IntelliJ IDEA 的安装、配置与使用-简化版

尚硅谷 Java 研究院-宋红康
www.atguigu.com

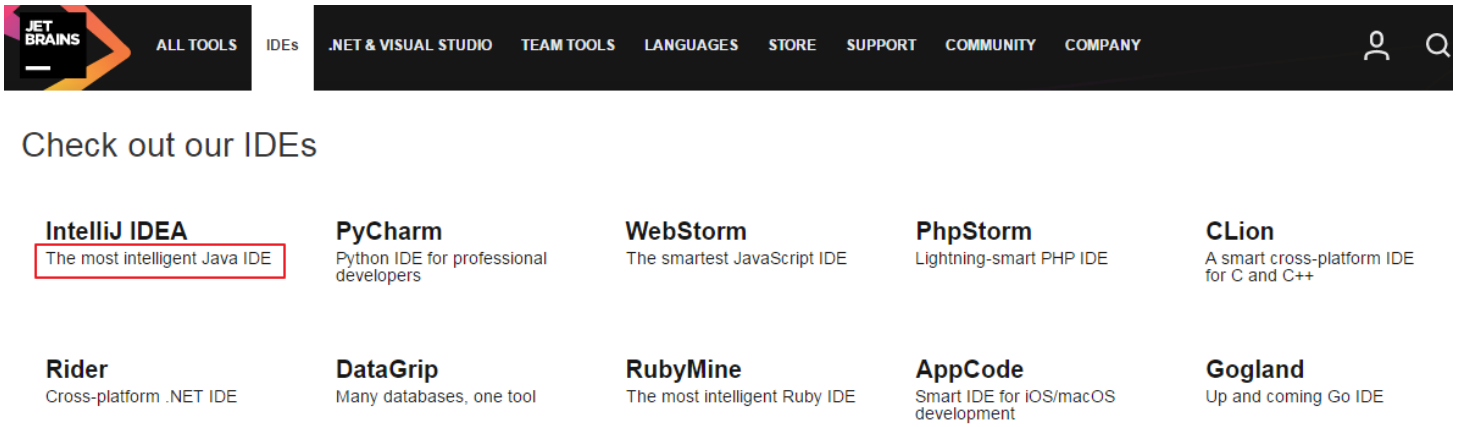
一、IntelliJ IDEA 介绍

1.JetBrains 公司介绍

IDEA(https://www.jetbrains.com/idea/)是 JetBrains 公司的产品，公司旗下还有其它产品，比如：

- WebStorm：用于开发 JavaScript、HTML5、CSS3 等前端技术；
- PyCharm：用于开发 python
- PhpStorm：用于开发 PHP
- RubyMine：用于开发 Ruby/Rails
- AppCode：用于开发 Objective - C/Swift
- CLion：用于开发 C/C++
- DataGrip：用于开发数据库和 SQL
- Rider：用于开发.NET
- GoLand：用于开发 Go

- Android Studio：用于开发 android(google 基于 IDEA 社区版进行迭代)



2. IntelliJ IDEA 介绍

IDEA，全称 IntelliJ IDEA，是 Java 语言的集成开发环境，IDEA 在业界被公认为是最好的 java 开发工具之一，尤其在智能代码助手、代码自动提示、重构、J2EE 支持、Ant、JUnit、CVS 整合、代码审查、创新的 GUI 设计等方面的功能可以说是超常的。

IntelliJ IDEA 在 2015 年的官网上这样介绍自己：

Excel at enterprise, mobile and web development with Java, Scala and Groovy, with all the latest modern technologies and frameworks available out of the box.

简明翻译：IntelliJ IDEA 主要用于支持 Java、Scala、Groovy 等语言的开发工具，同时具备支持目前主流的技术和框架，擅长于企业应用、移动应用和 Web 应用的开发。

3.IDEA 的主要功能介绍

语言支持上：

安装插件后支持	SQL类	基本JVM
PHP	PostgreSQL	Java
Python	MySQL	Groovy
Ruby	Oracle	
Scala	SQL Server	
Kotlin		
Clojure		

其他支持：

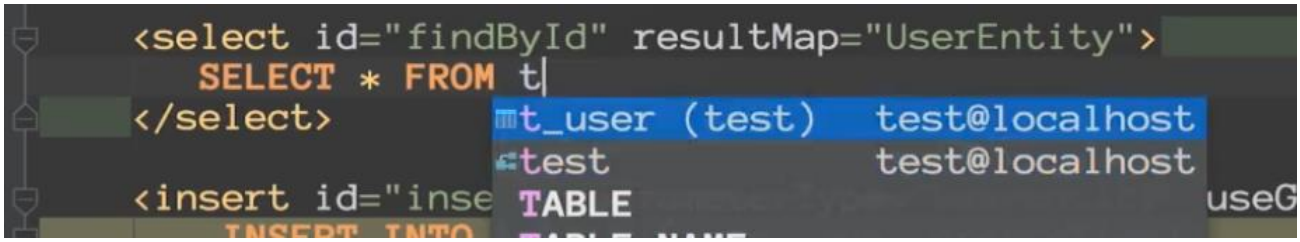
支持的框架	额外支持的语言代码提示	支持的容器
Spring MVC	HTML5	Tomcat
GWT	CSS3	TomEE
Vaadin	SASS	WebLogin
Play	LESS	JBoss
Grails	JavaScript	Jetty
Web Services	CoffeeScript	WebSphere
JSF	Node.js	
Struts	ActionScript	
Hibernate		
Flex		

4.IDEA 的主要优势：(相较于 Eclipse 而言)

① 强大的整合能力。比如：Git、Maven、Spring 等

② 提示功能的快速、便捷

③ 提示功能的范围广



④ 好用的快捷键和代码模板 private static final psf

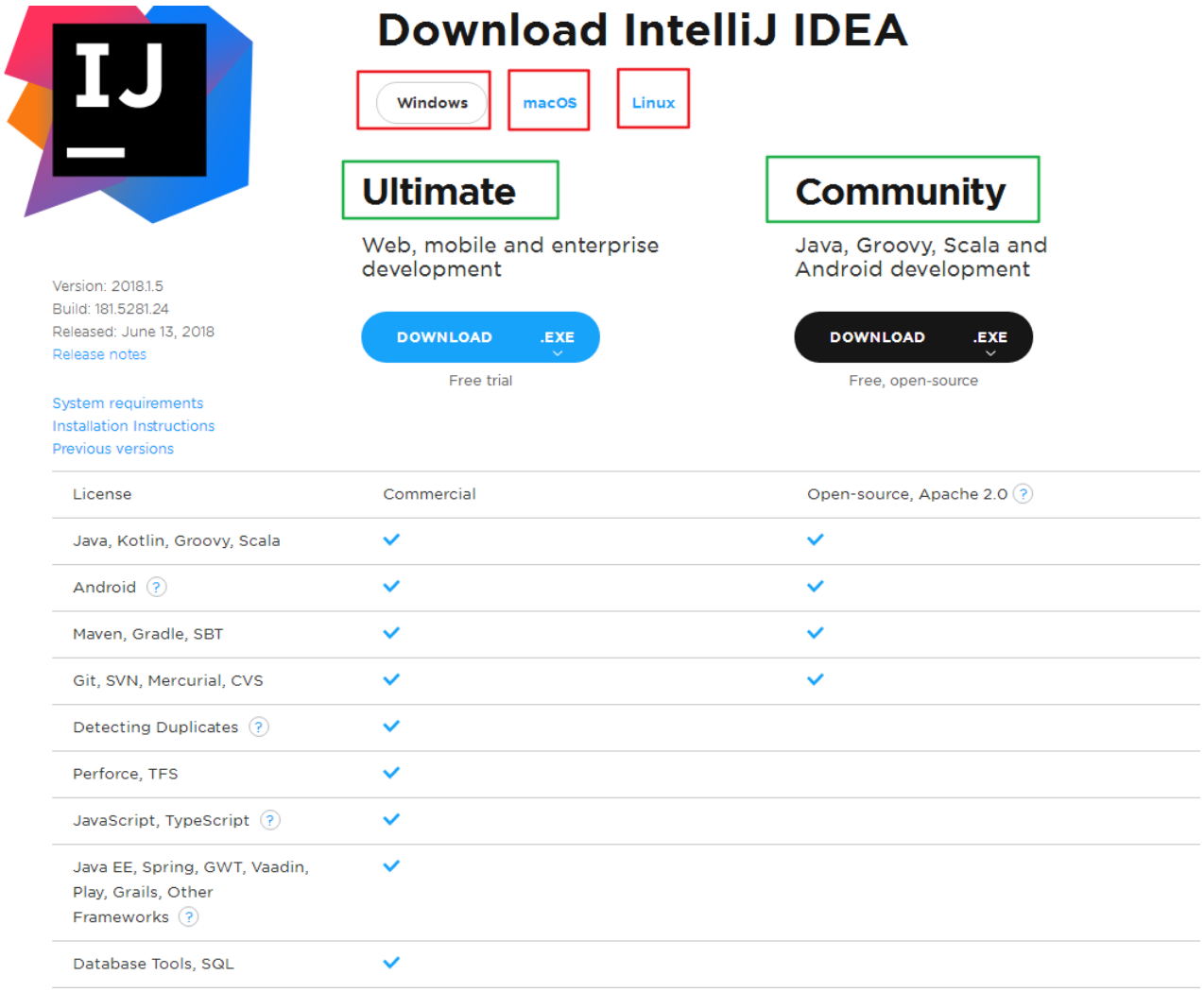
⑤ 精准搜索

5.IDEA 的下载地址：(官网)

https://www.jetbrains.com/idea/download/#section=windows

IDEA 分为两个版本：旗舰版(Ultimate)和社区版(Community)。

旗舰版收费(限 30 天免费试用)，社区版免费，这和 Eclipse 有很大区别。



这里提供了不同操作系统下的两个不同版本的安装文件。

两个不同版本的详细对比，可以参照官网：

https://www.jetbrains.com/idea/features/editions_comparison_matrix.html

6. 官网提供的详细使用文档：

https://www.jetbrains.com/help/idea/meet-intellij-idea.html

二、windows 下安装过程

1. 安装前的准备

1.1 硬件要求(Hardware requirements)

内存：2 GB RAM minimum, 4 GB RAM recommended

硬盘：1.5 GB hard disk space + at least 1 GB for caches

屏幕：1024x768 minimum screen resolution

个人建议配置：内存 8G 或以上，CPU 最好 i5 以上，最好安装块固态硬盘(SSD)，将 IDEA 安装在固态硬盘上，这样流畅度会加快很多。

1.2 软件要求(Software requirements)

操作系统： Microsoft Windows 10/8/7/Vista/2003/XP (32 or 64 bit)

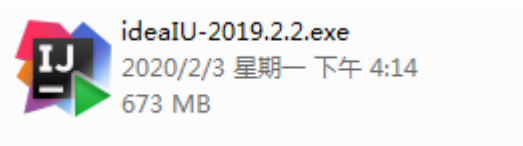
软件环境：

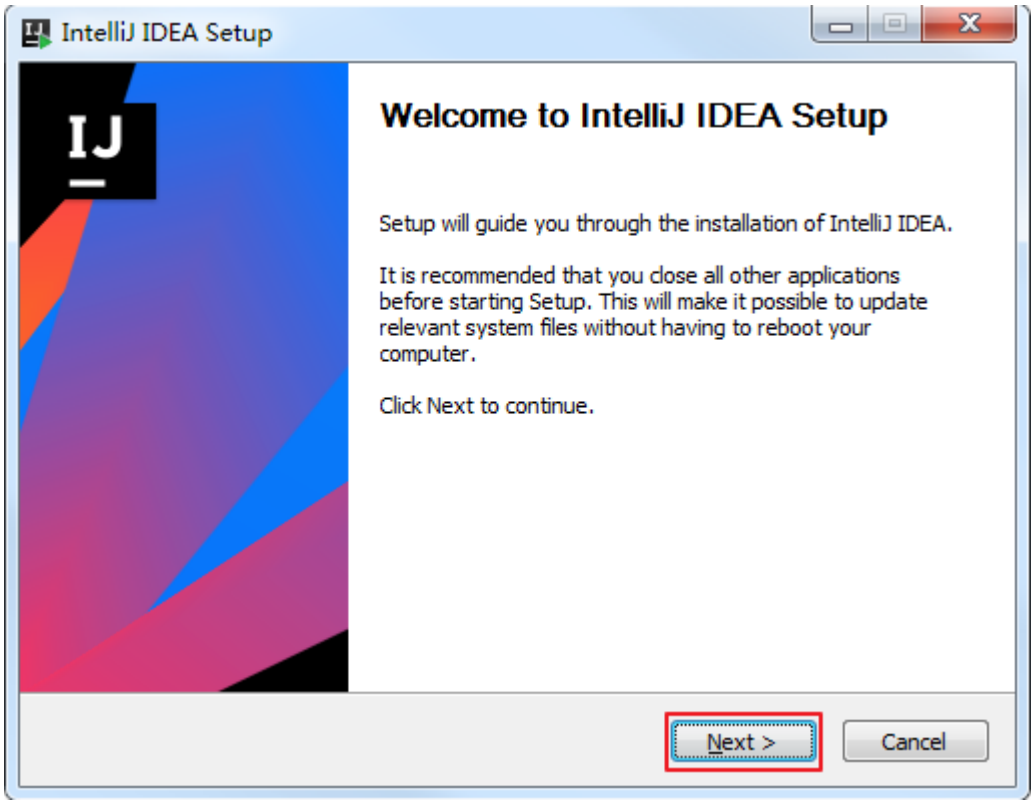
- JRE 1.8 is bundled with the IntelliJ IDEA distribution. You do not need to install Java on your computer to run IntelliJ IDEA.
- A standalone JDK is required for Java development.

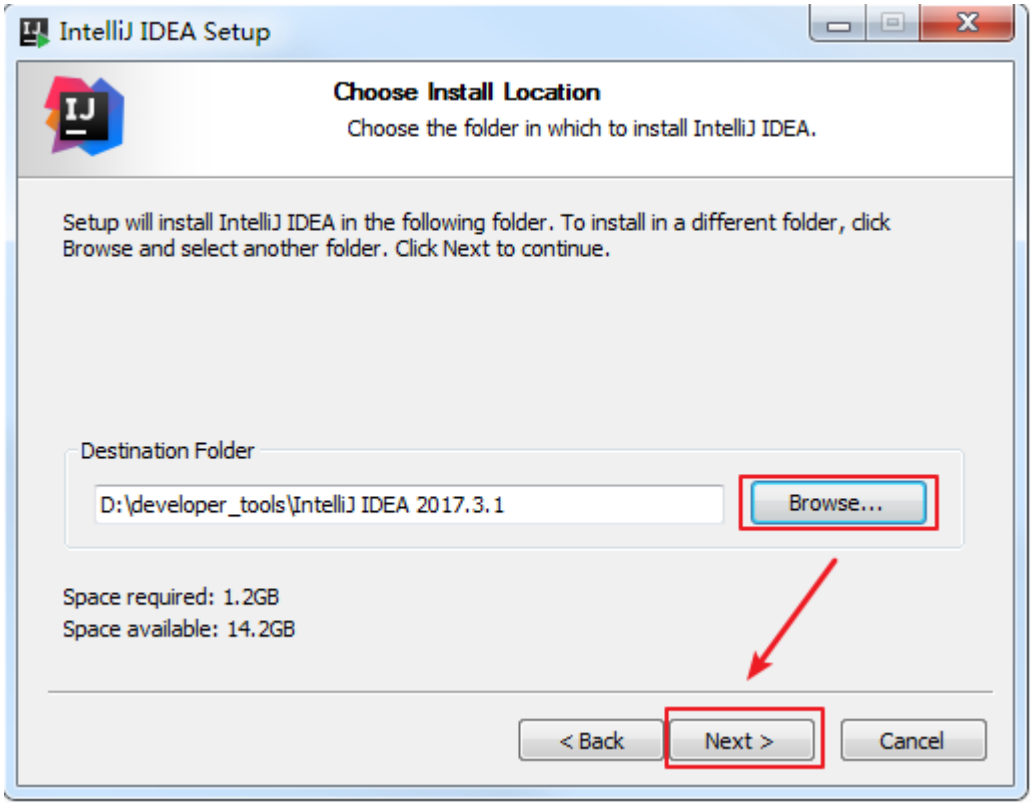
注意：这里如果没有安装 JDK 的话，请参考提供的文档《尚硅谷_宋红康_JDK8 的下载_安装_配置.pdf》进行安装配置。

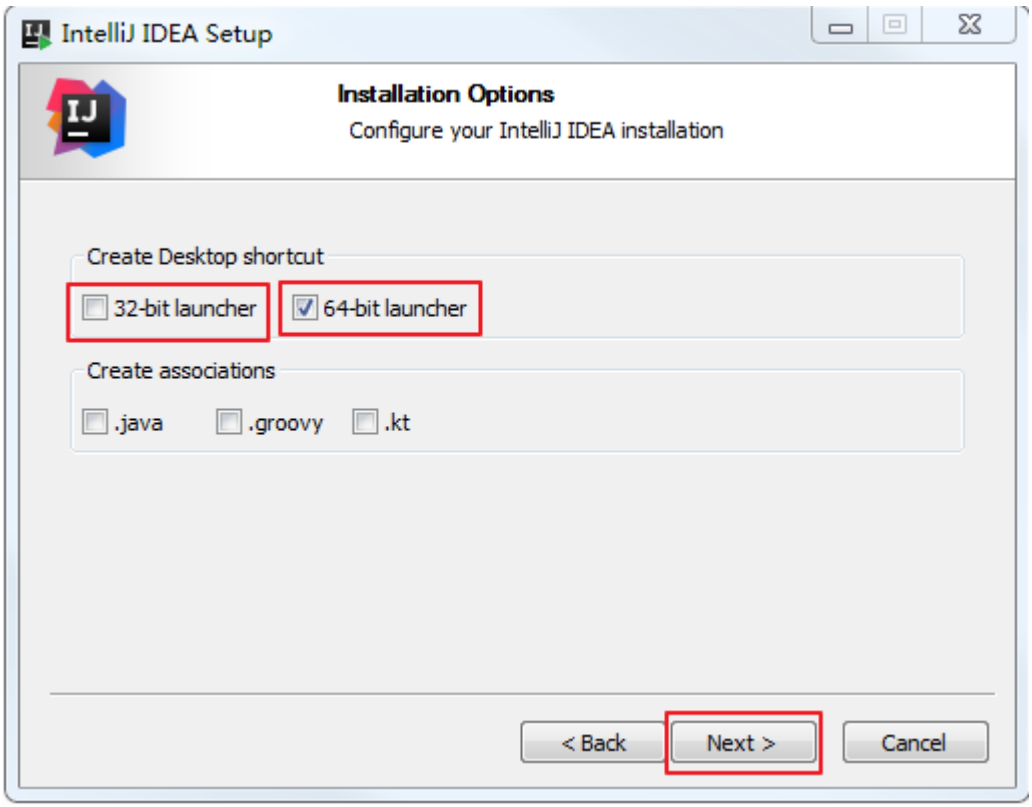
2.具体安装过程

双击：

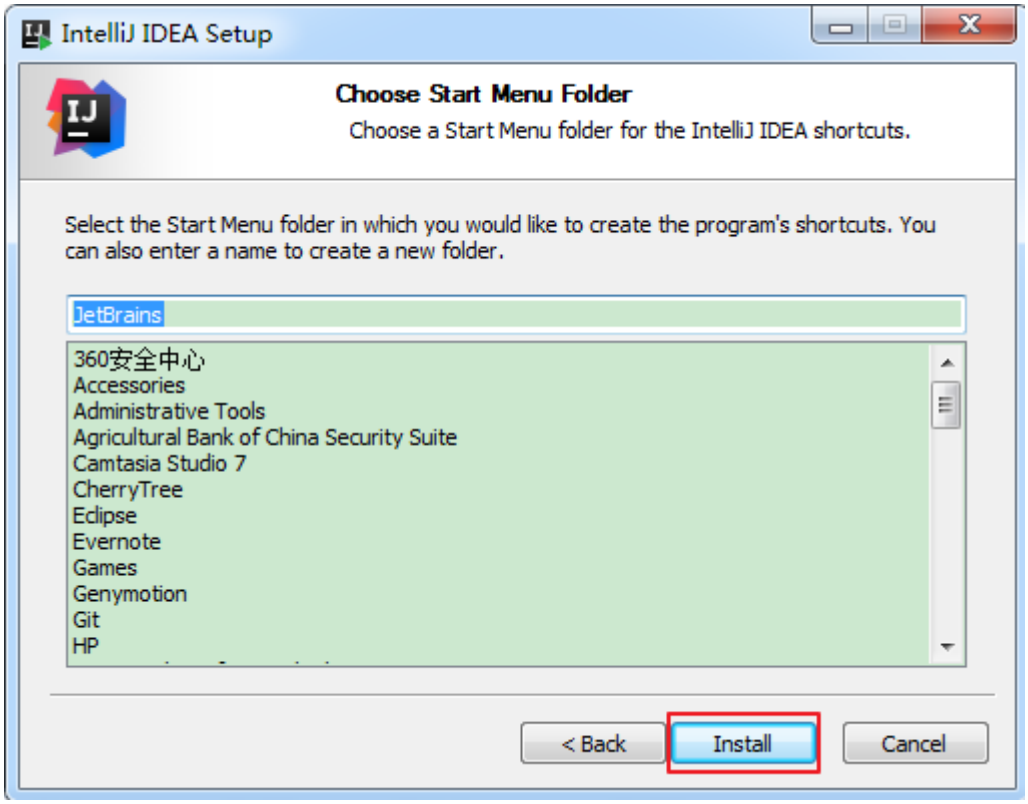


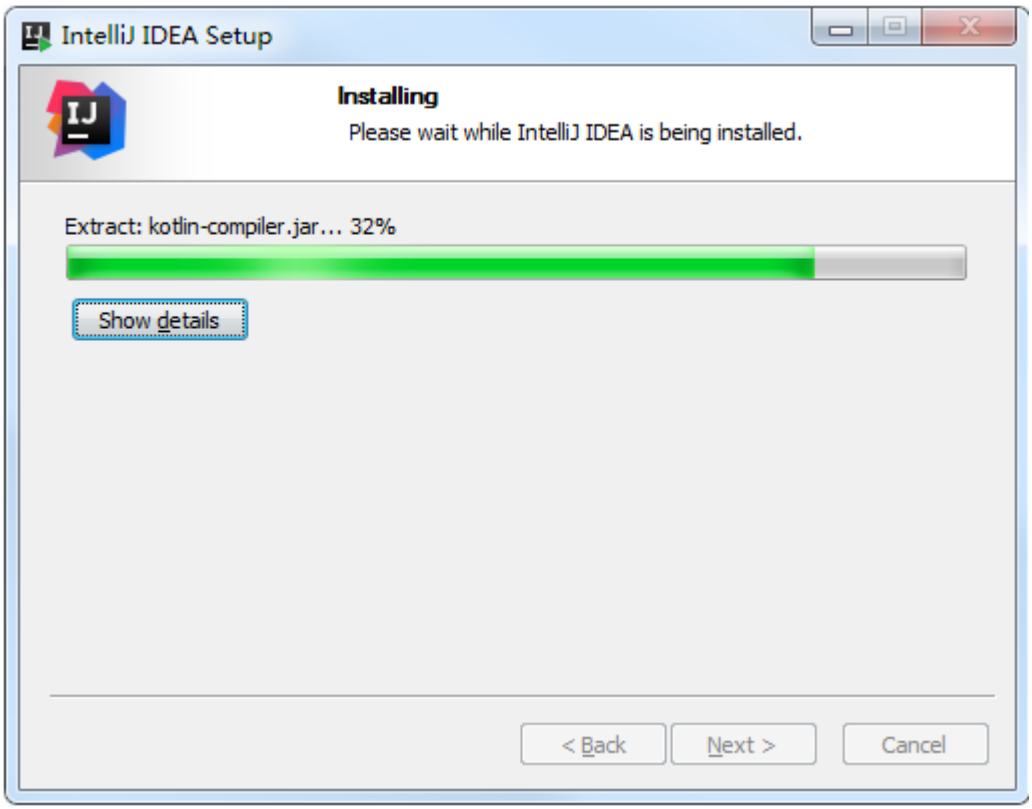

IntelliJ IDEA Setup
IJ
Welcome to IntelliJ IDEA Setup
Setup will guide you through the installation of IntelliJ IDEA.
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.
Click Next to continue.
Next >
Cancel

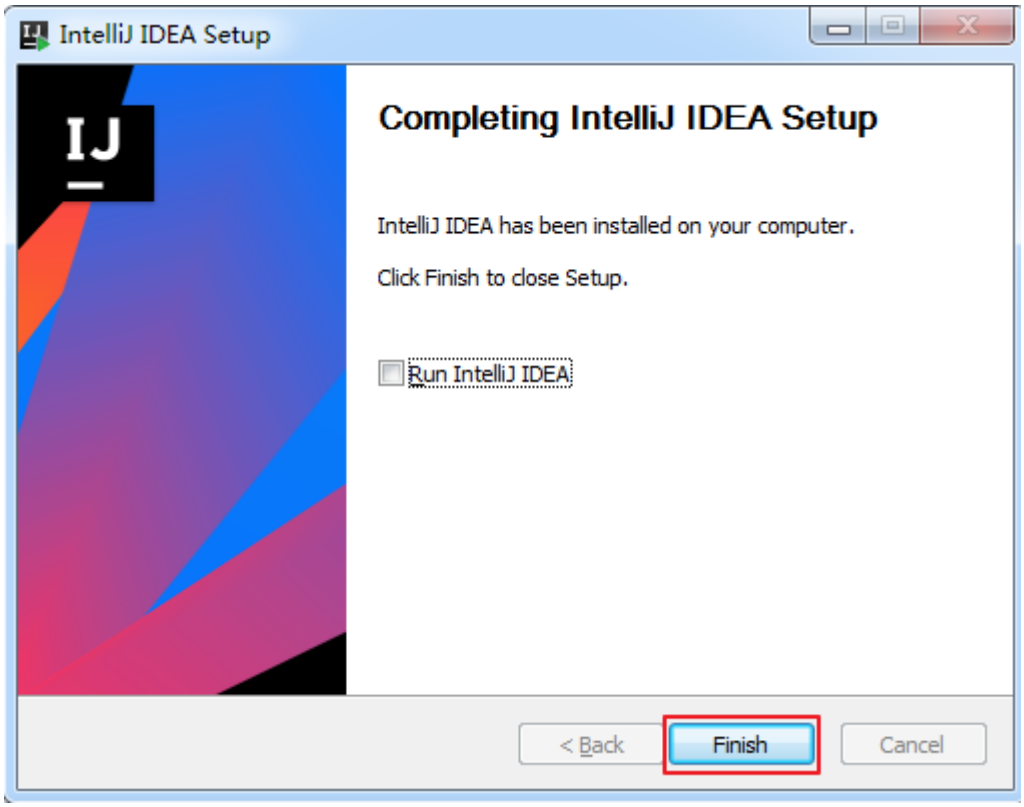

IntelliJ IDEA Setup
Choose Install Location
Choose the folder in which to install IntelliJ IDEA.
Setup will install IntelliJ IDEA in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.
Destination Folder
D:\developer_tools\IntelliJ IDEA 2017.3.1
Browse...
Space required: 1.2GB
Space available: 14.2GB
< Back
Next >
Cancel



