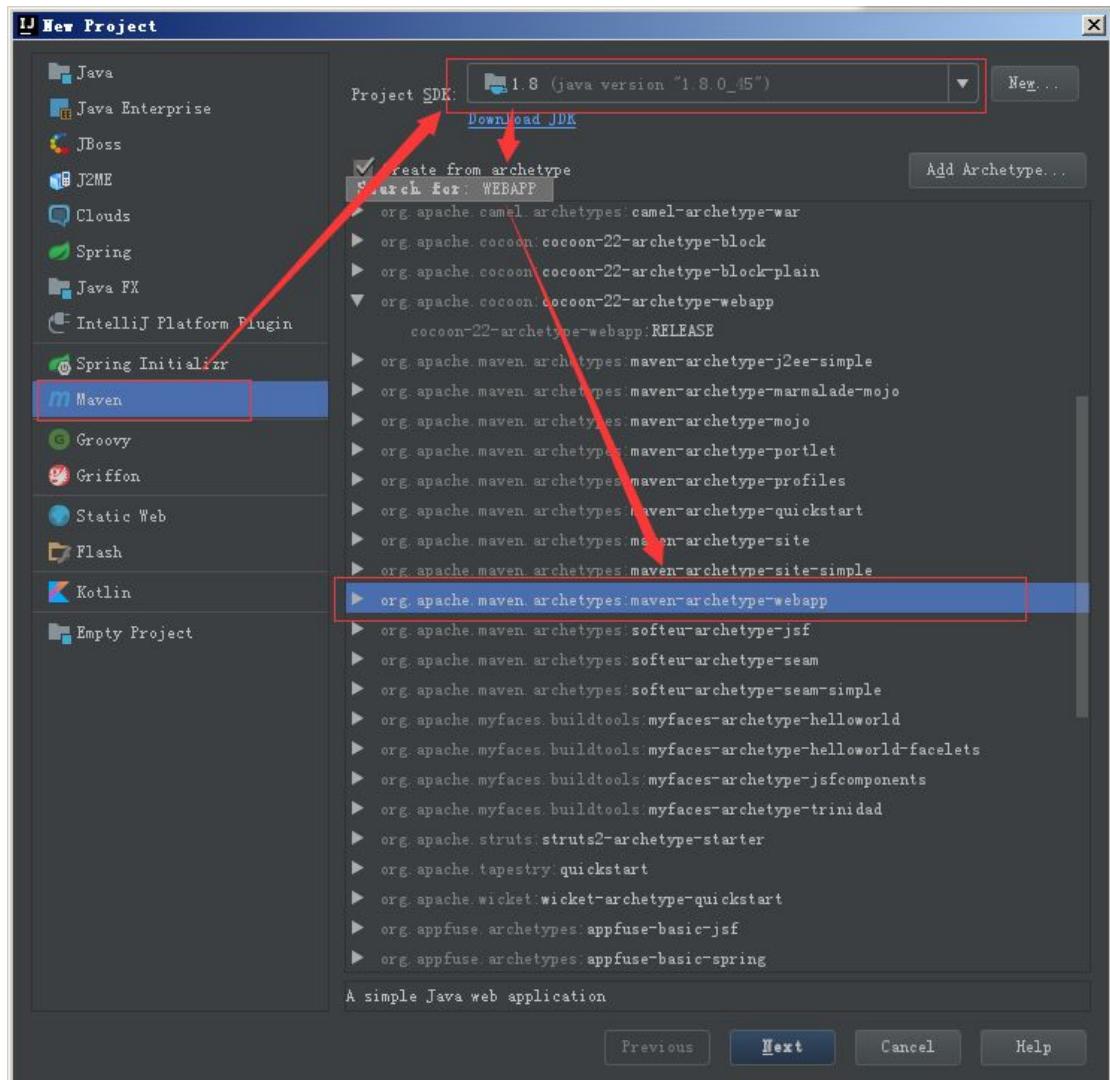
这里只做一个简单的入门指南，如有不对之处，还望指正。
这里是 maven 版本的 ssm，jdk 和 maven 需要提前配置好。

