

环境搭建

一、安装 scala

1. scala 的安装与其它编程语言相比有一些特殊：scala 并不是安装在系统里，而是安装在你的每一个 scala project 中。

2. 安装步骤：

- 首先确保安装了 Java 8 JDK：通过命令 `java -version` 查看。

```
huaxz@huaxz-PC:~$ java -version
openjdk version "1.8.0_191"
OpenJDK Runtime Environment (build 1.8.0_191-8u191-b12-2ubuntu0.16.04.1-b12)
OpenJDK 64-Bit Server VM (build 25.191-b12, mixed mode)
```

- 有 scala+IntelliJ 和 scala+sbt 两种方式。区别在于：
 - scala+IntelliJ：集成了 IntelliJ 这个 IDE，对新手友好。
 - scala+sbt：在命令行中执行 scala，以及使用命令行来利用 sbt 编译。当前 (2019-01-09) 最新版本 sbt 1.2.8。

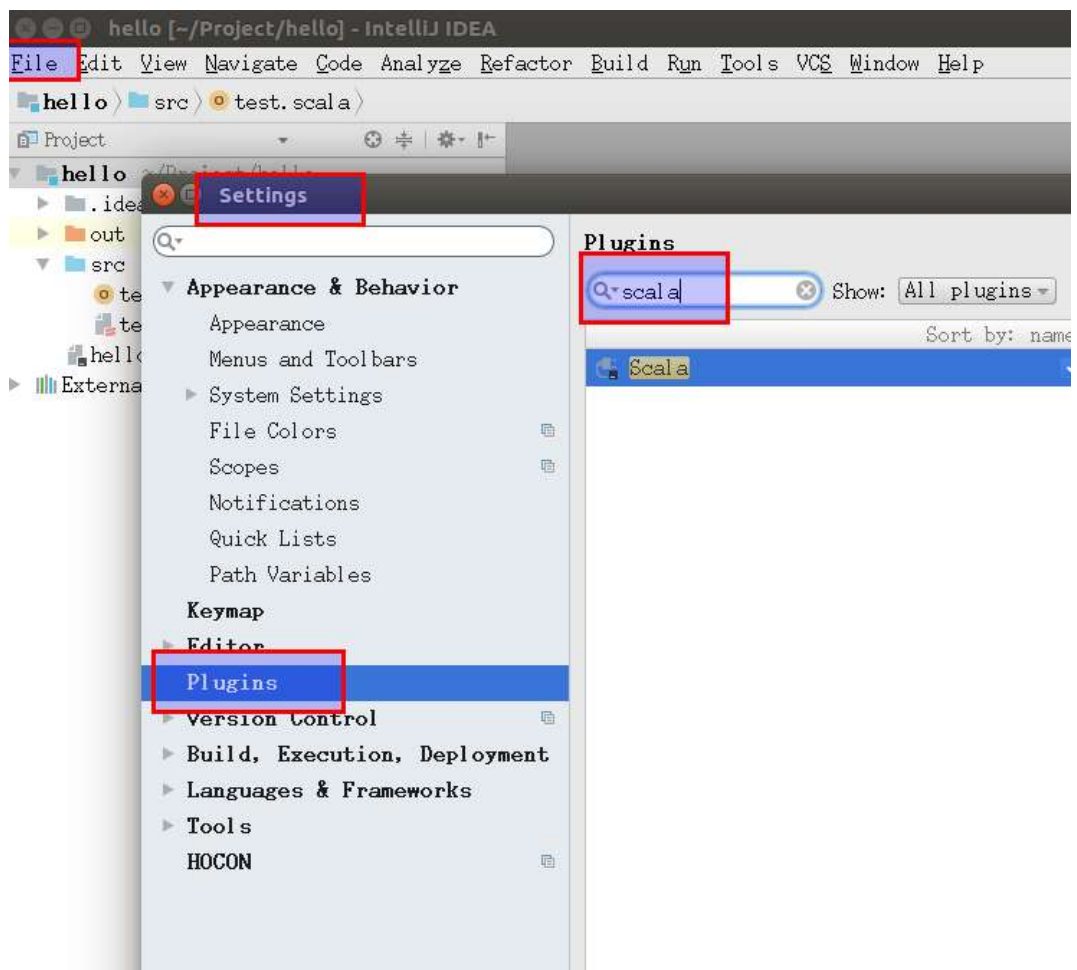
3. scala+sbt 安装方式：执行脚本（参考 <https://www.scala-lang.org/download/>）

```
echo "deb https://dl.bintray.com/sbt/debian /" | sudo tee -a \
/etc/apt/sources.list.d/sbt.list
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv \
2EE0EA64E40A89B84B2DF73499E82A75642AC823
sudo apt-get update
sudo apt-get install sbt
```