- 确认 32 位版还是 64 位版
- 确认是否与.java、.groovy、.kt 格式文件进行关联，这里也可以选择不关联。



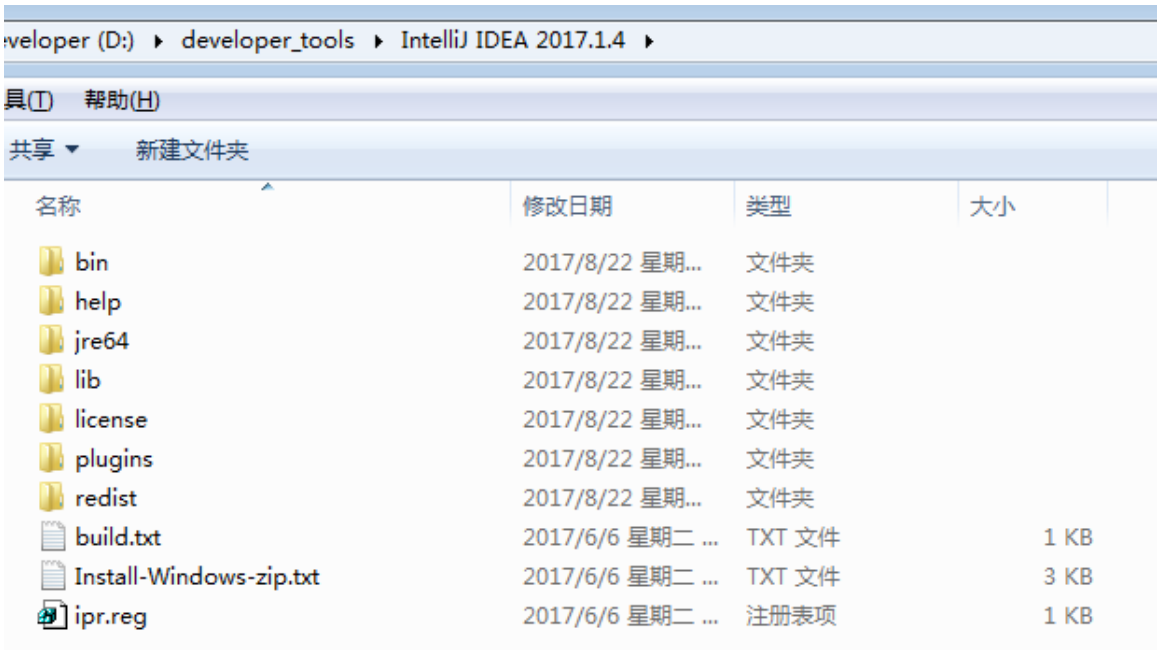

IntelliJ IDEA Setup
Installing
Please wait while IntelliJ IDEA is being installed.
Extract: kotlin-compiler.jar... 32%
Show details
< Back
Next >
Cancel


IntelliJ IDEA Setup
Completing IntelliJ IDEA Setup
IntelliJ IDEA has been installed on your computer.
Click Finish to close Setup.
Run IntelliJ IDEA
< Back
Finish
Cancel

3.安装总结

从安装上来看，IntelliJ IDEA 对硬件的要求似乎不是很高。可是实际在开发中其实并不是这样的，因为 IntelliJ IDEA 执行时会有大量的缓存、索引文件，所以如果你正在使用 Eclipse / MyEclipse，想通过 IntelliJ IDEA 来解决计算机的卡、慢等问题，这基本上是不可能的，本质上你应该对自己的硬件设备进行升级。

4.查看安装目录结构



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

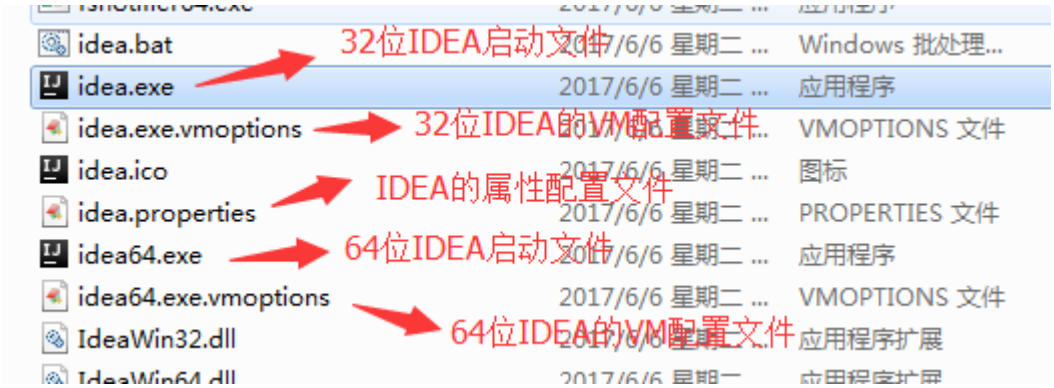
jre64：64 位java 运行环境

lib：idea 依赖的类库

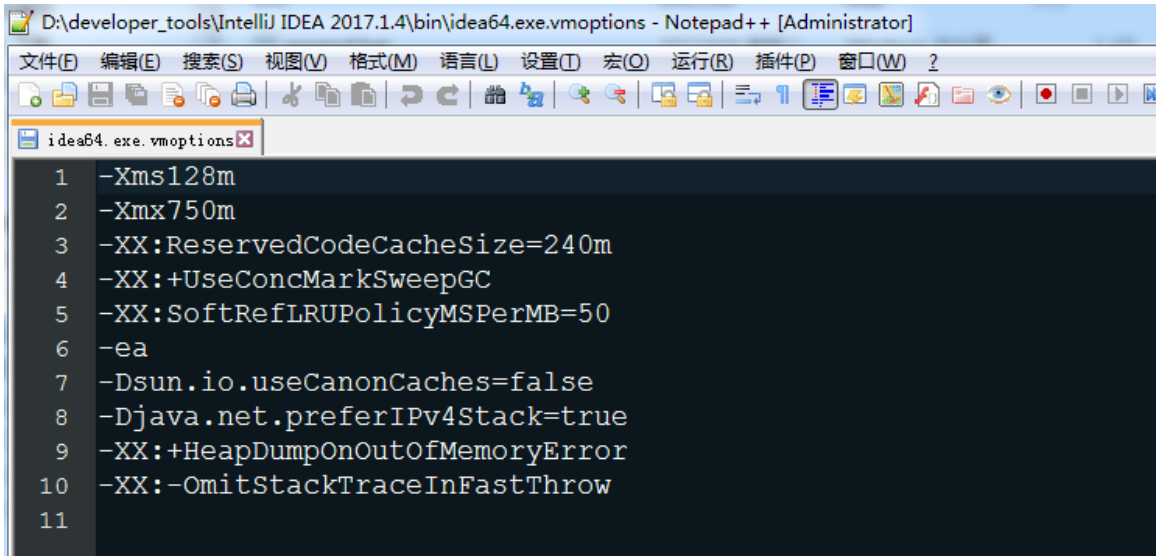
license：各个插件许可

plugin：插件

其中：bin 目录下：



这里以我的电脑系统(64 位 windows7，16G 内存)为例，说明一下如何调整 VM 配置文件：



1. 大家根据电脑系统的位数，选择 32 位的 VM 配置文件或者 64 位的 VM 配置文件
2. 32 位操作系统内存不会超过 4G，所以没有多大空间可以调整，建议不用调整了
3. 64 位操作系统中 8G 内存以下的机子或是静态页面开发者是无需修改的。
4. 64 位操作系统且内存大于 8G 的，如果你是开发大型项目、Java 项目或是 Android 项目，建议进行修改，常修改的就是下面 3 个参数：

`-Xms128m`，16 G 内存的机器可尝试设置为 `-Xms512m`
(设置初始的内存数，增加该值可以提高 Java 程序的启动速度。)

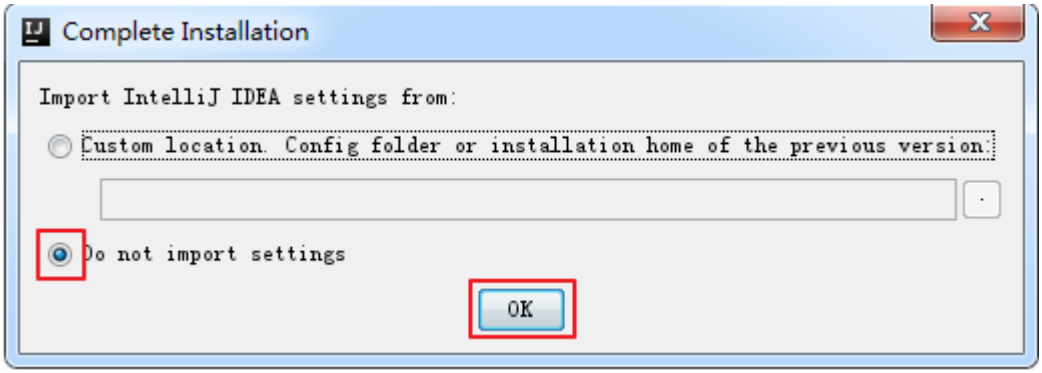
`-Xmx750m`，16 G 内存的机器可尝试设置为 `-Xmx1500m`
(设置最大内存数，提高该值，可以减少内存 Garage 收集的频率，提高程序性能)

`-XX:ReservedCodeCacheSize=240m`，16G 内存的机器可尝试设置为 `-XX:ReservedCodeCacheSize=500m`
(保留代码占用的内存容量)

三、启动应用后简单配置

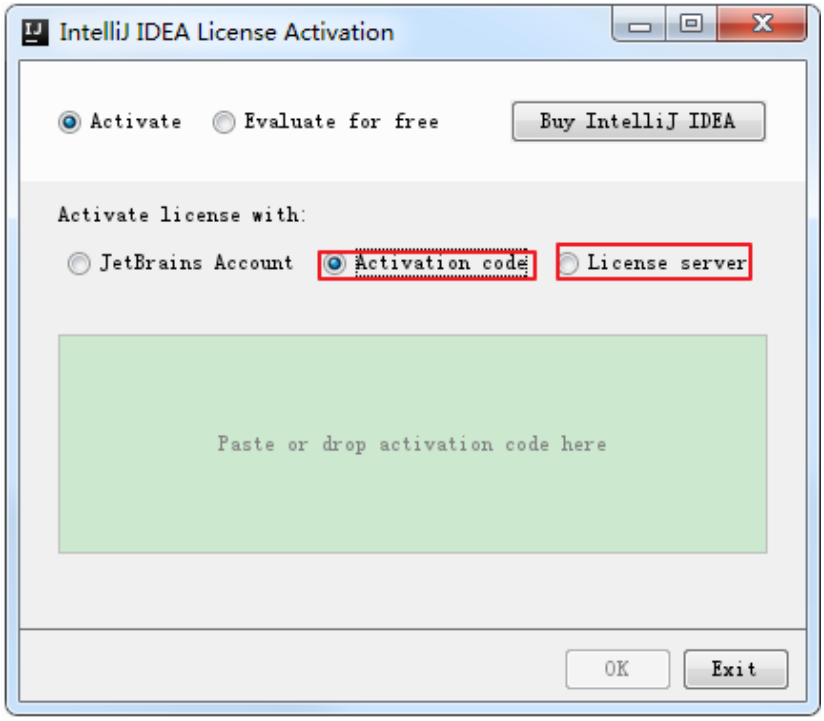
1.是否导入已有设置

首次启动，会弹出如下的对话框。选择不导入已有的设置。



2.激活

然后根据提供的激活文档《IDEA2017-2018_激活方法》或百度：idea 破解码，填入：lisence server 的具体值即可。（需要联网）或者 选择 Activation code，根据文档提供的激活码，同样可以激活。（不需要联网）

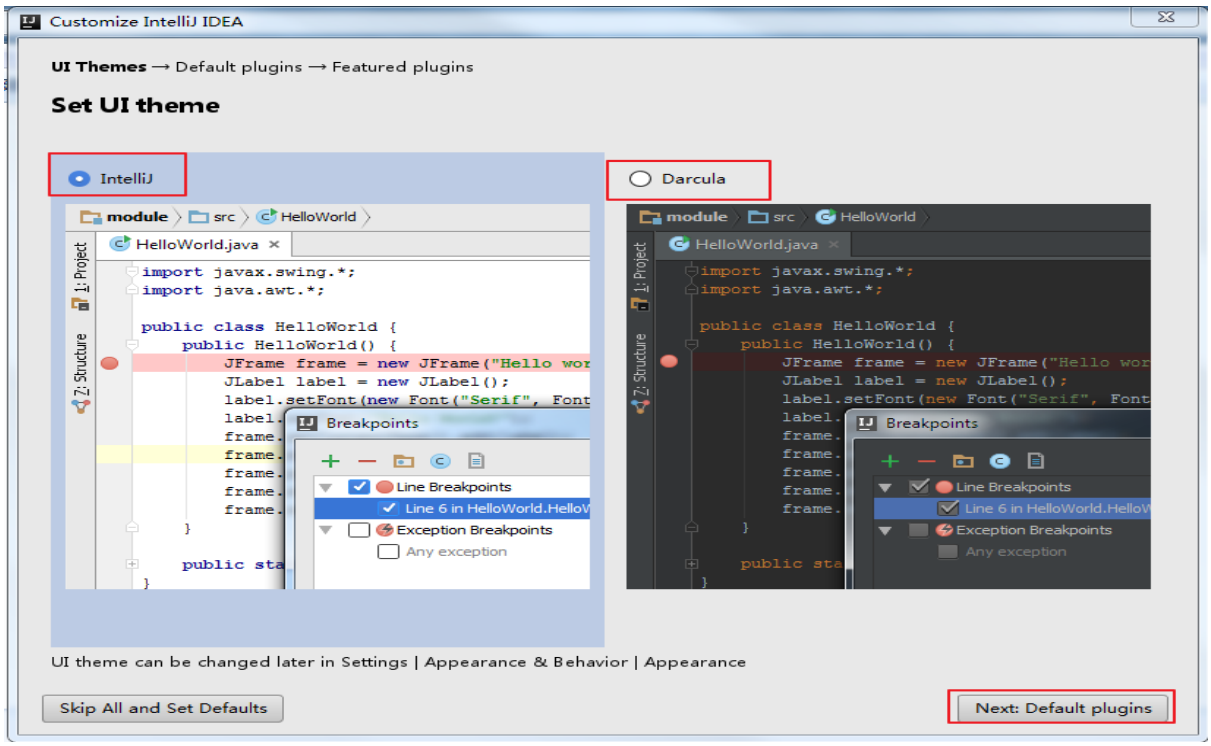


补充：

对于 IDEA 2019.2 月版本，需要按照如下的方式激活：

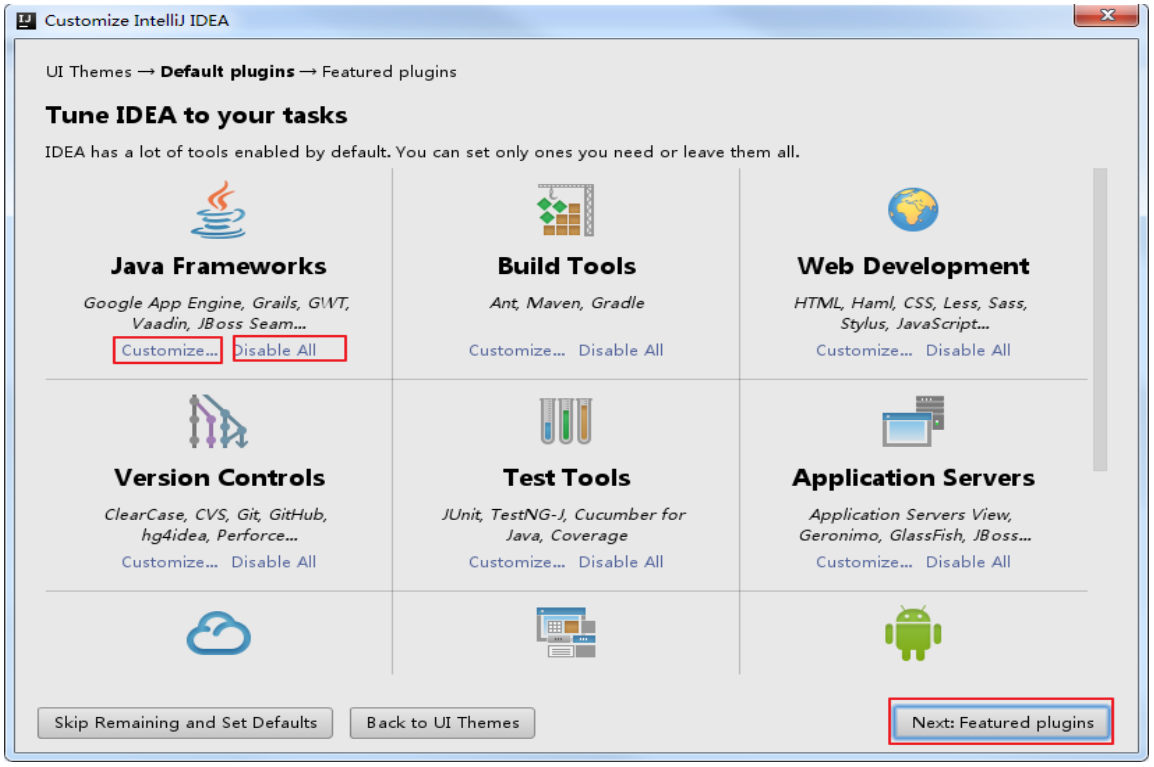
《参见 2019.2 注册文档》

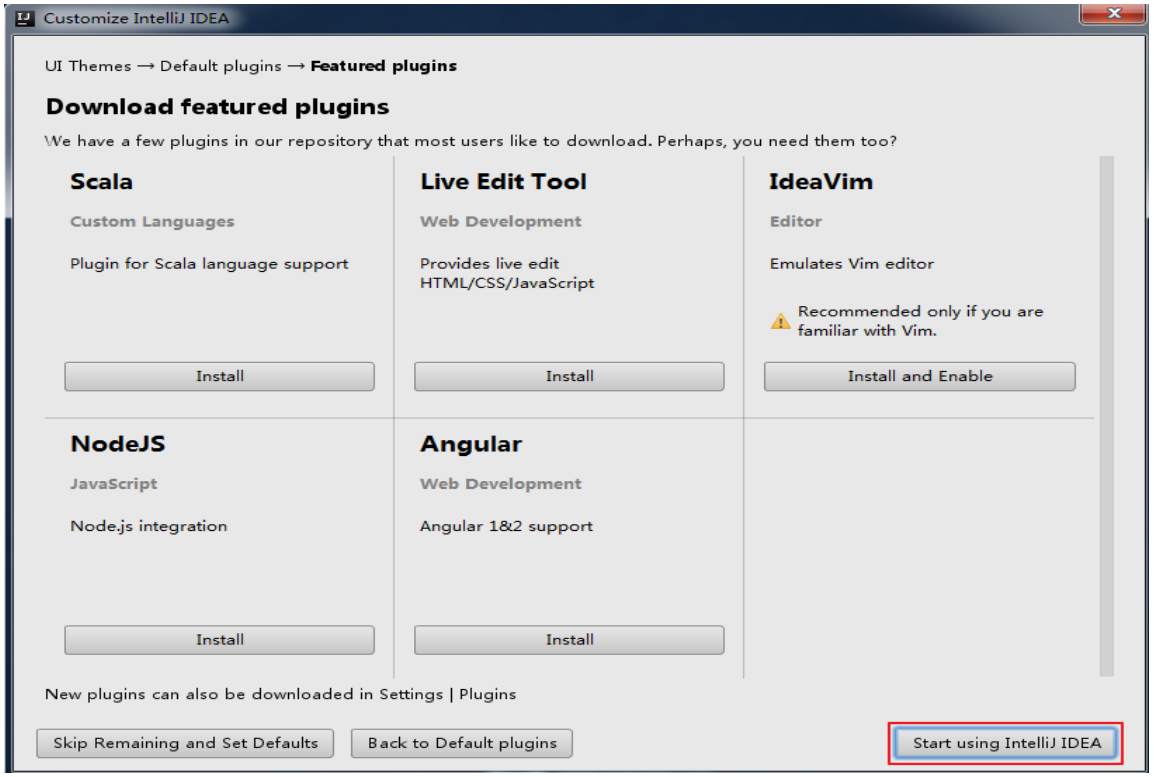
3.设置主题



这里根据个人喜好，进行选择，也可以选择跳过(skip all and set defaults)。后面在 settings 里也可以再设置主题等。这里选择：Next:Default plugins

4.设置插件





设置 IDEA 中的各种插件，可以选择自定义设置、删除，或者安装本身不存在的插件（比如：支持 Scala 的插件）。这里不设置，后面也可以通过界面菜单栏的

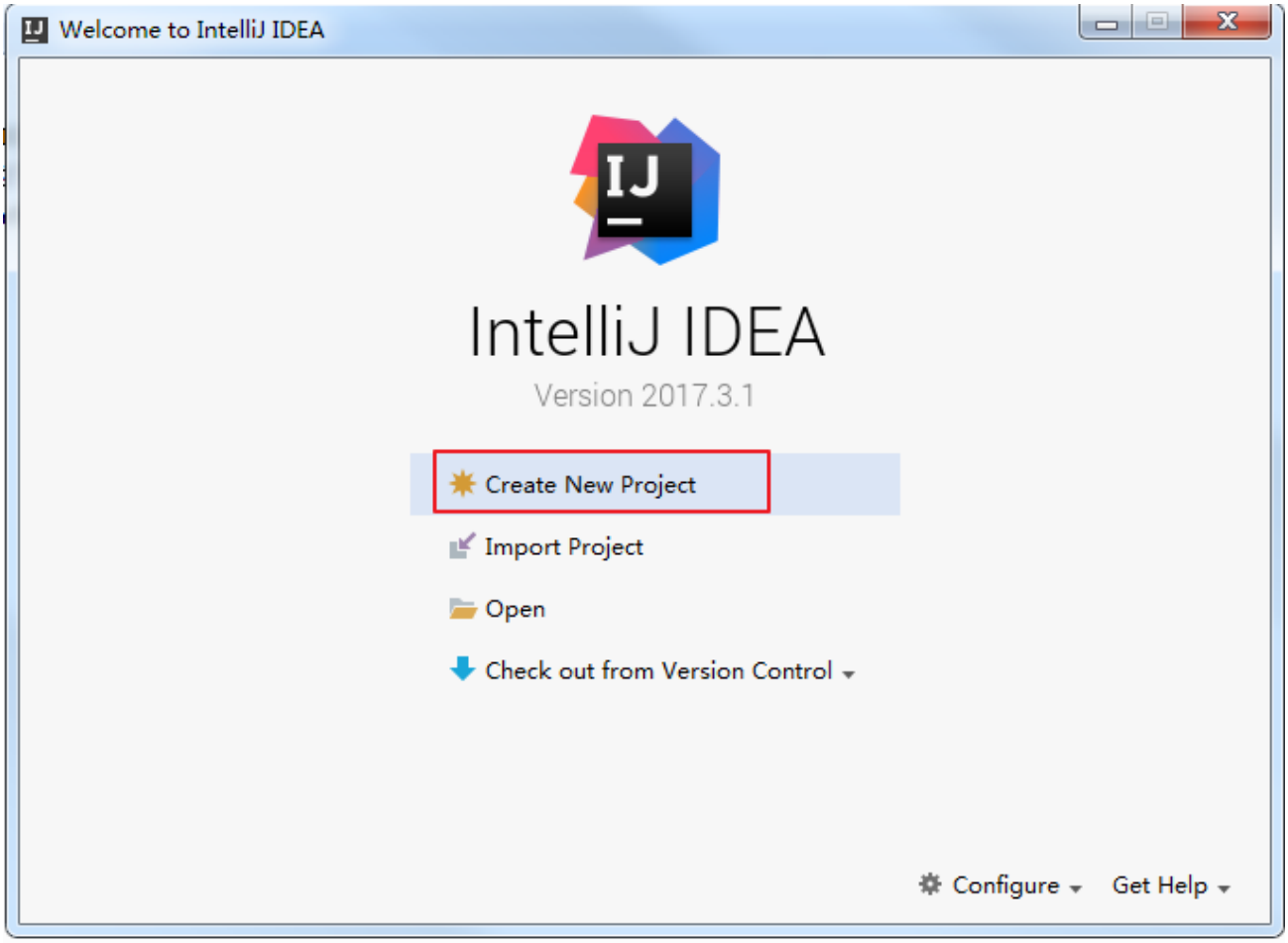
settings 进行设置。

IDEA 插件官方下载地址：https://plugins.jetbrains.com/idea

5.启动页面

四、创建 Java 工程，运行 HelloWorld

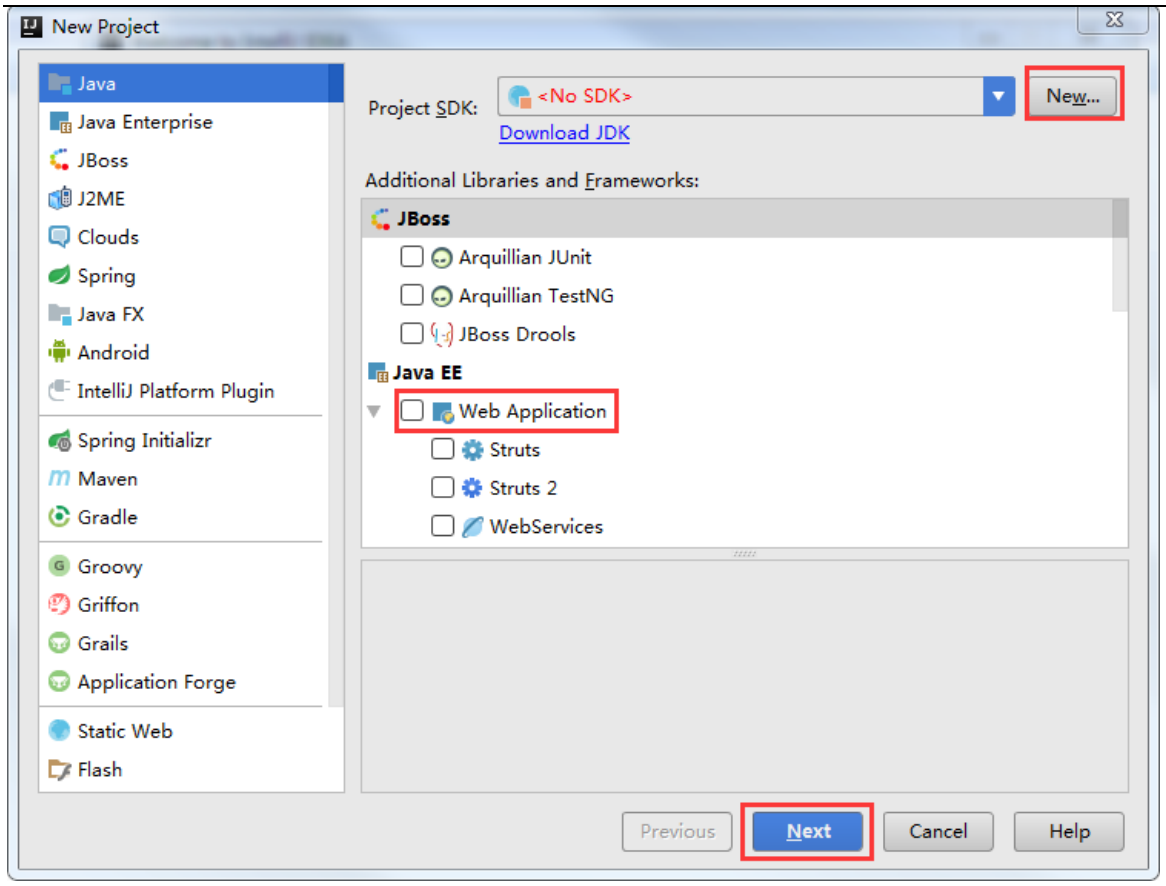
1.创建 Java 工程



- Create New Project:创建一个新的工程
- Import Project:导入一个现有的工程
- Open:打开一个已有工程。比如：可以打开 Eclipse 项目。
- Check out from Version Control:可以通过服务器上的项目地址 check out Github 上面项目或其他 Git 托管服务器上的项目

这里选择 Create New Project，需要明确一下概念：

IntelliJ IDEA 没有类似 Eclipse 的工作空间的概念（Workspaces），最大单元就是 Project。这里可以把 Project 理解为 Eclipse 中的 Workspace。

