

本教程包括以下几个方面：

- 1) java1.8 sdk 和 java11 sdk 安装和配置
- 2) eclipse 和 IntelliJ IDEA 安装和配置-怎么用某个 jdk 版本来编程
- 3) 命令行中 java8 和 java11 的切换配置
- 4) javafx 之于 11 的安装-下载、目录、library 和添加到项目中
- 5) jdk1.8 中运行 javafx; jdk11 中编译运行 javafx

## 安装 jdk1.8 以及 jdk11

此次实验是在 windows 下完成，mac 下的教程链接为：

jdk1.8 的安装与配置：

<https://blog.csdn.net/deliciousion/article/details/78046007>

jdk11 的安装与配置：

[https://blog.csdn.net/weixin\\_42095500/article/details/83576667](https://blog.csdn.net/weixin_42095500/article/details/83576667)

问题一：jdk1.8 的安装与配置

参考链接：<https://blog.csdn.net/xiegongmiao/article/details/81206975>

第一步：下载 jdk1.8，下载地址为

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

先选择 accept，然后根据自己的电脑选择对应版本，例如本机为 window10 的 64 位操作系统，因此选择 jdk-8u131-windows-x64.exe

**Java SE Development Kit 8u131**

You must accept the **Oracle Binary Code License Agreement for Java SE** to download this software.

☒ Accept License Agreement ☐ Decline License Agreement

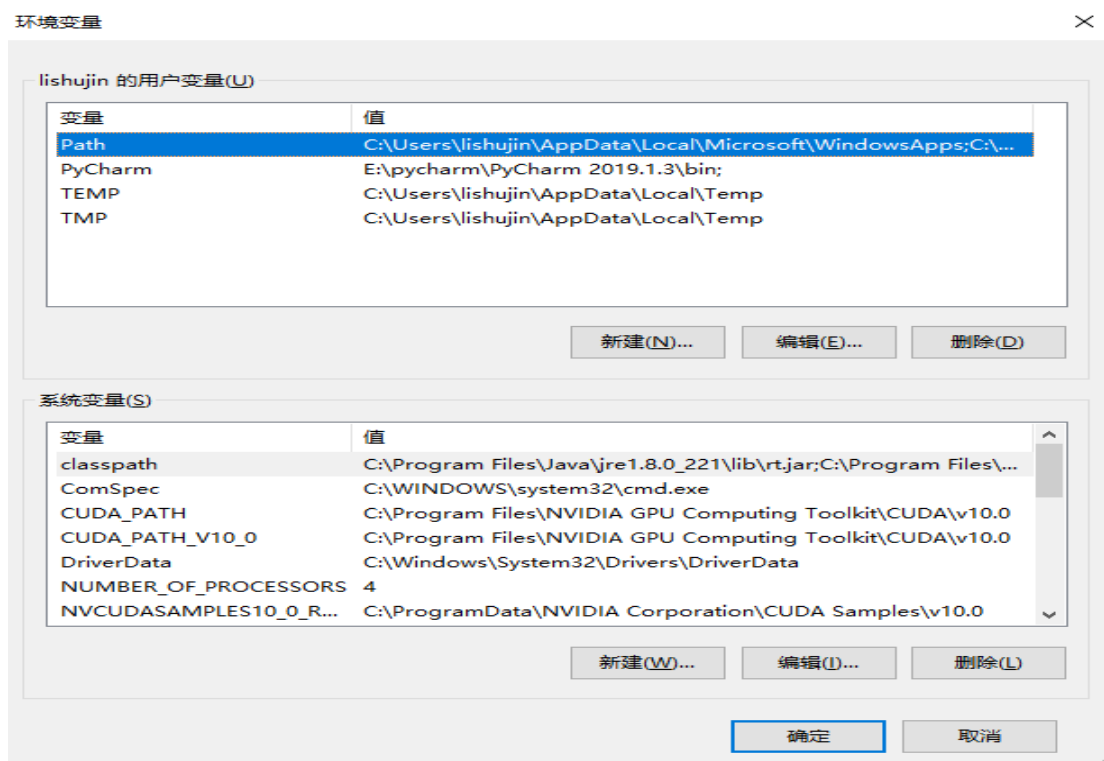
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.87 MB	<a href="#">jdk-8u131-linux-arm32-vfp-hflt.tar.gz</a>
Linux ARM 64 Hard Float ABI	74.81 MB	<a href="#">jdk-8u131-linux-arm64-vfp-hflt.tar.gz</a>
Linux x86	164.66 MB	<a href="#">jdk-8u131-linux-i586.rpm</a>
Linux x86	179.39 MB	<a href="#">jdk-8u131-linux-i586.tar.gz</a>
Linux x64	162.11 MB	<a href="#">jdk-8u131-linux-x64.rpm</a>
Linux x64	176.95 MB	<a href="#">jdk-8u131-linux-x64.tar.gz</a>
Mac OS X	226.57 MB	<a href="#">jdk-8u131-macosx-x64.dmg</a>
Solaris SPARC 64-bit	139.79 MB	<a href="#">jdk-8u131-solaris-sparcv9.tar.Z</a>
Solaris SPARC 64-bit	99.13 MB	<a href="#">jdk-8u131-solaris-sparcv9.tar.gz</a>
Solaris x64	140.51 MB	<a href="#">jdk-8u131-solaris-x64.tar.Z</a>
Solaris x64	96.96 MB	<a href="#">jdk-8u131-solaris-x64.tar.gz</a>
Windows x86	191.22 MB	<a href="#">jdk-8u131-windows-i586.exe</a>
Windows x64	198.03 MB	<a href="#">jdk-8u131-windows-x64.exe</a>

第二步：安装 jdk1.8，可以视情况选择安装路径（请记住安装路径），jdk 和 jre 建议安装在同一路径下。



第三步：配置环境变量，也就是配置 path 以及 classpath。（PS：请记住环境变量没有大小写之分，所以 Path，PATH，path 代表同一个变量，如果你系统已存在 Path 变量，你新建一个 path 变量然后赋予变量值后会覆盖掉原来的 Path 变量，造成未知错误）

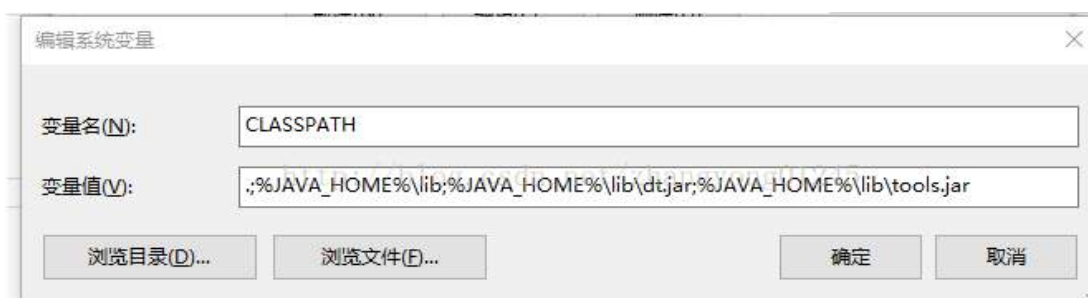
Windows 环境变量获取方式为进入我的电脑→右键→属性→高级系统设置→环境变量，得到以下界面。