Maven 项目

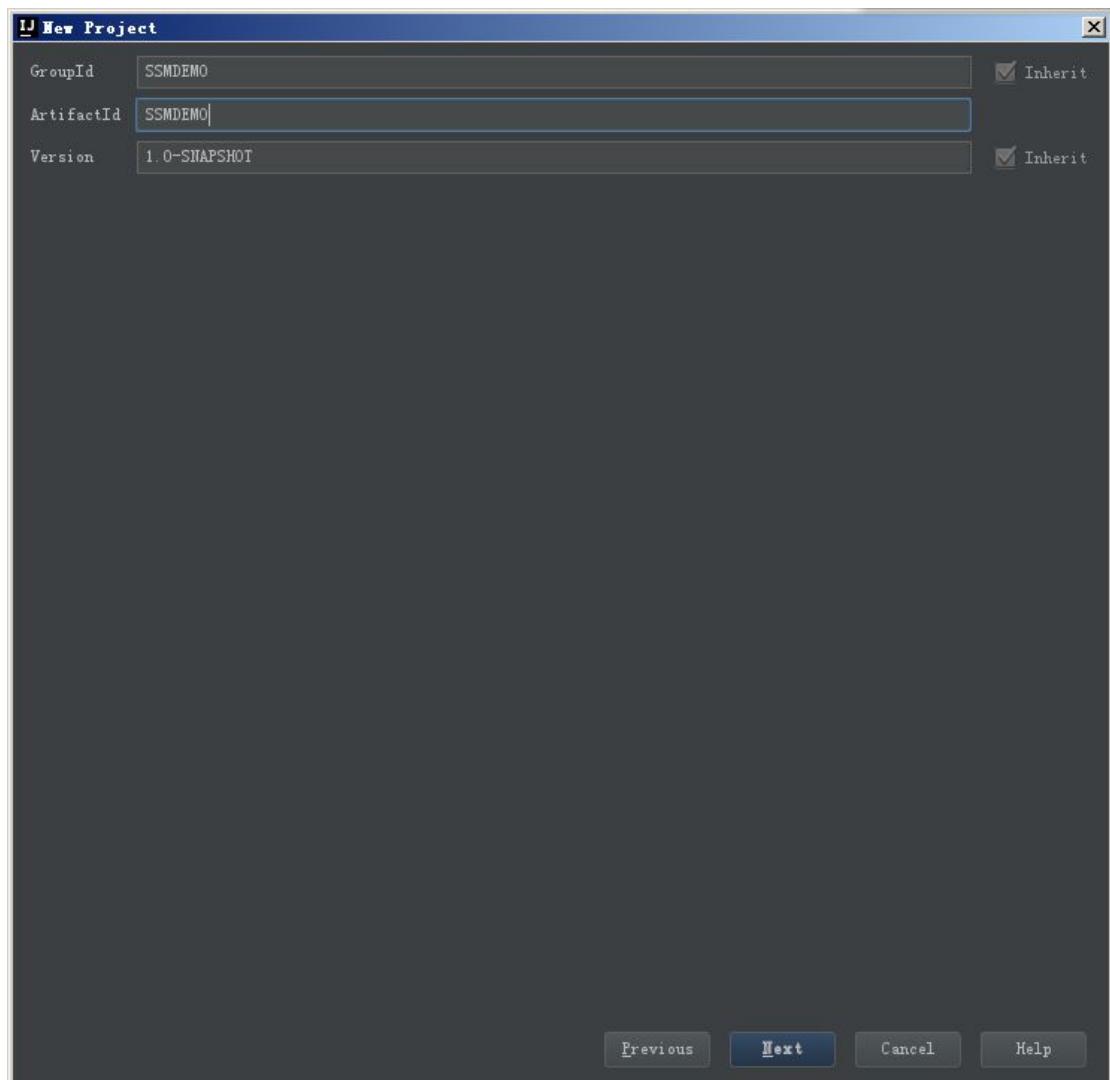
新建项目



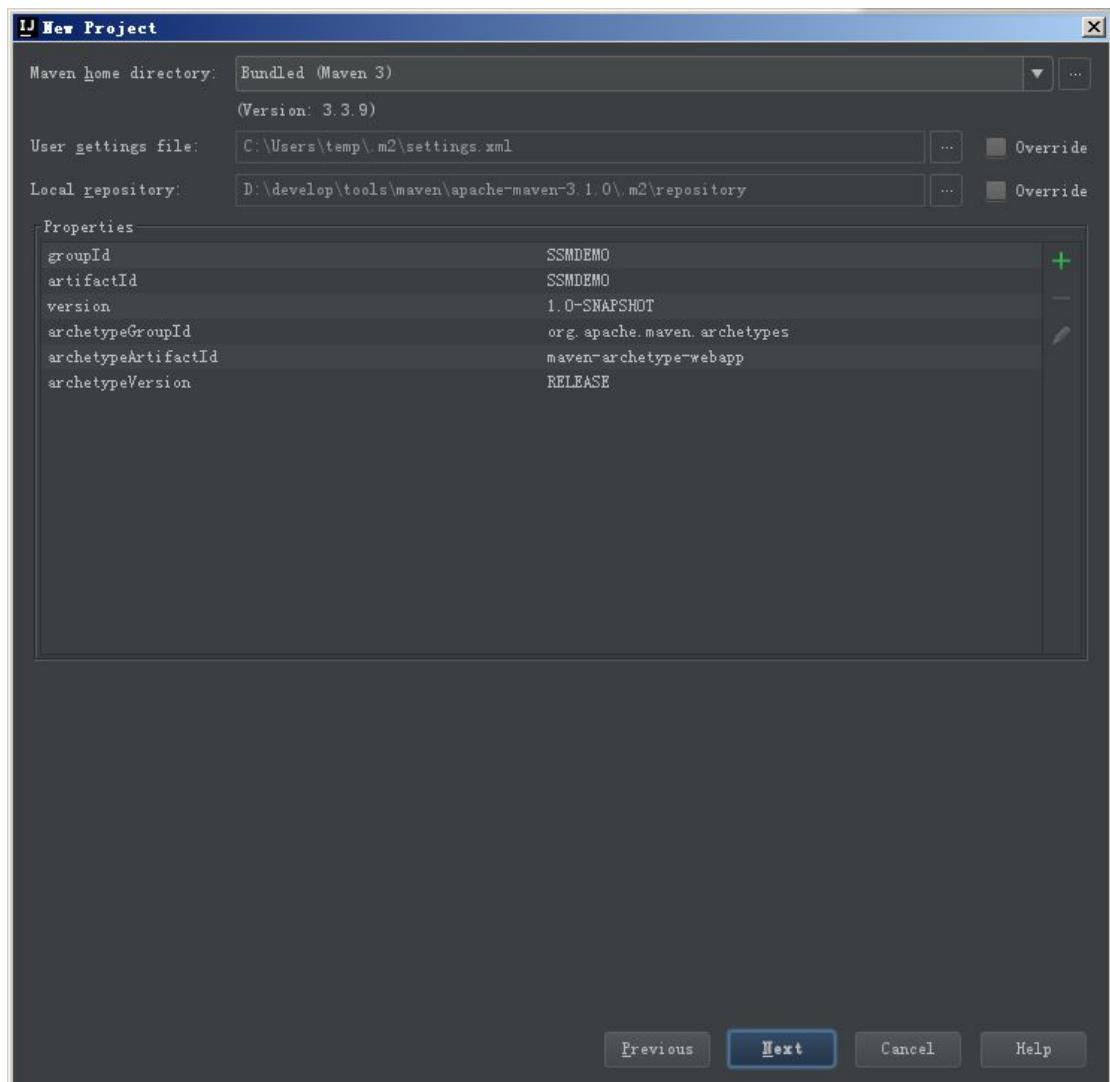
选择 maven 项目，配置 webapp 和 jdk



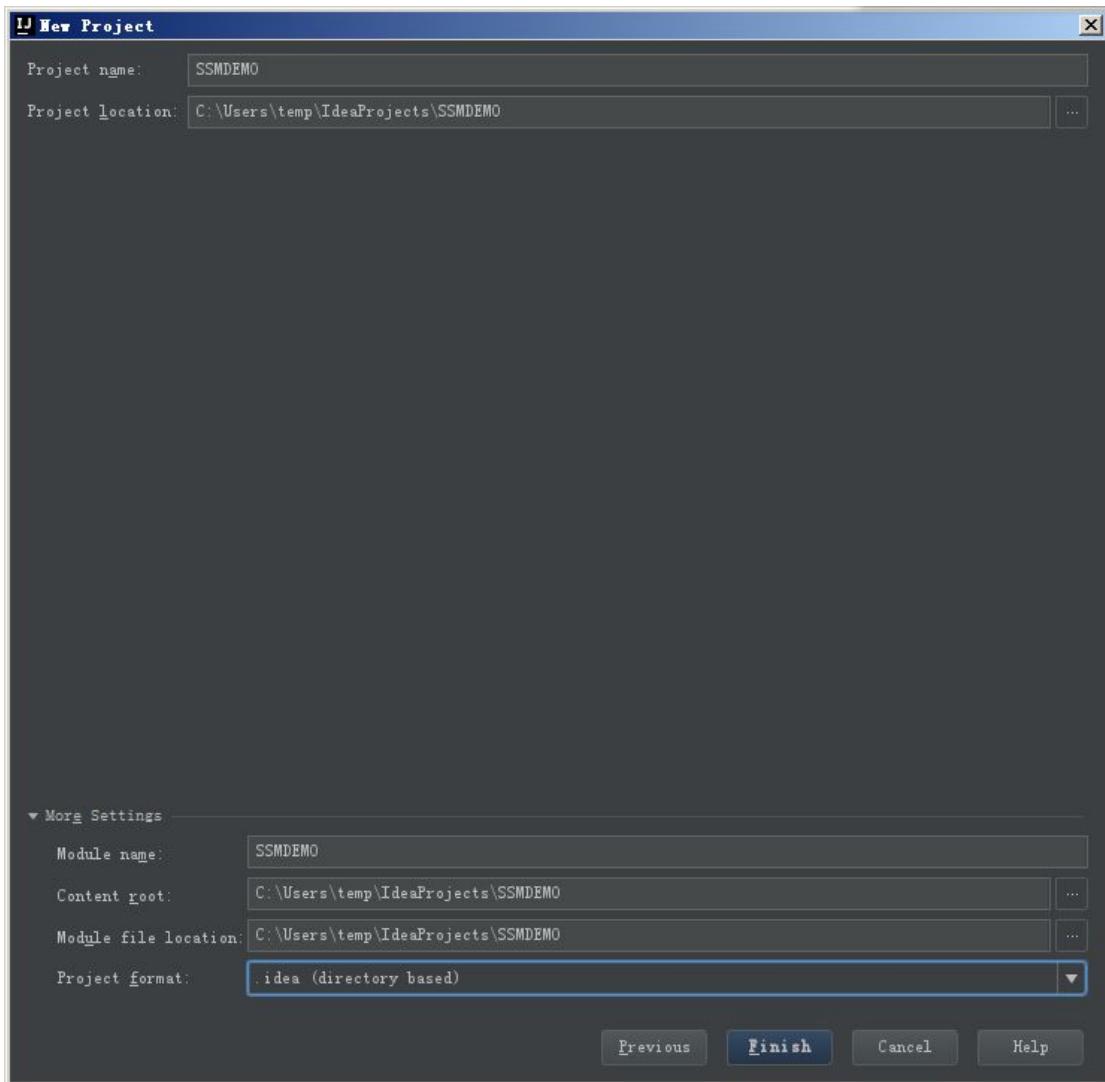
下一步，配置 groupId 和 ArtifactId



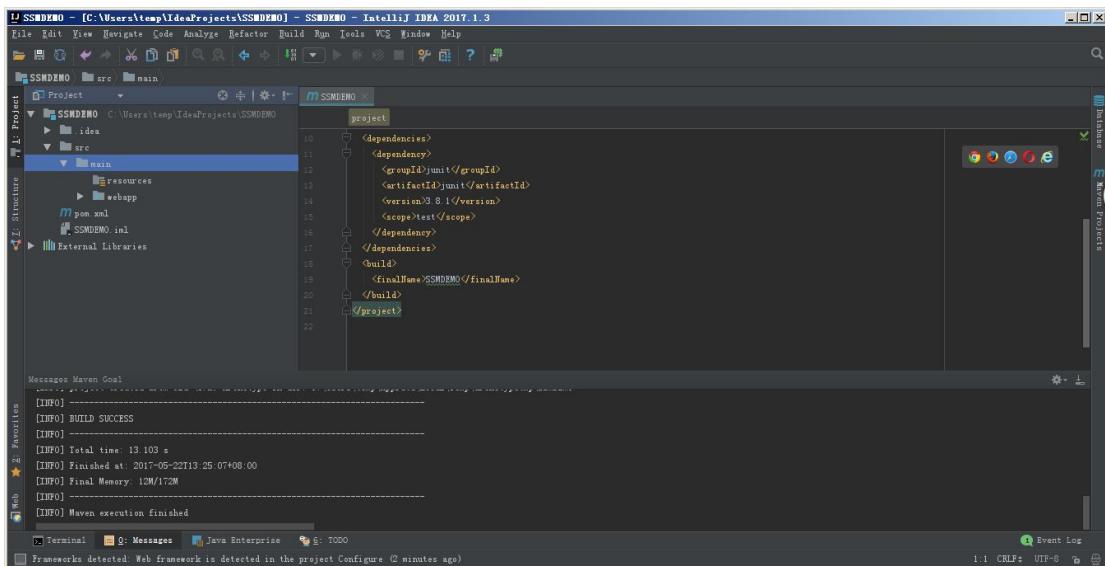
下一步，配置 maven 构建工具信息



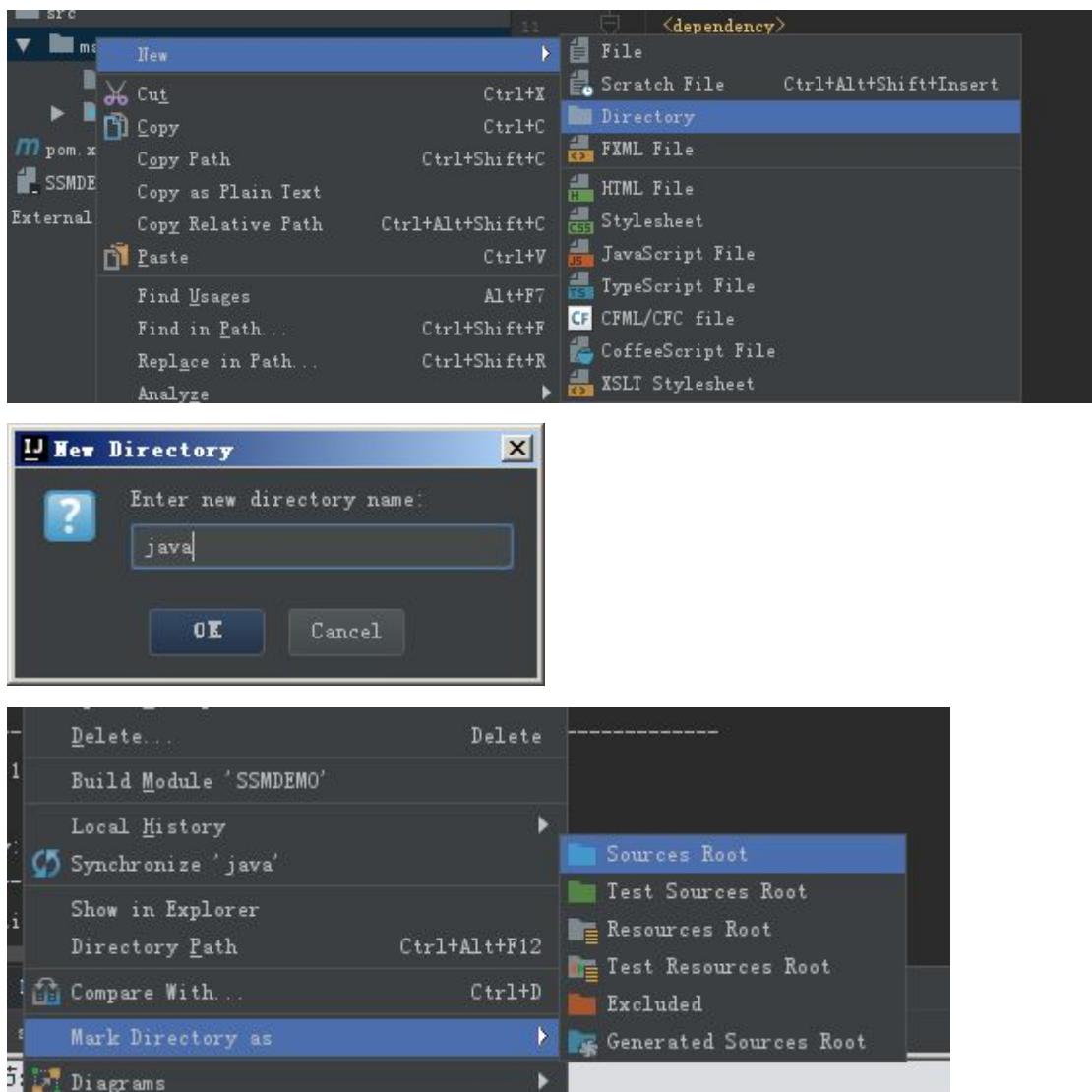
下一步，配置项目信息，module



Finish, 完成。稍等片刻, 选择右下角, enable auto import (maven 选项)



默认没有 java 目录, 新建一个 java 目录, 并设置为 root 源目录



Jar 包

添加 Spring 支持

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>4.3.7.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-beans</artifactId>
    <version>4.3.7.RELEASE</version>
</dependency>
<dependency>
```

```

<groupId>org.springframework</groupId>
<artifactId>spring-context</artifactId>
<version>4.3.7.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-aspects</artifactId>
    <version>4.3.7.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-aop</artifactId>
    <version>4.3.7.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-web</artifactId>
    <version>4.3.7.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-tx</artifactId>
    <version>4.3.7.RELEASE</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>4.3.7.RELEASE</version>
</dependency>

```

数据库和 mybatis

```

<!-- mysql 的数据库驱动包 -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.21</version>
</dependency>

<!-- Mybatis 开发包 -->
<dependency>
    <groupId>org.mybatis</groupId>

```

```

<artifactId>mybatis-spring</artifactId>
<version>1.2.0</version>
</dependency>

<!-- Mybatis 和 Spring 的 整合包, 是 mybatis 出的-->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.2.3</version>
</dependency>

```

其他 jar

```

<!-- 下面两个包 commons-dbcp,commons-pool 是配置数据源的包-->
<dependency>
    <groupId>commons-dbcp</groupId>
    <artifactId>commons-dbcp</artifactId>
    <version>1.4</version>
</dependency>

<dependency>
    <groupId>commons-fileupload</groupId>
    <artifactId>commons-fileupload</artifactId>
    <version>1.3</version>
</dependency>

<dependency>
    <groupId>commons-pool</groupId>
    <artifactId>commons-pool</artifactId>
    <version>1.4</version>
</dependency>

<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-log4j12</artifactId>
    <version>1.7.2</version>
</dependency>

<!-- 下面的三个包是在配置事务的时候用到的 spring 的依赖包 -->
<dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjweaver</artifactId>

```

```

<version>1.7.0</version>
</dependency>
<dependency>
    <groupId>aopalliance</groupId>
    <artifactId>aopalliance</artifactId>
    <version>1.0</version>
</dependency>
<dependency>
    <groupId>cglib</groupId>
    <artifactId>cglib</artifactId>
    <version>3.1</version>
</dependency>

<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.7.4</version>
</dependency>
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-core</artifactId>
    <version>2.7.4</version>
</dependency>
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-annotations</artifactId>
    <version>2.7.4</version>
</dependency>

```

配置文件

Web.xml

```

<web-app version="2.4"
    xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
    http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
    <display-name>Archetype Created Web Application</display-name>
    <servlet>
        <servlet-name>mvc-dispatcher</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

```

```

<init-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath:/Spring-mvc.xml</param-value>
</init-param>
<load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>mvc-dispatcher</servlet-name>
    <url-pattern>/</url-pattern>
</servlet-mapping>
<!-- Spring context startup -->
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath*:./Spring-service.xml</param-value>
</context-param>

<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
</web-app>

```

Spring-service.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xsi:schemaLocation="
           http://www.springframework.org/schema/tx
           http://www.springframework.org/schema/tx/spring-tx.xsd
           http://www.springframework.org/schema/aop
           http://www.springframework.org/schema/aop/spring-aop.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context.xsd
           http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd">
    <context:component-scan base-package="com.spring">
        <context:exclude-filter type="annotation"
                               expression="org.springframework.stereotype.Controller"></context:exclude-filter>
    
```

```

</context:component-scan>
<!-- 引入参数配置文件 -->
<bean id="propertyConfigure"
      class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
    <property name="locations">
      <list>
        <value>classpath:jdbc.properties</value>
      </list>
    </property>
</bean>

<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
      destroy-method="close">
    <property name="driverClassName" value="${jdbc.driver}"/>
    <property name="url" value="${jdbc.url}"/>
    <property name="username" value="${jdbc.user}"/>
    <property name="password" value="${jdbc.password}"/>
    <!-- data source configuration -->
    <property name="initialSize" value="20"/>
    <!-- initial connections -->
    <property name="maxActive" value="100"/>
    <!-- MAX connections -->
    <property name="maxIdle" value="30"/>
    <!-- MAX idle connections -->
    <property name="minIdle" value="5"/>
    <!-- MIN idle connections -->
    <property name="testWhileIdle" value="true"/>
    <property name="testOnBorrow" value="false"/>
    <property name="testOnReturn" value="false"/>
    <property name="validationQuery" value="select 1"/>
    <property name="timeBetweenEvictionRunsMillis" value="20000"/>
    <property name="numTestsPerEvictionRun" value="100"/>
</bean>
<!-- ====== 事务相关控制 ====== -->
<bean id="transactionManager"
      class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
    <property name="dataSource" ref="dataSource"/>
</bean>
<tx:annotation-driven transaction-manager="transactionManager"/>

<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
    <property name="mapperLocations" value="classpath:/com/*.xml"/>
    <property name="dataSource" ref="dataSource"/>
</bean>

```

```

<bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
    <property name="basePackage" value="com.spring.learn.dao"/>
    <property name="sqlSessionFactoryBeanName" value="sqlSessionFactory" />
</bean>
<aop:config>
    <aop:advisor          pointcut="execution(*           com.spring.learn.*service.*(..))"
advice-ref="txAdvice" />
</aop:config>

<tx:advice id="txAdvice">
    <tx:attributes>
        <tx:method name="get*" read-only="true" />
        <tx:method name="find*" read-only="true" />
        <tx:method name="query*" read-only="true" />
        <tx:method name="is*" read-only="true" />
        <tx:method name="*" propagation="REQUIRED" />
    </tx:attributes>
</tx:advice>
</beans>