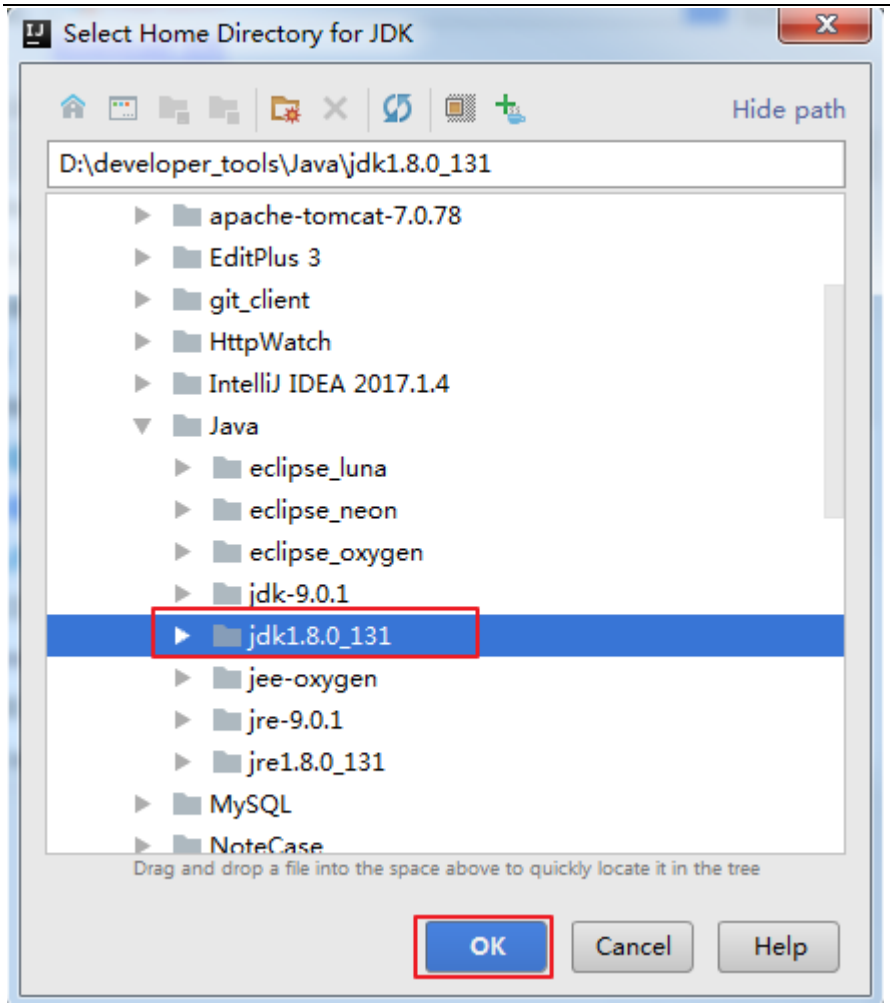


选择指定目录下的 JDK 作为 Project SDK。

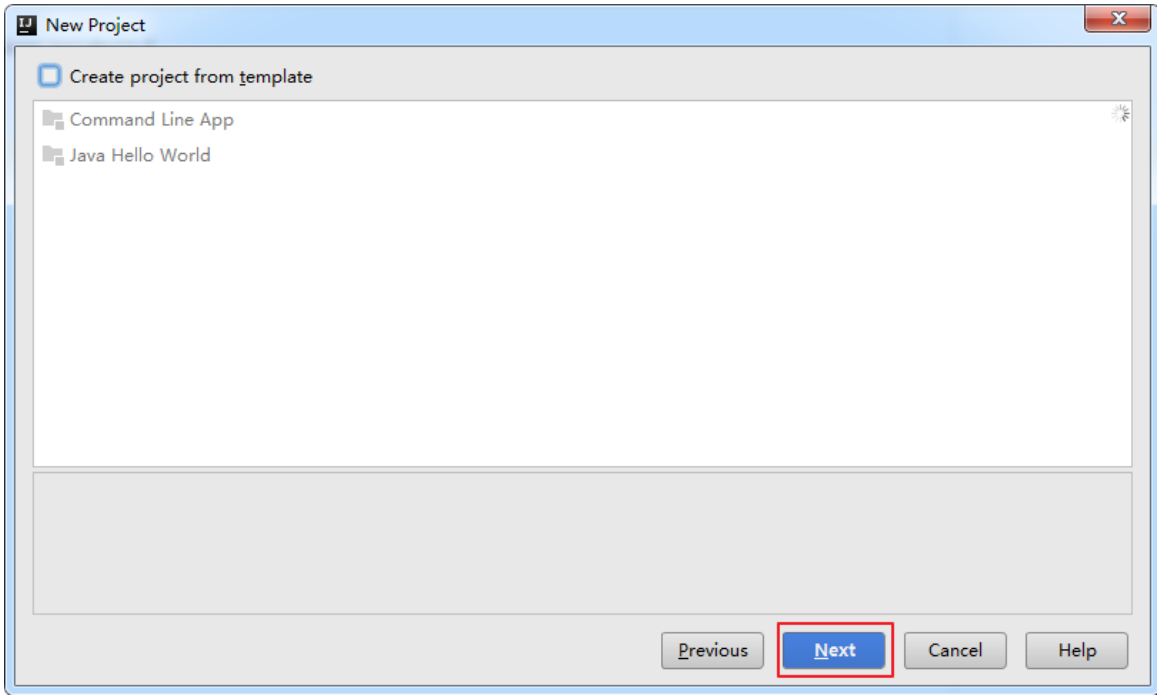
如果要创建 Web 工程，则需要勾选上面的 Web Application。如果不需要创建 Web 工程的话，则不需要勾选。这里先不勾选，只是创建简单的 Java 工程。

其中，选择 New：

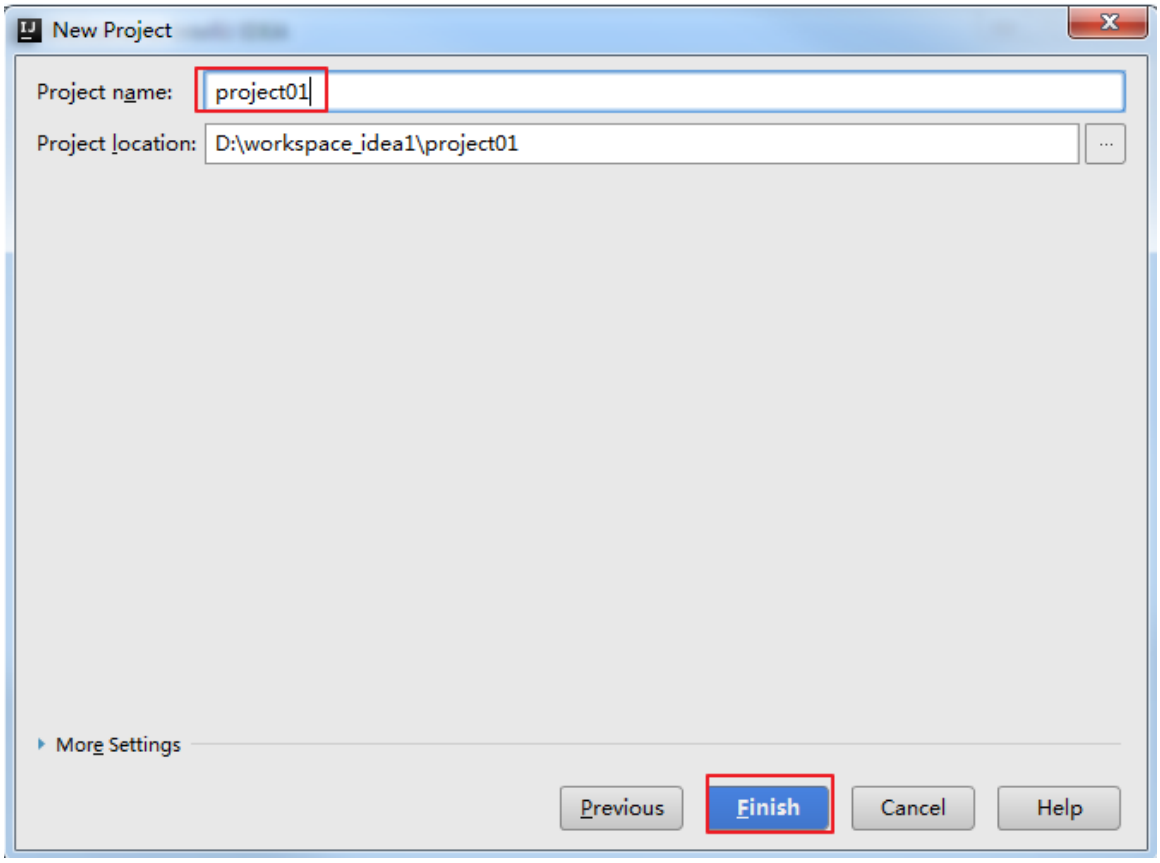
选择 jdk 的安装路径所在位置：



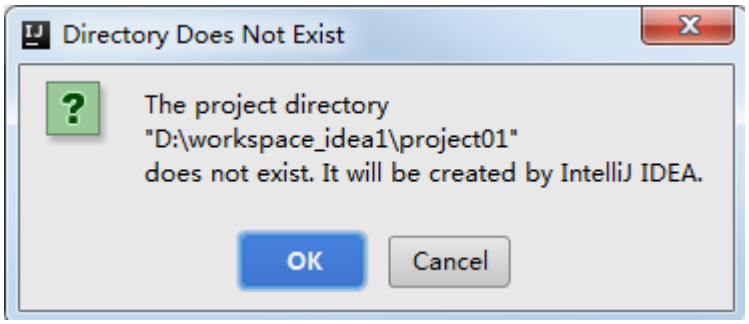
点击 OK 以后，选择 Next:



这里不用勾选。选择 Next，进入下一个页面：

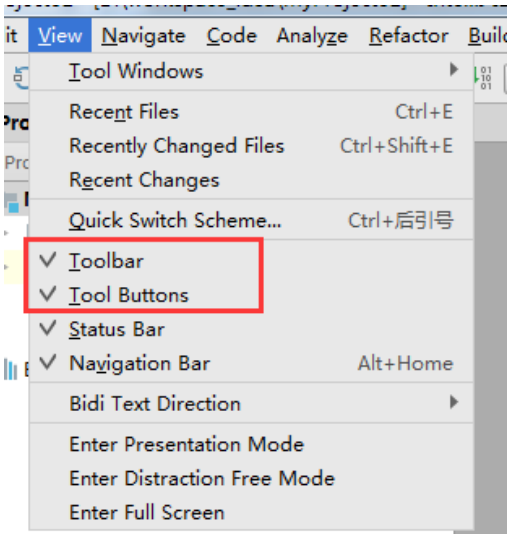


给创建的工程起一个名字，点击 finish。



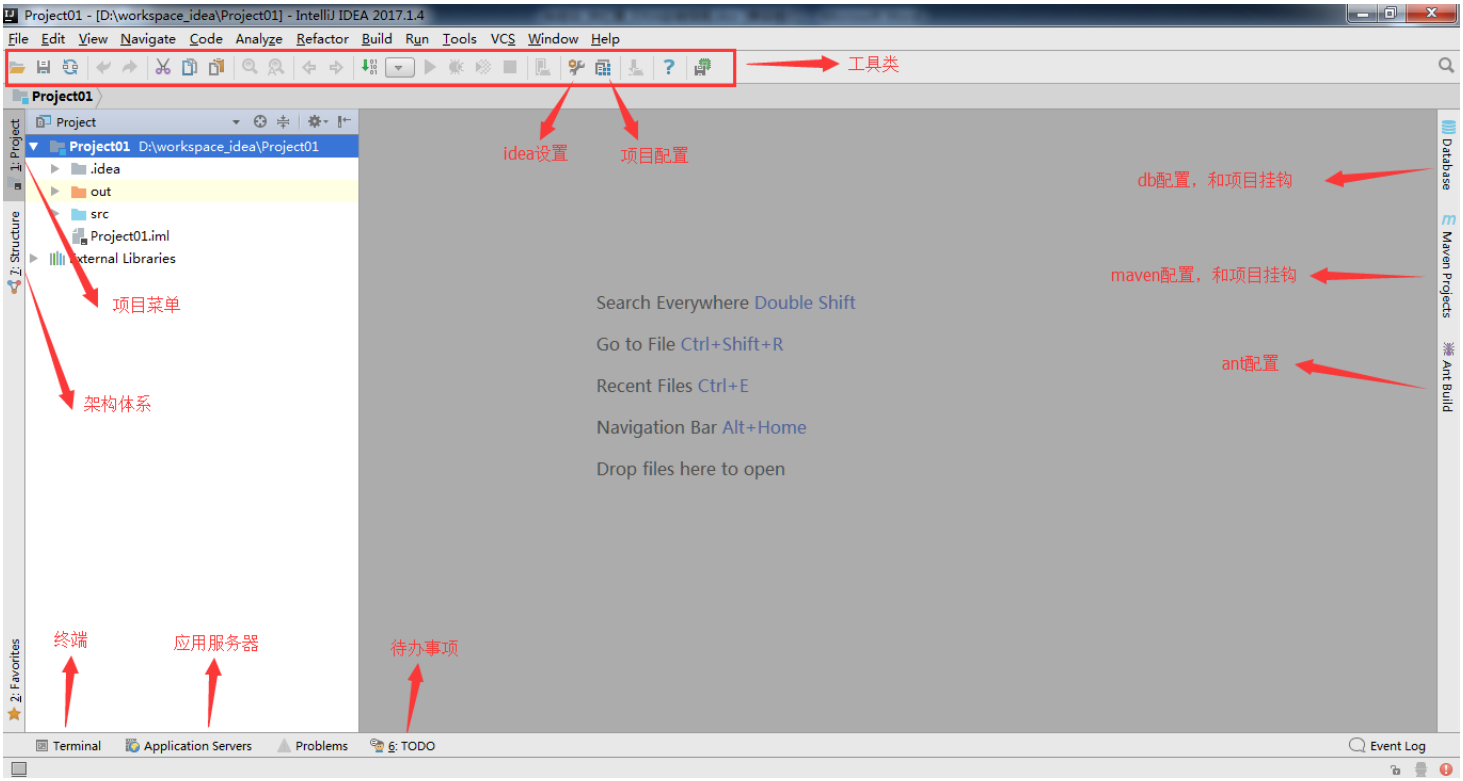
点击 OK 即可。

2.设置显示常见的视图



调出工具条和按钮组

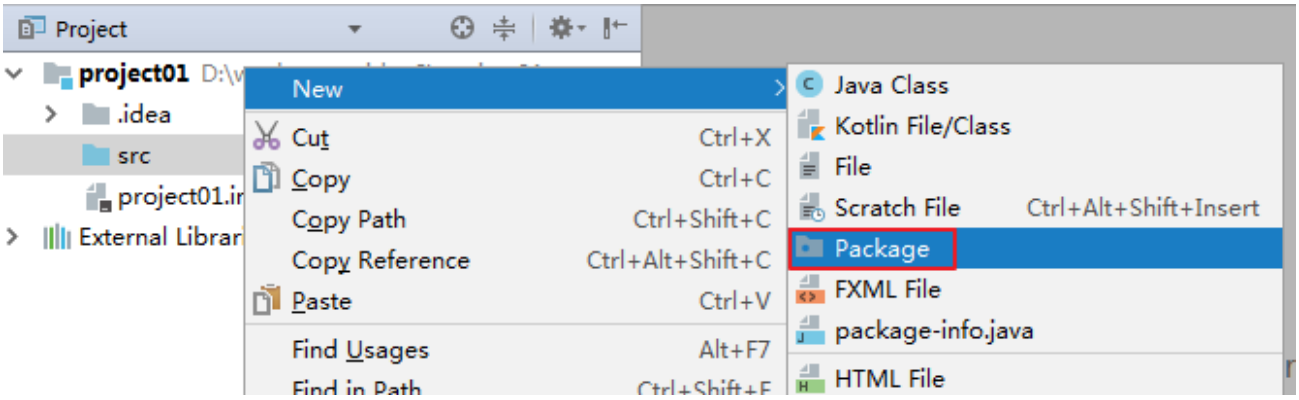
3.工程界面展示

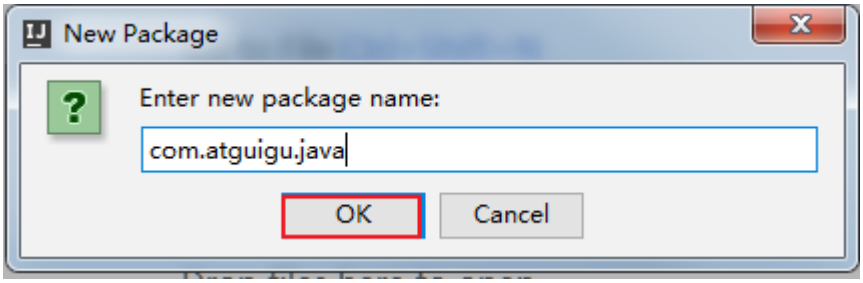


- 工程下的 src 类似于 Eclipse 下的 src 目录，用于存放代码。
- 工程下的.idea 和 project01.iml 文件都是 IDEA 工程特有的。类似于 Eclipse 工程下的.settings、.classpath、.project 等。

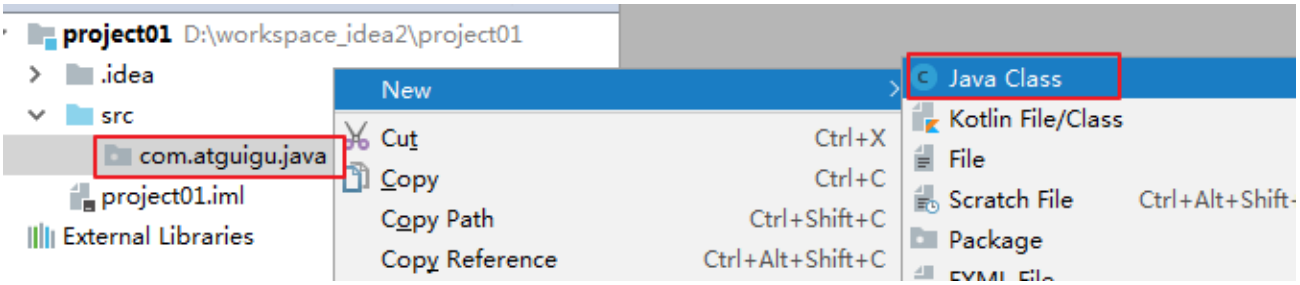
4.创建 package 和 class

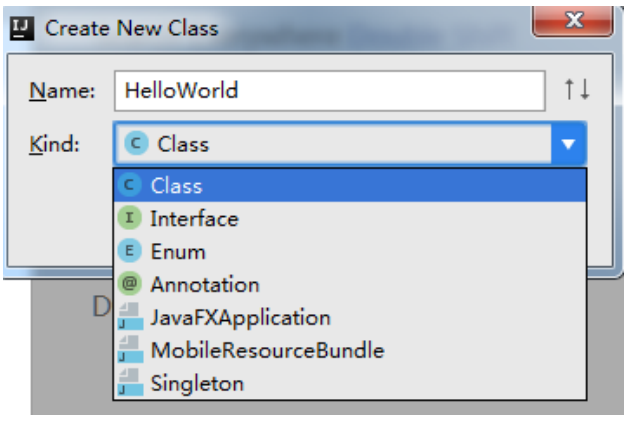
接着在 src 目录下创建一个 package：





在包下 new-class：

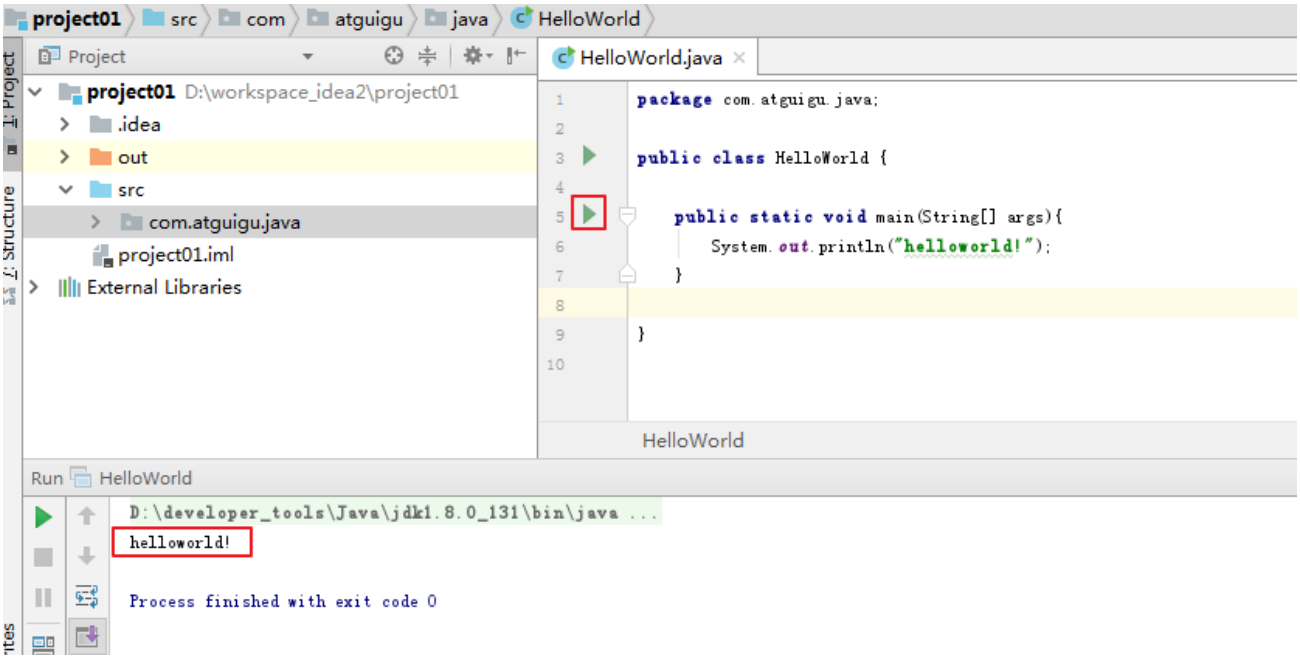




不管是创建 class，还是 interface，还是 annotation，都是选择 new – java class，

然后在下拉框中选择创建的结构的类型。

接着在类 HelloWorld 里声明主方法，输出 helloworld，完成测试。



说明：在 IDEA 里要说的是，写完代码，不用点击保存。IDEA 会自动保存代码。

5.创建模块(Module)

1. 在 Eclipse 中我们有 Workspace（工作空间）和 Project（工程）的概念，在 IDEA 中只有 Project（工程）和 Module（模块）的概念。这里的对应关系为：

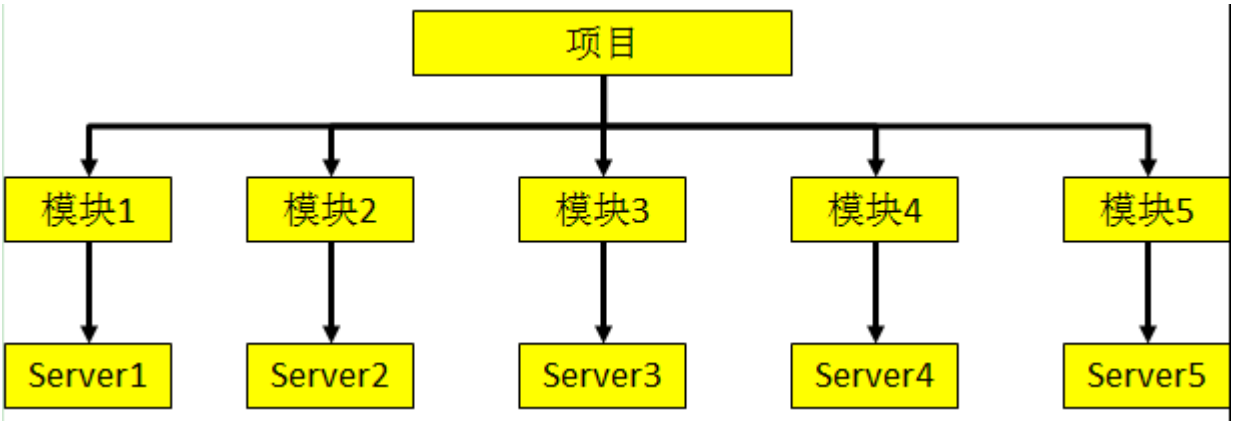
IDEA 官网说明： **An Eclipse workspace is similar to a project in IntelliJ IDEA** **An Eclipse project maps to a module in IntelliJ IDEA**
翻译： **Eclipse 中 workspace 相当于 IDEA 中的 Project** **Eclipse 中 Project 相当于 IDEA 中的 Module**

这个地方刚开始用的时候会很容易理不清它们之间的关系。

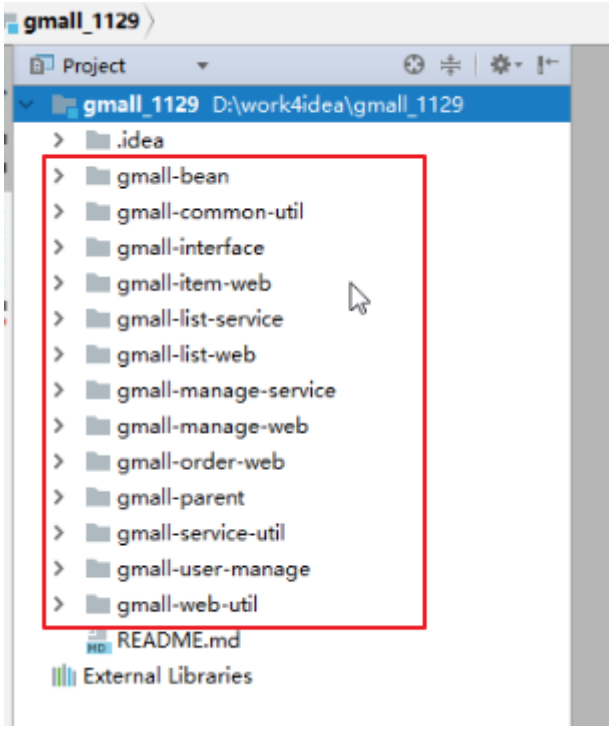
2. 从 Eclipse 转过来的人总是下意识地要在同一个窗口管理 n 个项目，这在 IntelliJ IDEA 是无法做到的。IntelliJ IDEA 提供的解决方案是打开多个项目实例，即打开多个项目窗口。即：一个 Project 打开一个 Window 窗口。

3. 在 IntelliJ IDEA 中 Project 是最顶级的级别，次级别是 Module。一个 Project

可以有多个 Module。目前主流的大型项目都是分布式部署的，结构都是类似这种多 Module 结构。



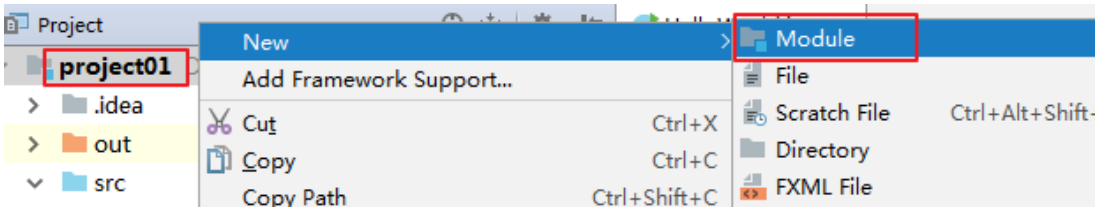
这类项目一般是这样划分的，比如：core Module、web Module、plugin Module、solr Module 等等，模块之间彼此可以相互依赖。通过这些 Module 的命名也可以看出，他们之间都是处于同一个项目业务下的模块，彼此之间是有不可分割的业务关系的。举例：