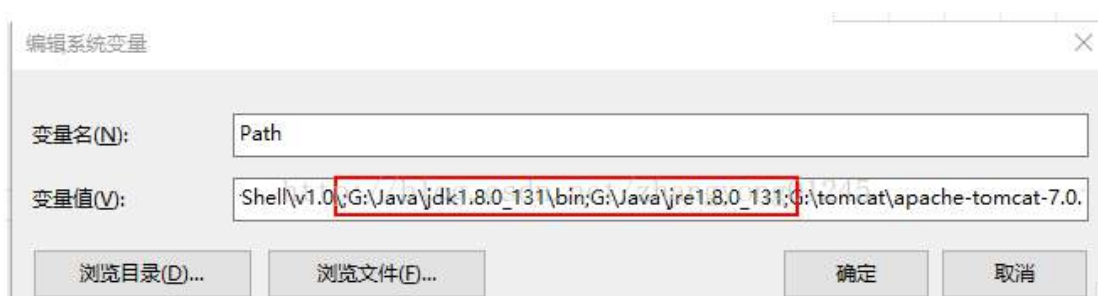
然后新增用户变量 JAVA\_HOME，变量值为你 jdk 的安装路径。



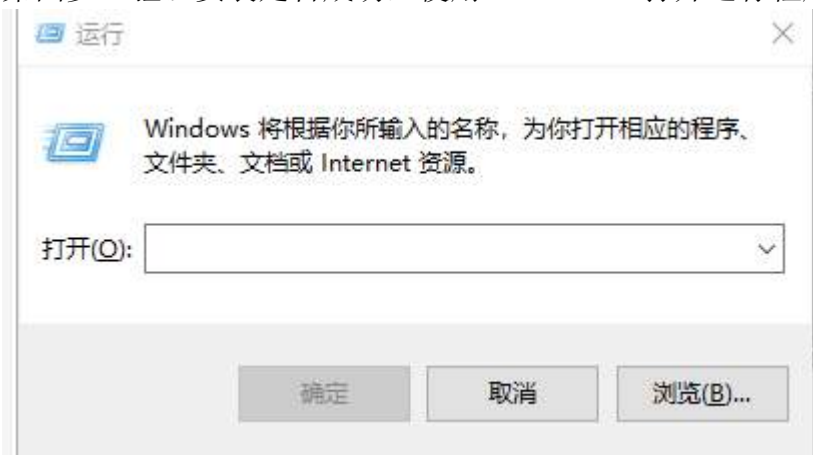
然后新增系统变量 CLASSPATH，变量值为  
.;%JAVA\_HOME%\lib;%JAVA\_HOME%\lib\dt.jar;%JAVA\_HOME%\lib\tools.jar  
(也就是 classpath，不分大小写的)。



然后修改系统变量 Path 为 G:\Java\jdk1.8.0\_131\bin;G:\Java\jre1.8.0\_131;，请注意前后分号；



第四步：验证安装是否成功，使用 windows+r 打开运行程序



输入 cmd，确定后新建一个命令行界面

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.17134.950]
(c) 2018 Microsoft Corporation. 保留所有权利。
C:\Users\lishuiin>
```

输入 `java -version`，显示了你所安装的 java 版本，确定安装成功。

## 问题二：jdk11 安装与配置

参考链接：[https://blog.csdn.net/qq\\_37905269/article/details/87442737](https://blog.csdn.net/qq_37905269/article/details/87442737)

jdk11 的下载与安装等同于 jdk1.8，但是环境变量的配置不一样。

第一步下载 jdk11，下载地址：

<https://www.oracle.com/technetwork/java/javase/downloads/jdk11-downloads-5066655.html>

第二步安装 jdk11，请记住安装地址。

第三步设置环境变量：

右击我的电脑——>属性——>高级——>环境变量

新建用户变量 `JAVA_HOME`，变量值为 jdk 下载路径

在系统变量中找到 `path`，变量值为 `%JAVA_HOME%\bin`；

第四步检查是否安装完成

使用 `cmd` 打开命令提示符，输入 `java -version`，输出其 jdk 信息为 jdk11，即为配置成功。