```

Spring-mvc.xml

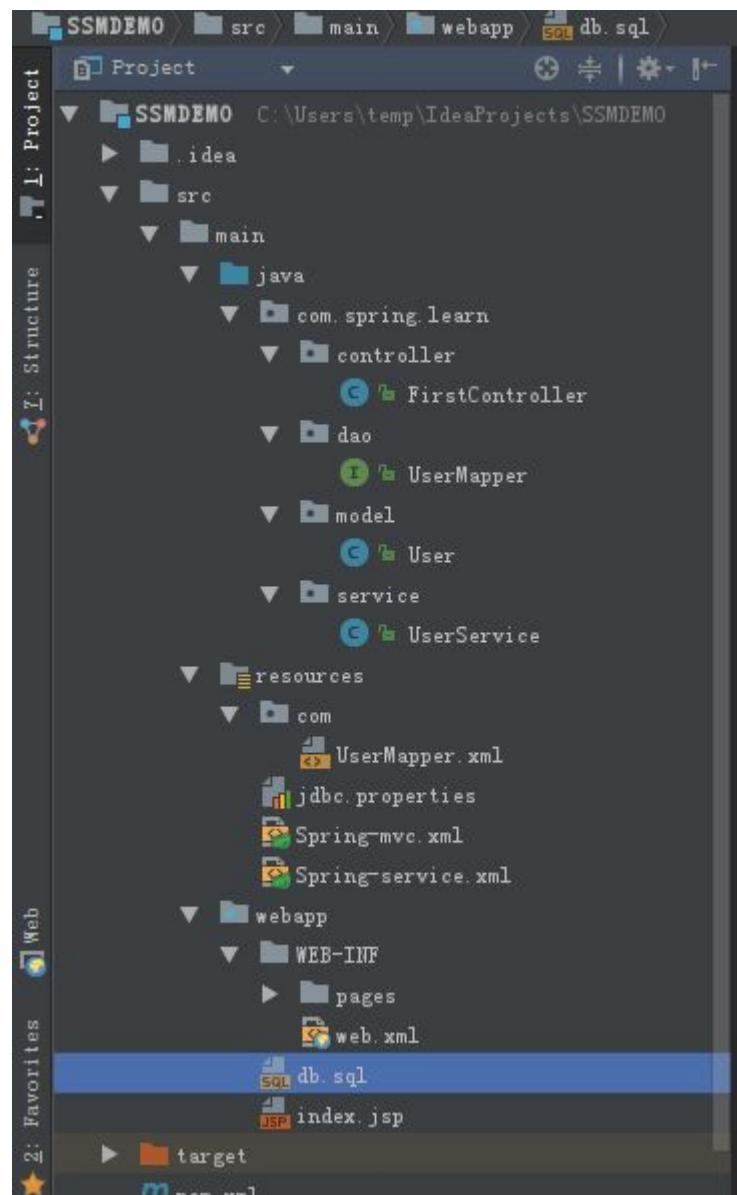
```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:mvc="http://www.springframework.org/schema/mvc"
       xsi:schemaLocation="
           http://www.springframework.org/schema/tx
           http://www.springframework.org/schema/tx/spring-tx.xsd
           http://www.springframework.org/schema/aop
           http://www.springframework.org/schema/aop/spring-aop.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context.xsd
           http://www.springframework.org/schema/mvc
           http://www.springframework.org/schema/mvc/spring-mvc.xsd
           http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd">
<context:component-scan base-package="com.spring.learn.controller"/>
<mvc:annotation-driven enable-matrix-variables="true"></mvc:annotation-driven>

```

```
<!--JSON 转换器-->
<bean id="mappingJacksonHttpMessageConverter"
      class="org.springframework.http.converter.json.MappingJackson2HttpMessageConverter">
    <property name="supportedMediaTypes">
      <list>
        <value>application/json;charset=UTF-8</value>
        <value>text/html;charset=UTF-8</value>
      </list>
    </property>
  </bean>
  <!-- 启动 SpringMVC 的注解功能，完成请求和注解 POJO 的映射 -->
  <bean
    class="org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping">
  </bean>
  <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
    <property name="prefix" value="/WEB-INF/pages/">
    <property name="suffix" value=".jsp"/>
  </bean>
  <bean id="multipartResolver"
        class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
    <property name="defaultEncoding" value="utf-8"/>
    <property name="maxUploadSize" value="1024102410241024"/>
    <property name="maxInMemorySize" value="40960"/>
  </bean>
</beans>
```

项目目录结构



说明

Mapper, service, Controller 这里就不赘述，比较基础，自己查阅 demo。

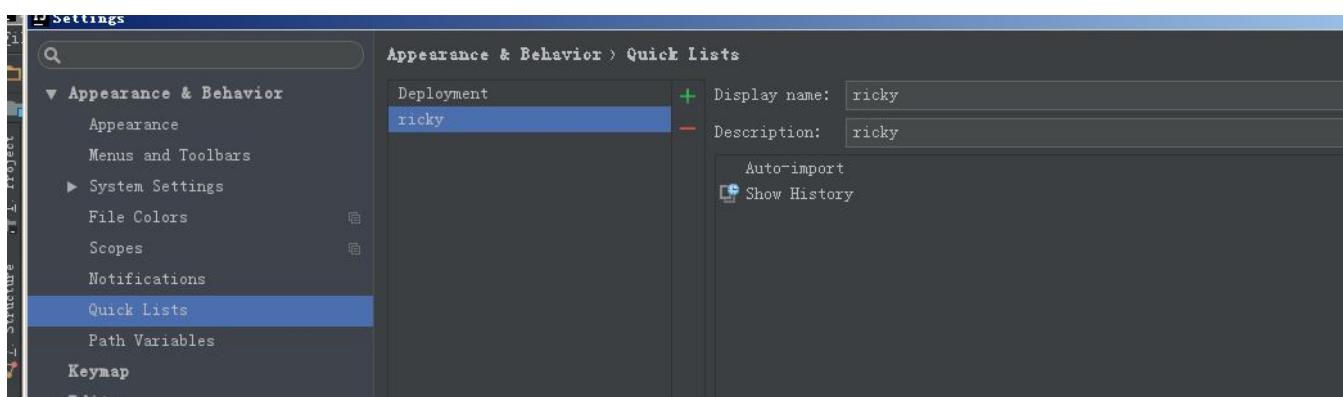
Demo 地址:

<http://pan.baidu.com/s/1ckwWG2>

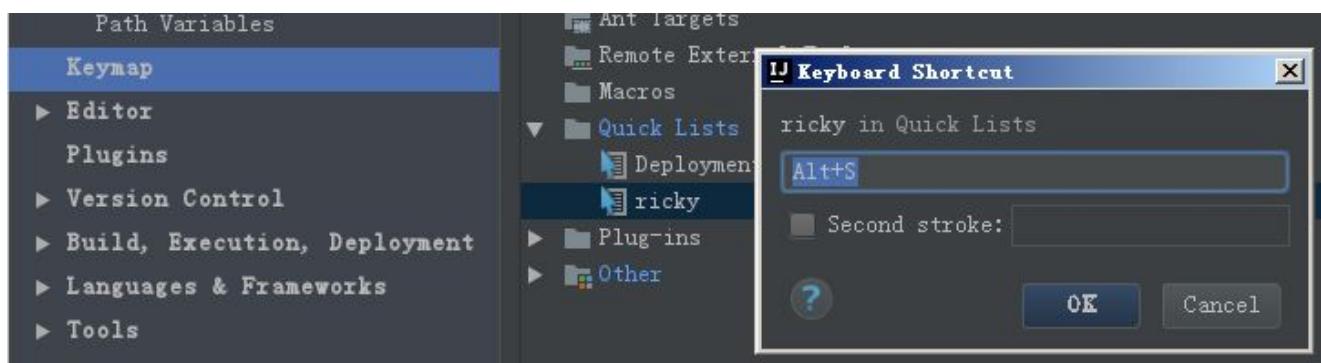
常用技巧&问题

创建自定义快捷列表

创建列表



设置快捷键



成品 (alt+s)

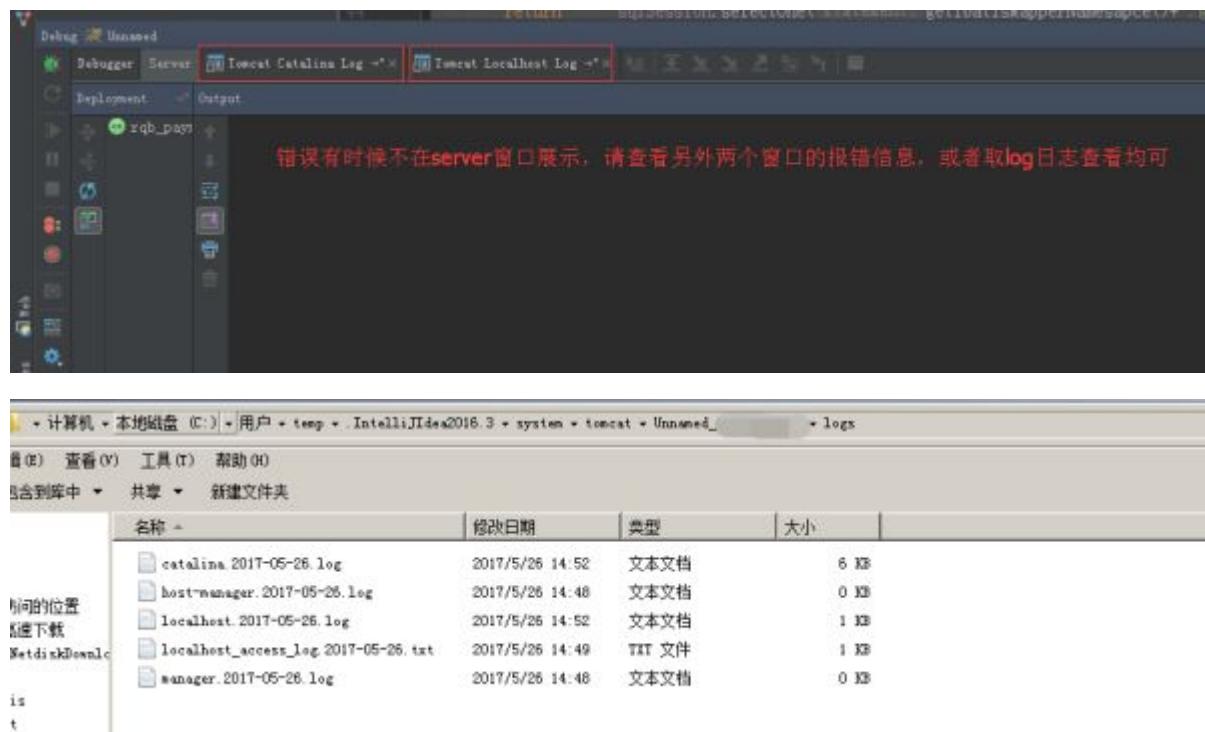
```
<!-- 根据loanId查询红包用户-->
<select id="getByLoanId" resultMap="coupon">
    SELECT <include refid="cou...
```

Tomcat 部署失败

情况 1

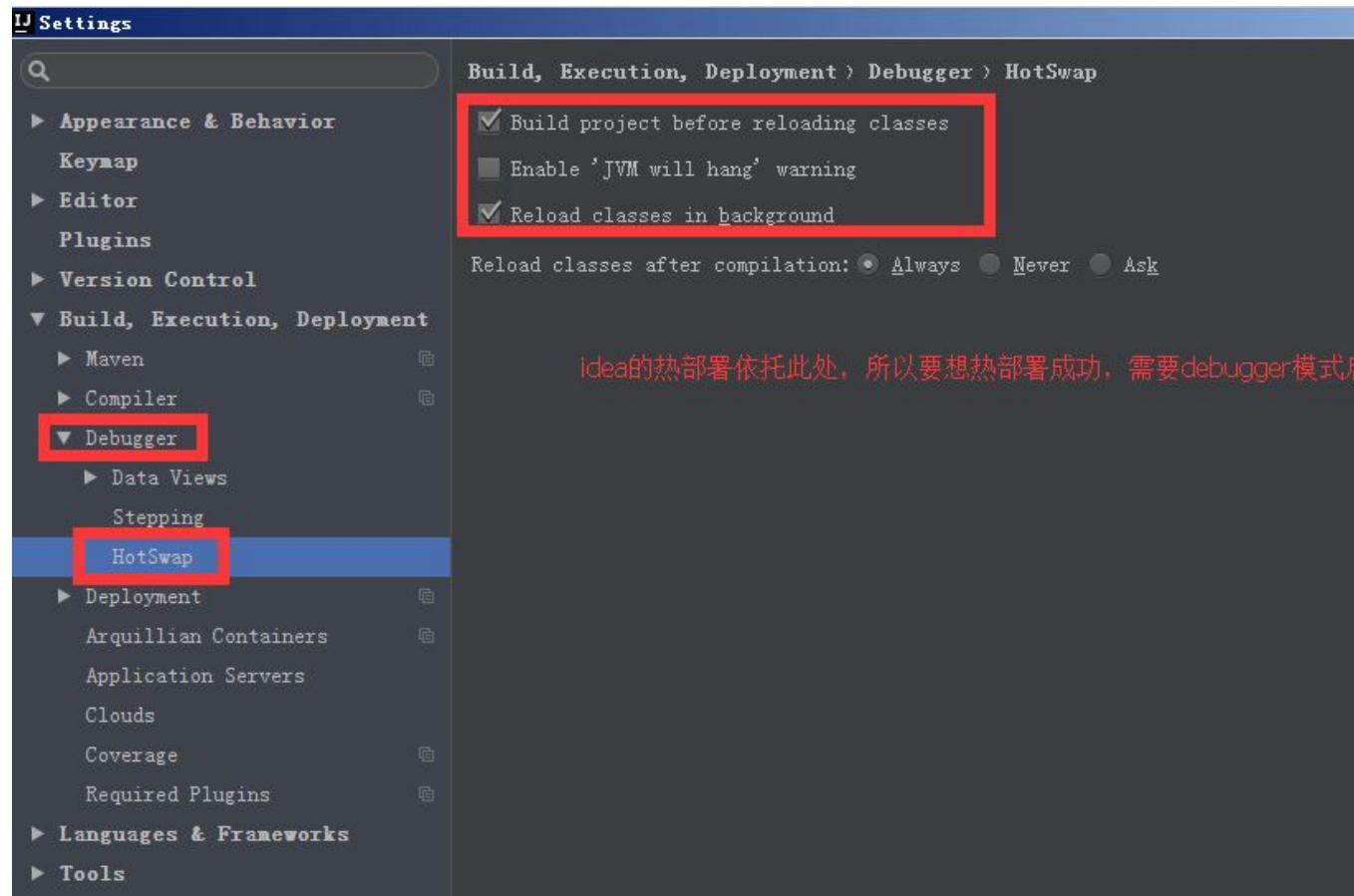
Artifact firstweb:war exploded: Error during artifact deployment. See server log for details.
错误有时候不在 server 窗口展示, 请查看另外两个窗口的报错信息, 或者取 log 日志查看均