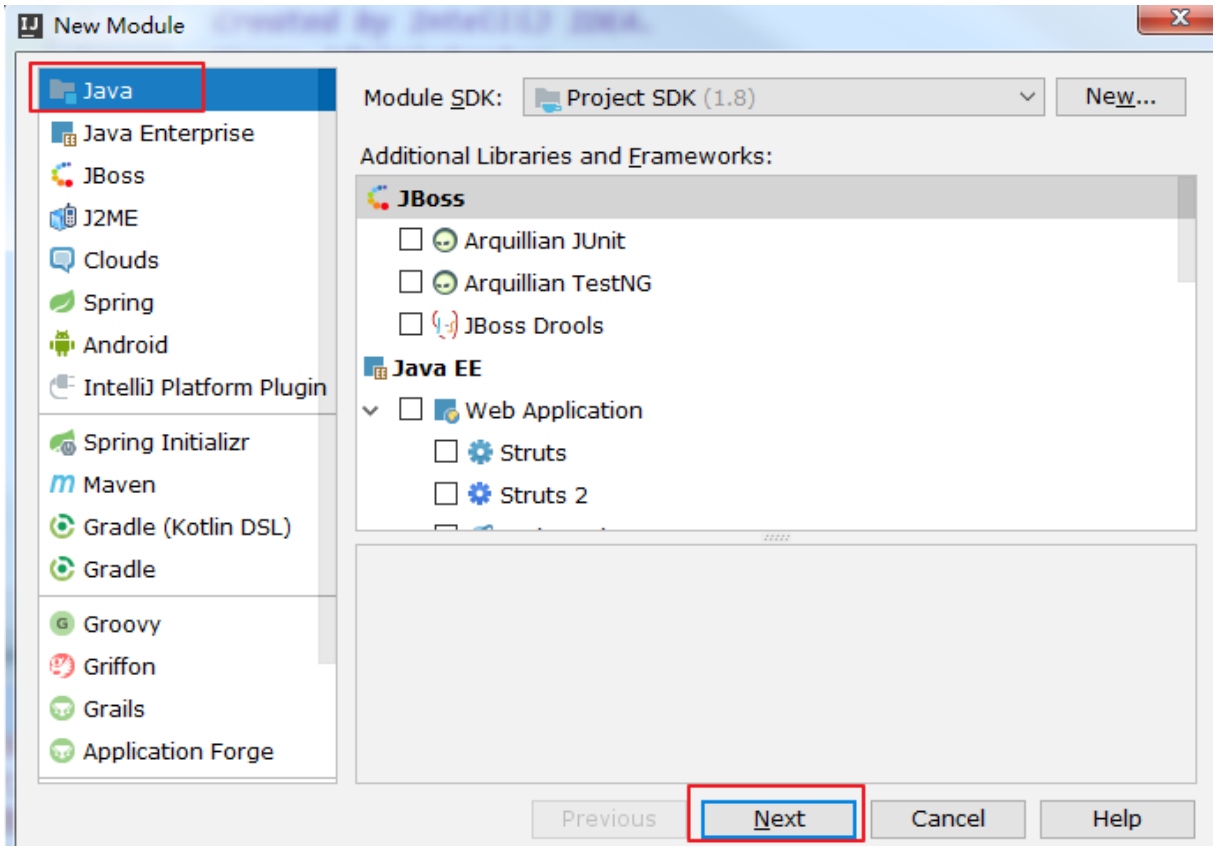


4. 相比较于多 Module 项目，小项目就无需搞得这么复杂。只有一个 Module 的结构 IntelliJ IDEA 也是支持的，并且 IntelliJ IDEA 创建项目的时候，默认就是单

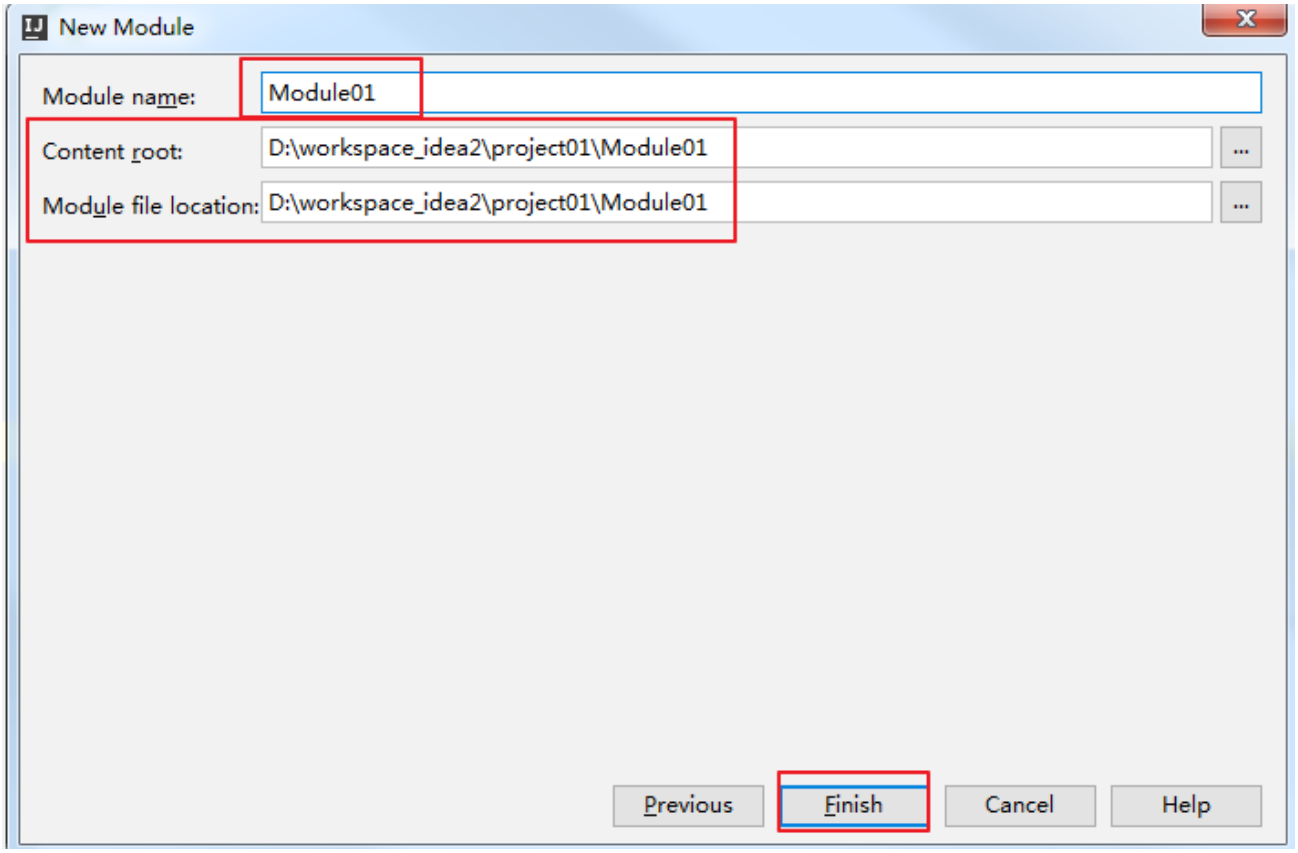
Module 的结构的。

下面，我们演示如何创建 Module:



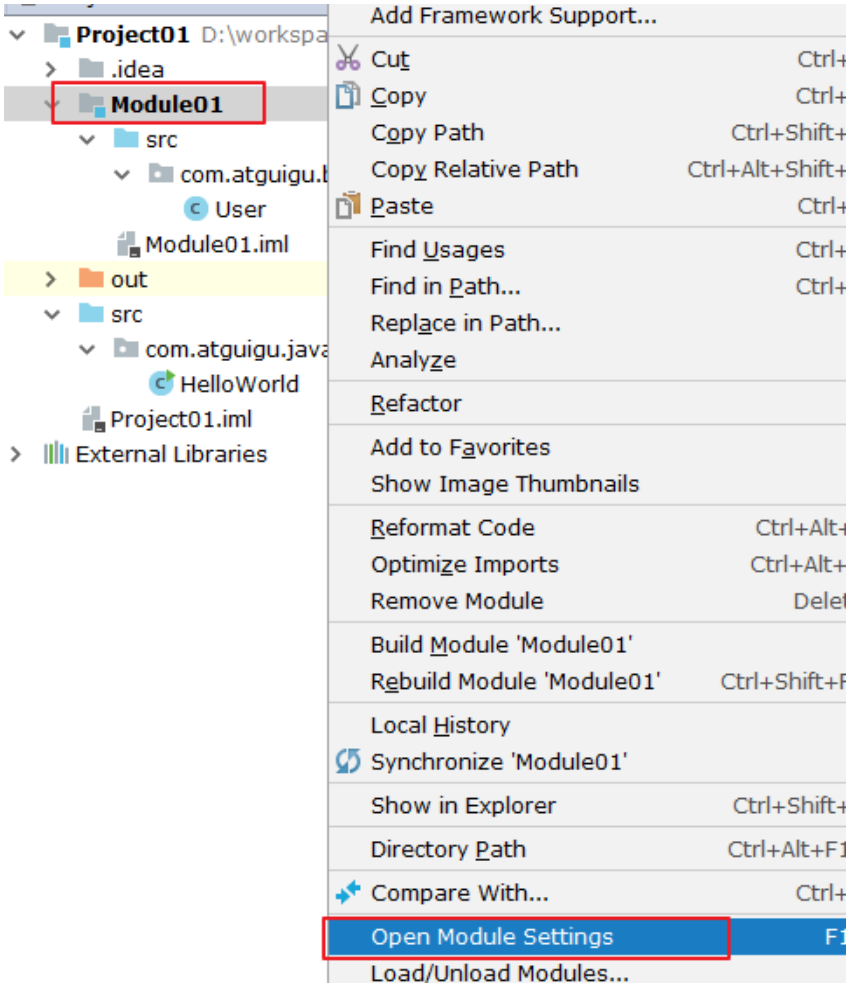


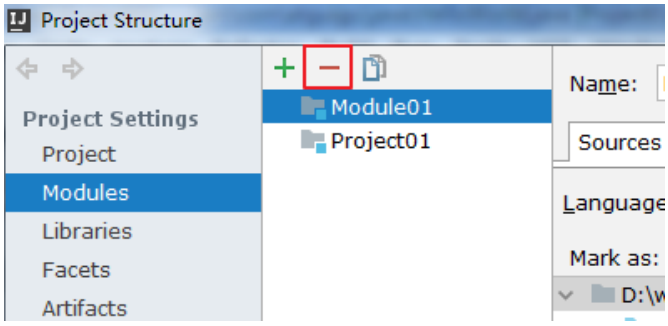
接着选择 Next:

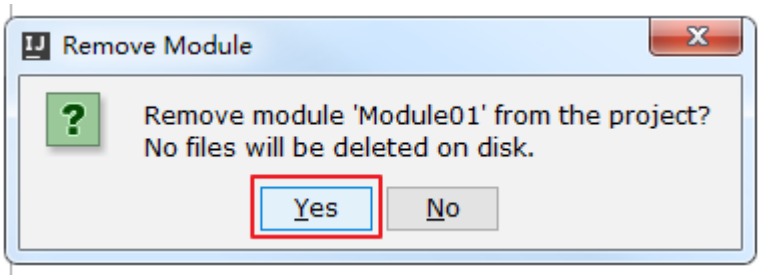


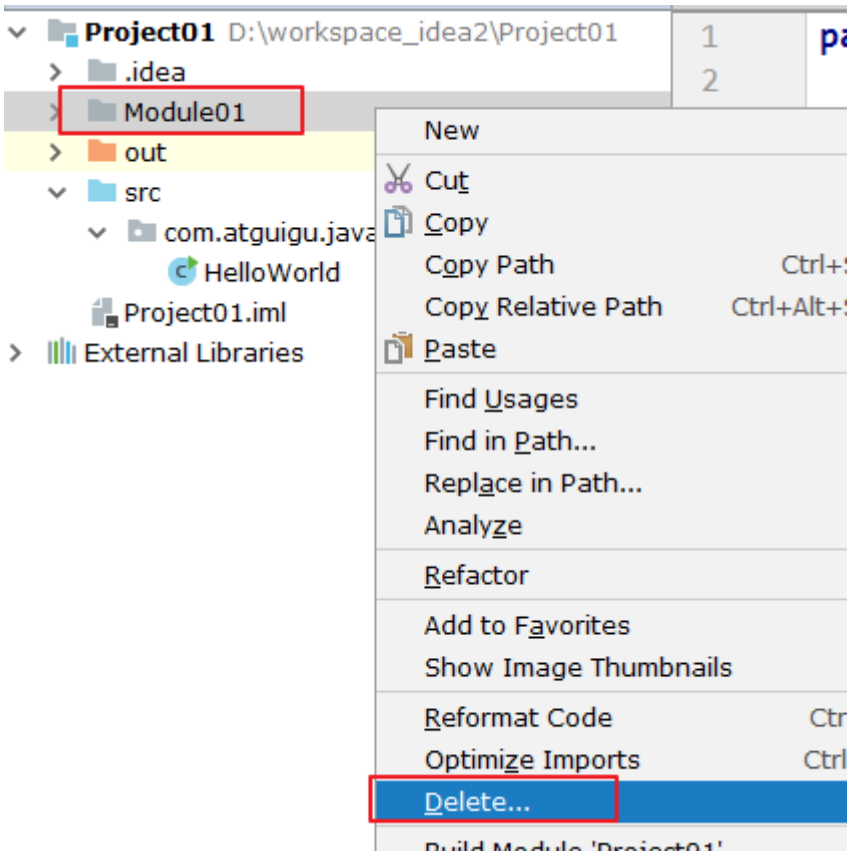
之后，我们可以在 Module 的 src 里写代码，此时 Project 工程下的 src 就没什么用了。可以删掉。

6. 如何删除模块


Project01 D:\workspa
.idea
Module01
src
com.atguigu.
User
Module01.iml
out
src
com.atguigu.java
HelloWorld
Project01.iml
External Libraries
Add Framework Support...
Cut Ctrl+
Copy Ctrl+
Copy Path Ctrl+Shift+
Copy Relative Path Ctrl+Alt+Shift+
Paste Ctrl+
Find Usages Ctrl+
Find in Path... Ctrl+
Replace in Path...
Analyze
Refactor
Add to Favorites
Show Image Thumbnails
Reformat Code Ctrl+Alt+
Optimize Imports Ctrl+Alt+
Remove Module Delet
Build Module 'Module01'
Rebuild Module 'Module01' Ctrl+Shift+F
Local History
Synchronize 'Module01'
Show in Explorer Ctrl+Shift+
Directory Path Ctrl+Alt+F1
Compare With... Ctrl+
Open Module Settings F1
Load/Unload Modules...

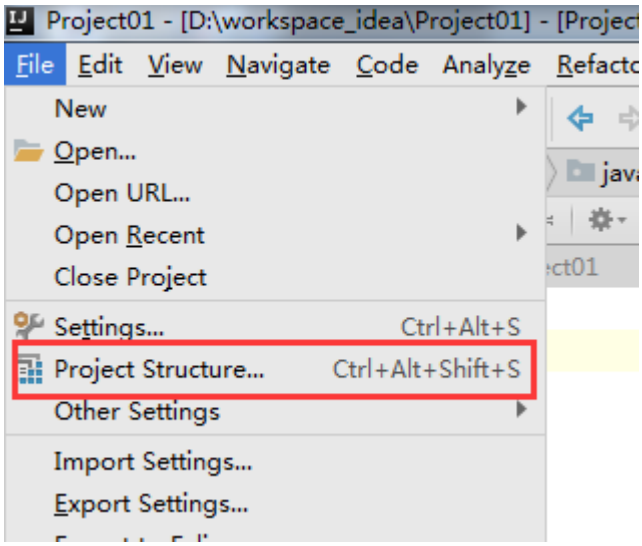

Project Structure
Project Settings
Project
Modules
Libraries
Facets
Artifacts
Module01
Project01
Name:
Sources
Language
Mark as:


Remove Module
Remove module 'Module01' from the project?
No files will be deleted on disk.
Yes
No



此时的删除，会从硬盘上将此 module 删除掉。

7.查看项目配置

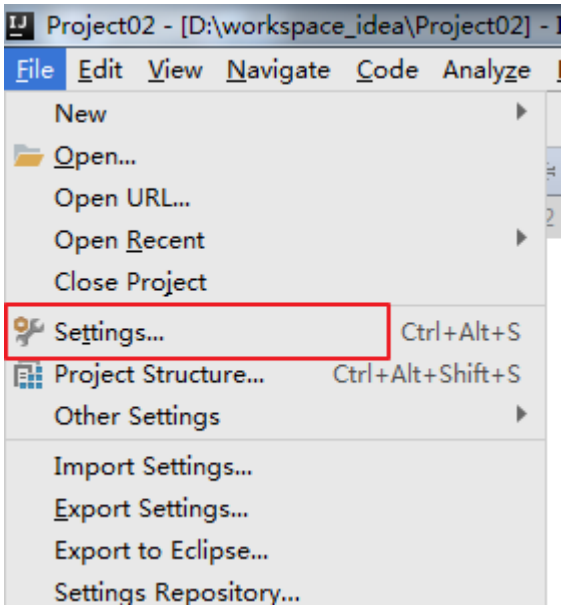


进入项目结构：

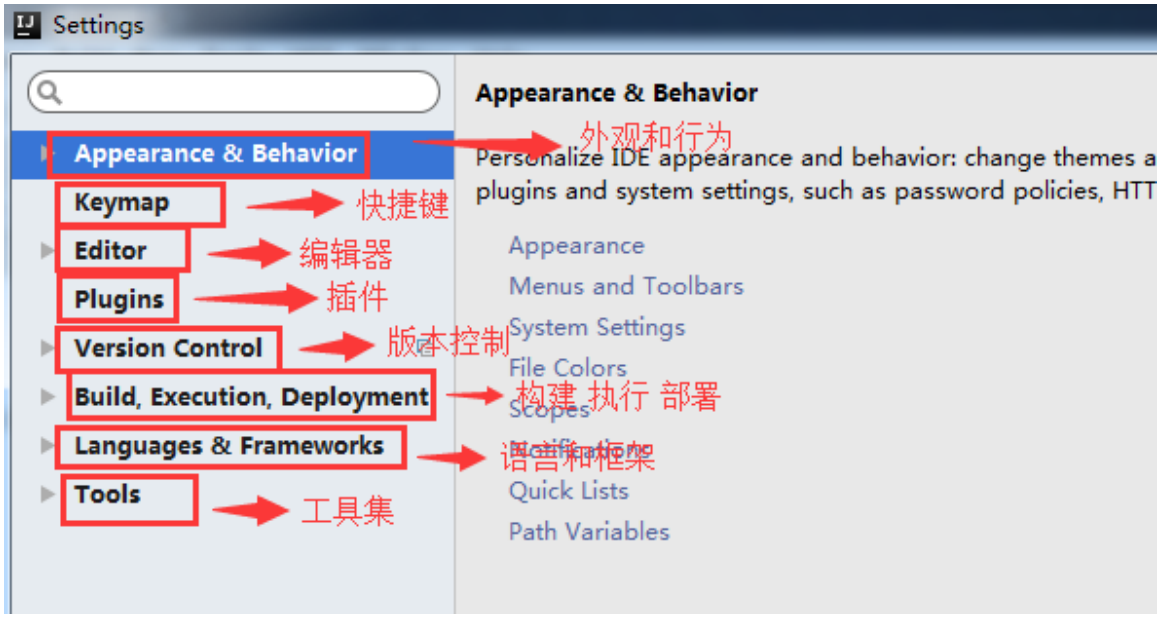
五、常用配置

IntelliJ IDEA 有很多人性化的设置我们必须单独拿出来讲解，也因为这些人性化的设置让那些 IntelliJ IDEA 死忠粉更加死心塌地使用它和分享它。

进入设置界面：

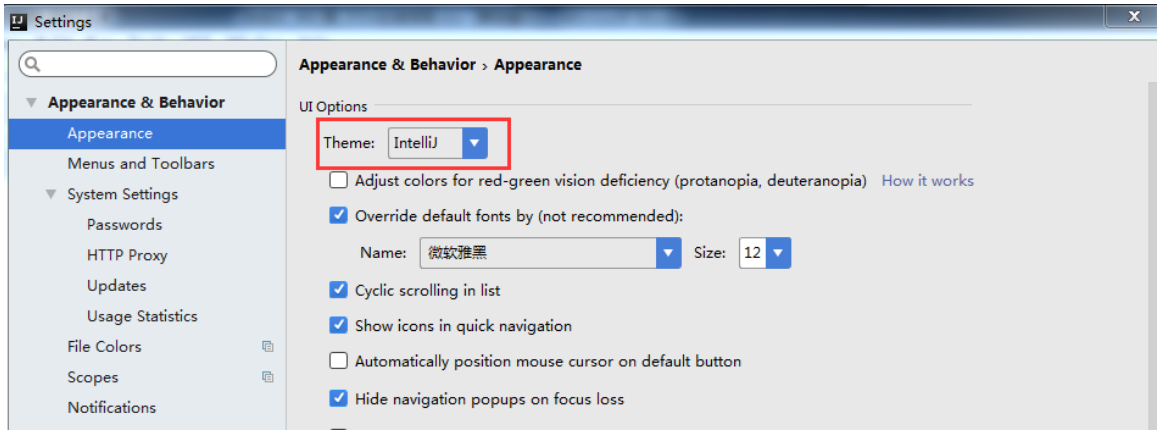


目录结构如下：



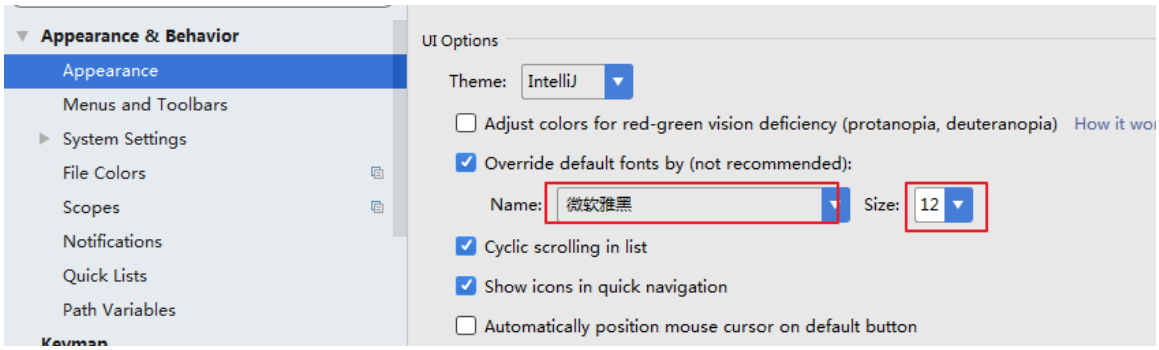
1.Appearance & Behavior

1.1 设置主题



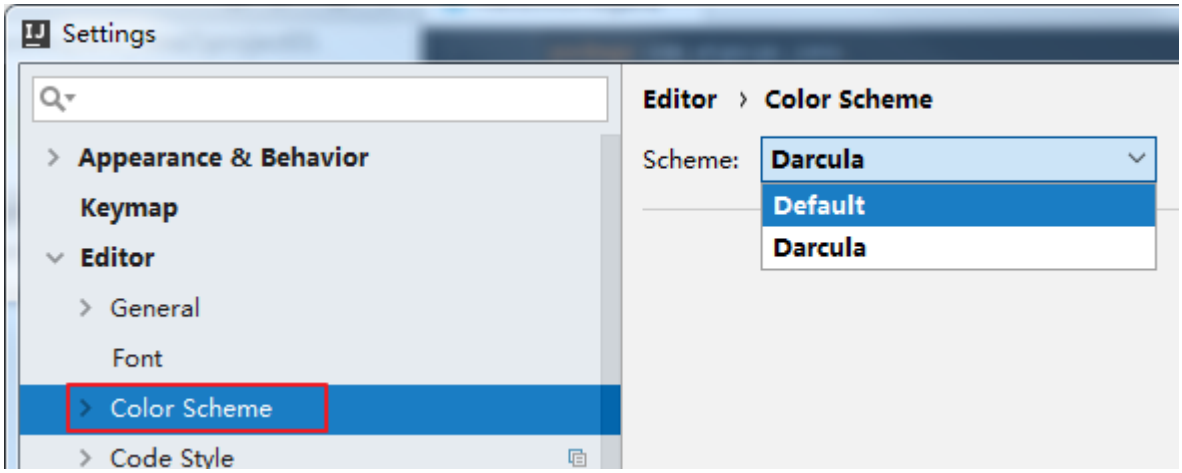
这里默认提供了三套主题：IntelliJ，Darcula，Windows。这里可以根据自己的喜好进行选择。

1.2 设置窗体及菜单的字体及字体大小 (可忽略)



1.3 补充:设置编辑区主题 (可忽略)