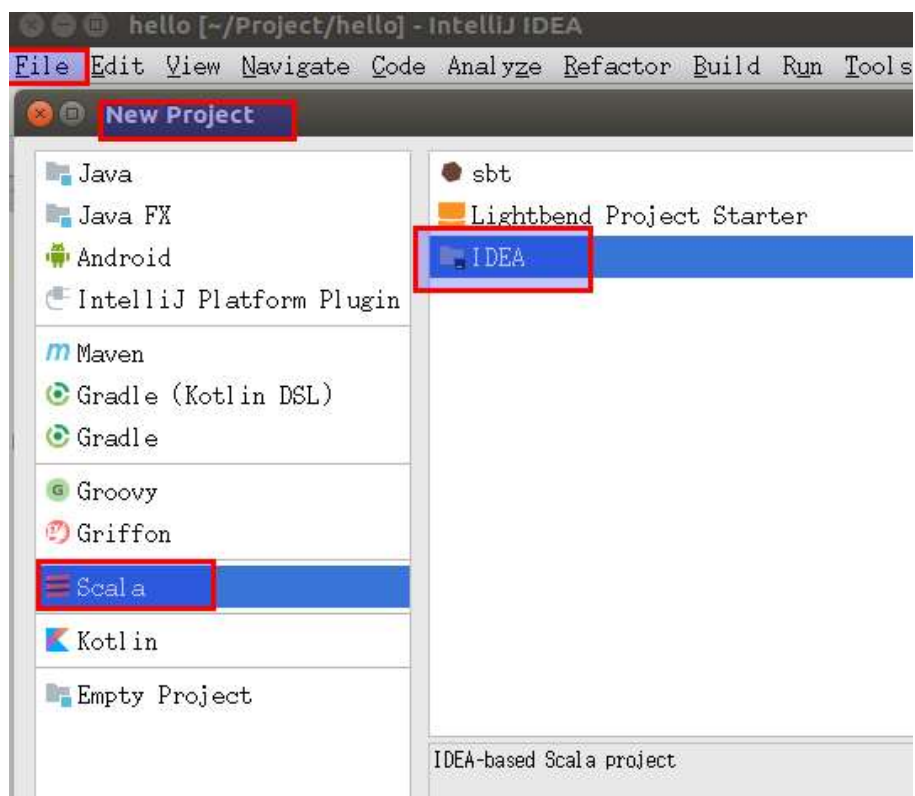
然后执行 sbt 命令，该命令会同步下载一些 jar 包。

4. scala+IntelliJ 安装方式：

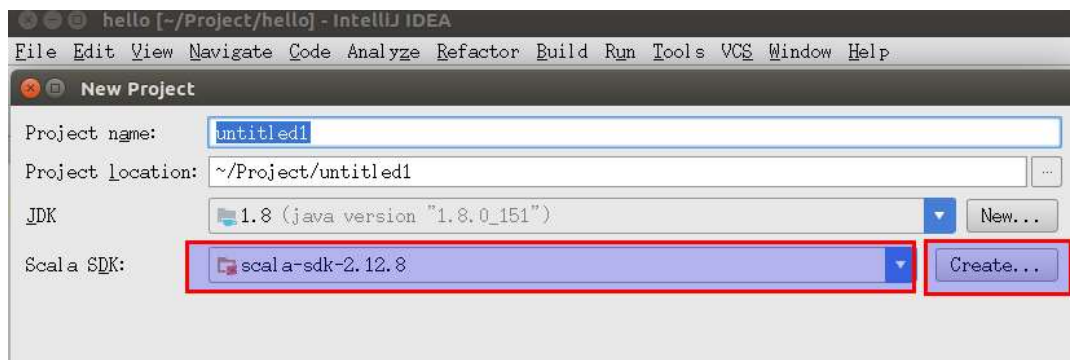
- 下载 IntelliJ Community Edition。
- 安装插件：打开 File->Settings->Plugins，搜索并安装 scala（不需要安装其它，如 SBT）。