可



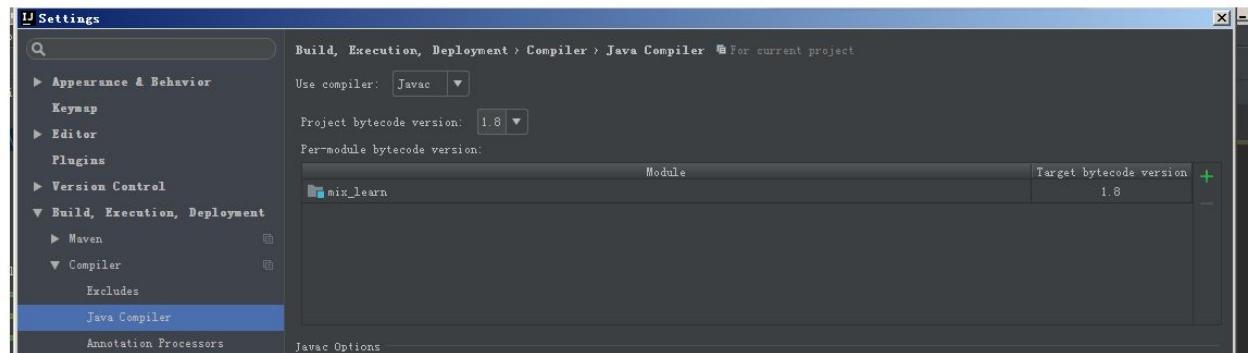
热部署注意事项

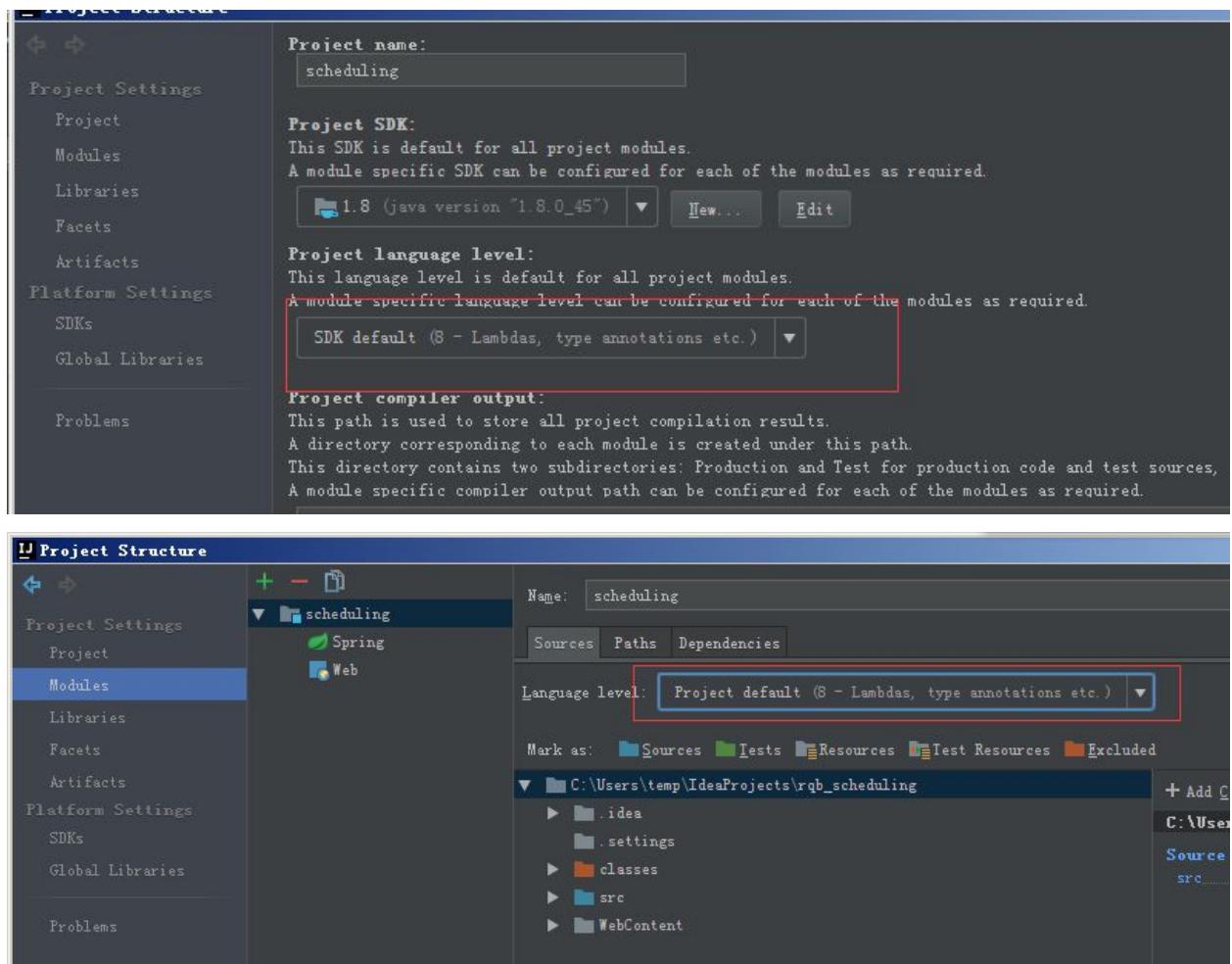
热部署依托于此处, 所以请**务必 debugger 模式启动**, 配合 **ctrl+F9**/或者设置失去焦点自动部署



修改 JAVA 编译版本

需要修改三个地方

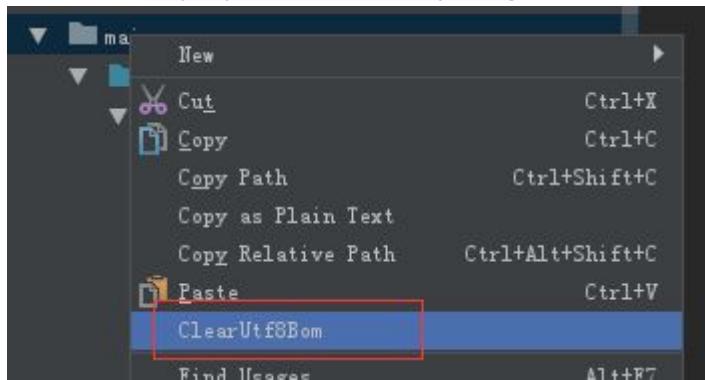




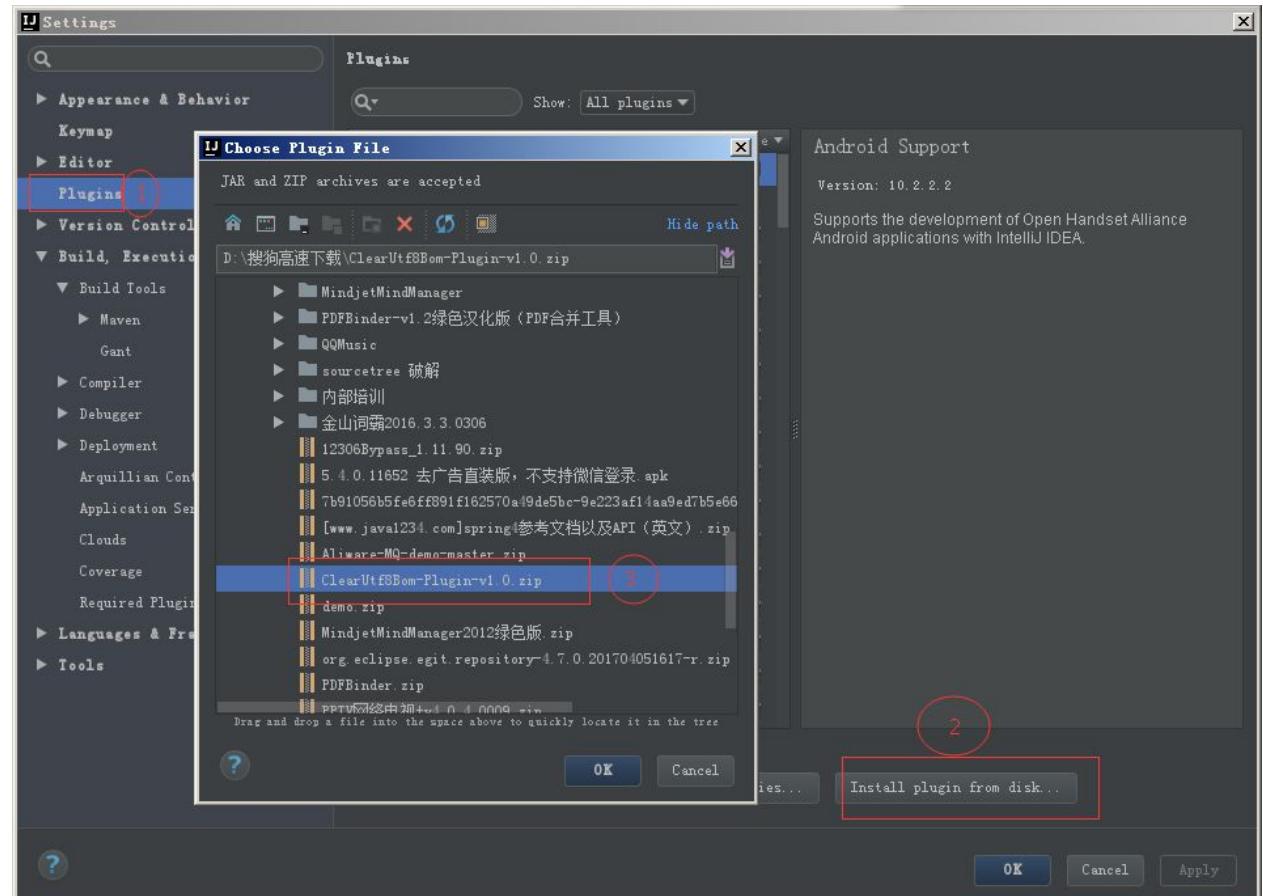
UTF8BOM 格式转 utf8

安装插件: ClearUtf8Bom-Plugin-v1.0

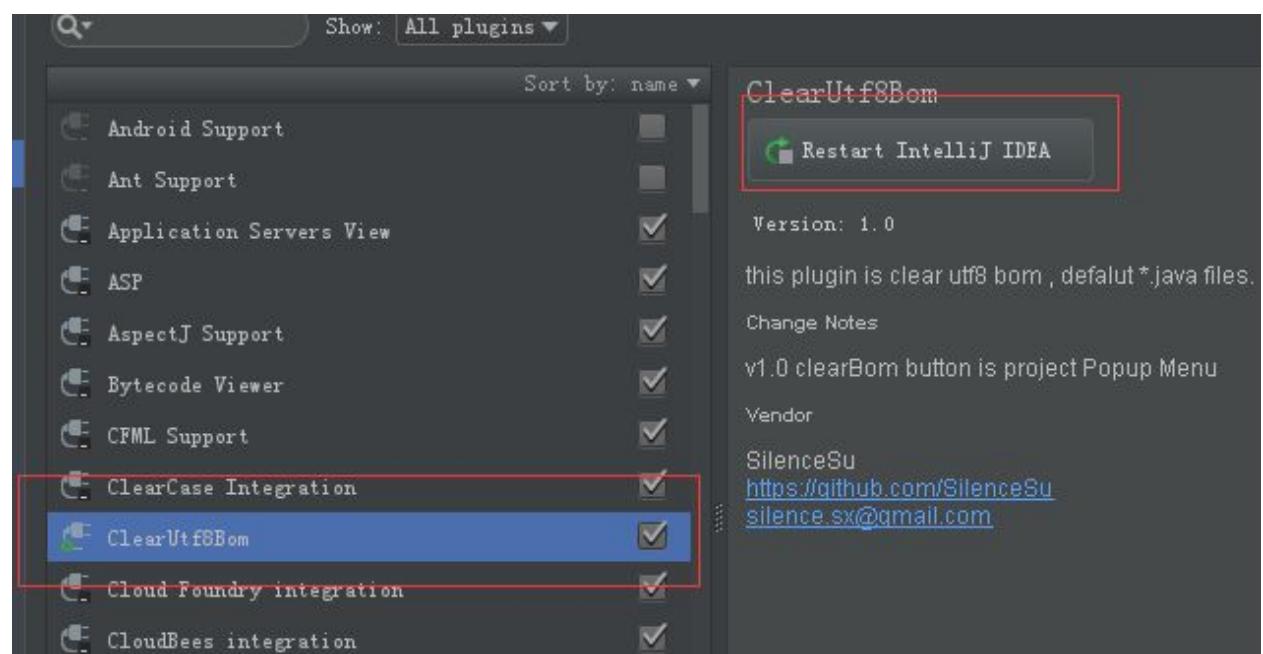
插件地址: <http://pan.baidu.com/s/1jIFUJFg>



插件安装(本地)