IDEA 默认提供了两个编辑区主题，可以通过如下的方式进行选择。



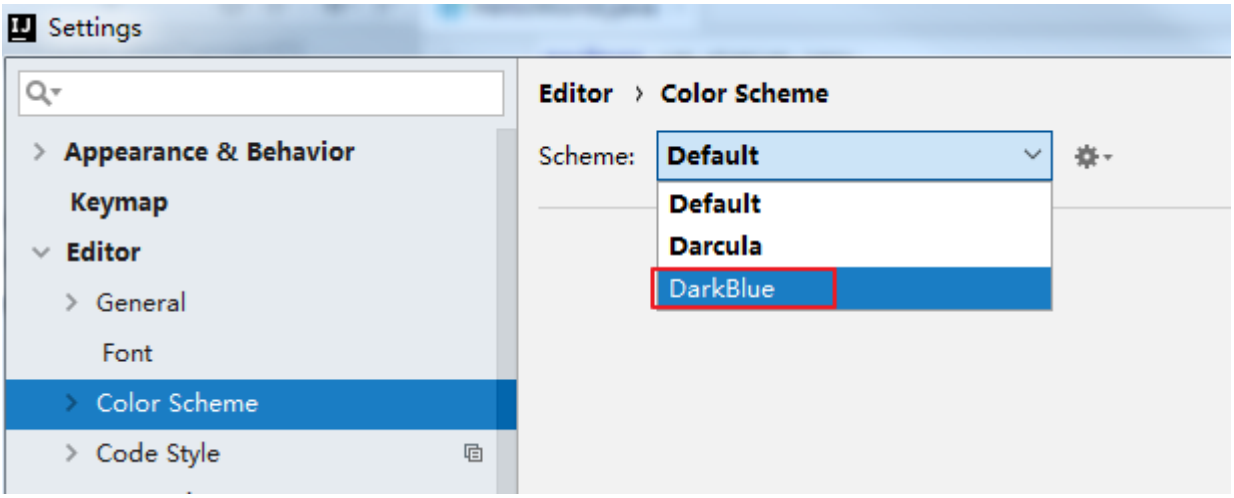
➢ 如果想要更多的主题效果的话，可以到如下的网站下载：

http://www.riaway.com/

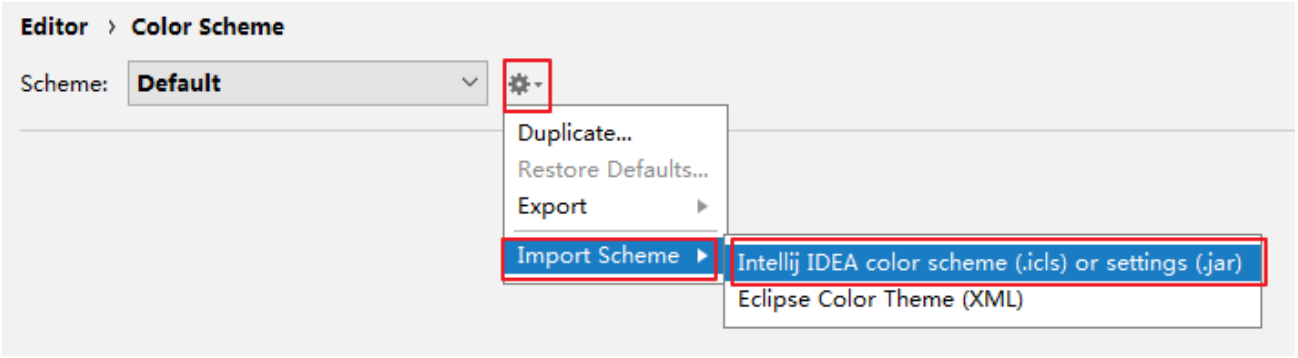
➢ 下载以后，导入主题：（方式一）

file －> import setttings －> 选中下载的主题 jar 文件 －> 一路确认 －> 重启。

重启以后，新主题会自动启用。如果没有启用，可以如下方式选择：

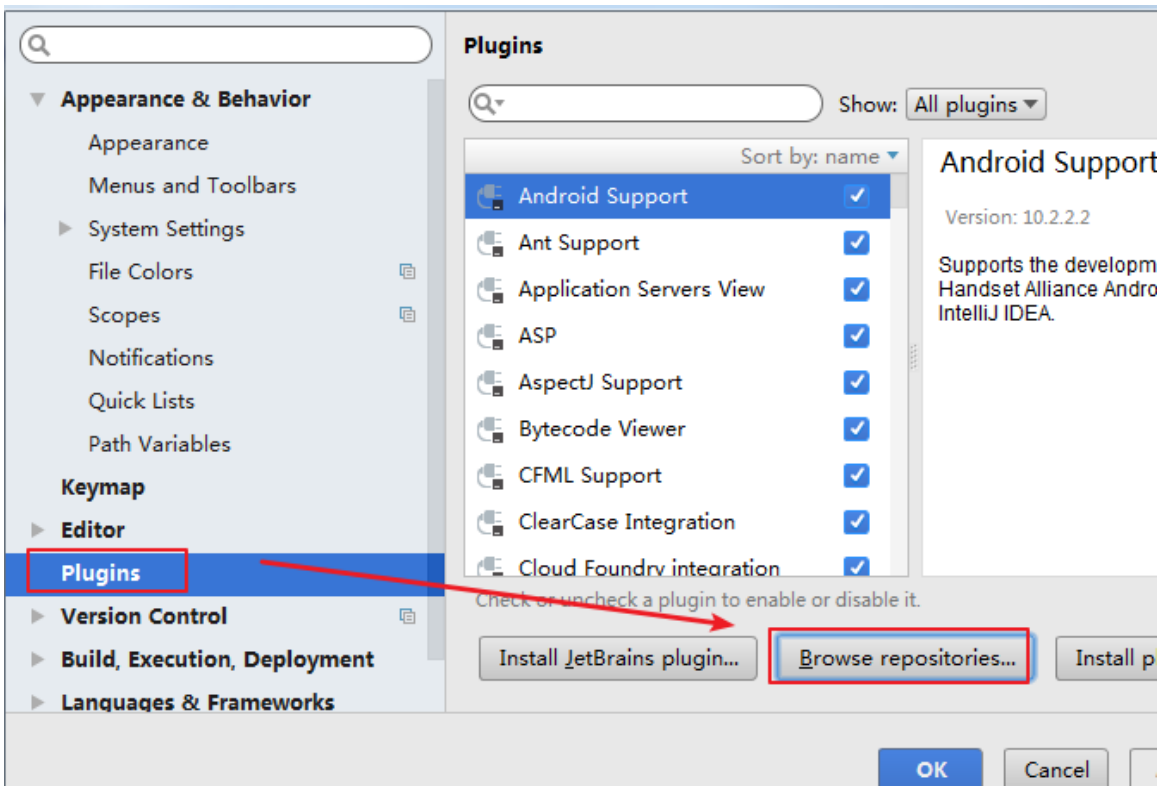


➢ 下载以后，导入主题：（方式二）

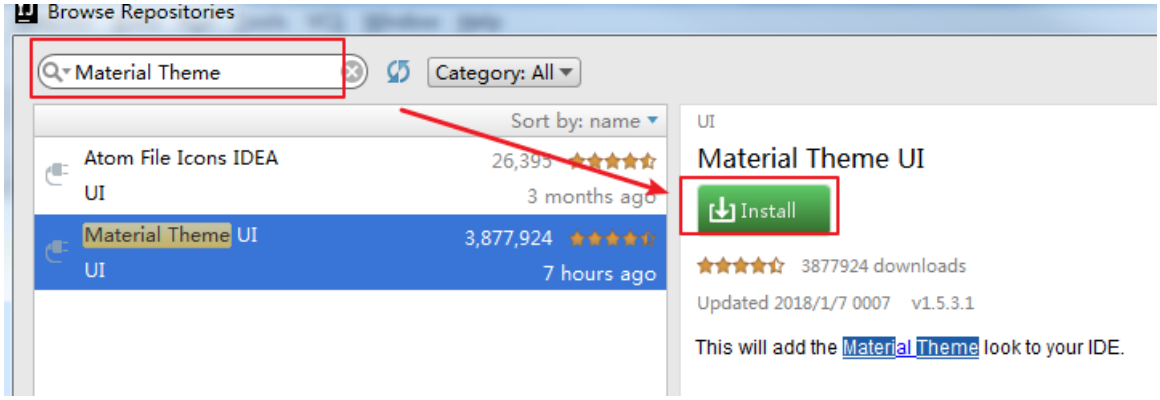


1.4 补充:通过插件(plugins)更换主题

喜欢黑色主题的话，还可以下载插件：Material Theme UI



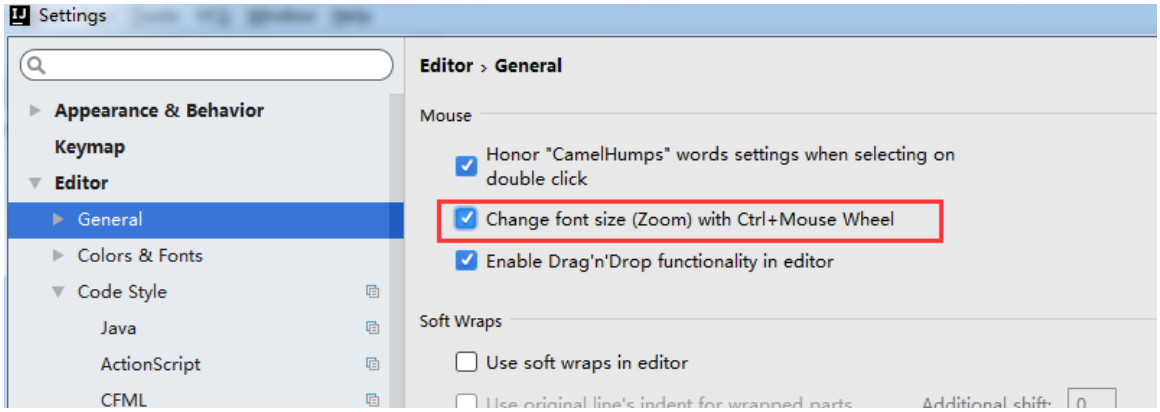
点击按钮以后，在联网环境下搜索如下的插件-安装-重启 IDEA 即可：



如果对安装的主题插件不满意，还可以找到此插件，进行卸载 – 重启 IDEA 即可。

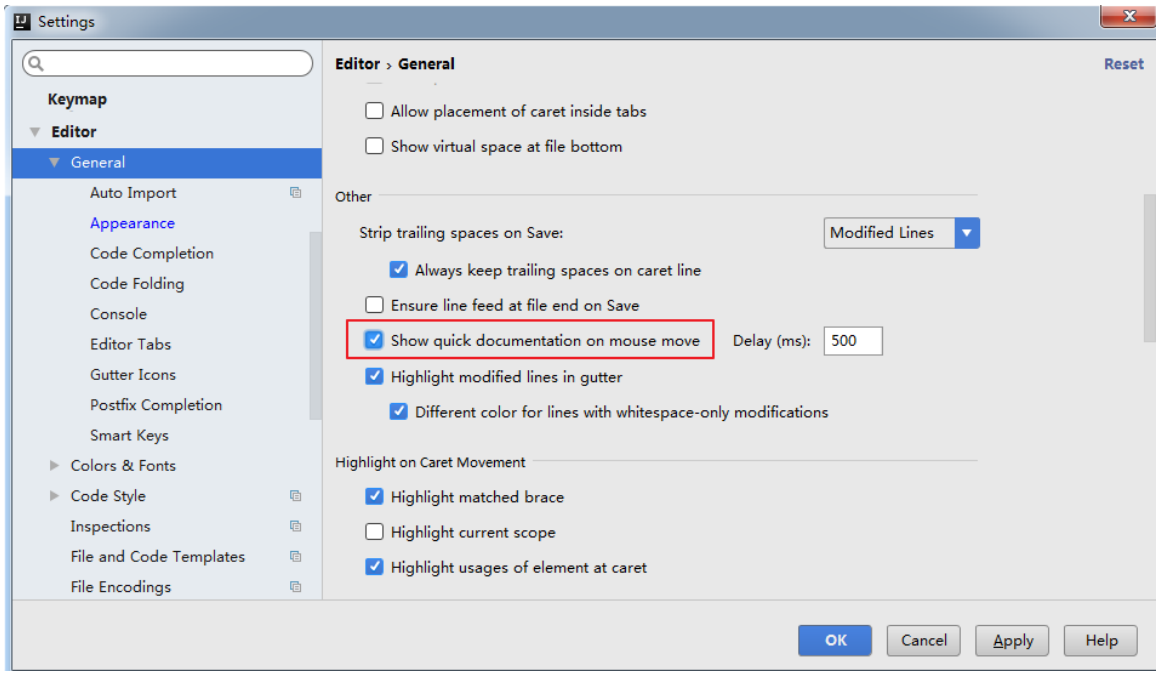
2. Editor - General

2.1 设置鼠标滚轮修改字体大小(可忽略)

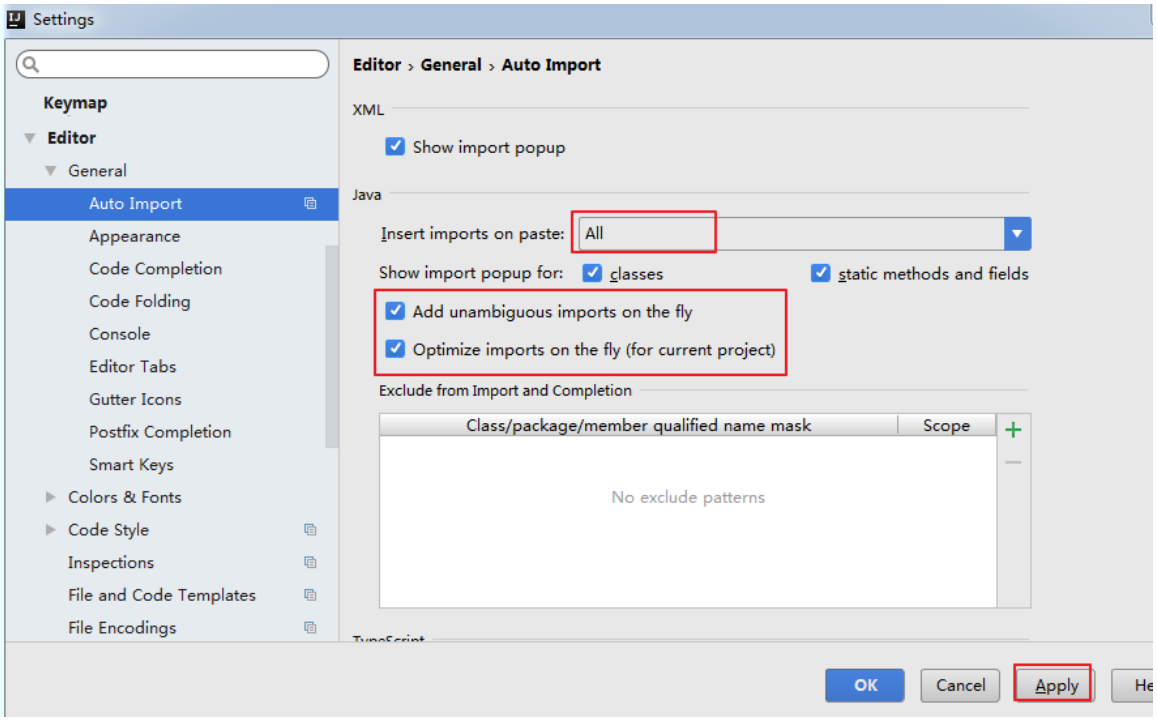


我们可以勾选此设置后，增加 Ctrl + 鼠标滚轮 快捷键来控制代码字体大小显示。

2.2 设置鼠标悬浮提示

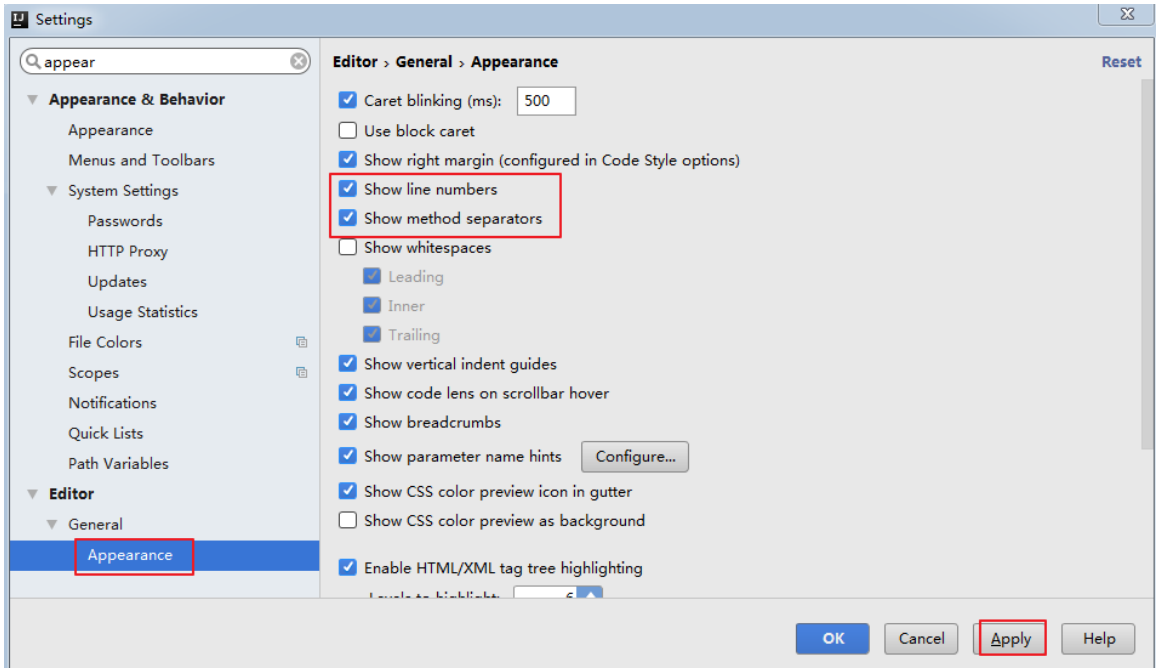


2.3 设置自动导包功能



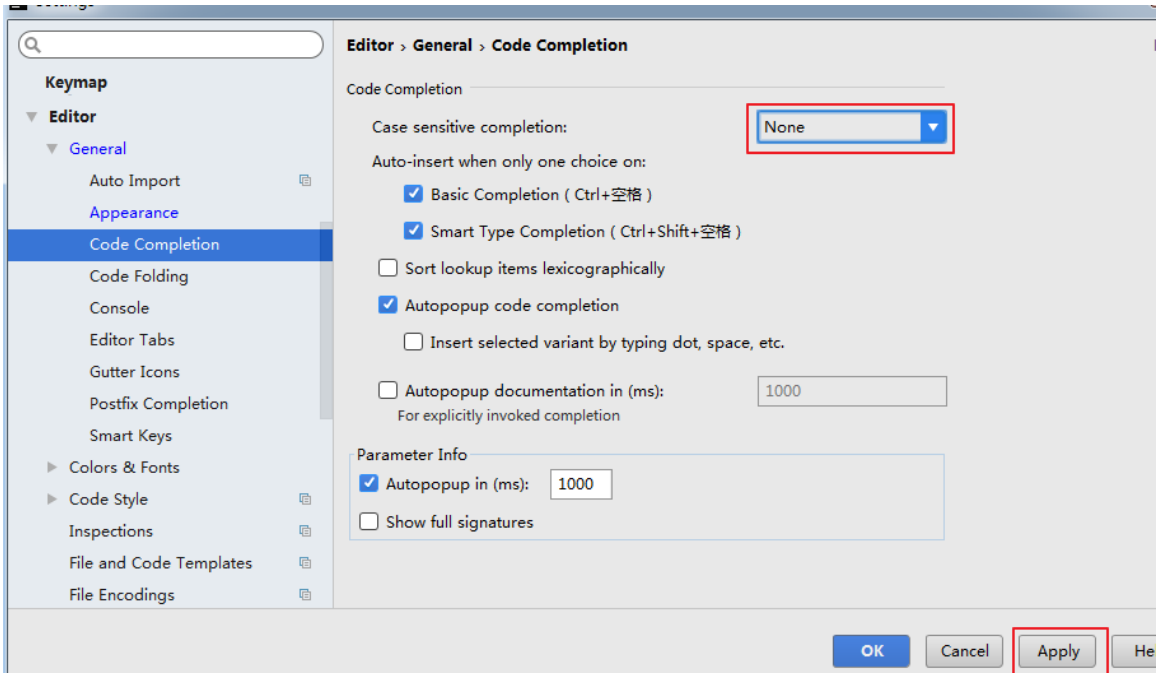
- Add unambiguous imports on the fly：自动导入不明确的结构
- Optimize imports on the fly：自动帮我们优化导入的包

2.4 设置显示行号和方法间的分隔符



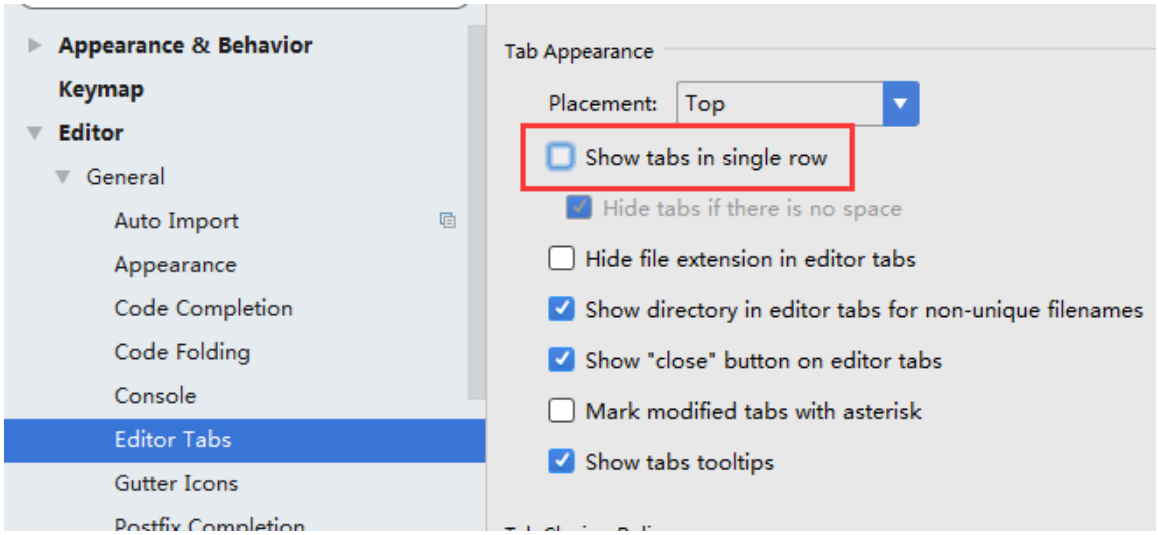
- 如上图红圈所示，可以勾选 Show line numbers：显示行数。我建议一般这个要勾选上。
- 如上图红圈所示，可以勾选 Show method separators： 显示方法分隔线。这种线有助于我们区分开方法，所以建议勾选上。

2.5 忽略大小写提示



- IntelliJ IDEA 的代码提示和补充功能有一个特性：区分大小写。如上图标注所示，默认就是 First letter 区分大小写的。
- 区分大小写的情况是这样的：比如我们在 Java 代码文件中输入 stringBuffer，IntelliJ IDEA 默认是不会帮我们提示或是代码补充的，但是如果我们输入 StringBuffer 就可以进行代码提示和补充。
- 如果想不区分大小写的话，改为 None 选项即可。

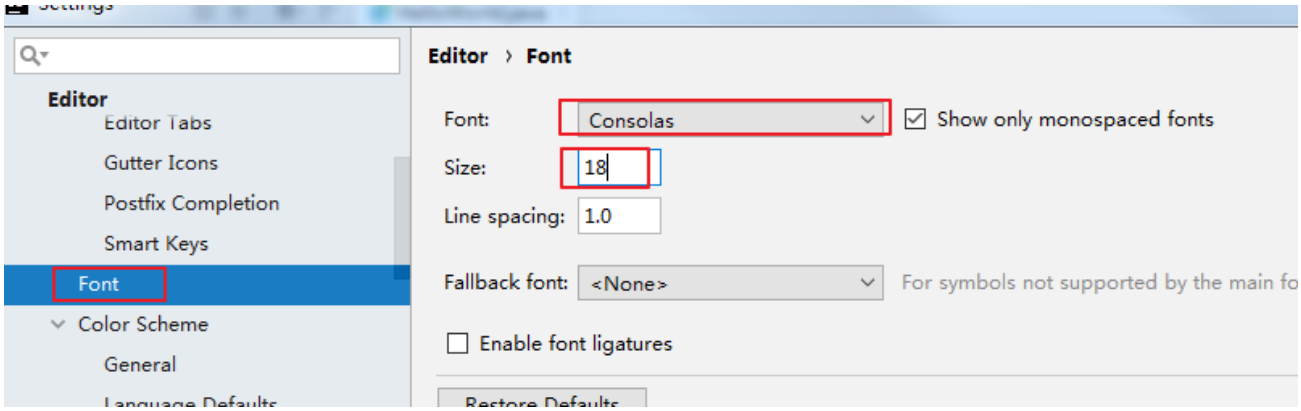
2.6 设置取消单行显示 tabs 的操作



如上图标注所示，在打开很多文件的时候，IntelliJ IDEA 默认是把所有打开的文件名 Tab 单行显示的。但是我个人现在的习惯是使用多行，多行效率比单行高，因为单行会隐藏超过界面部分 Tab，这样找文件不方便。

3. Editor – Font

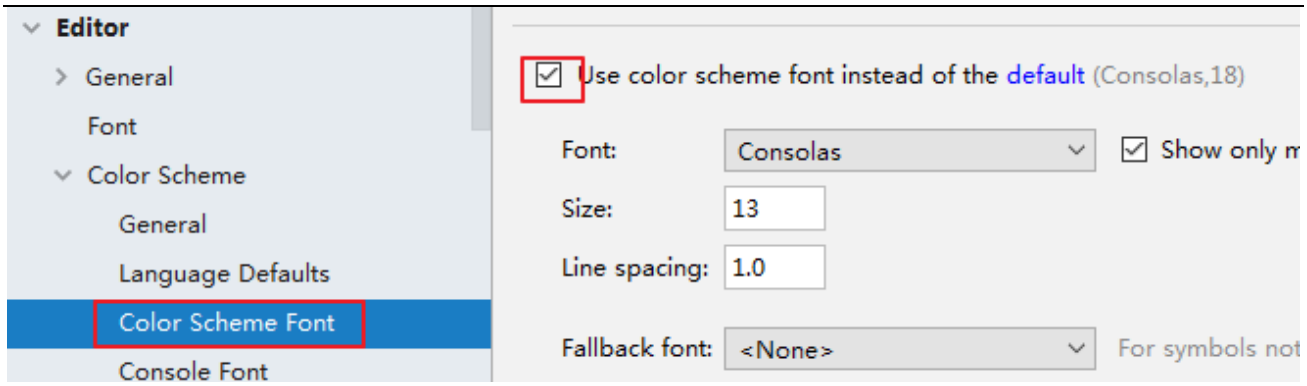
3.1 设置默认的字体、字体大小、字体行间距



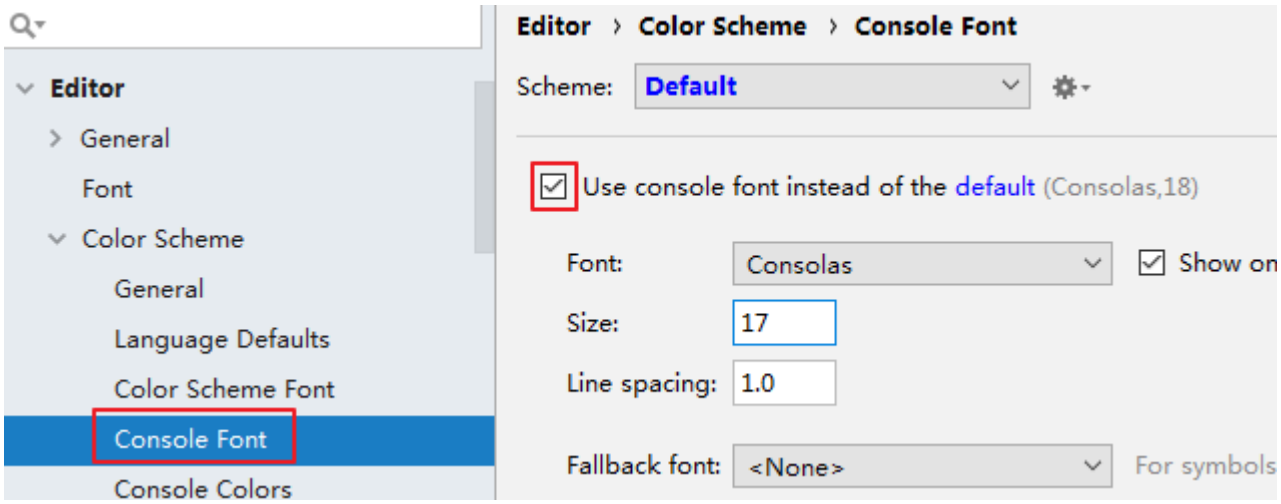
4. Editor – Color Scheme

4.1 修改当前主题的字体、字体大小、字体行间距(可忽略)

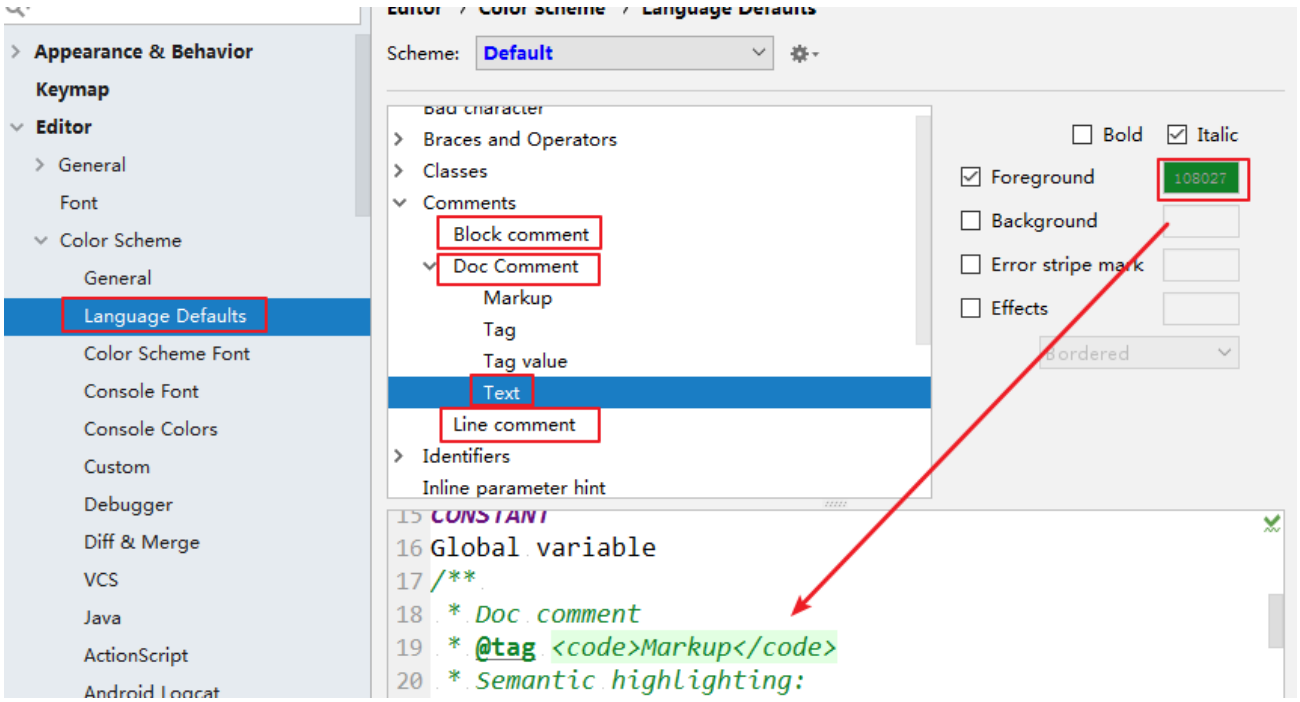
如果当前主题不希望使用默认字体、字体大小、字体行间距，还可以单独设置：



4.2 修改当前主题的控制台输出的字体及字体大小(可忽略)