```
java version "11.0.4" 2019-07-16 LTS
```

## Eclipse 和 IntelliJ IDEA 安装和配置 jdk 版本

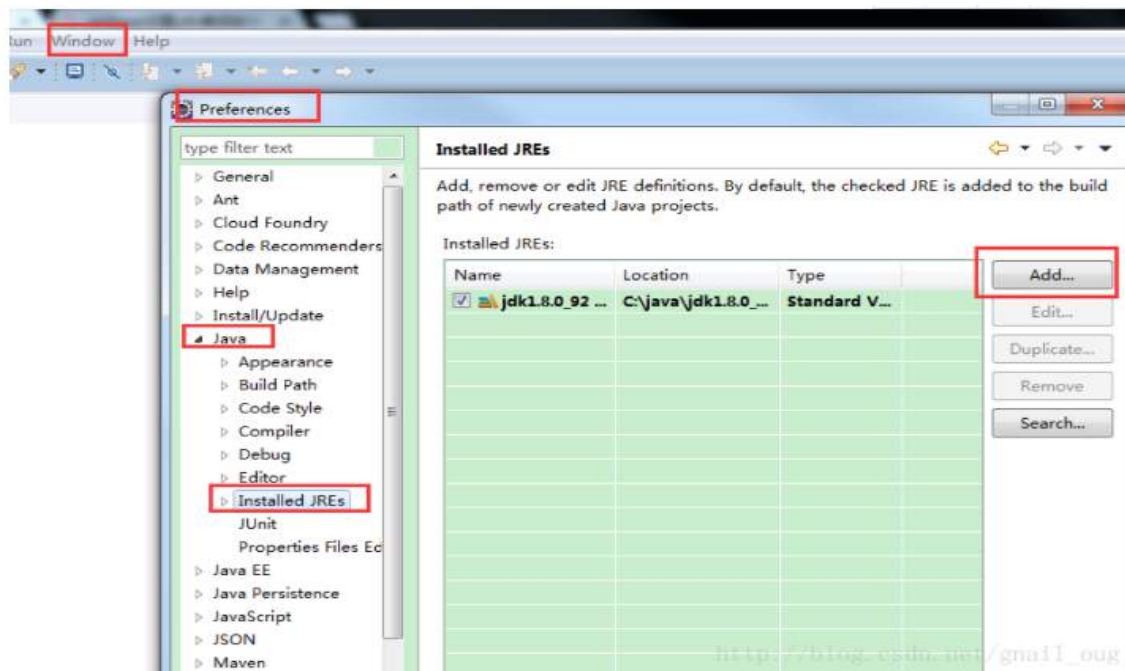
此次实验只安装了 eclipse，使用 IntelliJ IDEA 配置 JDK 可参考链接：

<https://blog.csdn.net/nobb111/article/details/77116259>

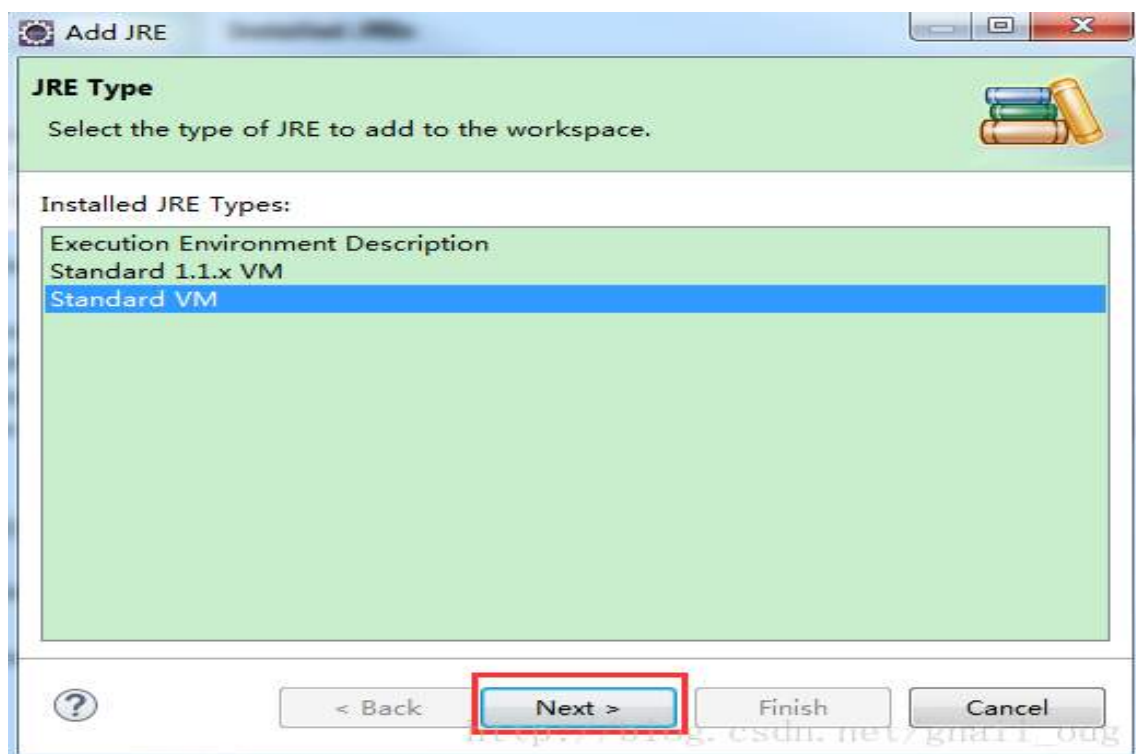
Eclipse 配置 jdk 版本

参考链接：[https://blog.csdn.net/gnail\\_oug/article/details/53610768](https://blog.csdn.net/gnail_oug/article/details/53610768)

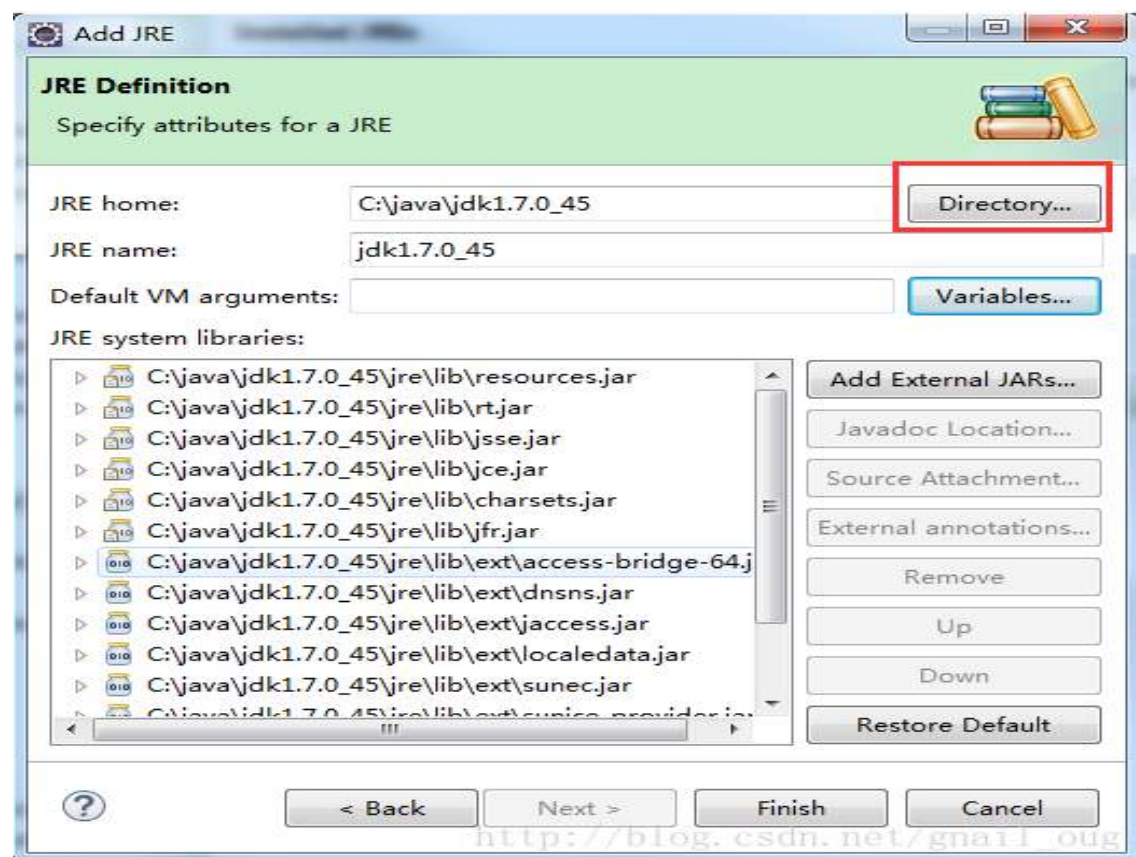
第一步：打开 windows->preferences,弹出页面里选择 java->Installed JREs



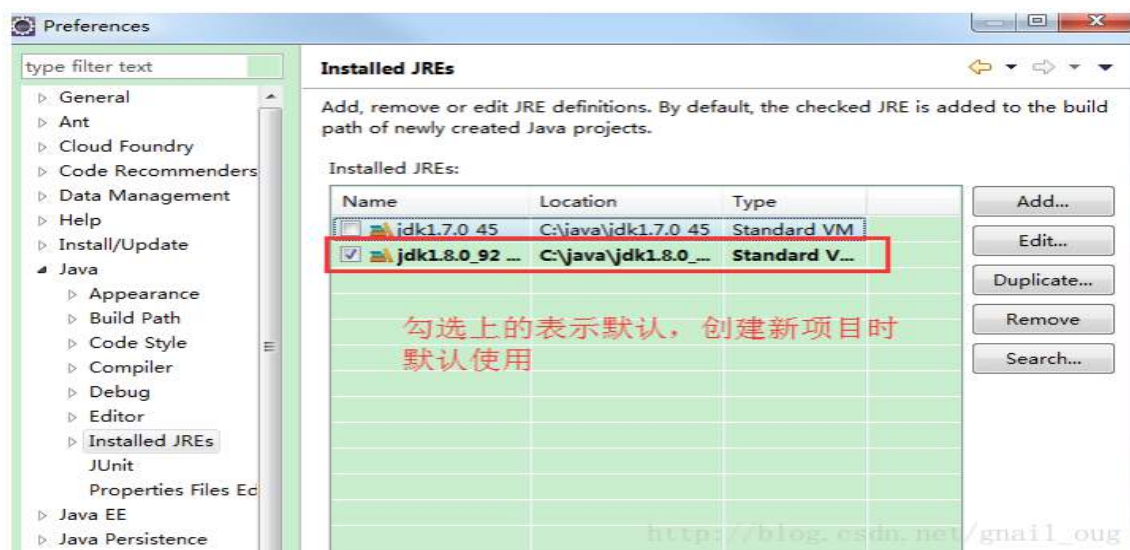
第二步：点击 add 按钮，打开 add jre 对话框