Ok 之后

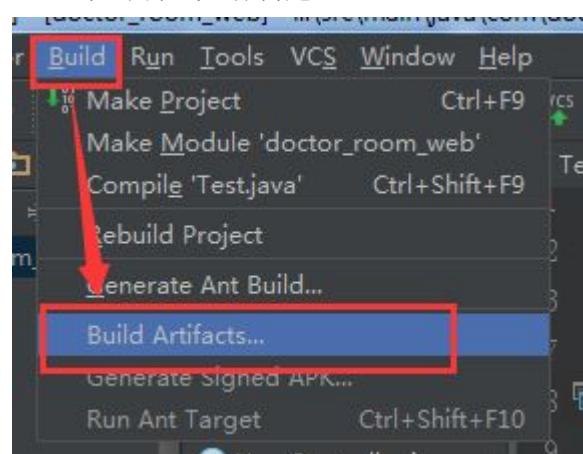


JSP 实时编译问题

在某些情况下，jsp 和 js 等静态资源更改时，不能实时生效。

1.可以通过 **ctrl+s** 达到实时编译。

2.通过如下方式进行构建。

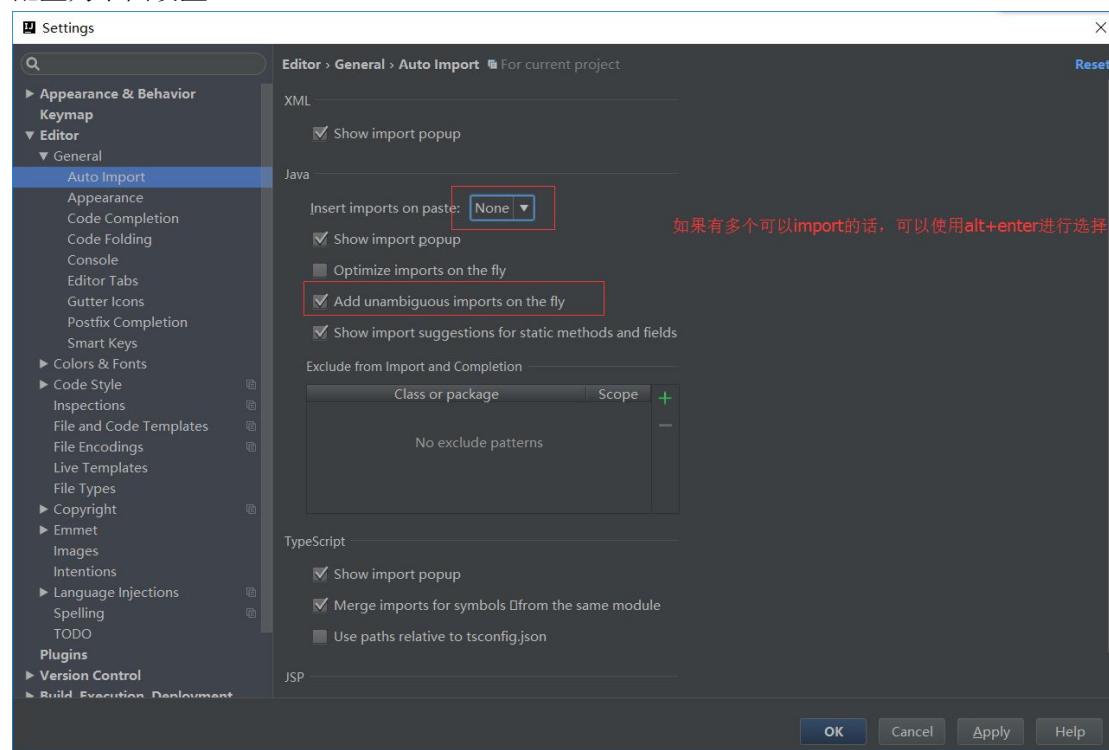


搜索功能失效

ctrl+shift+f 失效，一般是快捷键冲突，比如和搜狗输入法的冲突，修改快捷键即可。

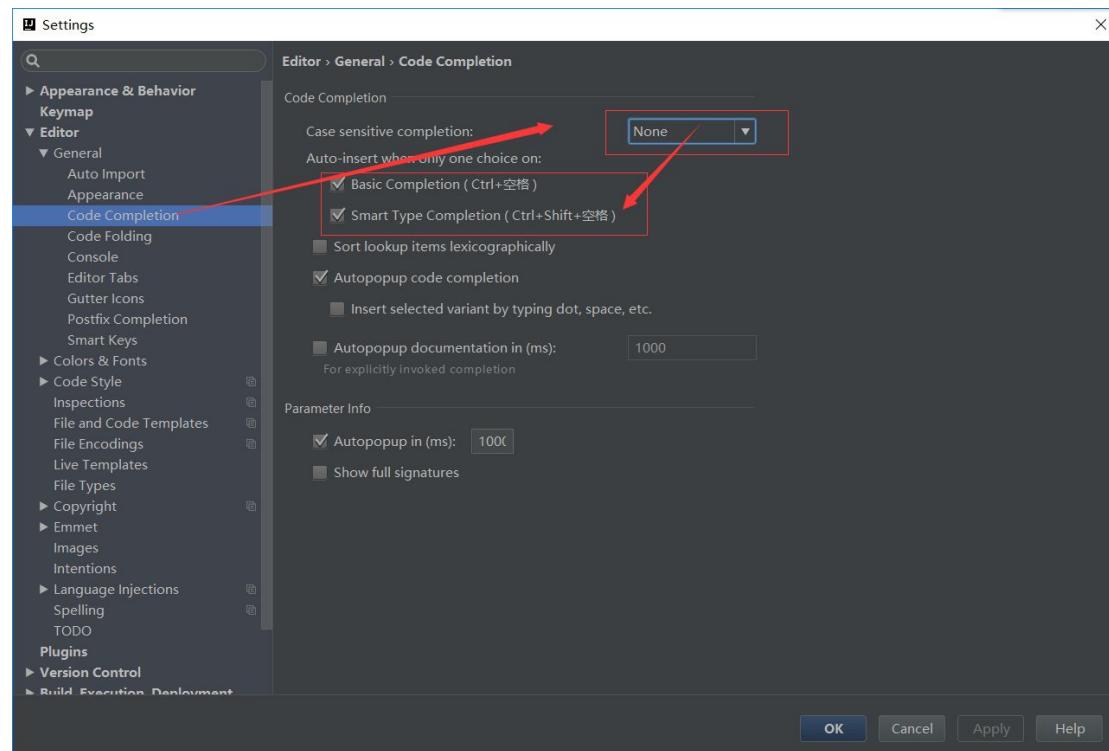
自动导入依赖

配置为下图设置，

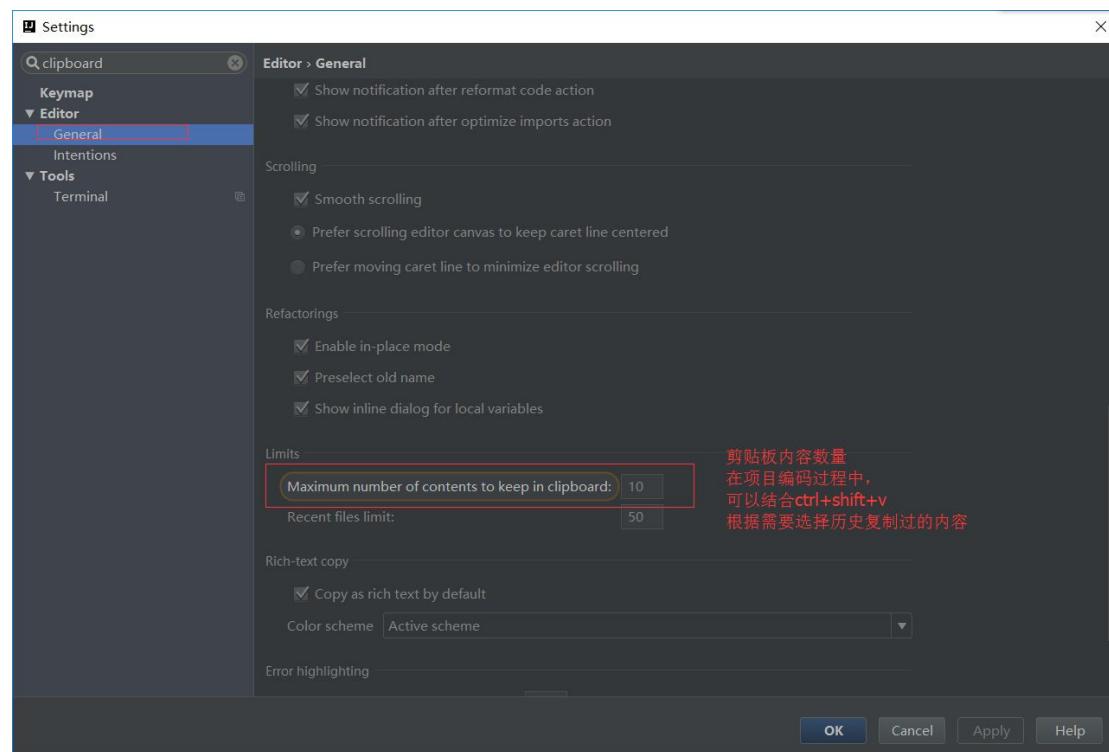


提示不区分大小写

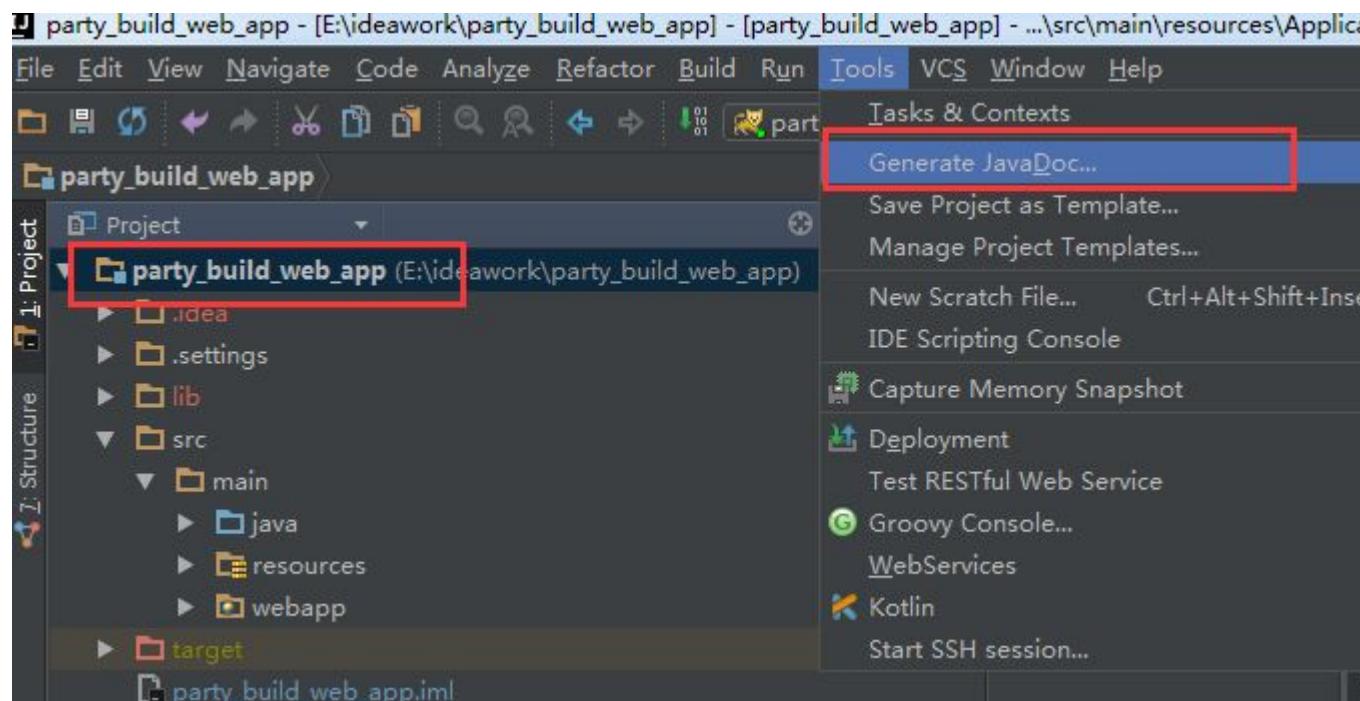
修改为如下图配置即可，

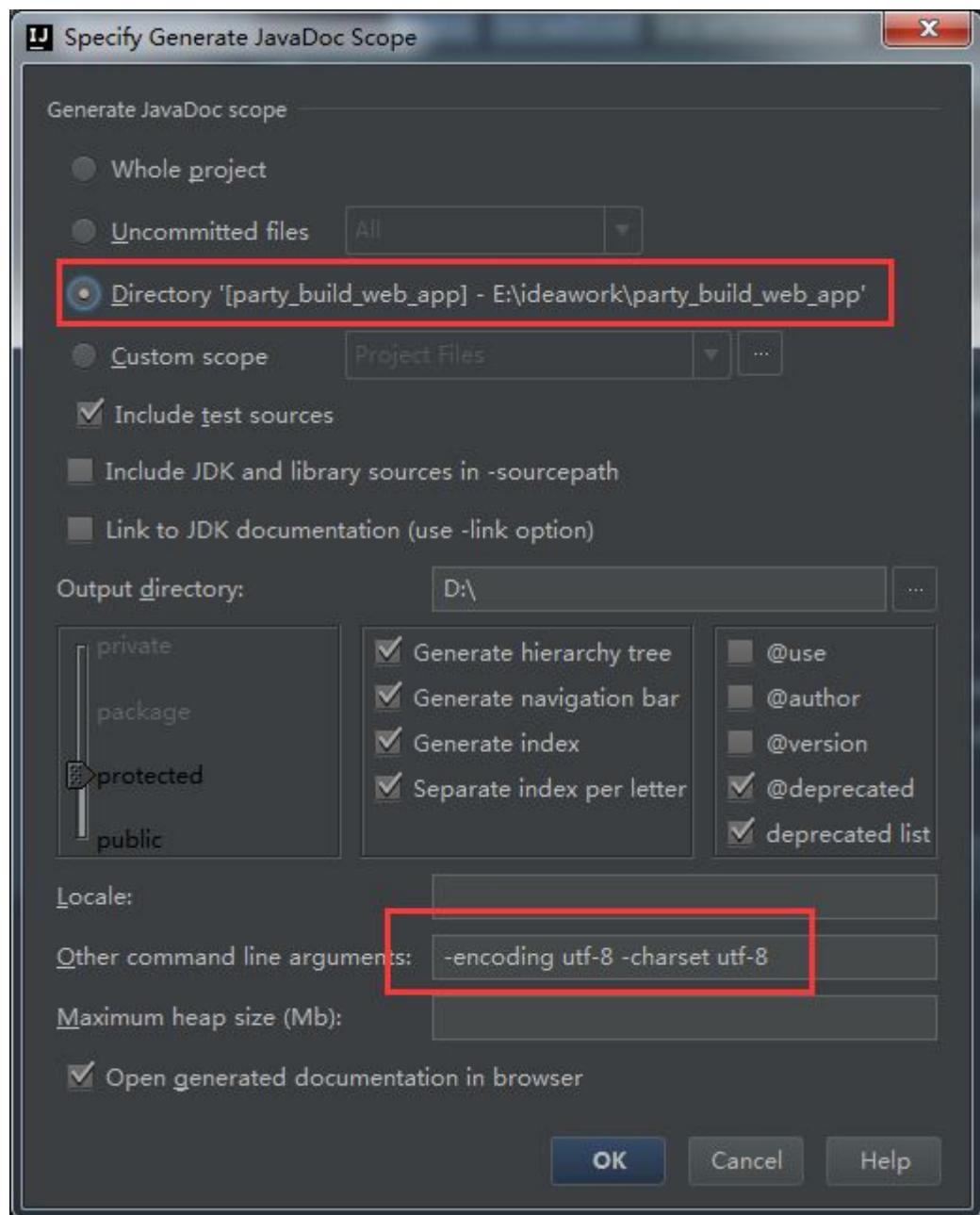


剪贴板数量设置



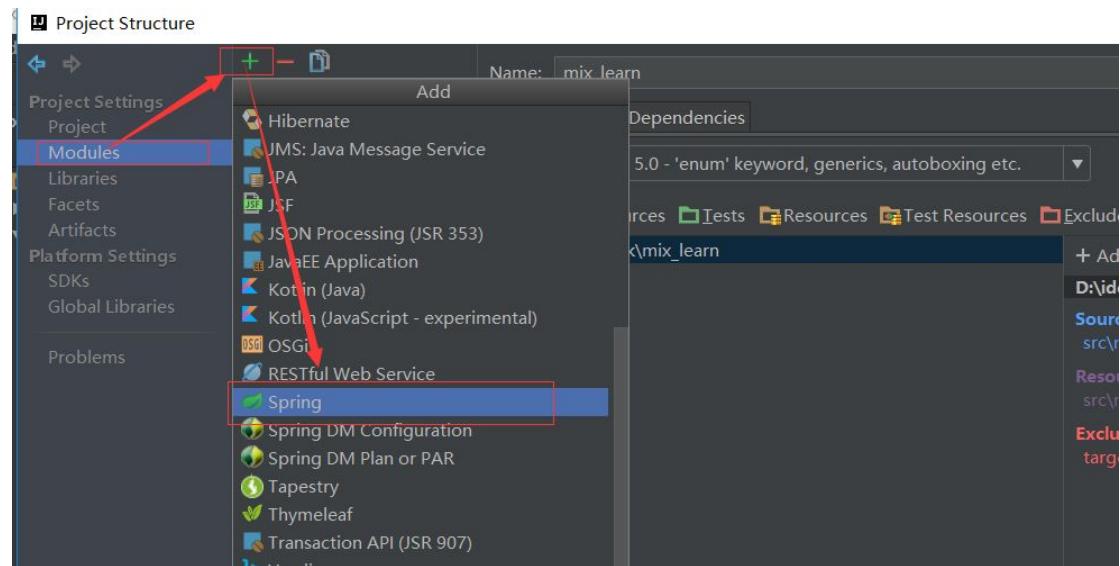
生成 JAVADOC



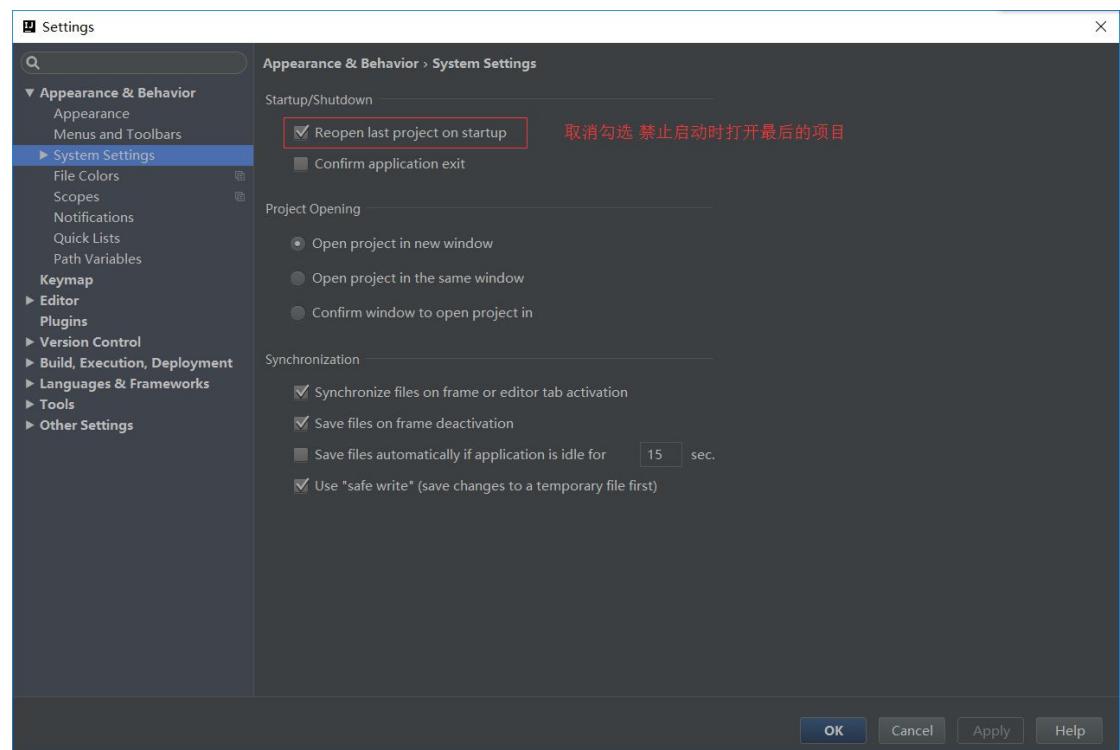


设置 Spring 支持

可以根据方法，直接调到对应的 jsp 页面

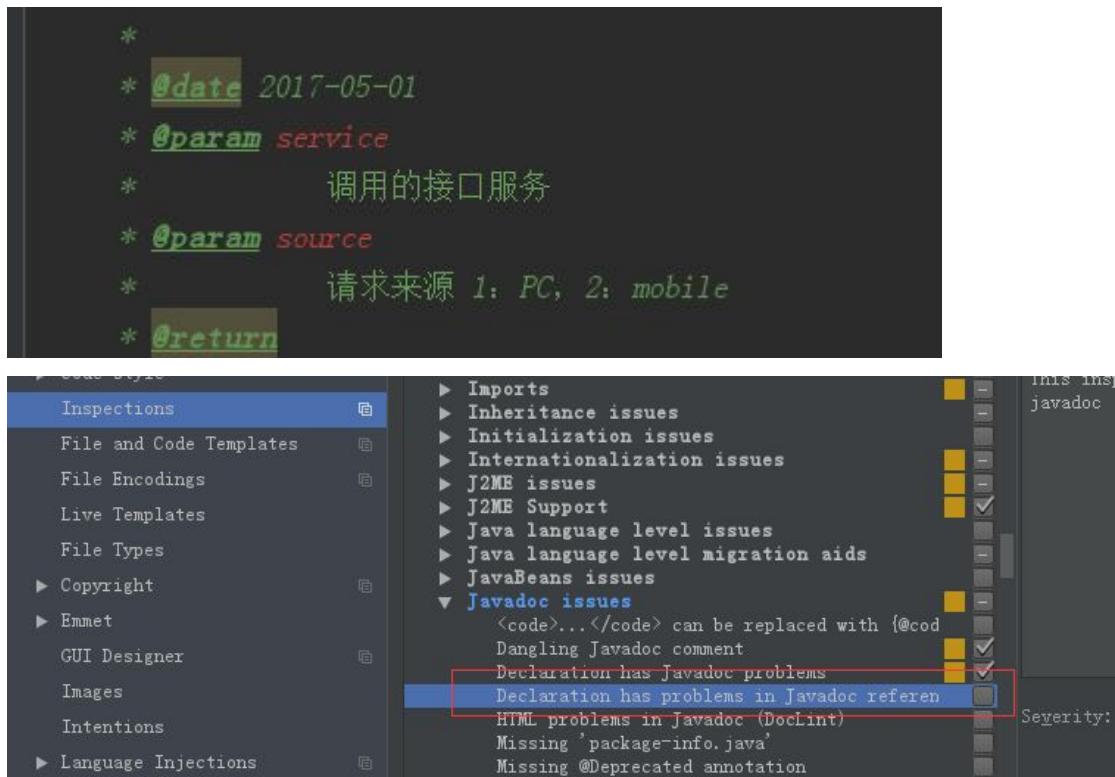


启动时不自动打开项目



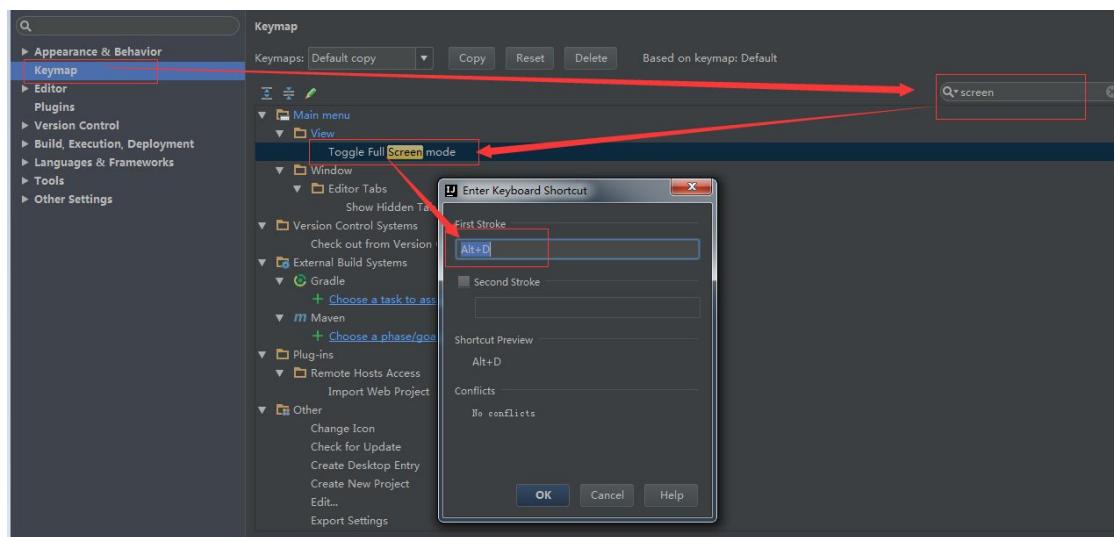
取消注释检查

注释中有时候会爆红，不习惯的可以取消勾选下面的。



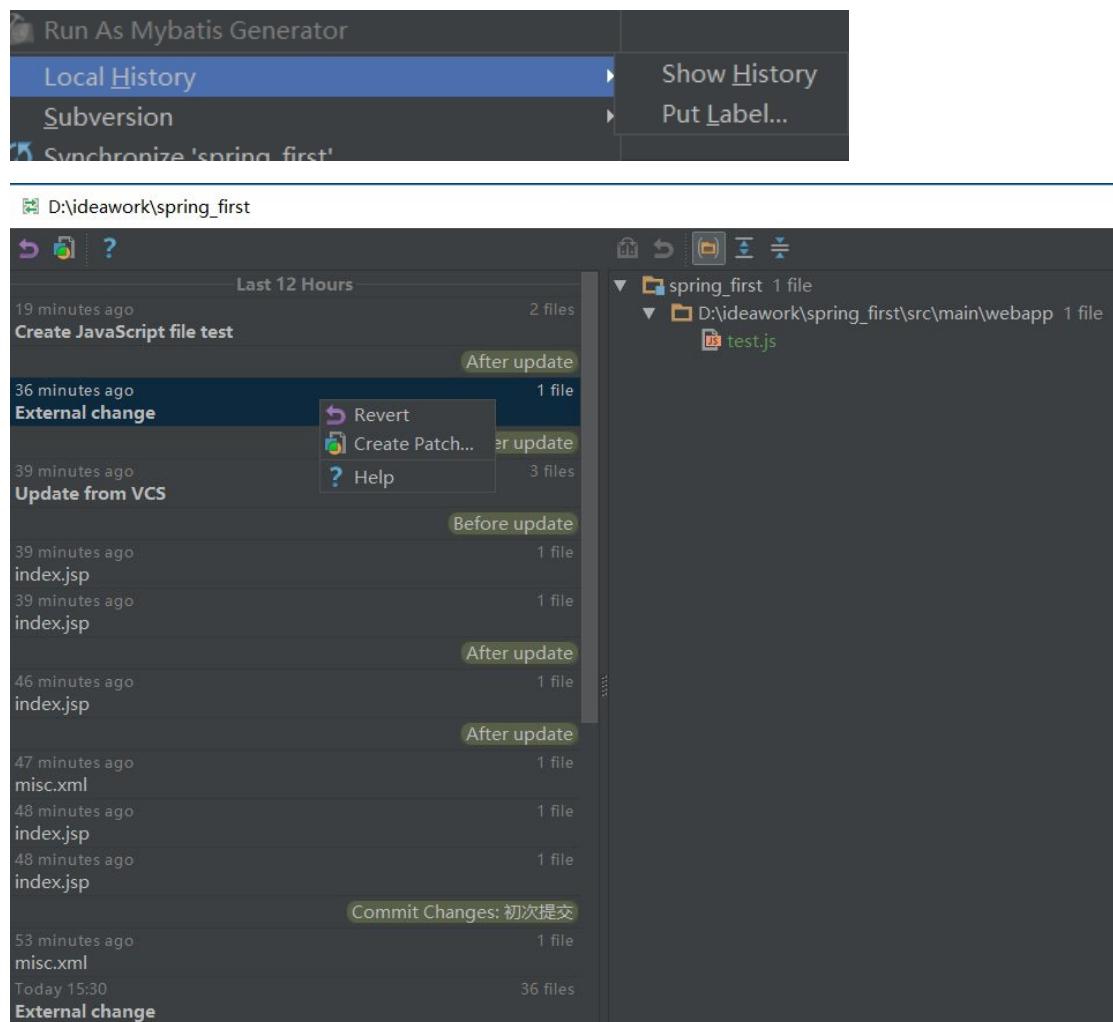
全屏设置

通过如下配置，可以使用 alt+d 切换全屏



本地历史

右键文件或者项目，可以进行历史找回和还原

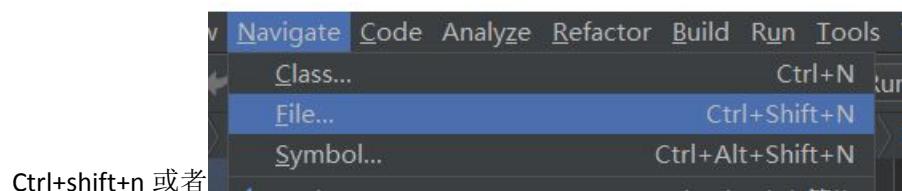


搜索

所有文件