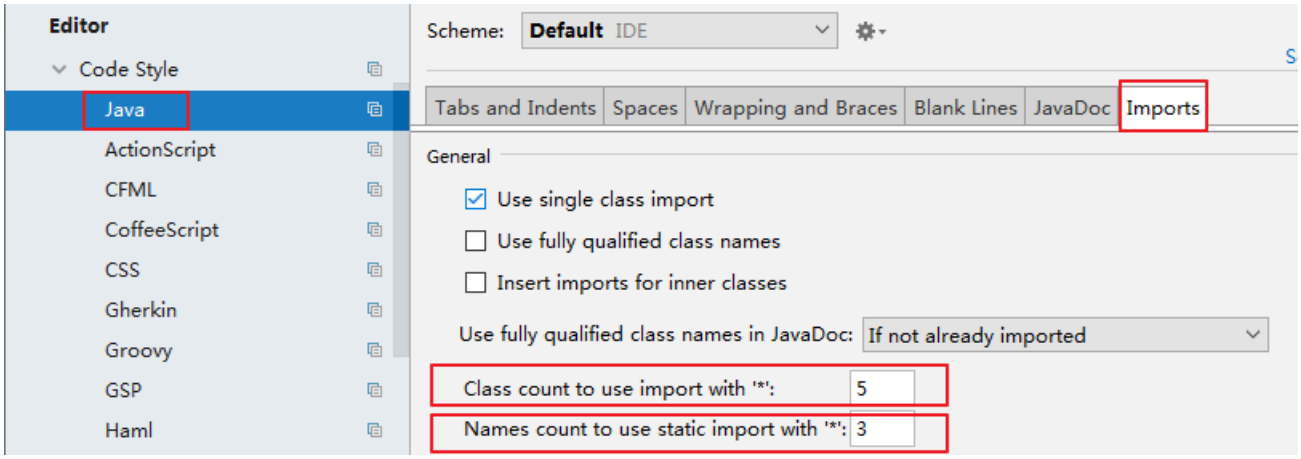
4.3 修改代码中注释的字体颜色



- Doc Comment – Text：修改文档注释的字体颜色
- Block comment：修改多行注释的字体颜色
- Line comment：修改单行注释的字体颜色

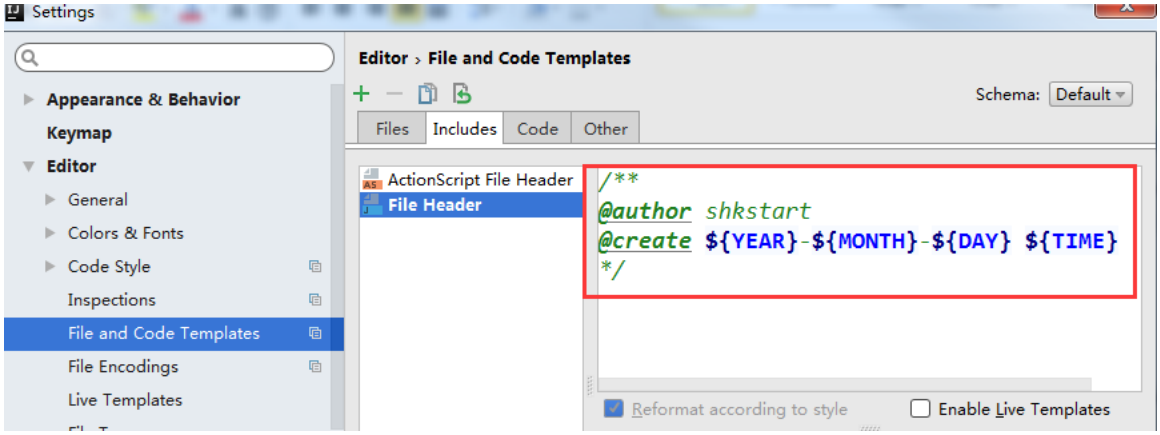
5. Editor – Code Style

5.1 设置超过指定 import 个数，改为* (可忽略)



6. Editor – File and Code Templates

6.1 修改类头的文档注释信息



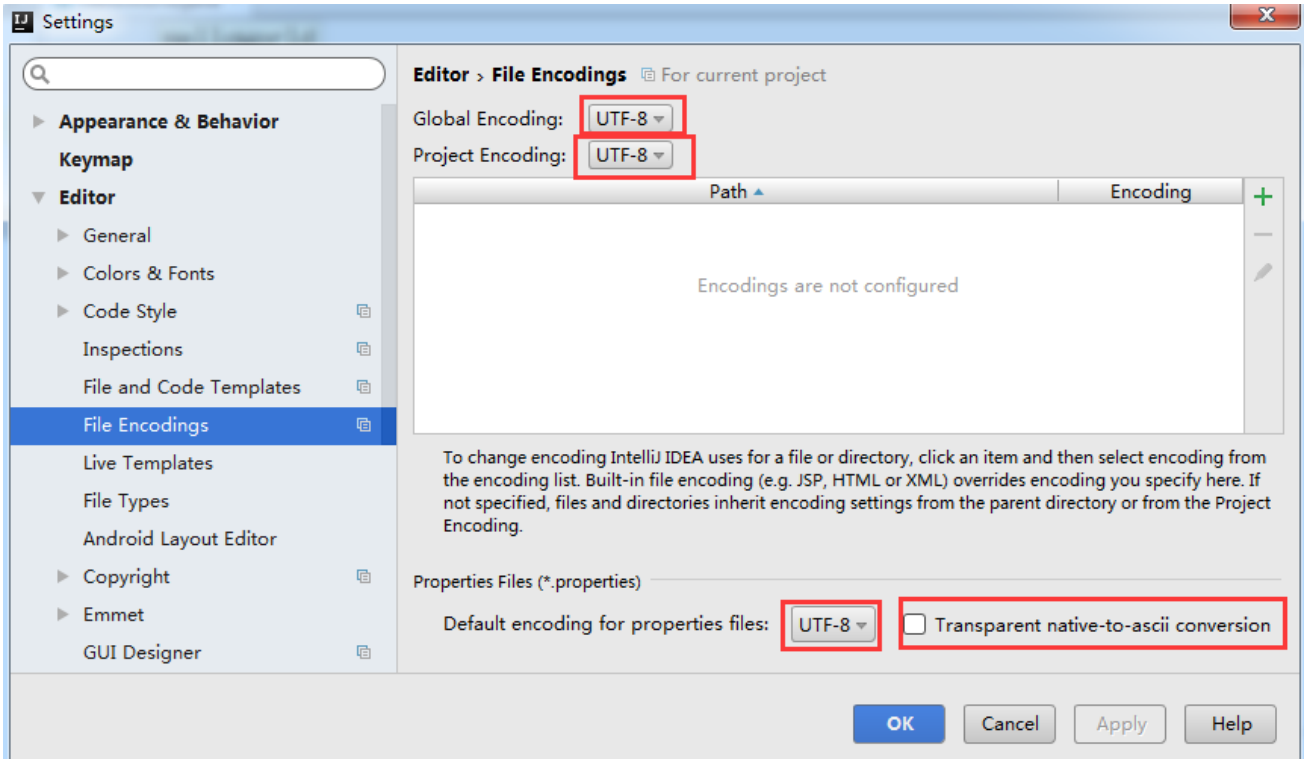

```
/**

@author shkstart

@create ${YEAR}-${MONTH}-${DAY} ${TIME}

*/
```

常用的预设的变量，这里直接贴出官网给的：

${PACKAGE_NAME} - the name of the target package where the new class or interface will be created.
${PROJECT_NAME} - the name of the current project.
${FILE_NAME} - the name of the PHP file that will be created.
${NAME} - the name of the new file which you specify in the New File dialog box during the file creation.
${USER} - the login name of the current user.
${DATE} - the current system date.
${TIME} - the current system time.
${YEAR} - the current year.
${MONTH} - the current month.
${DAY} - the current day of the month.
${HOUR} - the current hour.
${MINUTE} - the current minute.
${PRODUCT_NAME} - the name of the IDE in which the file will be created.
${MONTH_NAME_SHORT} - the first 3 letters of the month name. Example: Jan, Feb, etc.
${MONTH_NAME_FULL} - full name of a month. Example: January, February, etc.

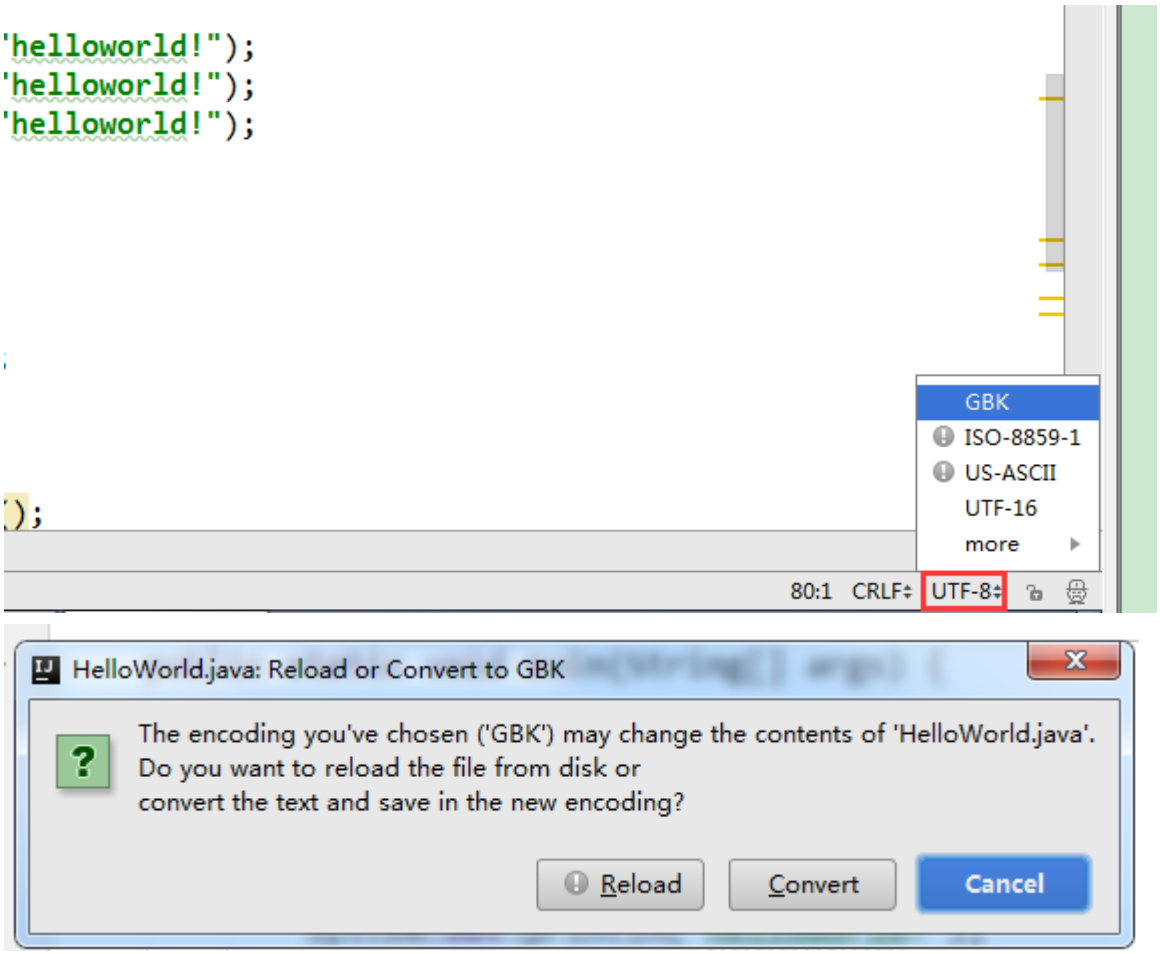
7. Editor – File Encodings

7.1 设置项目文件编码



说明：Transparent native-to-ascii conversion 主要用于转换 ascii，一般都要勾选，不然 Properties 文件中的注释显示的都不会是中文。

7.2 设置当前源文件的编码(可忽略)



对单独文件的编码修改还可以点击右下角的编码设置区。如果代码内容中包含中文，则会弹出如上的操作选择。其中：

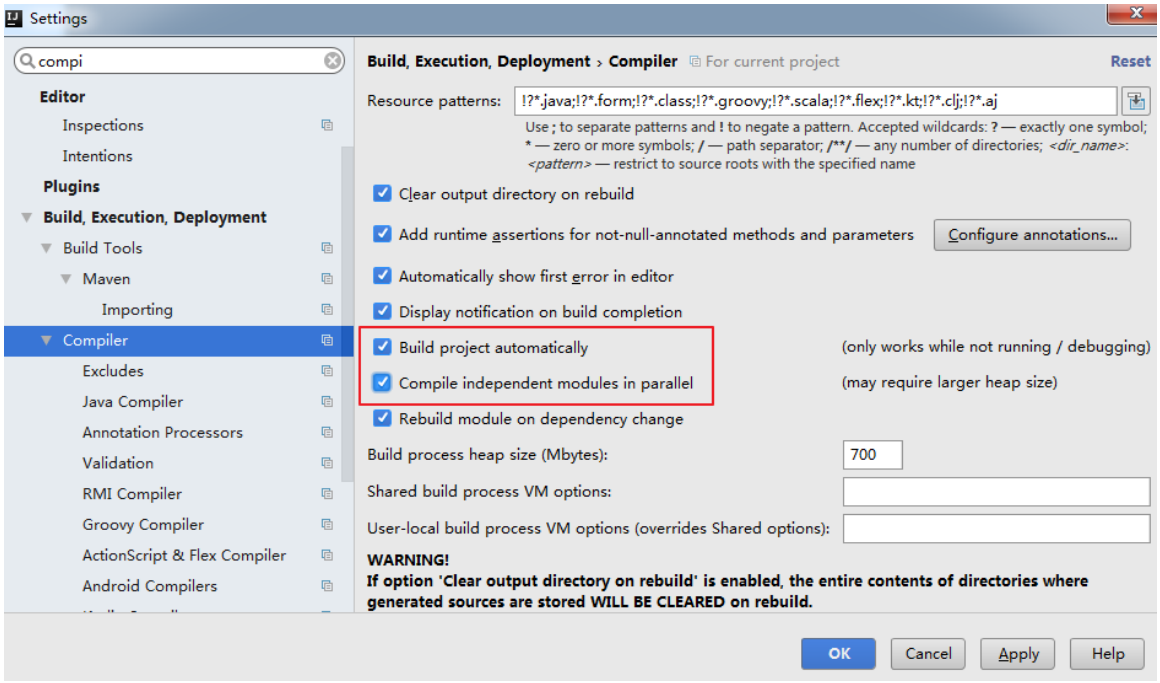
①Reload 表示使用新编码重新加载，新编码不会保存到文件中，重新打开此文件，旧编码是什么依旧还是什么。

②Convert 表示使用新编码进行转换，新编码会保存到文件中，重新打开此文件，新编码是什么则是什么。

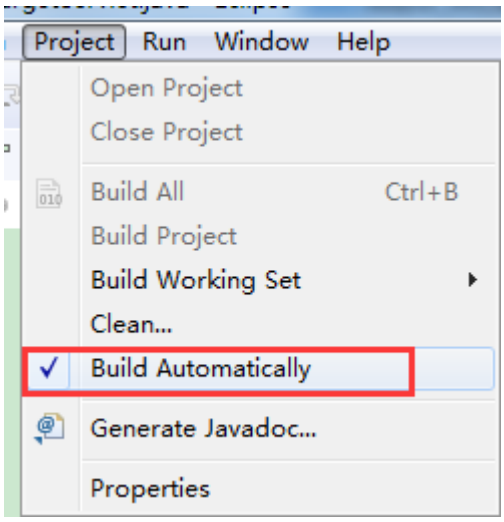
③含有中文的代码文件，Convert 之后可能会使中文变成乱码，所以在转换成请做好备份，不然可能出现转换过程变成乱码，无法还原。

8. Build,Execution,Deployment

8.1 设置自动编译

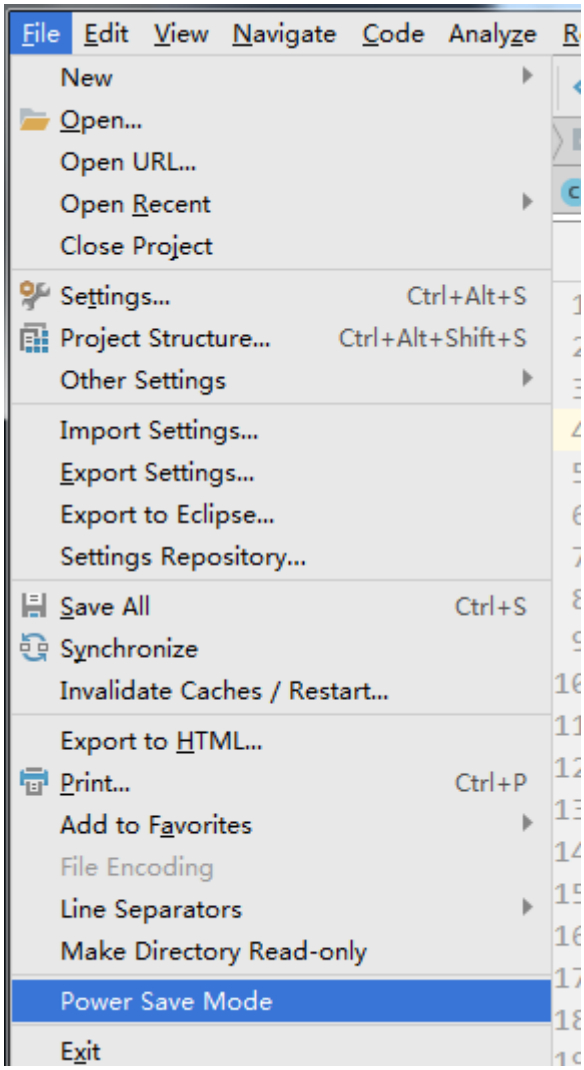


- 构建就是以我们编写的 java 代码、框架配置文件、国际化等其他资源文件、JSP 页面和图片等资源作为“原材料”，去“生产”出一个可以运行的项目的过程。
- Intellij Idea 默认状态为不自动编译状态，Eclipse 默认为自动编译：



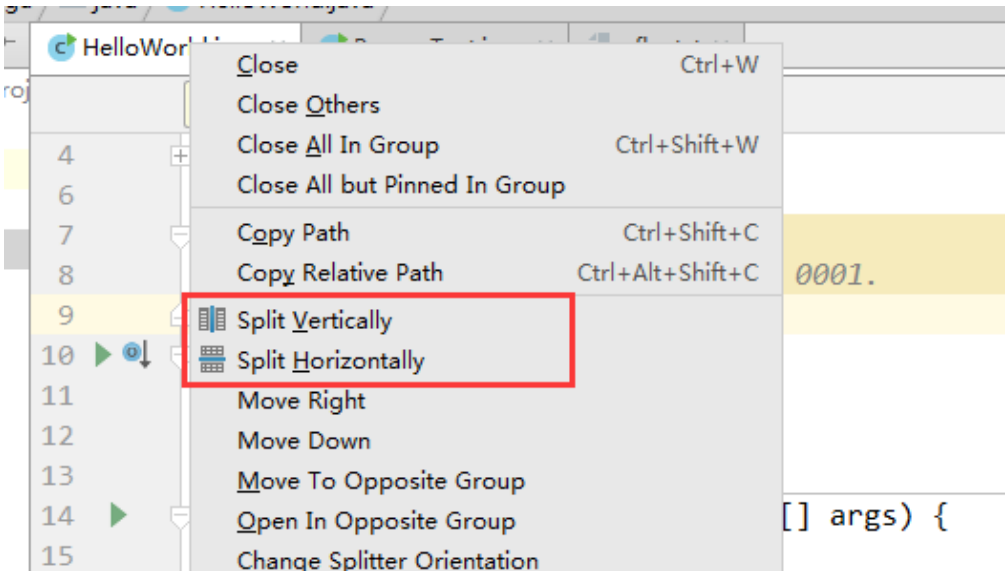
很多朋友都是从 Eclipse 转到 Intellij 的，这常常导致我们在需要操作 class 文件时忘记对修改后的 java 类文件进行重新编译，从而对旧文件进行了操作。

9. 设置为省电模式 (可忽略)



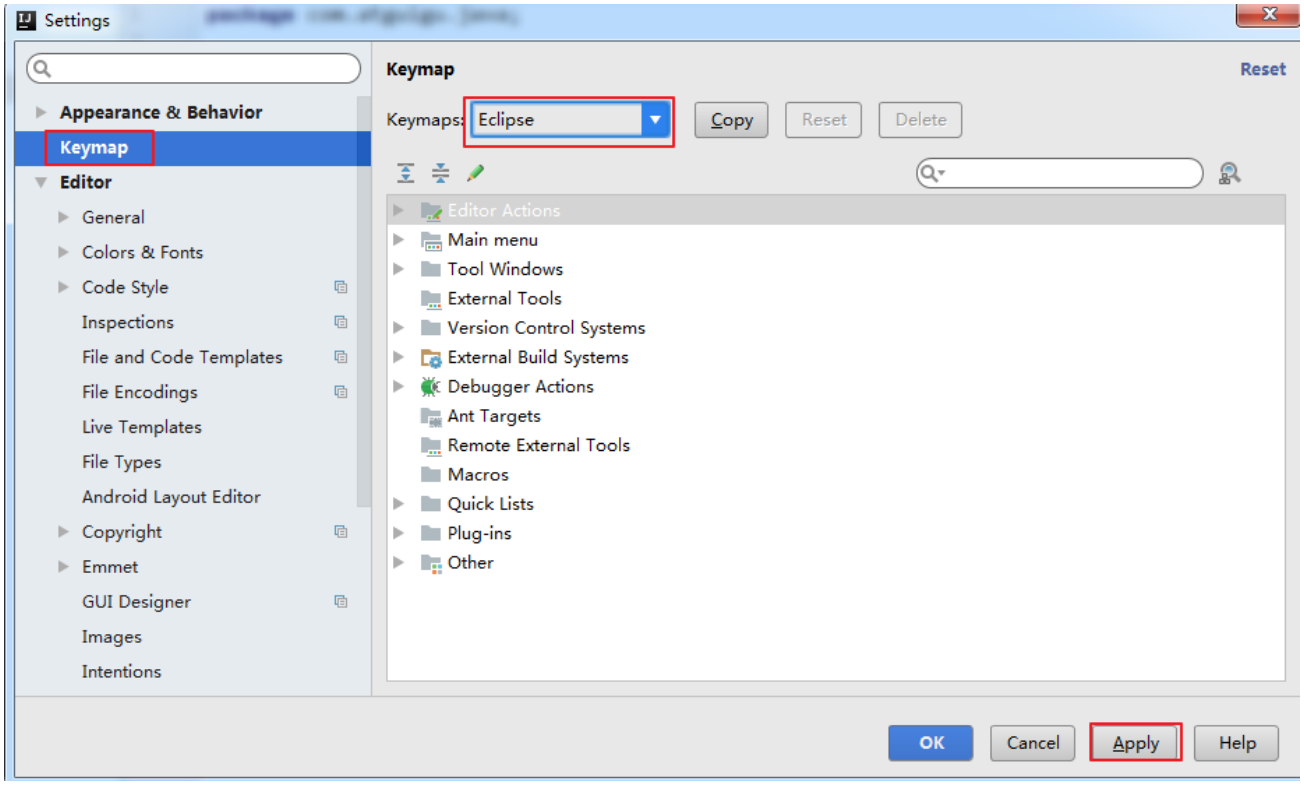
如上图所示，IntelliJ IDEA 有一种叫做 省电模式 的状态，开启这种模式之后 IntelliJ IDEA 会关掉代码检查和代码提示等功能。所以一般也可认为这是一种 阅读模式，如果你在开发过程中遇到突然代码文件不能进行检查和提示，可以来看看这里是否有开启该功能。

10. 设置代码水平或垂直显示

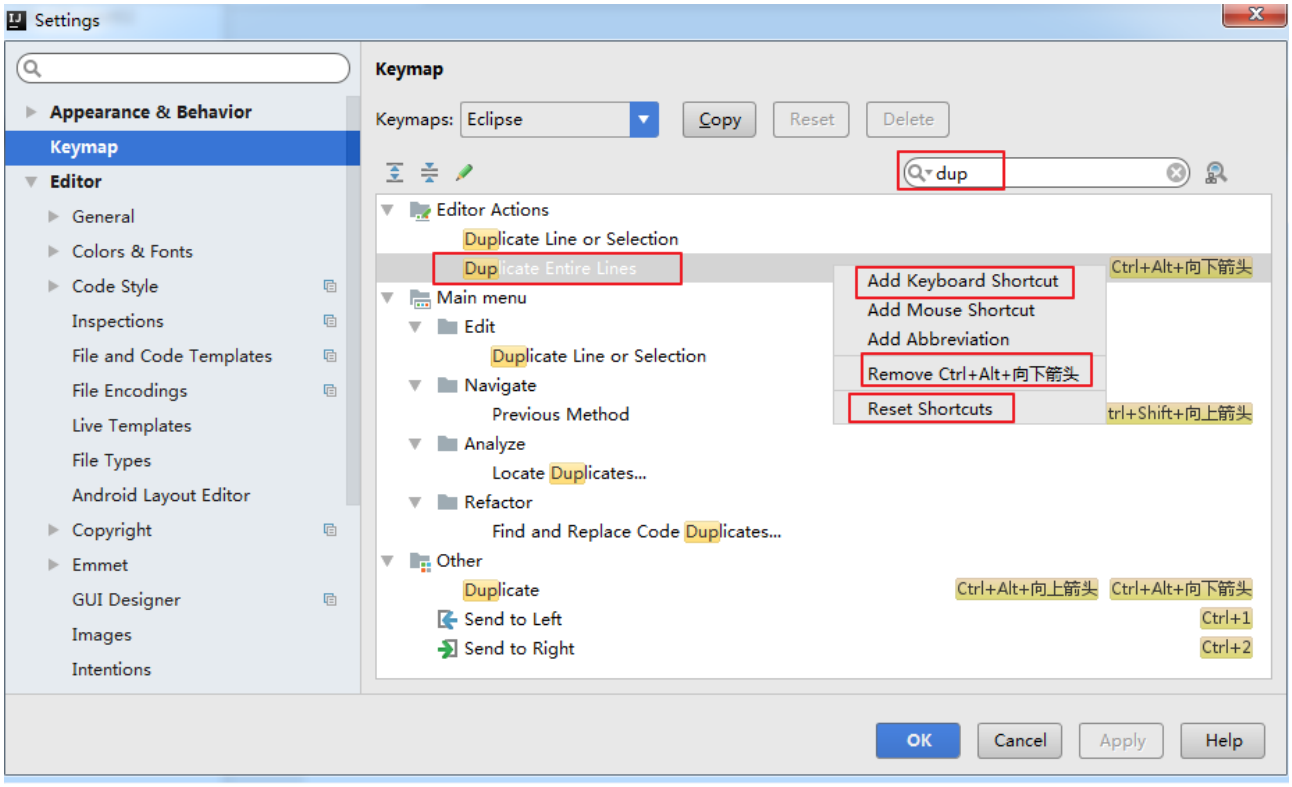


六、设置快捷键(Keymap)