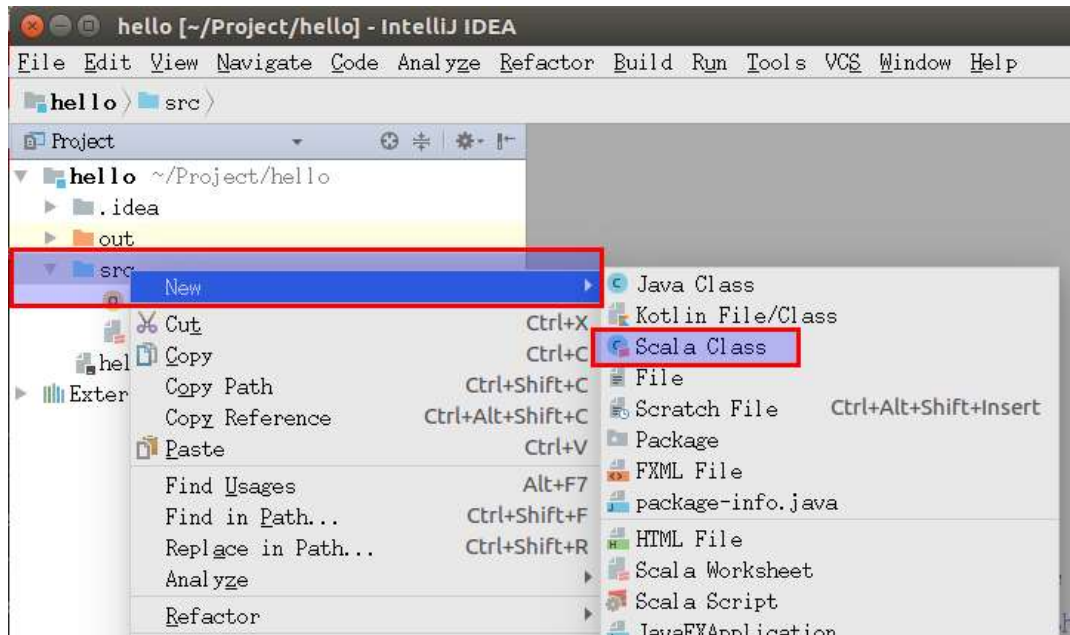
- 新建 Project，选择 scala，在面板右侧选择 IDEA（而不是 sbt 或者 Lightbend Project Starter）。



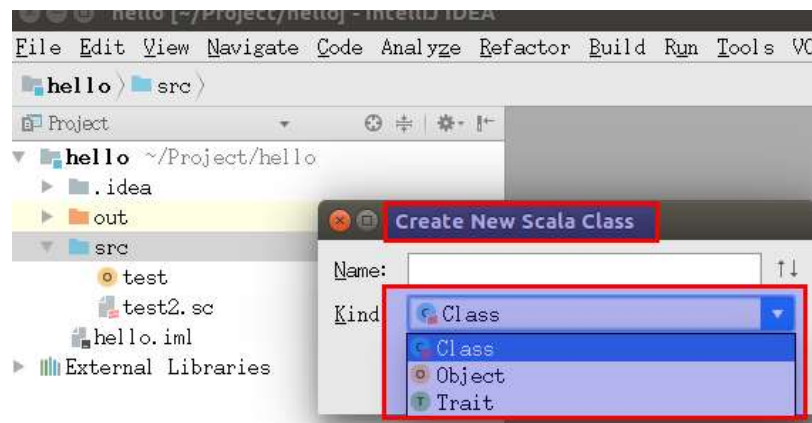
- 选择下一步，在填写 Project name 和 JDK/Scala SDK 的地方。点击 Scala SDK 右侧的 Create，可以选择下载或者切换到指定版本的 Scala SDK。