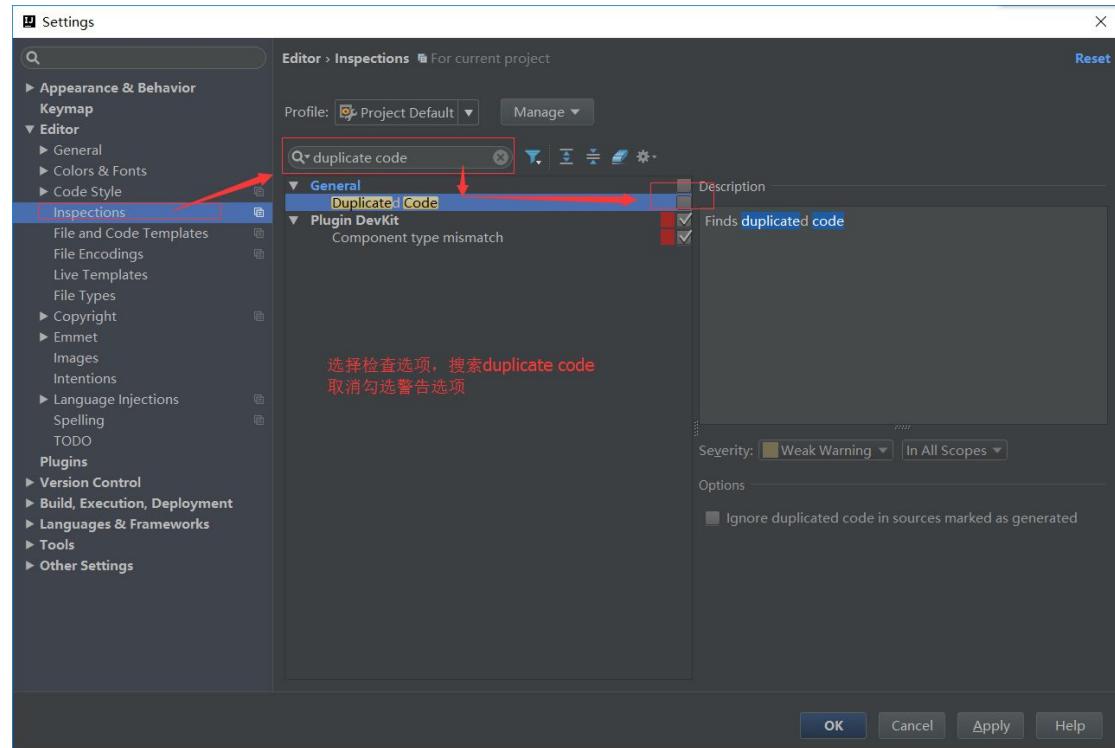
双击 shift, 或者选择右上角面板搜索按钮。

项目文件



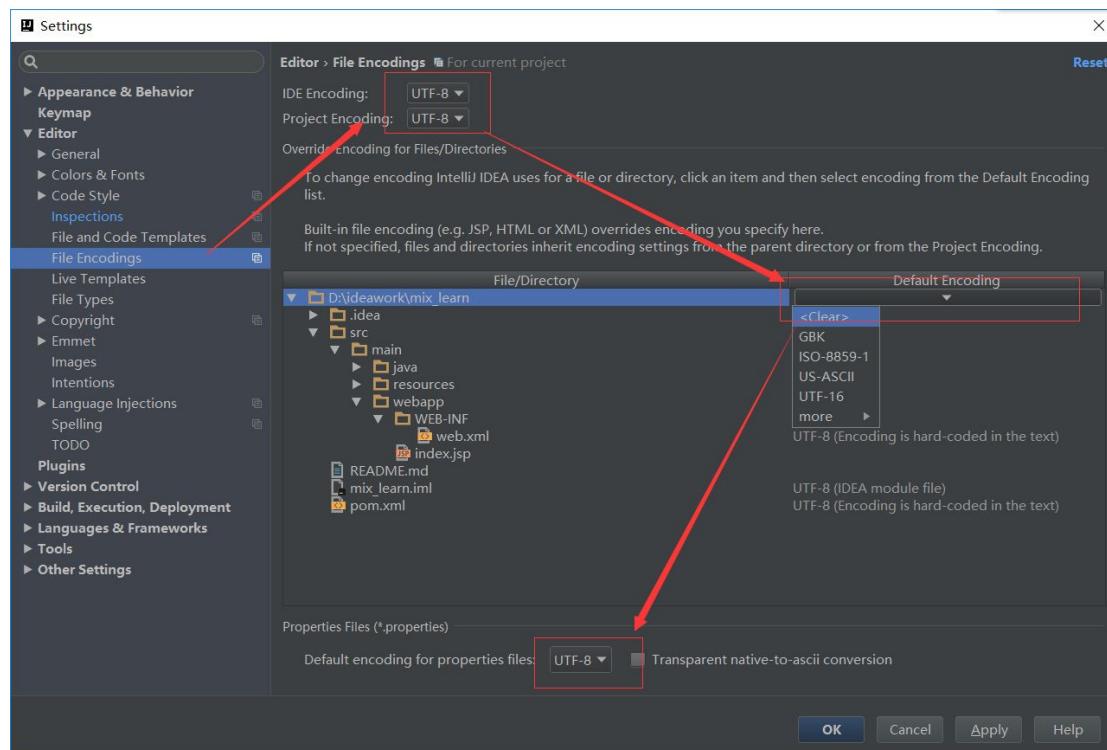
取消重复代码提示

配置如下图



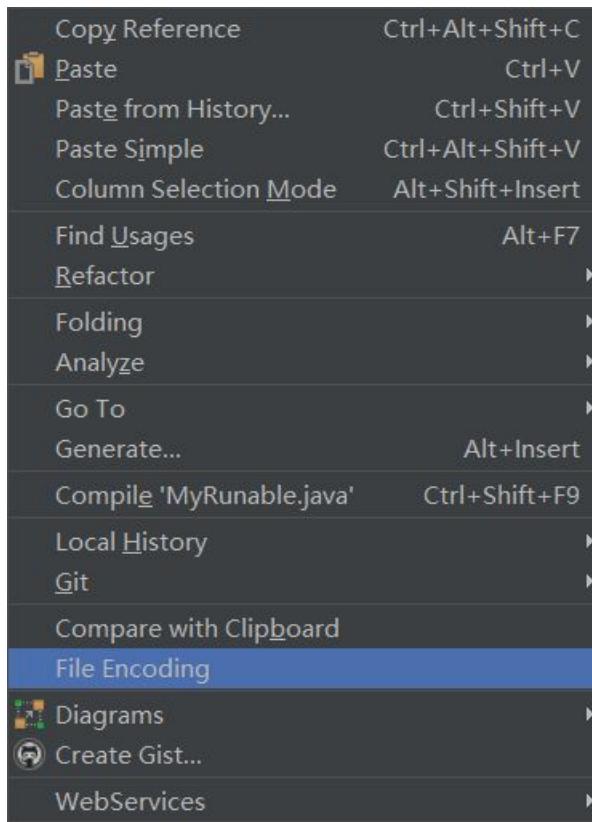
设置字符集

项目字符集



单个文件

右键，选择菜单中的 file encoding



Idea 优化配置

启动参数优化

Red arrows point to the configuration options in the `idea.properties` file:

```

1 -server
2 -Xms512m
3 -Xmx1024m
4 -XX:ReservedCodeCacheSize=240m
5 -XX:+UseConcMarkSweepGC
6 -XX:SoftRefLRUPolicyMSPerMB=50
7 -ea
8 -Dsun.io.useCanonCaches=false
9 -Djava.net.preferIPv4Stack=true
10 -XX:+HeapDumpOnOutOfMemoryError
11 -XX:-OmitStackTraceInFastThrow
12
13 #优化配置
14 #字节码校验策略
15 -Xverify:none
16 #关闭CLASS的垃圾回收功能，就是虚拟机加载的类，即便是不使用，没有实例也不会回收
17 -XnoClassGC
18 #可以让IDEA最小化到任务栏时依然保持以占有的内存，当你重新回到IDEA，能够被快速显示，而不是由灰白的界面逐渐显现整个界面，加快回复到原界面的速度
19 -Dsun.awt.keepWorkingSetOnMinimize=true
20 -XX:+UseParNewGC
21 -XX:CMSInitiatingOccupancyFraction=85