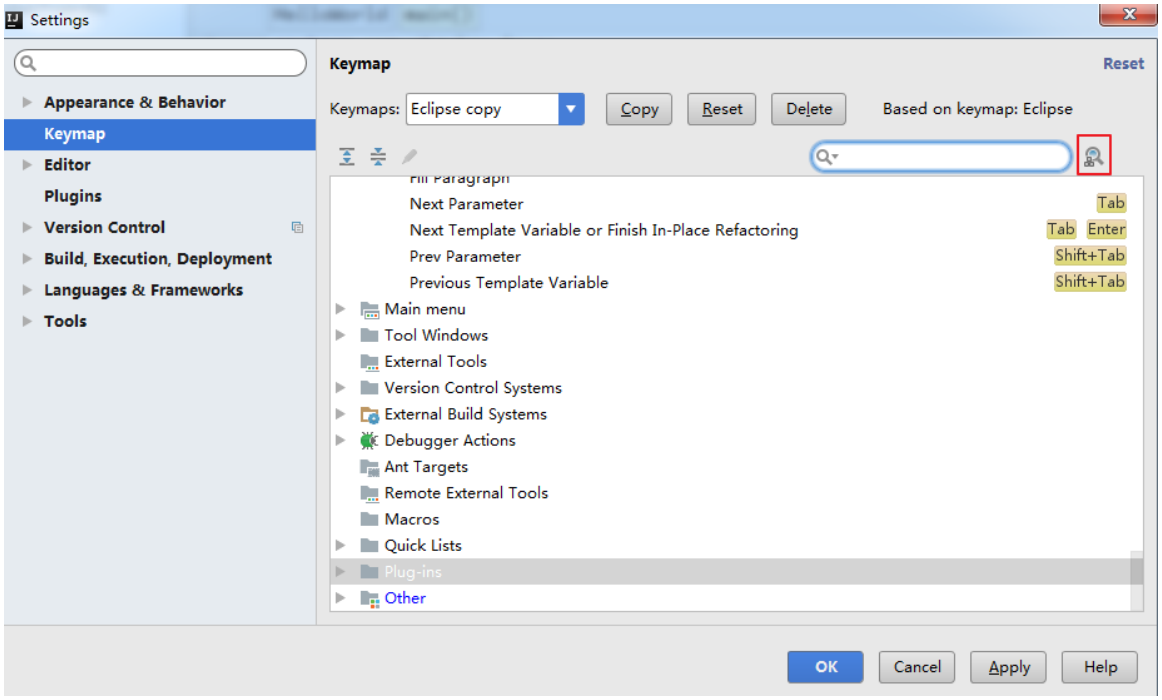
1. 设置快捷为 Eclipse 的快捷键



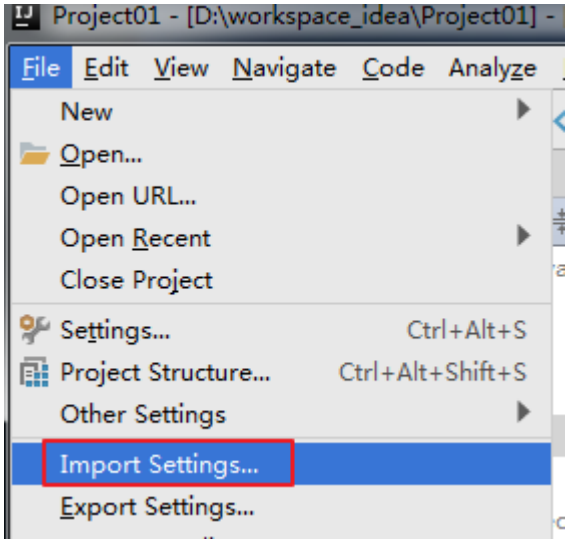
2.通过快捷键功能修改快捷键设置

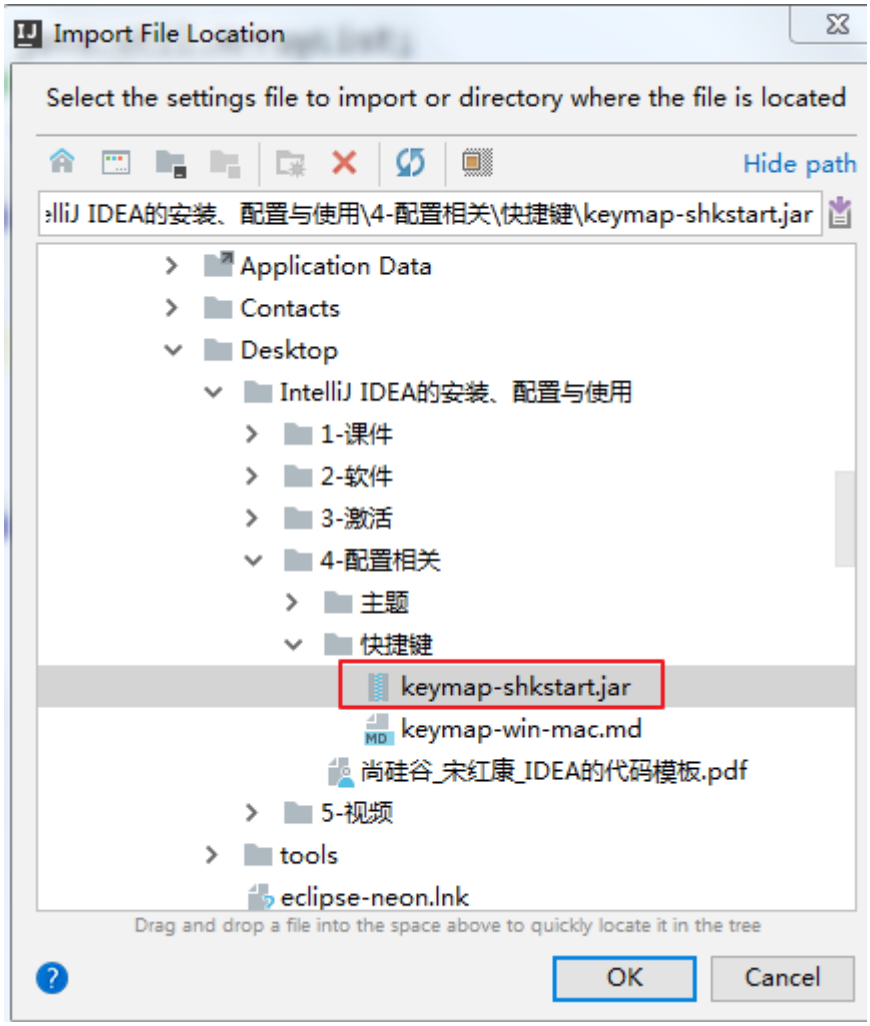


3.通过指定快捷键，查看或修改其功能



4.导入已有的设置





点击 OK 之后，重启 IDEA 即可。

5.常用快捷键

尚硅谷・宋红康 设置版		
1	执行(run)	alt+r
2	提示补全 (Class Name Completion)	alt+/
3	单行注释	ctrl + /
4	多行注释	ctrl + shift + /
5	向下复制一行 (Duplicate Lines)	ctrl+alt+down
6	删除一行或选中行 (delete line)	ctrl+d
7	向下移动行(move statement down)	alt+down
8	向上移动行(move statement up)	alt+up
9	向下开始新的一行(start new line)	shift+enter
10	向上开始新的一行 (Start New Line before current)	ctrl+shift+enter
11	如何查看源码 (class)	ctrl + 选中指定的结构 或 ctrl + shift + t
12	万能解错/生成返回值变量	alt + enter
13	退回到前一个编辑的页面 (back)	alt + left
14	进入到下一个编辑的页面(针对于上条) (forward)	alt + right
15	查看继承关系(type hierarchy)	F4
16	格式化代码(reformat code)	ctrl+shift+F
17	提示方法参数类型(Parameter Info)	ctrl+alt+/
18	复制代码	ctrl + c
19	撤销	ctrl + z
20	反撤销	ctrl + y
21	剪切	ctrl + x
22	粘贴	ctrl + v
23	保存	ctrl + s
24	全选	ctrl + a
25	选中数行，整体往后移动	tab
26	选中数行，整体往前移动	shift + tab
27	查看类的结构：类似于 eclipse 的 outline	ctrl+o
28	重构：修改变量名与方法名(rename)	alt+shift+r
29	大写转小写/小写转大写(toggle case)	ctrl+shift+y

30	生成构造器/get/set/toString	alt +shift + s
31	查看文档说明(quick documentation)	F2
32	收起所有的方法(collapse all)	alt + shift + c
33	打开所有方法(expand all)	alt+shift+x
34	打开代码所在硬盘文件夹(show in explorer)	ctrl+shift+x
35	生成 try-catch 等(surround with)	alt+shift+z
36	局部变量抽取为成员变量(introduce field)	alt+shift+f
37	查找/替换(当前)	ctrl+f
38	查找(全局)	ctrl+h
39	查找文件	double Shift
40	查看类的继承结构图(Show UML Diagram)	ctrl + shift + u
41	查看方法的多层重写结构(method hierarchy)	ctrl+alt+h
42	添加到收藏(add to favorites)	ctrl+alt+f
43	抽取方法(Extract Method)	alt+shift+m
44	打开最近修改的文件(Recently Files)	ctrl+E
45	关闭当前打开的代码栏(close)	ctrl + w
46	关闭打开的所有代码栏(close all)	ctrl + shift + w
47	快速搜索类中的错误(next highlighted error)	ctrl + shift + q
48	选择要粘贴的内容(Show in Explorer)	ctrl+shift+v
49	查找方法在哪里被调用(Call Hierarchy)	ctrl+shift+h

七、关于模板(Templates)

(Editor – Live Templates 和 Editor – General – Postfix Completion)

1.Live Templates(实时代码模板)功能介绍

它的原理就是配置一些常用代码字母缩写，在输入简写时可以出现你预定义的固

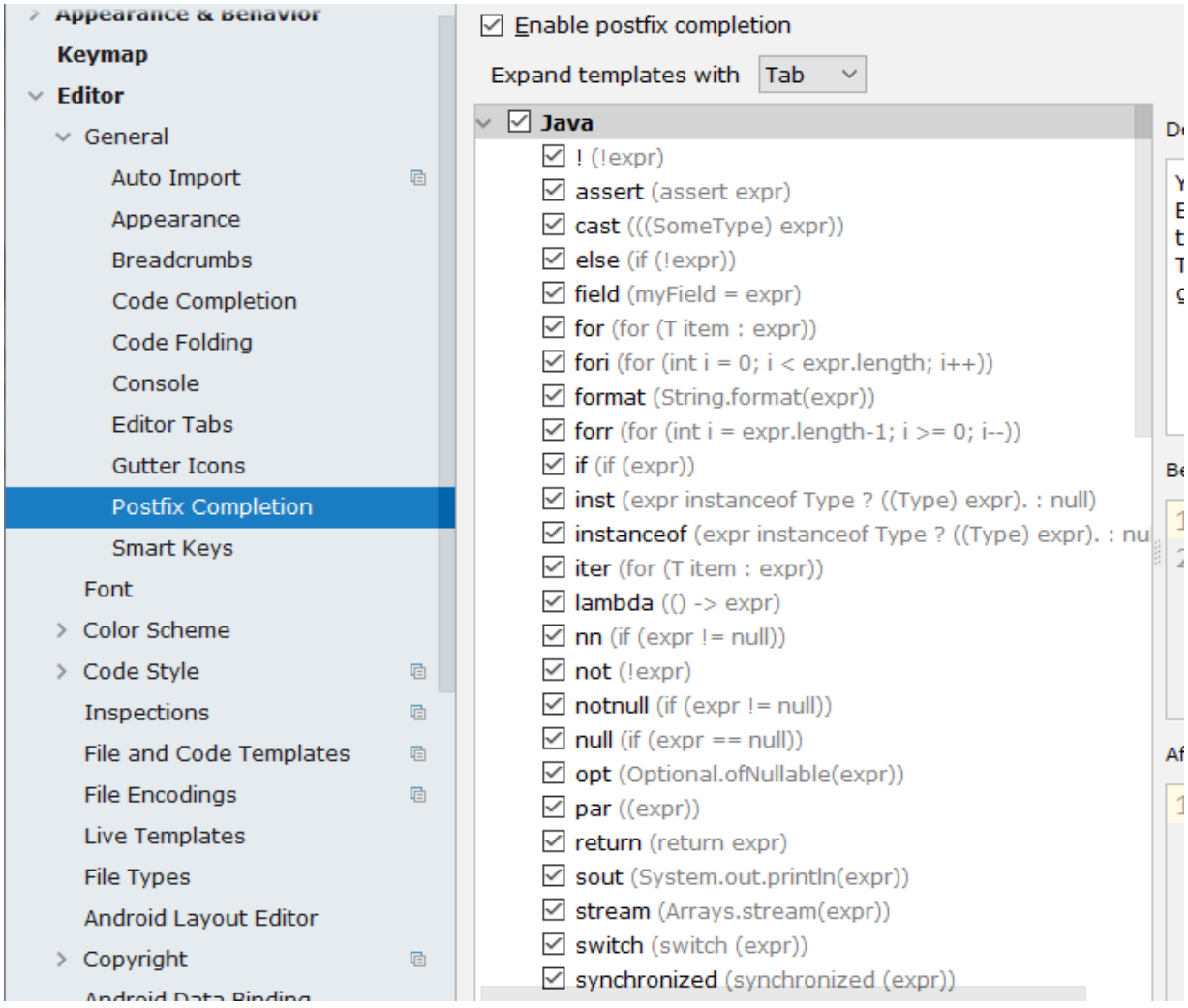
定模式的代码，使得开发效率大大提高，同时也可以增加个性化。最简单的例子就是在 Java 中输入 sout 会出现 System.out.println();

官方介绍 Live Templates：

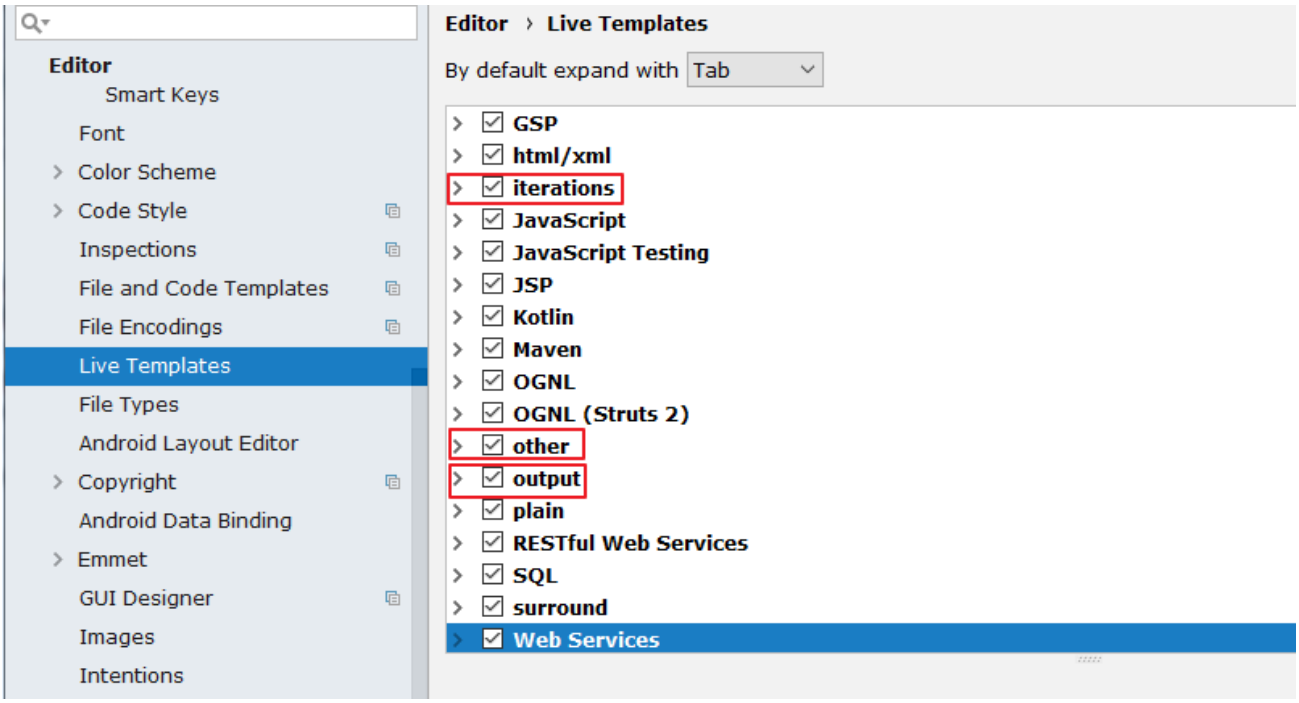
https://www.jetbrains.com/help/idea/using-live-templates.html

2.已有的常用模板

Postfix Completion 默认如下：



Live Templates 默认如下：



二者的区别：Live Templates 可以自定义，而 Postfix Completion 不可以。同时，有些操作二者都提供了模板，Postfix Templates 较 Live Templates 能快 0.01 秒

举例：

2.1 psvm： 可生成 main 方法

2.2 sout : System.out.println() 快捷输出

类似的：

soutp=System.out.println("方法形参名 = " + 形参名);

soutv=System.out.println("变量名 = " + 变量);

soutm=System.out.println("当前类名.当前方法");

“abc”.sout => System.out.println("abc");

2.3 fori： 可生成 for 循环

类似的：

iter：可生成增强 for 循环

itar：可生成普通 for 循环

2.4 list.for：可生成集合 list 的 for 循环

List<String> list = new ArrayList<String>();

输入: list.for 即可输出

for(String s:list){

}

又如：list.fori 或 list.forr

2.5 ifn：可生成 if(xxx = null)

类似的：

inn：可生成 if(xxx != null) 或 xxx.nn 或 xxx.null

2.6 prsf：可生成 private static final

类似的：

psf：可生成 public static final

psfi：可生成 public static final int

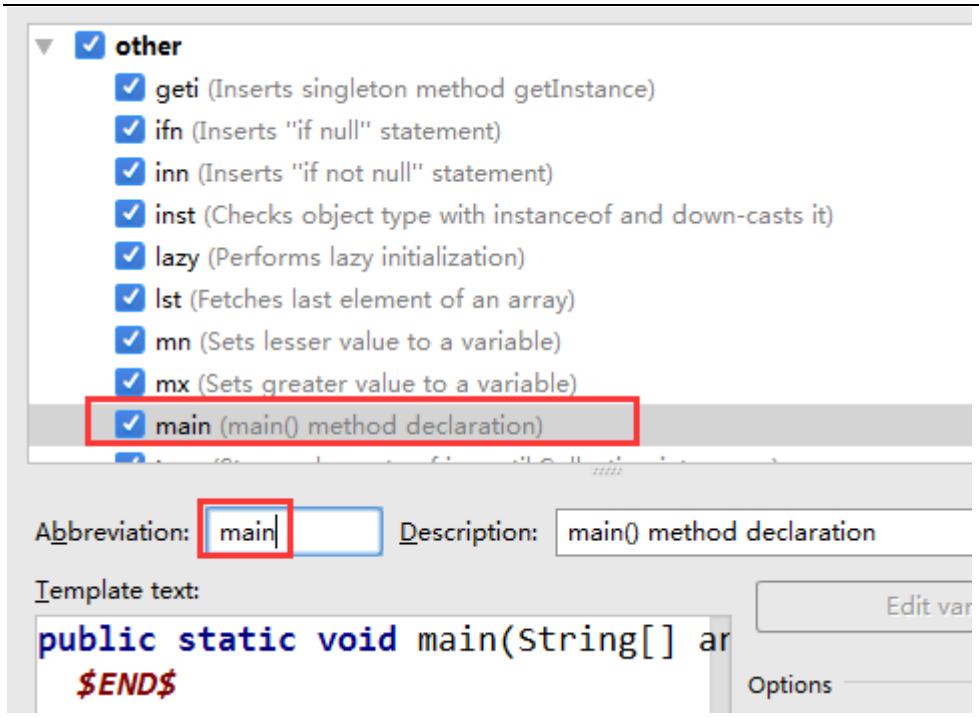
psfs：可生成 public static final String

3.修改现有模板:Live Templates

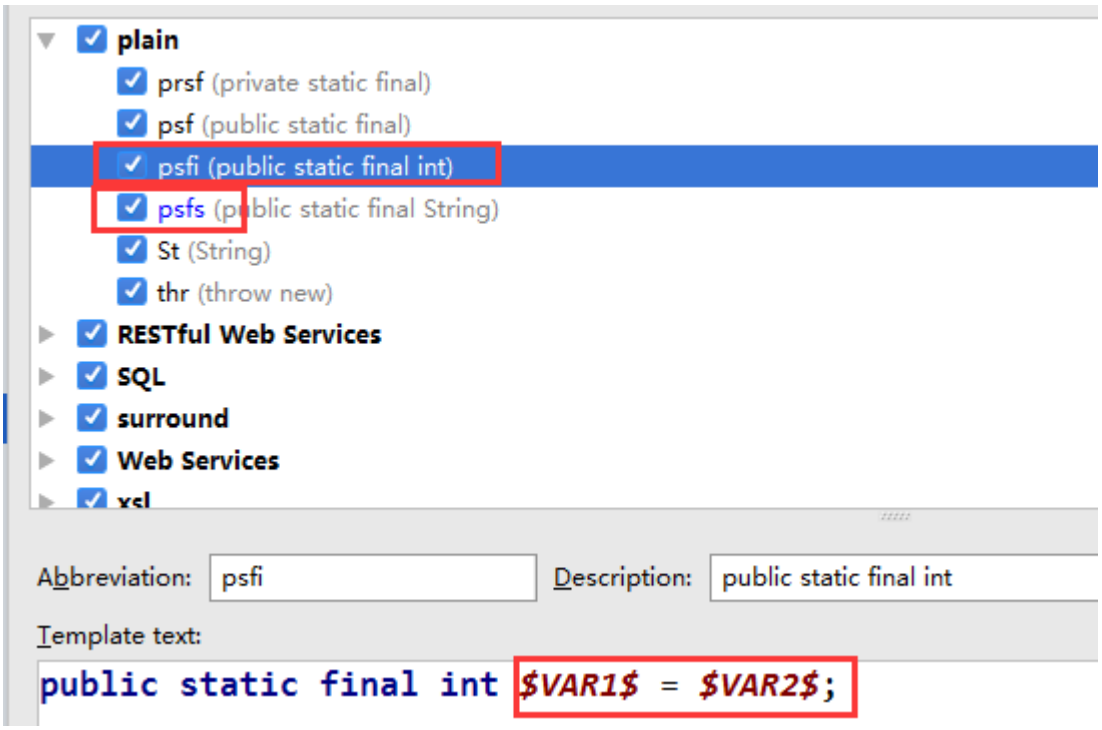
如果对于现有的模板，感觉不习惯、不适应的，可以修改：

修改 1：

通过调用 psvm 调用 main 方法不习惯，可以改为跟 Eclipse 一样，使用 main 调取。



修改 2：

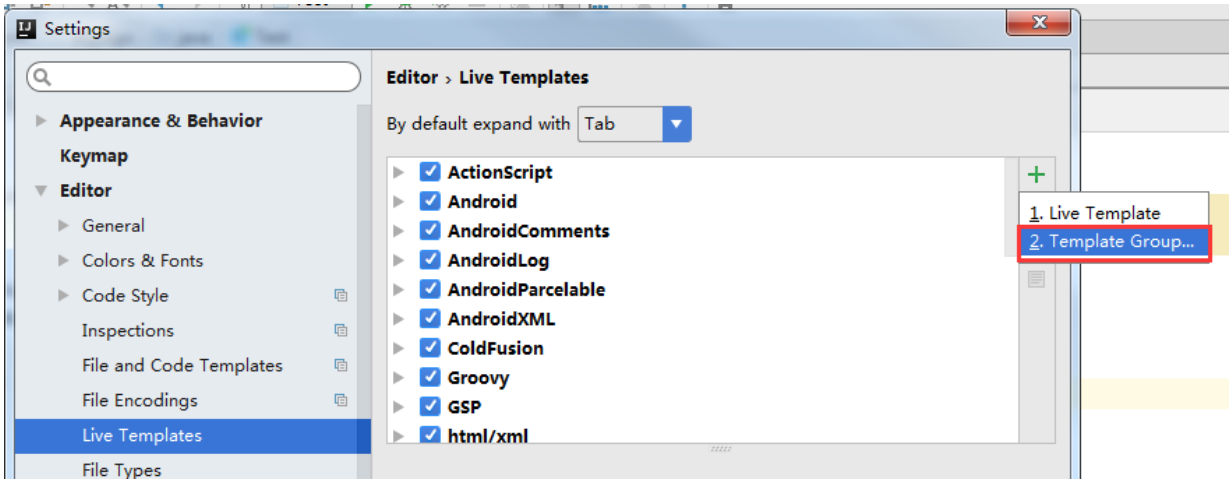


类似的还可以修改 psfs。

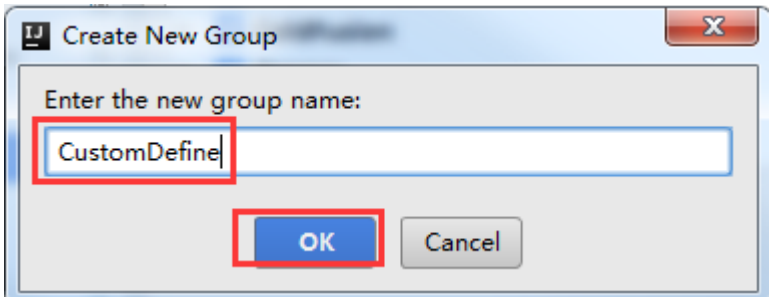
4.自定义模板

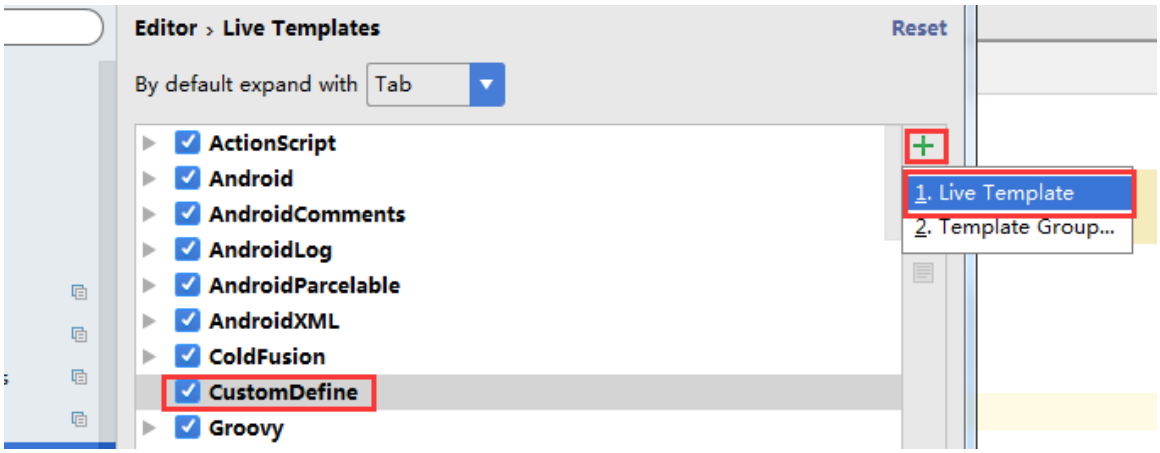
IDEA 提供了很多现成的 Templates。但你也可以根据自己的需要创建新的

Template。

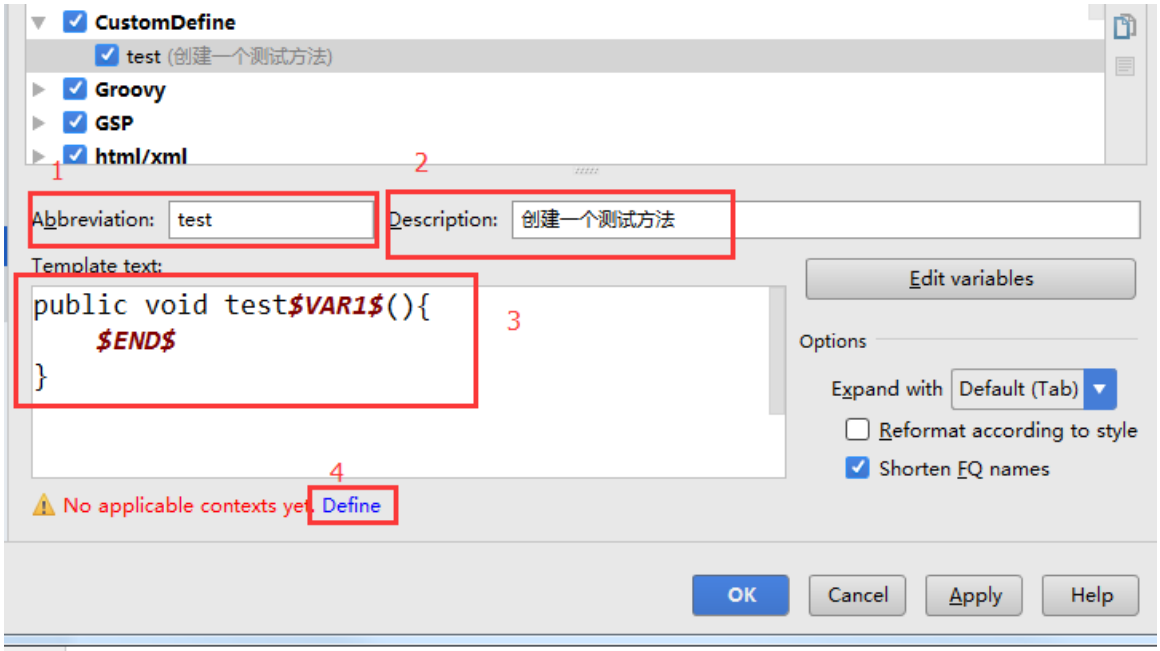


先定义一个模板的组：

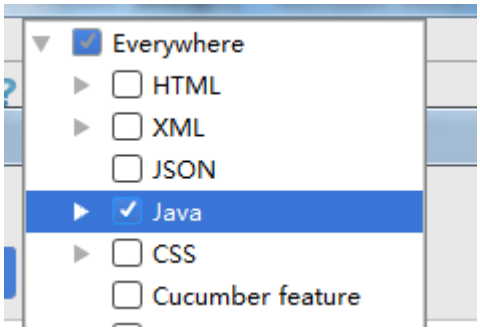




选中自定义的模板组，点击”+”来定义模板。



1. Abbreviation:模板的缩略名称
2. Description:模板的描述
3. Template text:模板的代码片段
4. 应用范围。比如点击 Define。选择如下：



可以如上的方式定义个测试方法，然后在 java 类文件中测试即可。

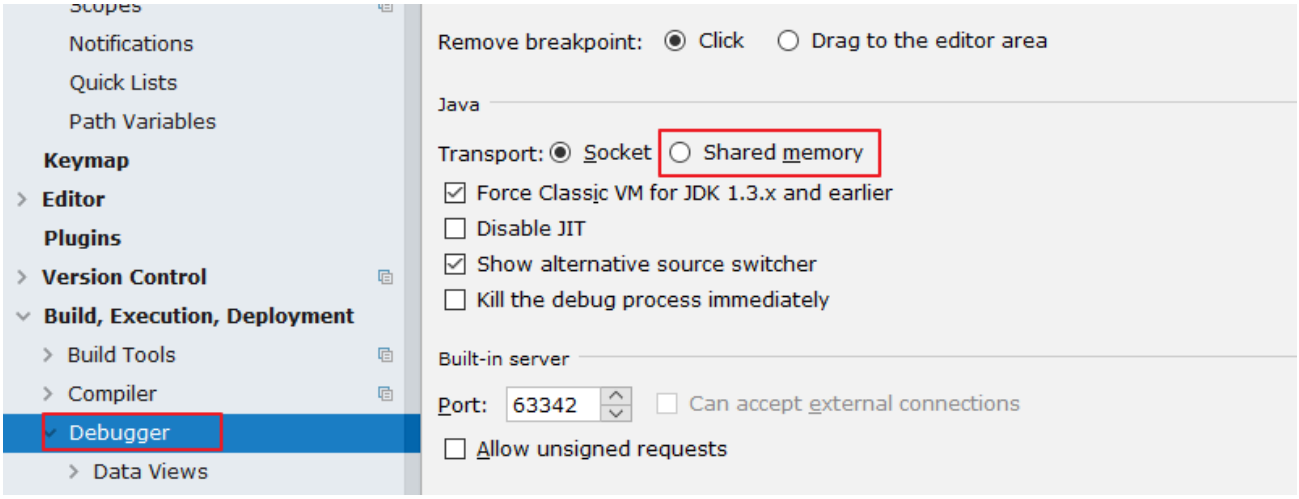
类似的可以再配置如下的几个 Template:

1.