- o 在 `src` 右键单击，选择 `New->Scala Class`。



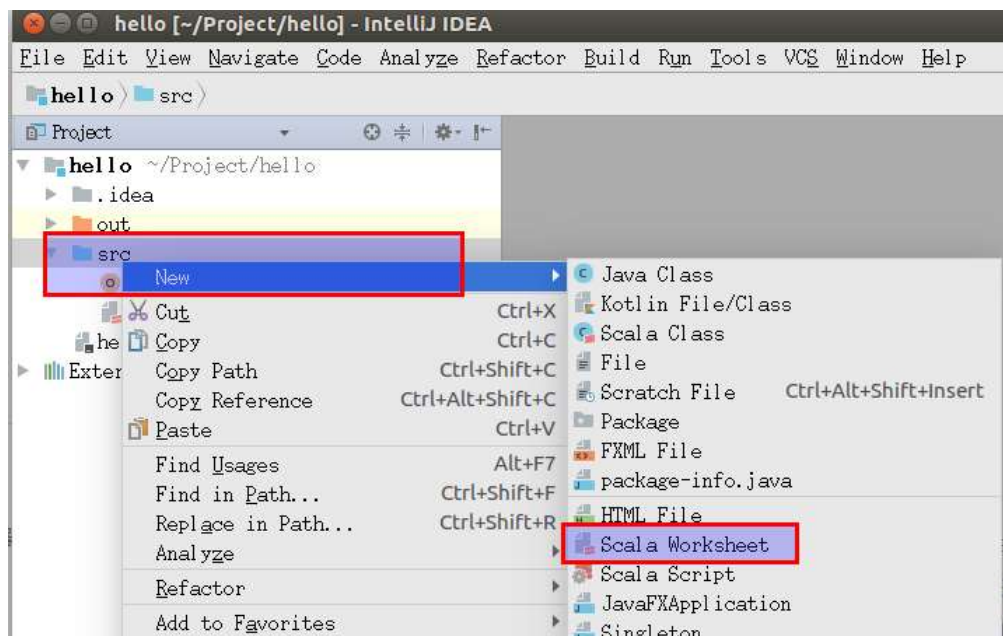
- 如果需要可执行代码，则 `Kind` 选择 `Object`，并新建一个包含 `main()` 方法的 `object`（或者选择 `extends App`）。在编辑界面右键单击，选择 `Run`，则代码编译并执行。



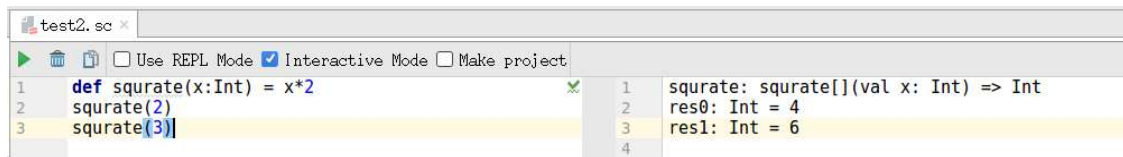
```
object Hello extends App {
  println("Hello, world!")
}
```

- 如果需要创建 `package`、`Class`，则选择常规方式。

5. 如果希望快速验证某些代码片断，则可以创建 `Scala worksheet`。方法是：在 `src` 右键单击，选择 `New->Scala worksheet`。



此时当你在编辑器修改了代码之后，代码立即会被求值并展示在编辑器右侧。



二、安装 Jupyter 支持

1. 通过 almond 安装 scala 的 jupyter 支持:

- o 下载 coursier :

```
curl -L -o coursier https://git.io/coursier && chmod +x coursier \
&& ./coursier --help
```

- o 创建一个 launcher :

```
SCALA_VERSION=2.12.8 ALMOND_VERSION=0.2.1 #环境变量
coursier bootstrap \
  -r jitpack \
  -i user -I user:sh.almond:scala-kernel-
api_$SCALA_VERSION:$ALMOND_VERSION \
  sh.almond:scala-kernel_$SCALA_VERSION:$ALMOND_VERSION \
  --sources --default=true \
  -o almond
```

- o 安装 kernel :

```
./almond --install
```

一旦安装完成，则可以安全的删除 launcher。删除方法为: `rm -f almond`。

- o 可以通过命令 `jupyter kernelspec list` 查看支持的 kernel :

```
huaxz@huaxz-PC:~$ jupyter kernelspec list
Available kernels:
scala          /home/huaxz/.local/share/jupyter/kernels/scala
pyspark3kernel /usr/local/share/jupyter/kernels/pyspark3kernel
pysparkkernel  /usr/local/share/jupyter/kernels/pysparkkernel
python3        /usr/local/share/jupyter/kernels/python3
sparkkernel    /usr/local/share/jupyter/kernels/sparkkernel
sparkrkernel   /usr/local/share/jupyter/kernels/sparkrkernel
```