22 #并发回收的时候进行内存压缩
23 -XX:+UseCMSCompactAtFullCollection
24 #5次full GC之后进行内存压缩
25 -XX:CMSFullGCsBeforeCompaction=5

```

Annotations:

- 用server方式启动，运行性能较好，启动较慢
调整堆内存初识和最大

```

-server
-Xms512m
-Xmx1024m
-XX:ReservedCodeCacheSize=240m
-XX:+UseConcMarkSweepGC
-XX:SoftRefLRUPolicyMSPerMB=50

```

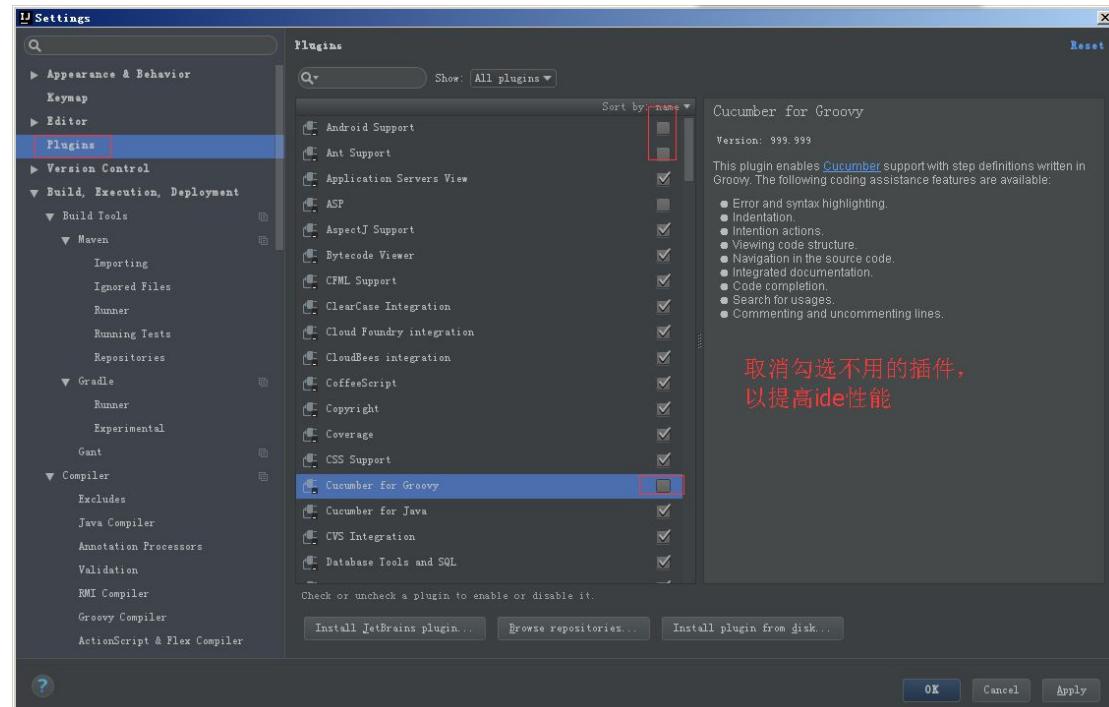
```

-ea
-Dsun.io.useCanonCaches=false
-Djava.net.preferIPv4Stack=true
-XX:+HeapDumpOnOutOfMemoryError
-XX:-OmitStackTraceInFastThrow

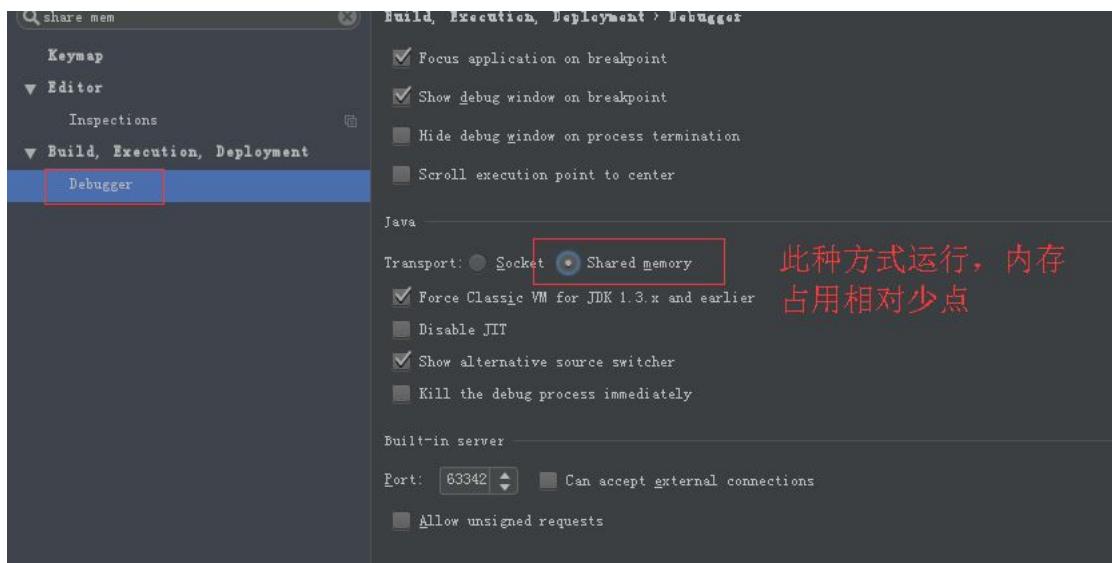
#优化配置
#字节码校验策略
-Xverify:none
#关闭 CLASS 的垃圾回收功能，就是虚拟机加载的类，即便是不使用，没有实例也不会回收
-Xnoclassgc
#可以让 IDEA 最小化到任务栏时依然保持以占有的内存，当你重新回到 IDEA，能够被快速显示，而不是由灰白的界面逐渐显现整个界面，加快回复到原界面的速度
-Dsun.awt.keepWorkingSetOnMinimize=true
-XX:+UseParNewGC
-XX:CMSInitiatingOccupancyFraction=85
#并发回收的时候进行内存压缩
-XX:+UseCMSCompactAtFullCollection
#5 次 full GC 之后进行内存压缩
-XX:CMSFullGCsBeforeCompaction=5