2.

八、断点调试

1. Debug 的设置



设置 Debug 连接方式，默认是 Socket。Shared memory 是 Windows 特有的一个属性，一般在 Windows 系统下建议使用此设置，内存占用相对较少。

2. 常用断点调试快捷键

step over	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
step into	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

force step into	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
step out	跳出
resume program	恢复程序运行，但如果该断点下面代码还有断点则停在下一个断点上
stop	停止
mute breakpoints	点中，使得所有的断点失效
view breakpoints	查看所有断点

对于常用的 Debug 的快捷键，需要大家熟练掌握。

3. 条件断点

说明：

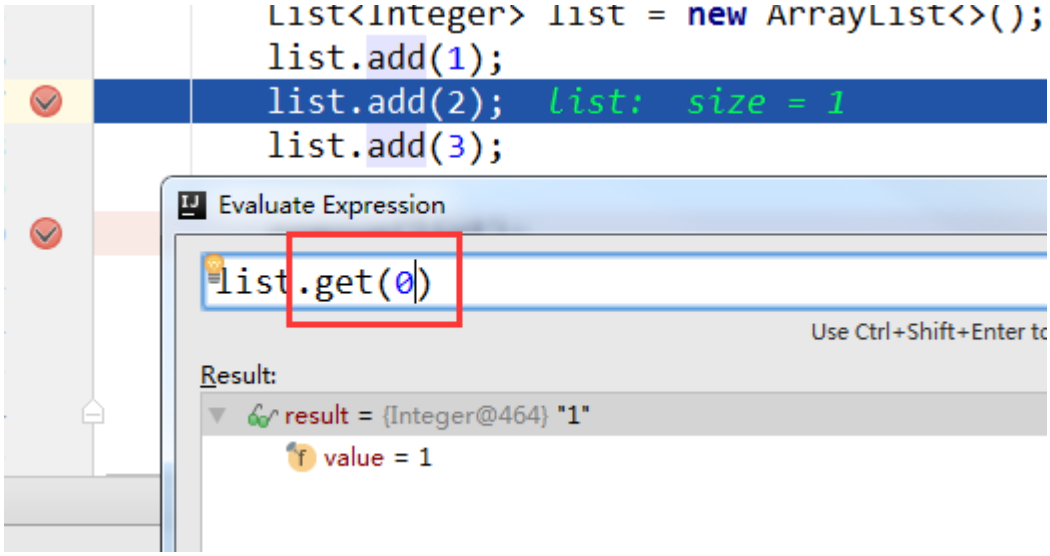
调试的时候，在循环里增加条件判断，可以极大的提高效率，心情也能愉悦。

具体操作：

在断点处右击调出条件断点。可以在满足某个条件下，实施断点。

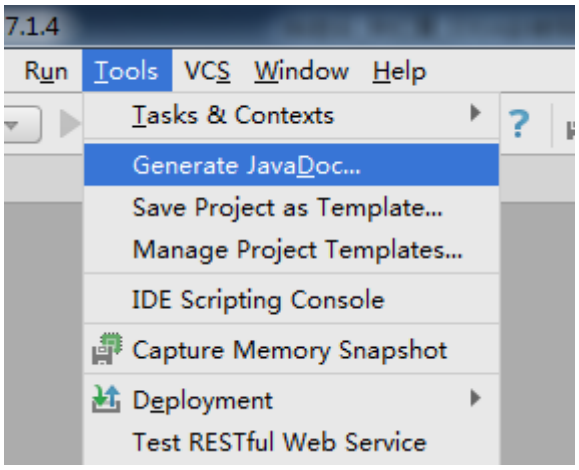
查看表达式的值(Ctrl + u)：

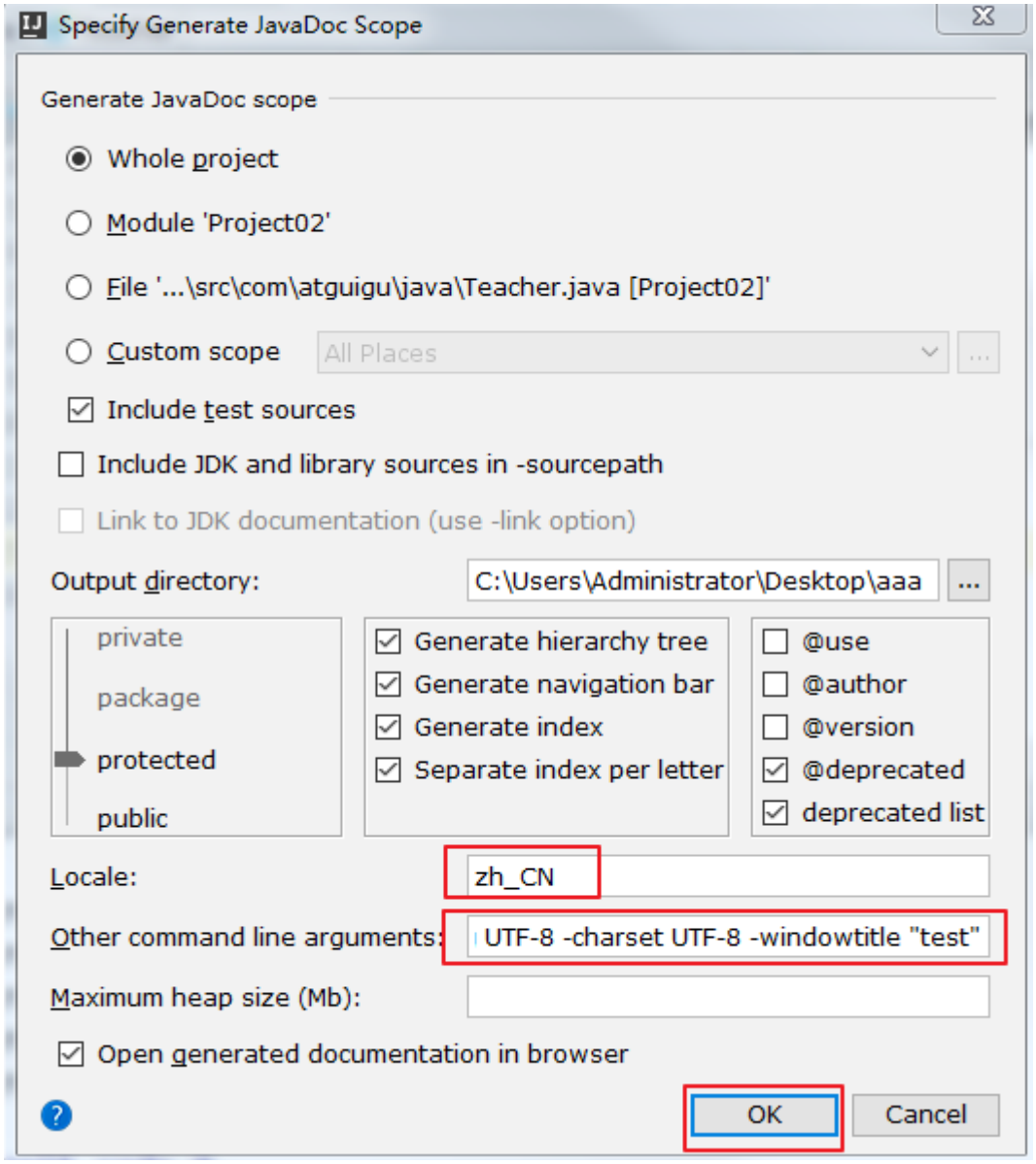
选择行，ctrl + u。还可以在查看框中输入编写代码时的其他方法：



九、其它设置

1.生成 javadoc





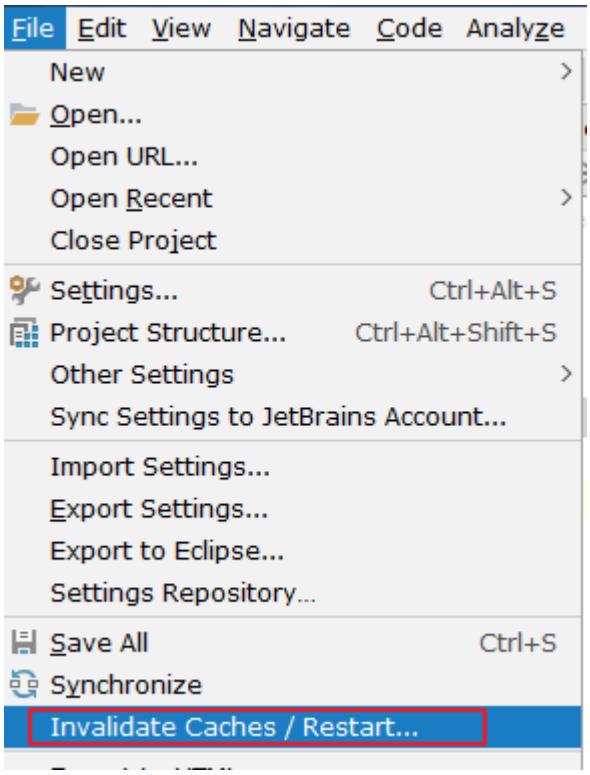
输入：

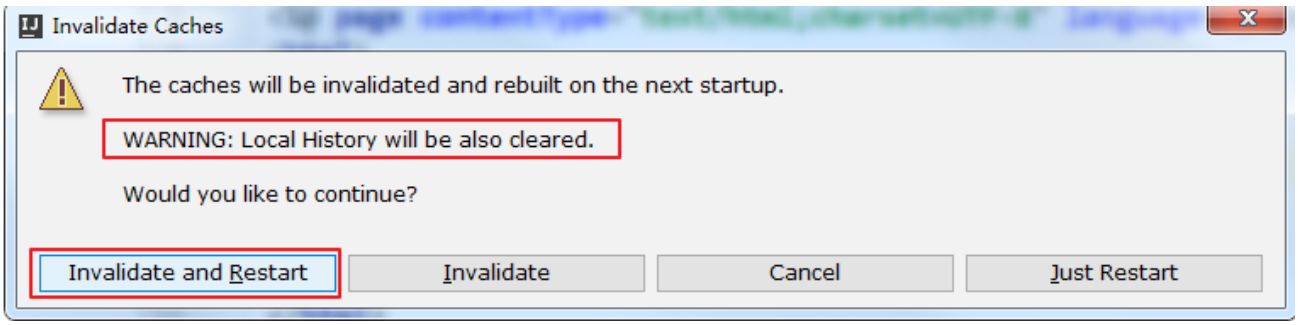
Locale：输入语言类型：zh_CN
Other command line arguments： -encoding UTF-8 -charset UTF-8

2. 缓存和索引的清理

IntelliJ IDEA 首次加载项目的时候，都会创建索引，而创建索引的时间跟项目的文件多少成正比。在 IntelliJ IDEA 创建索引过程中即使你编辑了代码也是编译不了、运行不起来的，所以还是安安静静等 IntelliJ IDEA 创建索引完成。

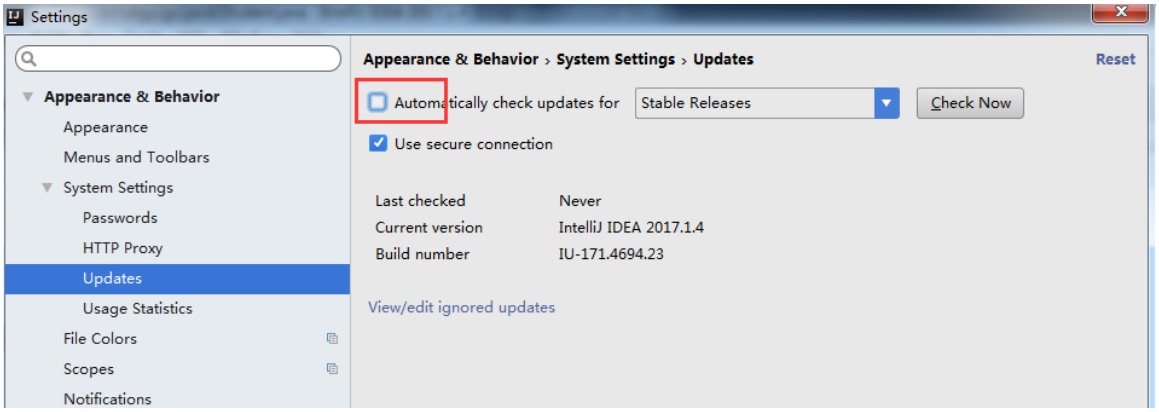
IntelliJ IDEA 的缓存和索引主要是用来加快文件查询，从而加快各种查找、代码提示等操作的速度，所以 IntelliJ IDEA 的索引的重要性再强调一次也不为过。但是，IntelliJ IDEA 的索引和缓存并不是一直会良好地支持 IntelliJ IDEA 的，某些特殊条件下，IntelliJ IDEA 的缓存和索引文件也是会损坏的，比如：断电、蓝屏引起的强制关机，当你重新打开 IntelliJ IDEA，很可能 IntelliJ IDEA 会报各种莫名其妙错误，甚至项目打不开，IntelliJ IDEA 主题还原成默认状态。即使没有断电、蓝屏，也会有莫名奇怪的问题的时候，也很有可能是 IntelliJ IDEA 缓存和索引出现了问题，这种情况还不少。遇到此类问题也不用过多担心。我们可以清理缓存和索引。如下：





- 一般建议点击 Invalidate and Restart，这样会比较干净。
- 上图警告：清除索引和缓存会使得 IntelliJ IDEA 的 Local History 丢失。所以如果你项目没有加入到版本控制，而你又需要你项目文件的历史更改记录，那你最好备份下你的 LocalHistory 目录。目录地址在：C:\Users\当前登录的系统用户名\.IntelliJIdea14\system\LocalHistory 建议使用硬盘的全文搜索，这样效率更高。
- 通过上面方式清除缓存、索引本质也就是去删除 C 盘下的 system 目录下的对应的文件而已，所以如果你不用上述方法也可以删除整个 system。当 IntelliJ IDEA 再次启动项目的时候会重新创建新的 system 目录以及对应项目缓存和索引。

3.取消更新

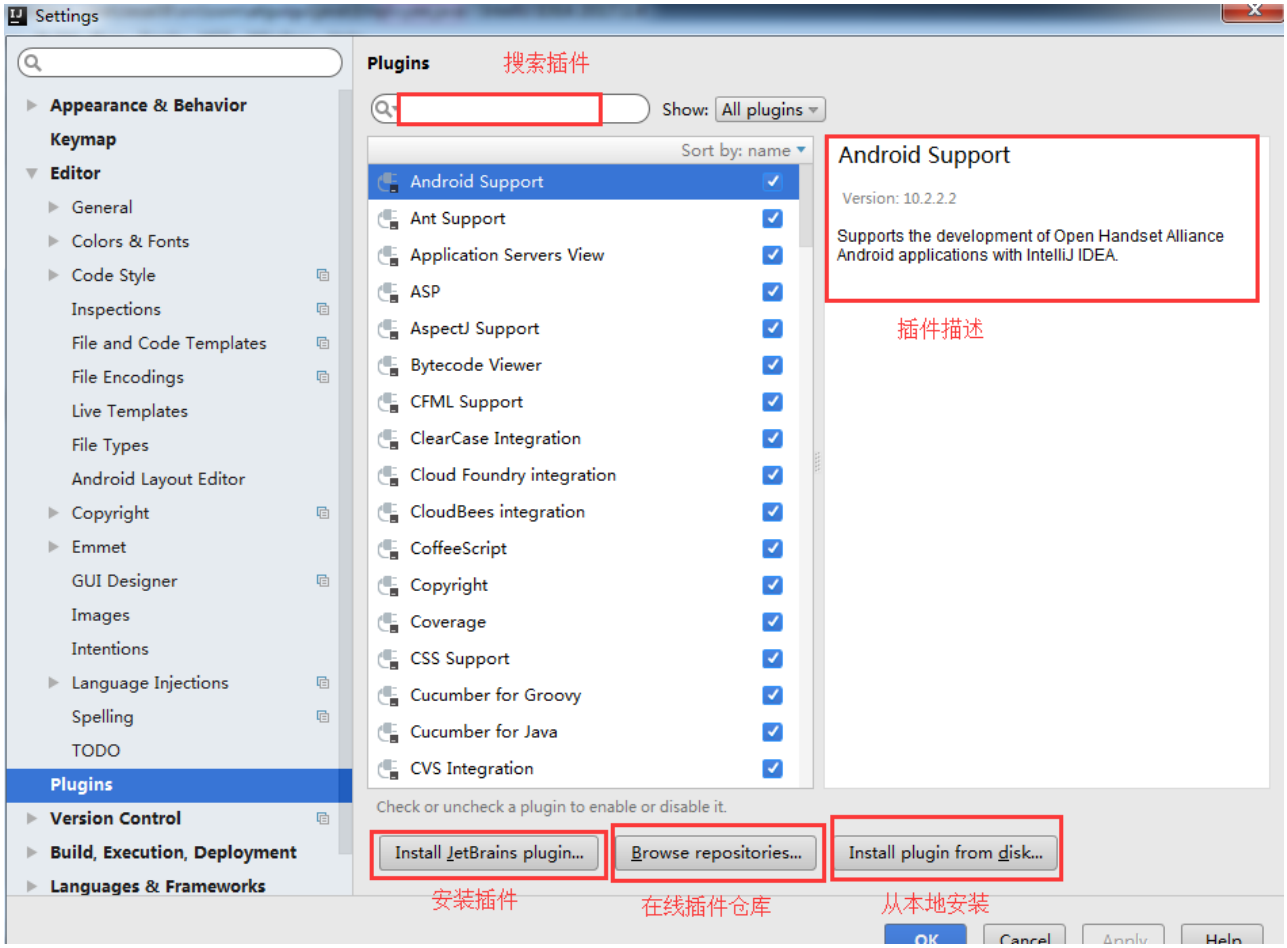


取消勾选：即可取消更新

4.插件的使用

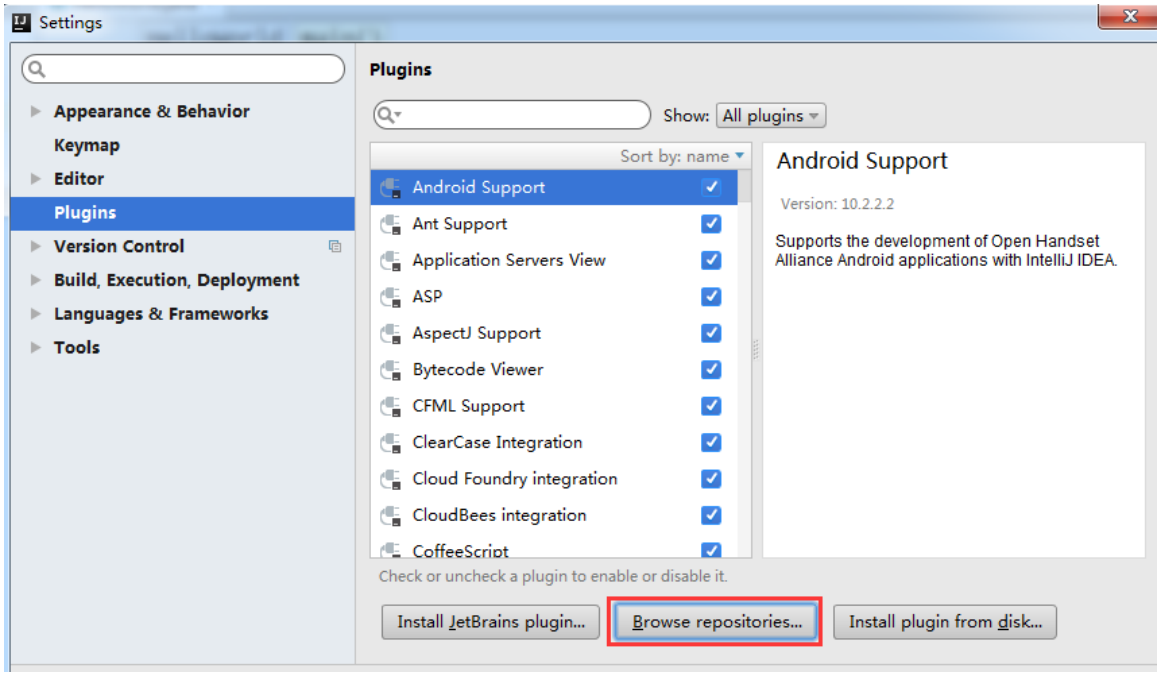
在 IntelliJ IDEA 的安装讲解中我们其实已经知道，IntelliJ IDEA 本身很多功能也都是通过插件的方式来实现的。

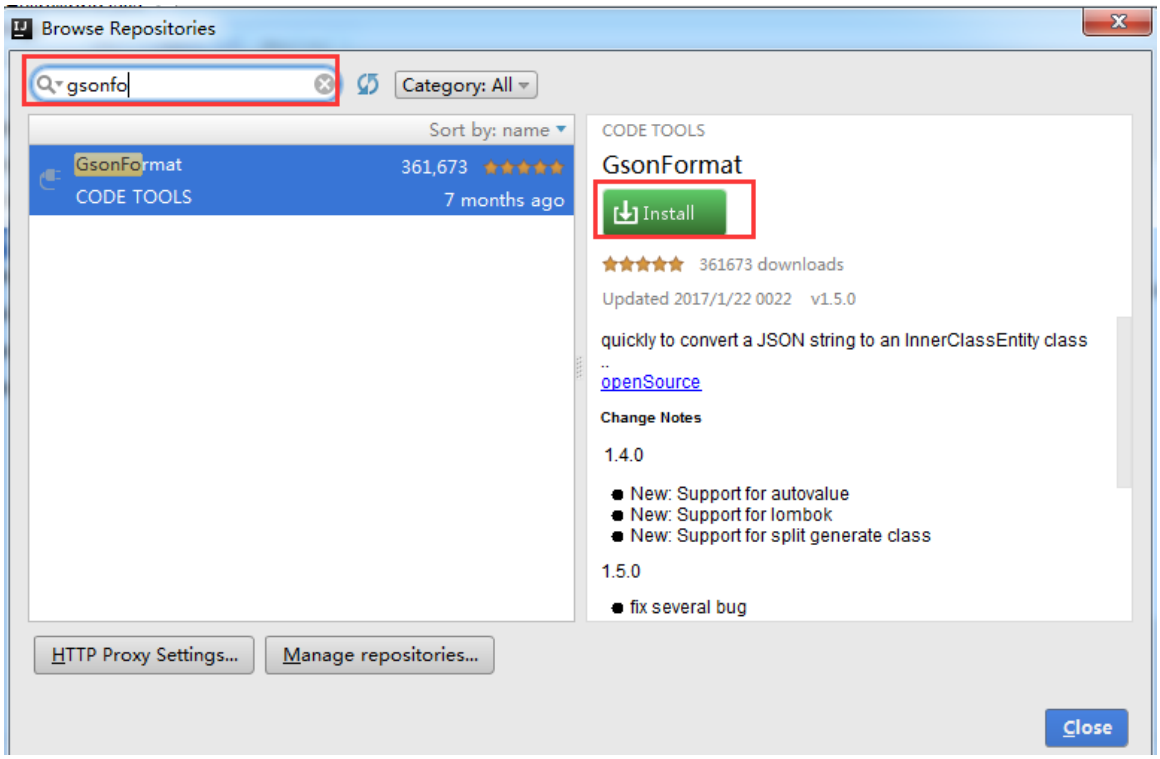
官网插件库：https://plugins.jetbrains.com/



- Install JetBrains plugin：弹出 IntelliJ IDEA 公司自行开发的插件仓库列表，供下载安装。
- Browse repositories：弹出插件仓库中所有插件列表供下载安装。
- Install plugin from disk：浏览本地的插件文件进行安装，而不是从服务器上下载并安装。

需要特别注意的是：在国内的网络下，经常出现显示不了插件列表，或是显示了插件列表，无法下载完成安装。这时候请自行打开 VPN，一般都可以得到解决。





如上图演示，在线安装 IntelliJ IDEA 插件库中的插件。安装完以后会提示重启，才可以使用插件。

常用插件推荐：

插件名称	插件介绍	官网地址

插件名称	插件介绍	官网地址
Key promoter	快捷键提示	https://plugins.jetbrains.com/plugin/4455?pr=idea
CamelCase	驼峰式命名和下划线命名交替变化	https://plugins.jetbrains.com/plugin/7160?pr=idea
CheckStyle-IDEA	代码样式检查	https://plugins.jetbrains.com/plugin/1065?pr=idea
FindBugs-IDEA	代码 Bug 检查	https://plugins.jetbrains.com/plugin/3847?pr=idea
Statistic	代码统计	https://plugins.jetbrains.com/plugin/4509?pr=idea
JRebel Plugin	热部署	https://plugins.jetbrains.com/plugin/?id=4441
CodeGlance	在编辑代码最右侧，显示一块代码小地图	https://plugins.jetbrains.com/plugin/7275?pr=idea
Eclipse Code Formatter	使用 Eclipse 的代码格式化风格，在一个团队中如果公司有规定格式化风格，这个可以使用。	https://plugins.jetbrains.com/plugin/6546?pr=idea
GsonFormat	把 JSON 字符串直接实例化成类	https://plugins.jetbrains.com/plugin/7654?pr=idea