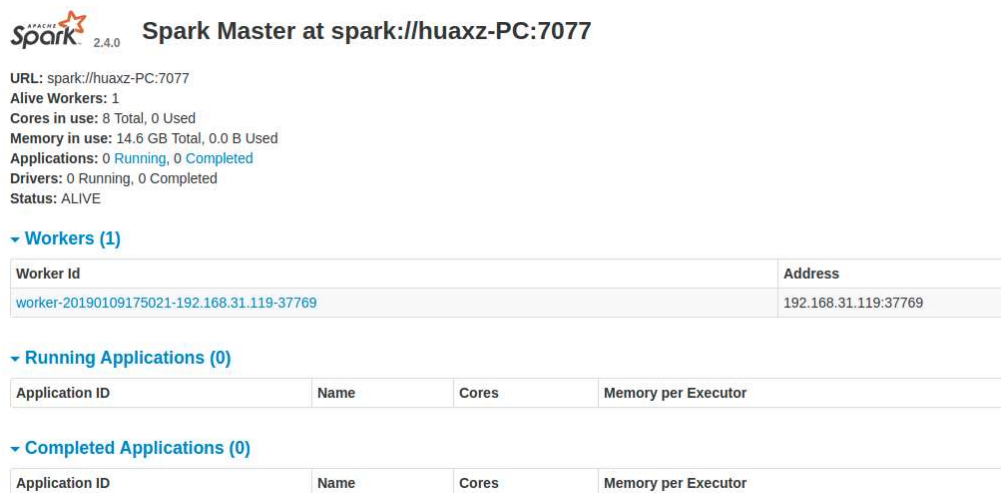
2. 可以通过 <https://github.com/jupyter/jupyter/wiki/Jupyter-kernels> 查看 jupyter 官方支持的 kernel 列表。

3. 通过 sparkmagic 安装 spark 的 jupyter 支持:

- 启用 spark (单机伪分布式) :

- 下载 spark : <http://spark.apache.org/downloads.html>
- 解压缩文件, 进入 spark 目录下的 sbin, 执行 start-all.sh。如果希望停止, 则执行 stop-all.sh。

默认的 spark ui : <http://127.0.0.1:8080/>。



Spark Master at spark://huaxz-PC:7077

URL: spark://huaxz-PC:7077
 Alive Workers: 1
 Cores in use: 8 Total, 0 Used
 Memory in use: 14.6 GB Total, 0.0 B Used
 Applications: 0 Running, 0 Completed
 Drivers: 0 Running, 0 Completed
 Status: ALIVE

▼ Workers (1)

Worker Id	Address
worker-20190109175021-192.168.31.119-37769	192.168.31.119:37769

▼ Running Applications (0)

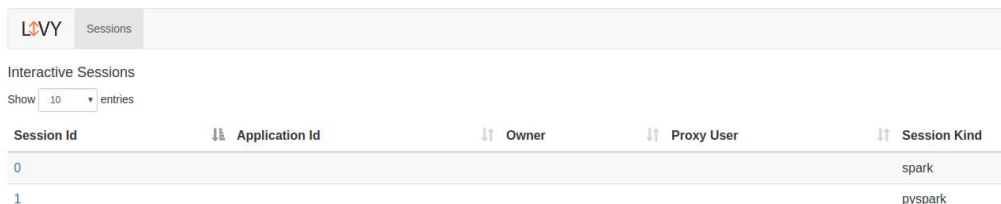
Application ID	Name	Cores	Memory per Executor
----------------	------	-------	---------------------

▼ Completed Applications (0)

Application ID	Name	Cores	Memory per Executor
----------------	------	-------	---------------------

- 安装 livy 依赖:

- 下载 livy : <https://livy.incubator.apache.org/download/>。
- 解压缩 livy, 进入 livy 文件夹, 然后运行 bin/livy-server。
 - 可以在 livy/conf 中配置 livy-server。
 - 需要 export SPARK_HOME 变量。
 - 默认的 livy ui : <http://localhost:8998>。



LIVY Sessions

Interactive Sessions
 Show 10 entries

Session Id	Application Id	Owner	Proxy User	Session Kind
0				spark
1				pyspark

- 安装必要组件:

```
python3.6 -m pip install sparkmagic # 安装 sparkmagic
jupyter nbextension enable --py --sys-prefix widgetsnbextension # 确保 ipywidgets 安装
```