```

插件优化



运行优化



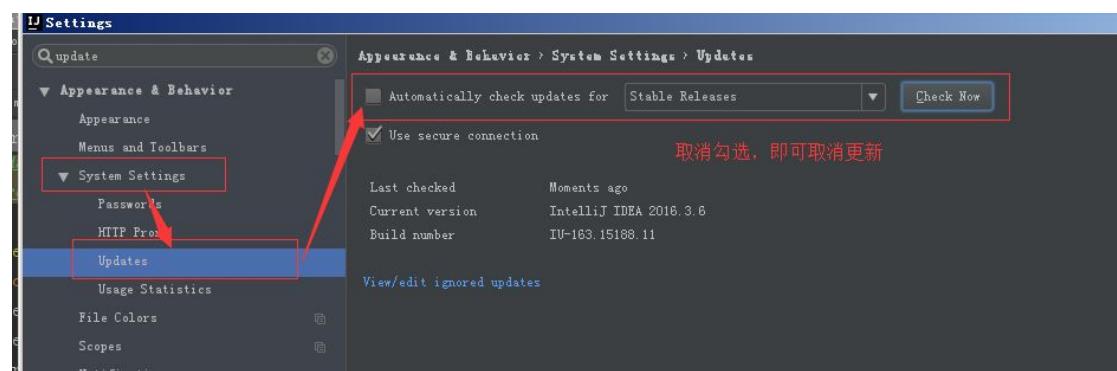
Git 证书失效

Fetch failed: unable to access 'https://gitlab.rqbaomidou.com/java/rqb_payment.git/': SSL certificate problem: certificate has expired

执行：

```
git config --global http.sslVerify false
```

取消更新



快捷键

Ctrl

快捷键	介绍
Ctrl + F	在当前文件进行文本查找 (必备)
Ctrl + R	在当前文件进行文本替换 (必备)
Ctrl + Z	撤销 (必备)
Ctrl + Y	删除光标所在行 或 删除选中的行 (必备)
Ctrl + X	剪切光标所在行 或 剪切选择内容
Ctrl + C	复制光标所在行 或 复制选择内容
Ctrl + D	复制光标所在行 或 复制选择内容，并把复制内容插入光标位置下面 (必备)
Ctrl + W	递进式选择代码块。可选中光标所在的单词或段落，连续按会在原有选中的基础上再扩展选中范围 (必备)
Ctrl + E	显示最近打开的文件记录列表 (必备)
Ctrl + N	根据输入的 类名 查找类文件 (必备)
Ctrl + G	在当前文件跳转到指定行处
Ctrl + J	插入自定义动态代码模板 (必备)
Ctrl + P	方法参数提示显示 (必备)
Ctrl + Q	光标所在的变量 / 类名 / 方法名等上面 (也可以在提示补充的时候按)， 显示文档内容
Ctrl + U	前往当前光标所在的方法的父类的方法 / 接口定义 (必备)
Ctrl + B	进入光标所在的方法/变量的接口或是定义处，等效于 Ctrl + 左键单击 (必备)
Ctrl + K	版本控制提交项目，需要此项目有加入到版本控制才可用
Ctrl + T	版本控制更新项目，需要此项目有加入到版本控制才可用
Ctrl + H	显示当前类的层次结构
Ctrl + O	选择可重写的方法
Ctrl + I	选择可继承的方法
Ctrl + +	展开代码
Ctrl + -	折叠代码
Ctrl + /	注释光标所在行代码，会根据当前不同文件类型使用不同的注释符号 (必备)
Ctrl + [移动光标到当前所在代码的花括号开始位置
Ctrl +]	移动光标到当前所在代码的花括号结束位置
Ctrl + F1	在光标所在的错误代码处显示错误信息 (必备)
Ctrl + F3	调转到所选中的词的下一个引用位置 (必备)
Ctrl + F4	关闭当前编辑文件
Ctrl + F8	在 Debug 模式下，设置光标当前行为断点，如果当前已经是断点则去掉断点
Ctrl + F9	执行 Make Project 操作
Ctrl + F11	选中文件 / 文件夹，使用助记符设定 / 取消书签 (必备)
Ctrl + F12	弹出当前文件结构层，可以在弹出的层上直接输入，进行筛选
Ctrl + Tab	编辑窗口切换，如果在切换的过程又加按上 delete，则是关闭对应选中的窗口
Ctrl + End	跳到文件尾
Ctrl + Home	跳到文件头

快捷键	介绍
Ctrl + Space	基础代码补全, 默认在 Windows 系统上被输入法占用, 需要进行修改, 建议修改为 Ctrl + 逗号 (必备)
Ctrl + Delete	删除光标后面的单词或是中文句 (必备)
Ctrl + BackSpace	删除光标前面的单词或是中文句 (必备)
Ctrl + 1, 2, 3...9	定位到对应数值的书签位置 (必备)
Ctrl + 左键单击在打开的文件标题上, 弹出该文件路径 (必备)	
Ctrl + 光标定位按 Ctrl 不要松开, 会显示光标所在的类信息摘要	
Ctrl + 左方向键光标跳转到当前单词 / 中文句的左侧开头位置 (必备)	
Ctrl + 右方向键光标跳转到当前单词 / 中文句的右侧开头位置 (必备)	
Ctrl + 前方向键等效于鼠标滚轮向前效果 (必备)	
Ctrl + 后方向键等效于鼠标滚轮向后效果 (必备)	

Alt

快捷键	介绍
Alt + `	显示版本控制常用操作菜单弹出层 (必备)
Alt + Q	弹出一个提示, 显示当前类的声明 / 上下文信息
Alt + F1	显示当前文件选择目标弹出层, 弹出层中有很多目标可以进行选择 (必备)
Alt + F2	对于前面页面, 显示各类浏览器打开目标选择弹出层
Alt + F3	选中文本, 逐个往下查找相同文本, 并高亮显示
Alt + F7	查找光标所在的方法 / 变量 / 类被调用的地方
Alt + F8	在 Debug 的状态下, 选中对象, 弹出可输入计算表达式调试框, 查看该输入内容的调试结果
Alt + Home	定位 / 显示到当前文件的 Navigation Bar
Alt + Enter	IntelliJ IDEA 根据光标所在问题, 提供快速修复选择, 光标放在的位置不同提示的结果也不同 (必备)
Alt + Insert	代码自动生成, 如生成对象的 set / get 方法, 构造函数, toString() 等 (必备)
Alt + 左方向键	切换当前已打开的窗口中的子视图, 比如 Debug 窗口中有 Output、Debugger 等子视图, 用此快捷键就可以在子视图中切换 (必备)
Alt + 右方向键	按切换当前已打开的窗口中的子视图, 比如 Debug 窗口中有 Output、Debugger 等子视图, 用此快捷键就可以在子视图中切换 (必备)
Alt + 前方向键	当前光标跳转到当前文件的前一个方法名位置 (必备)
Alt + 后方向键	当前光标跳转到当前文件的后一个方法名位置 (必备)
Alt + 1, 2, 3...9	显示对应数值的选项卡, 其中 1 是 Project 用得最多 (必备)

Shift

快捷键	介绍
-----	----

快捷键	介绍
Shift + F1	如果有外部文档可以连接外部文档
Shift + F2	跳转到上一个高亮错误 或 警告位置
Shift + F3	在查找模式下, 查找匹配上一个
Shift + F4	对当前打开的文件, 使用新 Windows 窗口打开, 旧窗口保留
Shift + F6	对文件 / 文件夹 重命名
Shift + F7	在 Debug 模式下, 智能步入。断点所在行上有多个方法调用, 会弹出进入哪个方法
Shift + F8	在 Debug 模式下, 跳出, 表现出来的效果跟 F9 一样
Shift + F9	等效于点击工具栏的 Debug 按钮
Shift + F10	等效于点击工具栏的 Run 按钮
Shift + F11	弹出书签显示层 (必备)
Shift + Tab	取消缩进 (必备)
Shift + ESC	隐藏当前 或 最后一个激活的工具窗口
Shift + End	选中光标到当前行尾位置
Shift + Home	选中光标到当前行头位置
Shift + Enter	开始新一行。光标所在行下空出一行, 光标定位到新行位置 (必备)
Shift + 左键单击	在打开的文件名上按此快捷键, 可以关闭当前打开文件 (必备)
Shift + 滚轮前后滚动	滚动当前文件的横向滚动轴滚动 (必备)

Ctrl + Alt

快捷键	介绍
Ctrl + Alt + L	格式化代码, 可以对当前文件和整个包目录使用 (必备)
Ctrl + Alt + O	优化导入的类, 可以对当前文件和整个包目录使用 (必备)
Ctrl + Alt + I	光标所在行 或 选中部分进行自动代码缩进, 有点类似格式化
Ctrl + Alt + T	对选中的代码弹出环绕选项弹出层 (必备)
Ctrl + Alt + J	弹出模板选择窗口, 将选定的代码加入动态模板中
Ctrl + Alt + H	调用层次
Ctrl + Alt + B	在某个调用的方法名上使用会跳到具体的实现处, 可以跳过接口
Ctrl + Alt + V	快速引进变量
Ctrl + Alt + Y	同步、刷新
Ctrl + Alt + S	打开 IntelliJ IDEA 系统设置 (必备)
Ctrl + Alt + F7	显示使用的地方。寻找被该类或是变量被调用的地方, 用弹出框的方式找出来
Ctrl + Alt + F11	切换全屏模式
Ctrl + Alt + Enter	光标所在行上空出一行, 光标定位到新行 (必备)
Ctrl + Alt + Home	弹出跟当前文件有关联的文件弹出层
Ctrl + Alt + Space	类名自动完成
Ctrl + Alt + 左方向键	退回到上一个操作的地方 (必备)
Ctrl + Alt + 右方向键	前进到上一个操作的地方 (必备)
Ctrl + Alt + 前方向键	在查找模式下, 跳到上个查找的文件
Ctrl + Alt + 后方向键	在查找模式下, 跳到下个查找的文件

Ctrl + Shift

快捷键	介绍
Ctrl + Shift + F	根据输入内容查找整个项目 或 指定目录内文件 (必备)
Ctrl + Shift + R	根据输入内容替换对应内容, 范围为整个项目 或 指定目录内文件 (必备)
Ctrl + Shift + J	自动将下一行合并到当前行末尾 (必备)
Ctrl + Shift + Z	取消撤销 (必备)
Ctrl + Shift + W	递进式取消选择代码块。可选中光标所在的单词或段落, 连续按会在原有选中的基础上再扩展取消选中范围 (必备)
Ctrl + Shift + N	通过文件名定位 / 打开文件 / 目录, 打开目录需要在输入的内容后面多加一个正斜杠 (必备)
Ctrl + Shift + U	对选中的代码进行大 / 小写轮流转换 (必备)
Ctrl + Shift + T	对当前类生成单元测试类, 如果已经存在的单元测试类则可以进行选择 (必备)
Ctrl + Shift + C	复制当前文件磁盘路径到剪贴板 (必备)
Ctrl + Shift + V	弹出缓存的最近拷贝的内容管理器弹出层
Ctrl + Shift + E	显示最近修改的文件列表的弹出层
Ctrl + Shift + H	显示方法层次结构
Ctrl + Shift + B	跳转到类型声明处 (必备)
Ctrl + Shift + I	快速查看光标所在的方法 或 类的定义
Ctrl + Shift + A	查找动作 / 设置
Ctrl + Shift + /	代码块注释 (必备)
Ctrl + Shift + [选中从光标所在位置到它的顶部中括号位置 (必备)
Ctrl + Shift +]	选中从光标所在位置到它的底部中括号位置 (必备)
Ctrl + Shift + +	展开所有代码 (必备)
Ctrl + Shift + -	折叠所有代码 (必备)
Ctrl + Shift + F7	高亮显示所有该选中文本, 按 Esc 高亮消失 (必备)
Ctrl + Shift + F8	在 Debug 模式下, 指定断点进入条件
Ctrl + Shift + F9	编译选中的文件 / 包 / Module
Ctrl + Shift + F12	编辑器最大化 (必备)
Ctrl + Shift + Space	智能代码提示
Ctrl + Shift + Enter	自动结束代码, 行末自动添加分号 (必备)
Ctrl + Shift + Backspace	退回到上次修改的地方 (必备)
Ctrl + Shift + 1, 2, 3...9	快速添加指定数值的书签 (必备)
Ctrl + Shift + 左键单击	把光标放在某个类变量上, 按此快捷键可以直接定位到该类中 (必备)
Ctrl + Shift + 左键	在代码文件上, 光标跳转到当前单词 / 中文句的左侧开头位置, 同时选中该单词 / 中文句 (必备)
Ctrl + Shift + 右键	在代码文件上, 光标跳转到当前单词 / 中文句的右侧开头位置, 同时选中该单词 / 中文句 (必备)
Ctrl + Shift + 前键	光标放在方法名上, 将方法移动到上一个方法前面, 调整方法排序 (必备)

快捷键	介绍
Ctrl + Shift + 后方向键	光标放在方法名上, 将方法移动到下一个方法前面, 调整方法排序 (必备)

Alt + Shift

快捷键	介绍
Alt + Shift + N	选择 / 添加 task (必备)
Alt + Shift + F	显示添加到收藏夹弹出层 / 添加到收藏夹
Alt + Shift + C	查看最近操作项目的变化情况列表
Alt + Shift + I	查看项目当前文件
Alt + Shift + F7	在 Debug 模式下, 下一步, 进入当前方法体内, 如果方法体还有方法, 则会进入该内嵌的方法中, 依此循环进入
Alt + Shift + F9	弹出 Debug 的可选择菜单
Alt + Shift + F10	弹出 Run 的可选择菜单
Alt + Shift + 左键双击	选择被双击的单词 / 中文句, 按住不放, 可以同时选择其他单词 / 中文句 (必备)
Alt + Shift + 前方向键	移动光标所在行向上移动 (必备)
Alt + Shift + 后方向键	移动光标所在行向下移动 (必备)

Ctrl + Shift + Alt

快捷键	介绍
Ctrl + Shift + Alt + V	无格式黏贴 (必备)
Ctrl + Shift + Alt + N	前往指定的变量 / 方法
Ctrl + Shift + Alt + S	打开当前项目设置 (必备)
Ctrl + Shift + Alt + C	复制参考信息

其他

快捷键	介绍
F2	跳转到下一个高亮错误 或 警告位置 (必备)
F3	在查找模式下, 定位到下一个匹配处
F4	编辑源 (必备)
F7	在 Debug 模式下, 进入下一步, 如果当前行断点是一个方法, 则进入当前方法体内, 如果该方法体还有方法, 则不会进入该内嵌的方法中
F8	在 Debug 模式下, 进入下一步, 如果当前行断点是一个方法, 则不进入当前方法体内
F9	在 Debug 模式下, 恢复程序运行, 但是如果该断点下面代码还有断点则停在下一个断点上
F11	添加书签 (必备)
F12	回到前一个工具窗口 (必备)

快捷键	介绍
Tab	缩进 (必备)
ESC	从工具窗口进入代码文件窗口 (必备)
连接两次	
Shift	弹出 Search Everywhere 弹出层