第三步：打开 jre 路径选择对话框

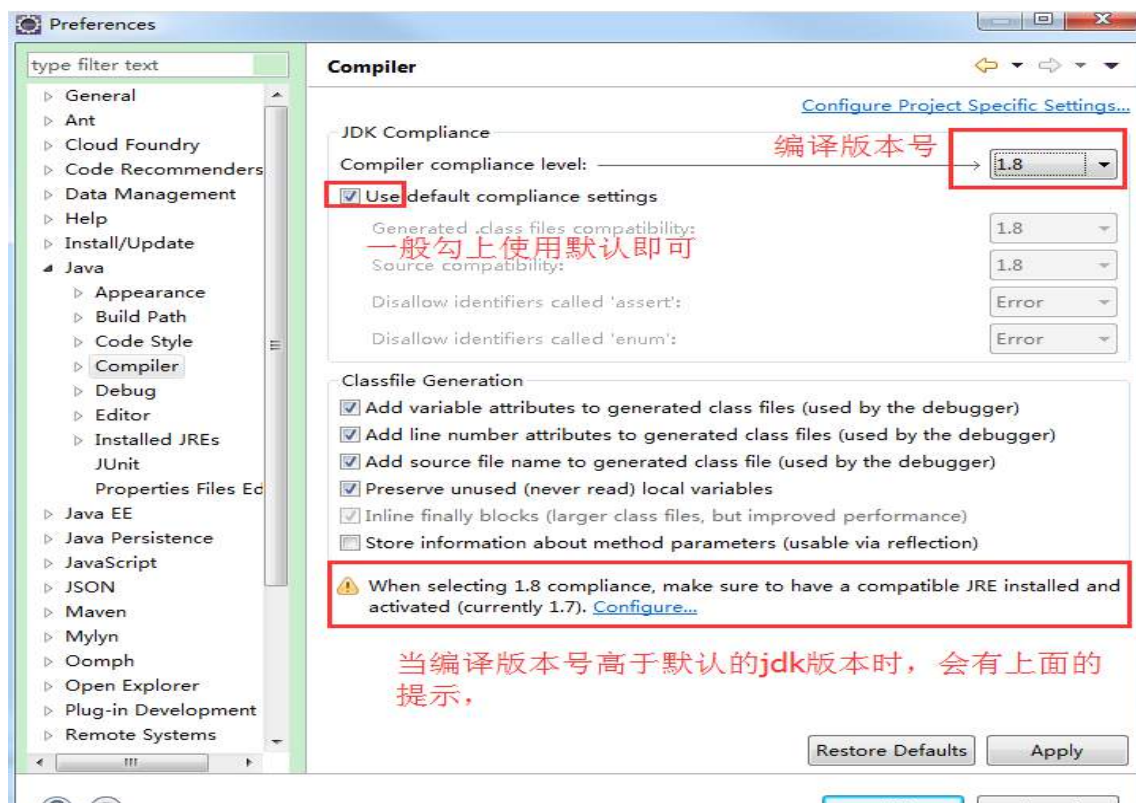


第四步：选择 JRE home

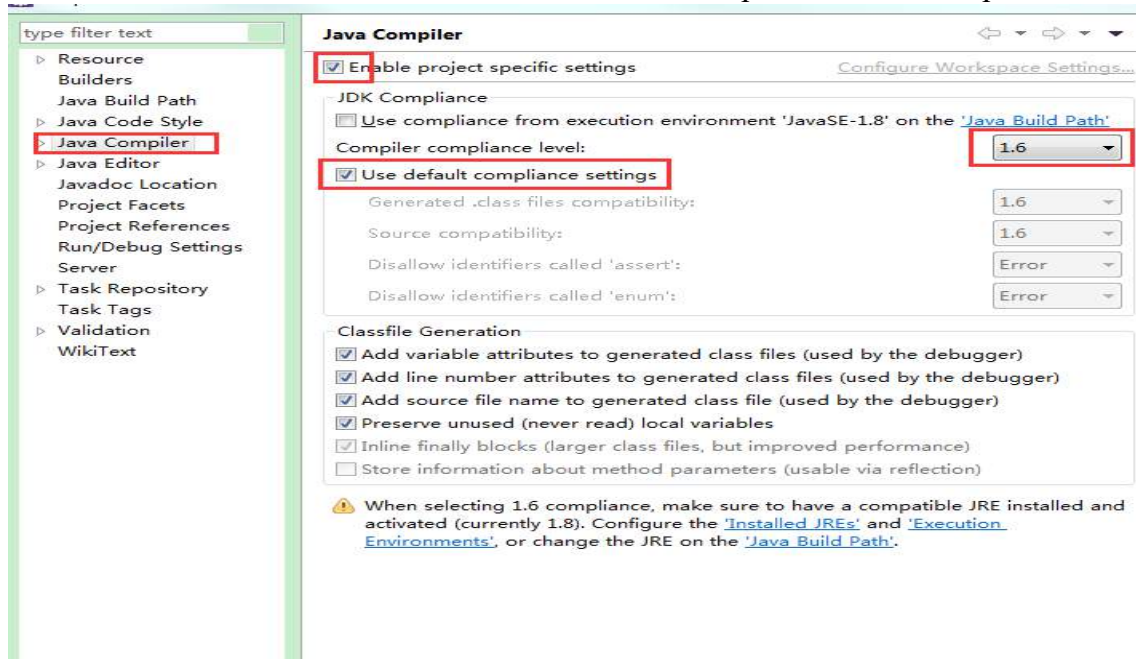




## 第五步：设置 eclipse 全局编译版本



(PS:针对某一个项目设置局部编译版本), 右键->Properties->Java Compiler



勾选上 Enable project specific settings，之后便可以针对该项目进行配置，比如把编译版本设置为 1.6，那么创建的 java 文件便以 1.6 编译成 class 文件。



## 命令行中 java8 和 java11 的切换配置

博客相关链接:

[https://blog.csdn.net/weixin\\_34233421/article/details/93561812](https://blog.csdn.net/weixin_34233421/article/details/93561812)

下面是我自己实验的部分截图

1、进入到指定文件夹。查看文件下的文件（jdk 是默认安装在 home 下的 /Library/Java/JavaVirtualMachines）