- 通过 `python3.6 -m pip show sparkmagic` 查看 sparkmagic 的安装位置, cd 到该位置。


```

huaxz@huaxz-PC:~$ python3.6 -m pip show sparkmagic
Name: sparkmagic
Version: 0.12.6
Summary: SparkMagic: Spark execution via Livy
Home-page: https://github.com/jupyter-incubator/sparkmagic
Author: Jupyter Development Team
Author-email: jupyter@googlegroups.org
License: BSD 3-clause
Location: /usr/local/lib/python3.6/dist-packages
Requires: hdijupyterutils, autovizwidget, ipython, nose, mock
Required-by:

```

- o 安装一些 kernel (如果某些功能不需要, 则不用安装) :

```

jupyter-kernelspec install sparkmagic/kernels/sparkkernel # scala
spark
jupyter-kernelspec install sparkmagic/kernels/pysparkkernel # pyspark
jupyter-kernelspec install sparkmagic/kernels/pyspark3kernel #
pyspark3
jupyter-kernelspec install sparkmagic/kernels/sparkrkernel # r spark

```

- o 可选: 可以修改 `~/.sparkmagic/config.json` 来修改配置, 其内容参考 https://github.com/jupyter-incubator/sparkmagic/blob/master/sparkmagic/example_config.json 。其中 `kernel_xx_credentials` 中的 `url` 给出了 `livy-server` 的 `host:port` 。

```

1 {
2   "kernel_python_credentials" : {
3     "username": "",
4     "password": "",
5     "url": "http://localhost:8998",
6     "auth": "None"
7   },
8
9   "kernel_scala_credentials" : {
10    "username": "",
11    "password": "",
12    "url": "http://localhost:8998"

```

- o 可选: 启用 `server` 扩展, 从而允许以编程的方式更改集群:

```

jupyter serverextension enable --py sparkmagic

```

- o 通过 `jupyter notebook` 启动 `jupyter`, 新建一个 `pyspark` 页面, 执行 `sc` 或者 `spark`。如果输出正常则安装成功。

```
In [1]: sc
```

Starting Spark application

ID	YARN Application ID	Kind	State	Spark UI	Driver log	Current session?
1	None	pyspark	idle			✓

SparkSession available as 'spark'.
<SparkContext master=local appName=livy-session-1>

```
In [2]: spark
```

<pyspark.sql.session.SparkSession object at 0x7efde3fd4410>

三、Scala 解释器

1. `scala` 解释器：一个编写 `scala` 的交互式 shell。调用方式为：在命令行中执行命令 `scala`。

`scala` 解释器会对你录入的表达式求值，并输出结果。结果格式为：

变量名 : 结果类型 = 结果值

如果表达式中定义了变量，则变量名就是该变量的名字；否则变量名就是 `resX`，你可以在后续表达式中使用该变量名。

- 如果希望在解释器中输入多行代码，只需要按回车。此时解释器会在下一行头部加上竖线 `|`。如果希望退出该模式，则连续按两次回车即可。
 - 退出解释器，则键入 `:q` 或者 `:quit`。
 - 如果希望查看对象包含的方法/属性，则可以输入 `obj.` 然后键入 `Tab`。
2. 如果希望在解释器中引入第三方包的依赖，则可以使用 `sbt` 来管理。
 - 首先创建一个目录，然后在该目录下创建一个名为 `build.sbt` 的文件。
 - 然后去 <https://search.maven.org> 中搜索第三方依赖包的坐标。
 - 修改该文件，在该文件中，你可以指定 `scala` 版本、依赖包：

```
name := "Scala Playground"
version := "1.0"
scalaVersion := "2.10.2" # scala 版本
libraryDependencies += "com.twitter" % "algebird-core_2.10" % "0.13.5"
# 第三方依赖
```

- 在该目录下，执行命令 `sbt console`。

四、IntelliJ 使用

1. `IntelliJ` 用于查看符号的快捷键：
 - `F1`：查看该符号的说明文档（如果有的话）。
 - `⌘Space`：查看该符号的定义代码。
 - `⌘F7`：查看该符号在哪些地方被用到。
 - `⌘B`：查看该符号的实现。
 - `Ctrl+Shift+P`：查看该符号的数据类型。
2. `⌘L`：格式化代码。

3. 如果项目使用了 `Maven`，则有可能 `IntelliJ` 未能实时跟踪到 `Maven` 下载的第三方依赖包，此时 `IntelliJ` 的代码提示找不到对应的符号。

解决方案：执行 `File | Invalidate Caches`。如果未能解决，则删除 `IDEA system directory`（Linux下位于 `~/.<PRODUCT><VERSION>`）。

五、Maven 使用