```
$ cd /Library/Java/JavaVirtualMachines
$ ls -al
```

```
Last login: Fri Sep 27 21:14:50 on ttys000
songzongyundeMacBook-Pro:~ songzongyun$ cd /Library/Java/JavaVirtualMachines
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$ ls -al
total 0
drwxr-xr-x  5 root  wheel  160  9 27 11:35  .
drwxr-xr-x  4 root  wheel  128  9 12 15:41  ..
drwxr-xr-x  3 root  wheel   96  9 27 11:35  jdk-11.0.4.jdk
drwxr-xr-x  3 root  wheel   96  9 15 11:42  jdk-12.0.2.jdk
drwxr-xr-x  3 root  wheel   96  9 12 15:41  jdk1.8.0_191.jdk
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$
```

## 2、修改 `bash_profile` 文本

```
# added by Anaconda3 5.0.0 installer
export PATH="/Users/songzongyun/anaconda3/bin:$PATH"

# added by Anaconda3 5.2.0 installer
export PATH="/Users/songzongyun/anaconda3/bin:$PATH"

export JAVA_8_HOME="$(/usr/libexec/java_home -v 1.8)"
export JAVA_11_HOME="$(/usr/libexec/java_home -v 11)"
alias jdk8='export JAVA_HOME=$JAVA_8_HOME'
alias jdk11='export JAVA_HOME=$JAVA_11_HOME'
export JAVA_HOME=$JAVA_8_HOME

~
~
~
~
~
~
~
~
~
~
~
~
```

:wq

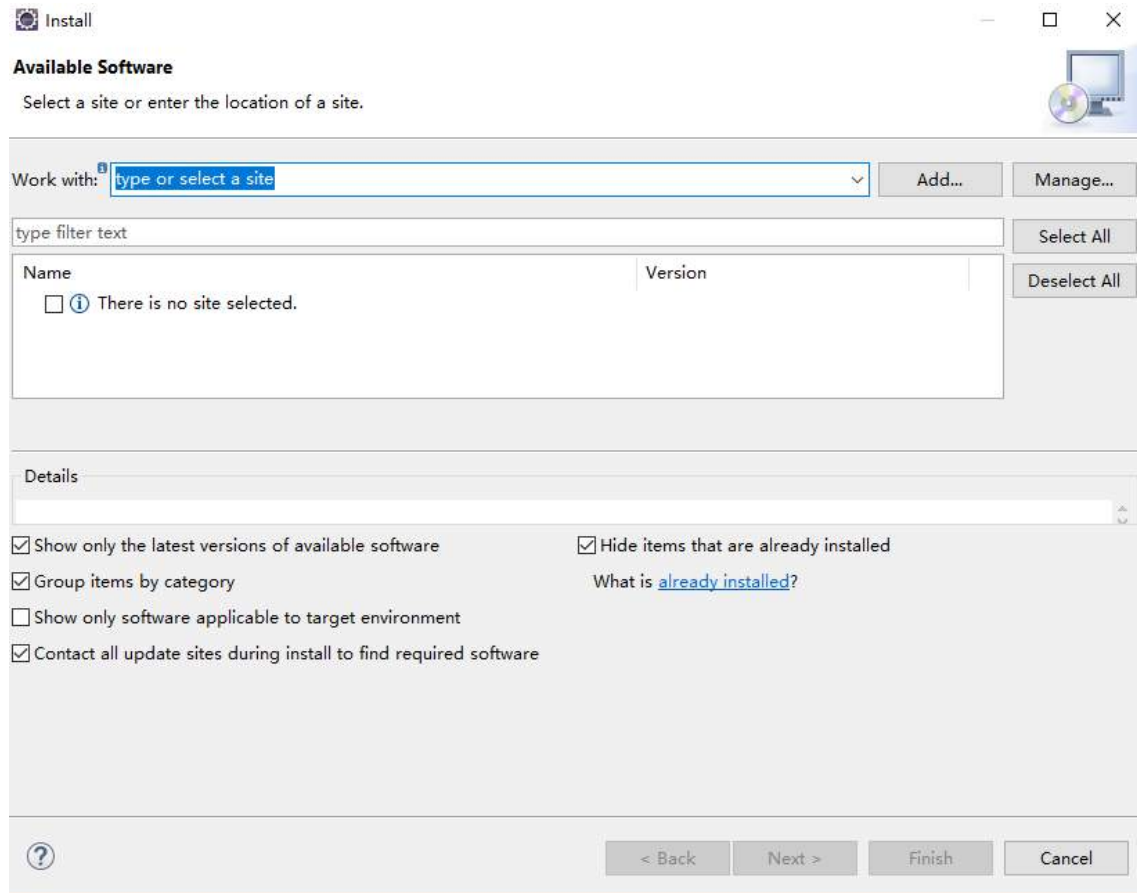
3、建议配置完激活环境后重新打开一次终端，按照下面指示在操作一次。（本人自己操作的时候 vim 修改文本之后激活环境，直接输入 jdk 转化命令发现版本并没有转化成功）

```
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$ source ~/.bash_profile
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$ ls -al
total 0
drwxr-xr-x  5 root  wheel  160  9 27 11:35 .
drwxr-xr-x  4 root  wheel  128  9 12 15:41 ..
drwxr-xr-x  3 root  wheel   96  9 27 11:35 jdk-11.0.4.jdk
drwxr-xr-x  3 root  wheel   96  9 15 11:42 jdk-12.0.2.jdk
drwxr-xr-x  3 root  wheel   96  9 12 15:41 jdk1.8.0_191.jdk
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$ vim ~/.bash_profile
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$ java -version
java version "12.0.2" 2019-07-16
Java(TM) SE Runtime Environment (build 12.0.2+10)
Java HotSpot(TM) 64-Bit Server VM (build 12.0.2+10, mixed mode, sharing)
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$ jdk11
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$ java -version
java version "11.0.4" 2019-07-16 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.4+10-LTS)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.4+10-LTS, mixed mode)
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$ jdk8
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$ java -version
java version "1.8.0_191"
Java(TM) SE Runtime Environment (build 1.8.0_191-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.191-b12, mixed mode)
songzongyundeMacBook-Pro:JavaVirtualMachines songzongyun$
```

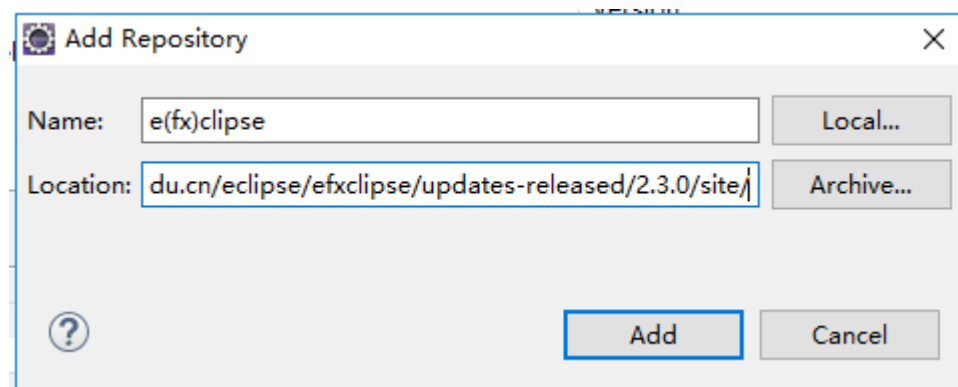
## Eclipse 安装 javafx

参考文献: [https://blog.csdn.net/weixin\\_42978870/article/details/83623435](https://blog.csdn.net/weixin_42978870/article/details/83623435)

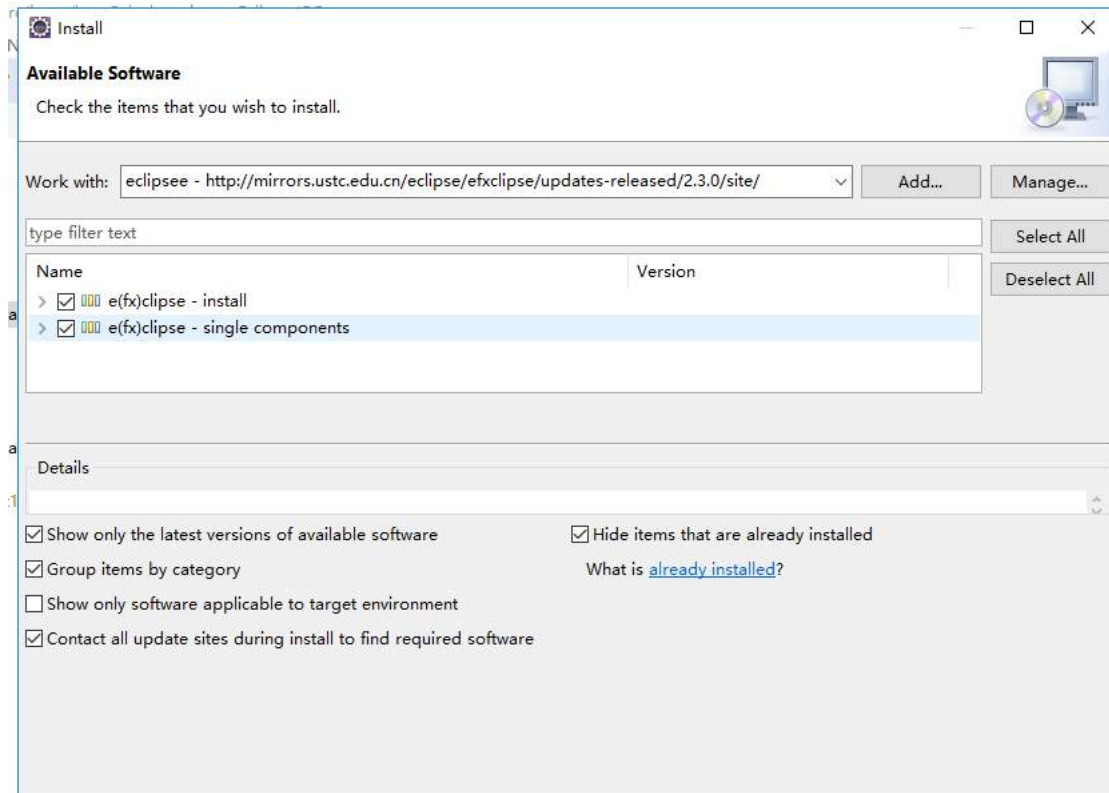
步骤一: 打开 eclipse, 选中菜单 help->install new software



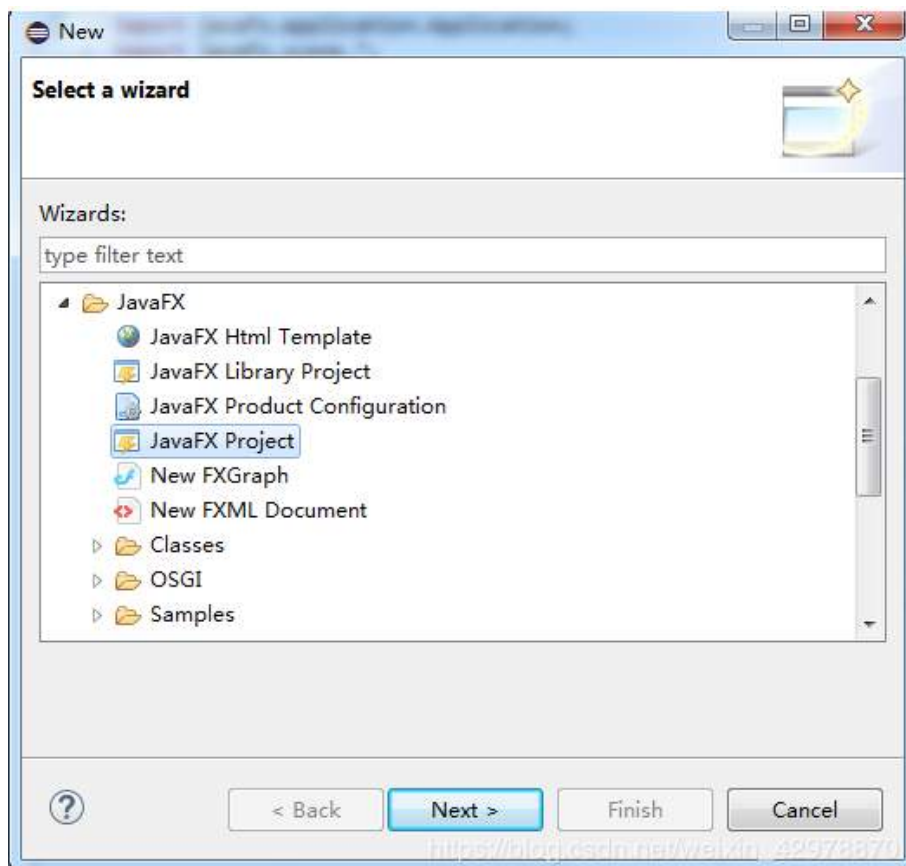
步骤二: 点击 add, 然后在 name 中填写 e(fx)clipse, 在 location 中填写 <http://mirrors.ustc.edu.cn/eclipse/efxclipse/updates-released/2.3.0/site/>



再勾选 name 下列选项, 得到以下界面。



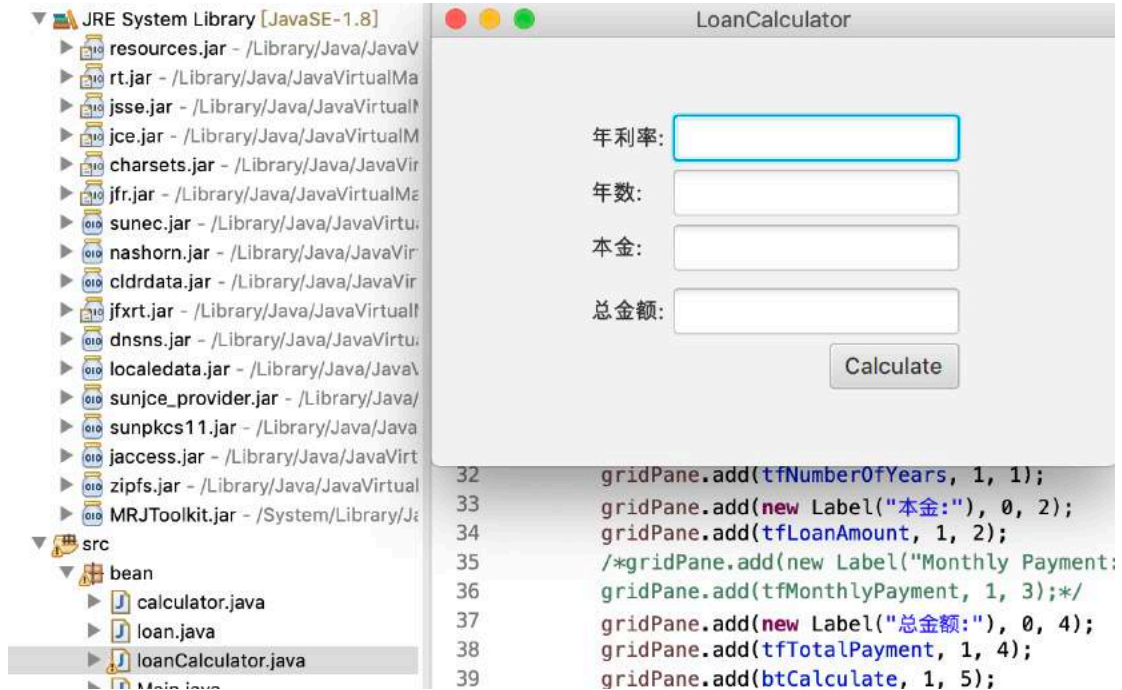
点击各类 next, I accept 以后, 等待一段时间后, eclipse 安装完成, 重启。  
在 new project 时选择 other, 可以看到



## Jdk1.8 和 jdk11 中编译运行 javafx

### 一、jdk1.8 中运行 javafx

在安装 jdk1.8 时，javafx 已经预安装好了，直接调用就行  
示例



### 二、Jdk11 中编译运行 javafx

问题一：jdk11 中不存在 javafx 编译包

解决参考文献一：[https://blog.csdn.net/Sky\\_Coolssy/article/details/90113638](https://blog.csdn.net/Sky_Coolssy/article/details/90113638)

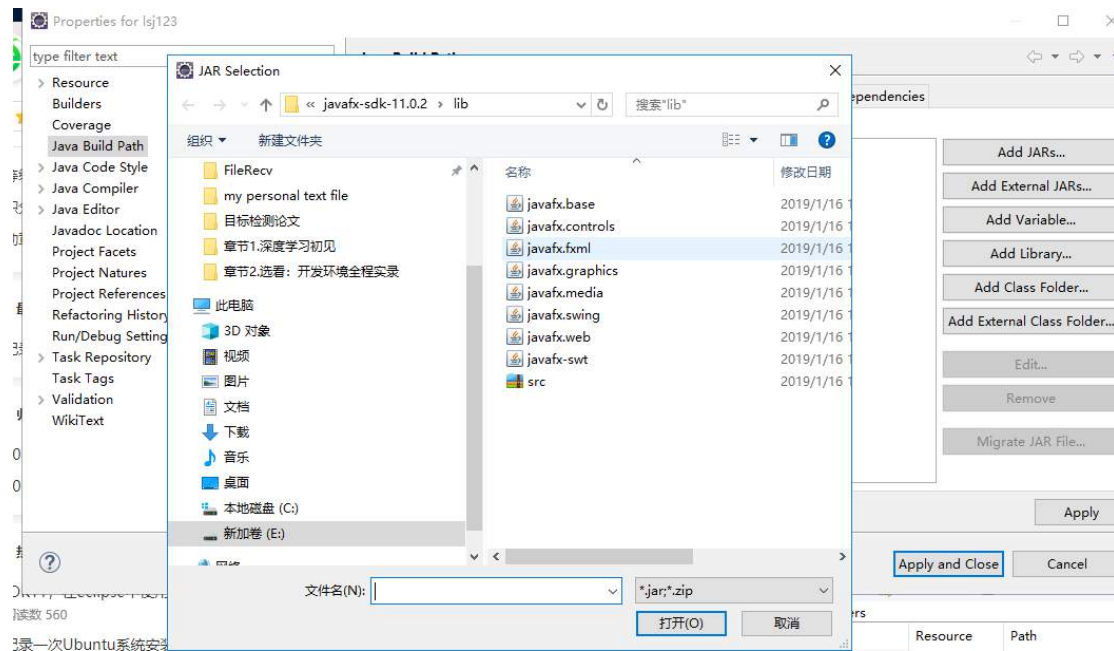
解决参考文献二：[https://blog.csdn.net/Sky\\_Coolssy/article/details/90113638](https://blog.csdn.net/Sky_Coolssy/article/details/90113638)

步骤一：下载 javafx 然后解压到文件夹，记住路径，例如

“D:\openjfx-11.0.2\_windows-x64\_bin-sdk\javafx-sdk-11.0.2”

步骤二：右键->Build Path->Configuration Build Path，然后 Add External JARS

把 javafx 文件夹里的 jar 包以及 src.zip 文件全部导入下来

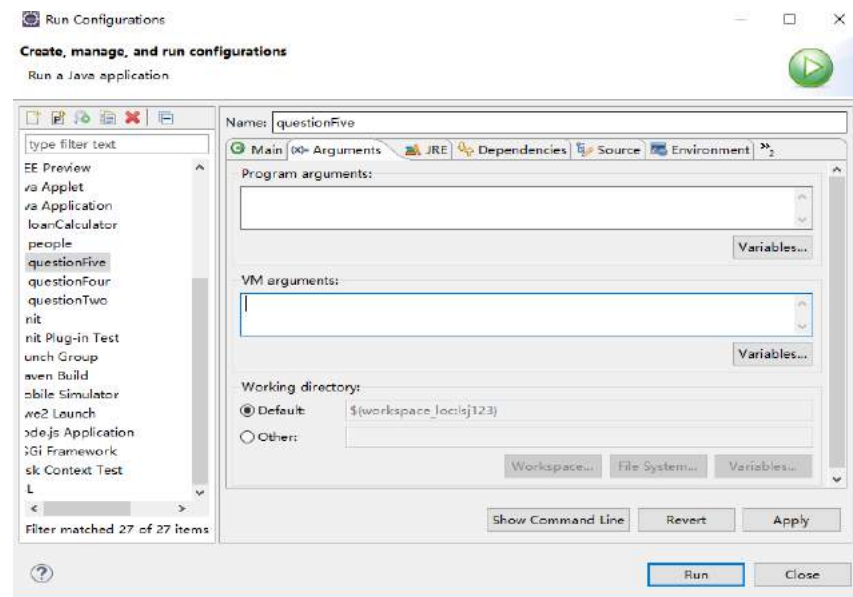


问题解决。

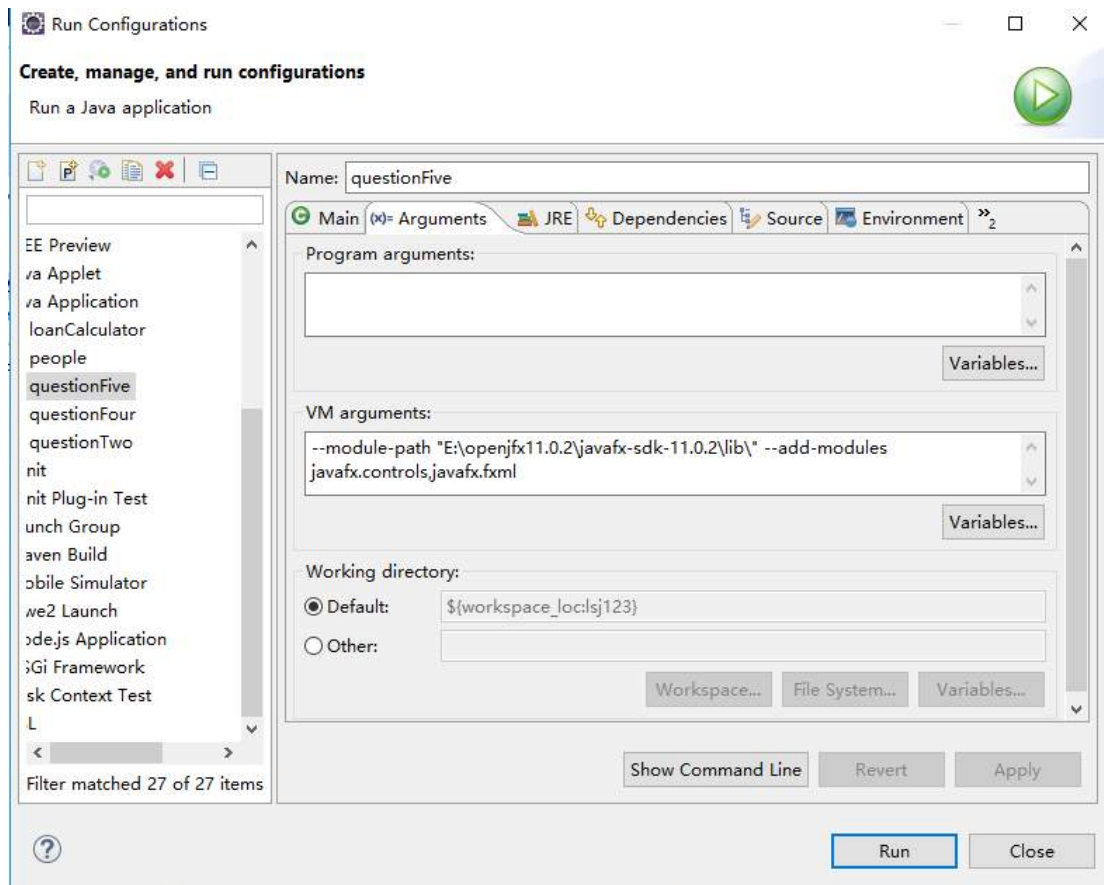
问题二：运行时缺少依赖组件

参考文献：[https://blog.csdn.net/Sky\\_Coolssy/article/details/90113638](https://blog.csdn.net/Sky_Coolssy/article/details/90113638)

步骤一:Run->Run Configurations,选择 Main.java 点击 Arguments







步骤二：在 VM Arguments 中添加参数

--module-path "\path\to\javafx-sdk-11\lib" --add-modules javafx.controls,javafx.fxml

其中 \path\to\javafx-sdk-11\lib 参数为 ipenjfx 文件解压的路径

