



主讲教师 张 智
计算机学院软件工程系
课程群: 421694618

1 Java概述

1.1 [Java语言简介](#)

1.2 [Java编程环境搭建](#)

1.3 [Hello Java](#)

1.4 [Java运行机制](#)

1.5 [打包Java程序](#)

1.6 [程序示例](#)


【附录】[Idea常用快捷键](#)

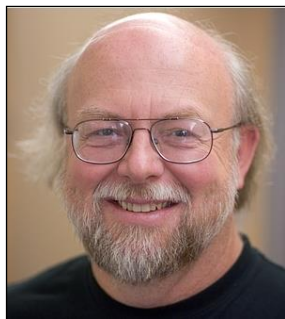
1.1 Java语言简介

- Java是全球排名前三的编程语言(20年之前多年排名第一)。
- 常年霸占着三大市场：
 - 互联网和企业应用，这是Java的长期优势和市场地位
 - 大数据平台：Hadoop、Spark、Flink平台主要用Java开发
 - Android移动平台

Java语言发展历史



- 1990年初， Sun公司基于家用电器编程需要，设计了一种**平台独立**的语言**Oak**。
- 随着互联网潮流的兴起，Sun公司预见互联网应用前景，于是改造了Oak，1995年5月，**Oak 语言正式更名为 Java**。
- 1996年，**JDK 1.0**发布，10个最主要的操作系统供应商声明将在其产品中嵌入Java技术。



James Gosling
Java之父

Java开发套件(Java Development Kit,JDK)

Java语言发展历史（续）

- Java伴随着互联网的迅猛发展而发展，逐渐成为重要的网络编程语言。
- 2006年SUN公司将Java开源，此时的JDK即为Open JDK。

2009年，**ORACLE** (甲骨文公司)以74亿美元现金收购Sun公司。

2014年，JDK8 发布，这是一个里程碑的版本。

2017年，**Oracle**不允许开源组织使用Java名号。

2019年，Oracle发布**Oracle JDK** 8u211和8u212两个版本(属于JDK8系列)，
并从这两个版本开始**商用收费**！

2021年，Oracle JDK16 发布。

Java平台的3个版本

1. 标准版：Java SE

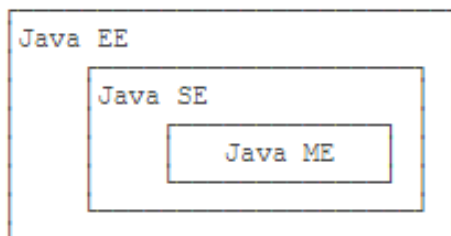
- 用于开发和部署在桌面、服务器等环境中的应用程序

2. 企业版：Java EE （ 俗称 J2EE ）

- 用于开发企业级Web应用程序

3. 微型版：Java ME

- 用于移动设备和嵌入式设备程序开发 → 已经被Android取代



Java的特点

Write Once, Run Anywhere

- **平台无关性**：采用**虚拟机(JVM)**技术实现平台无关性。Java软件是真正跨平台可移植的。
- **面向对象**：具有良好的代码重用性。
- **解释性**：采用预编译将源程序生成**.class字节码**，减轻运行时解释工作。
- **简单、健壮**：**不需要进行指针运算和存储器管理**，简化设计，减少出错可能性。

Java的特点(续)

- **安全**：拒绝执行非法的内存访问，超越权限的访问等。
- **多线程**：允许一个应用程序同时做多个任务。
- **分布式**：拥有网络协议对象库，可以象访问本地文件一样访问Internet上的对象。
- **动态性**：允许下载代码模块，当程序运行时也能动态升级。
- **高性能**：经过实际的综合评测得出结论，Java是高性能的。

[【返回】](#)

1.2 Java编程环境搭建

- 安装IDE：Idea（推荐Ultimate英文版）
- 安装JDK：自动安装（推荐）
- 附：手工安装JDK

基本概念：

- IDE：集成开发环境 (Integrated Development Environment)，集成了代码编写、编译、调试等功能的程序开发环境的应用程序
- JDK：Java开发工具集 (Java Development Kit)，包含了Java运行环境(JRE)和开发工具(编译器，调试器，javadoc等)。我们就是依靠JDK来开发和运行Java程序的。

[【返回】](#)

1. 安装Idea Ultimate版

- 官网：<https://www.jetbrains.com/idea/download/#section=windows>



Idea Ultimate安装参考：

■ 免费申请：

- <https://www.cnblogs.com/xicyannn/p/10505846.html>

- <https://blog.csdn.net/kanchaishaonian/article/details/81214904>

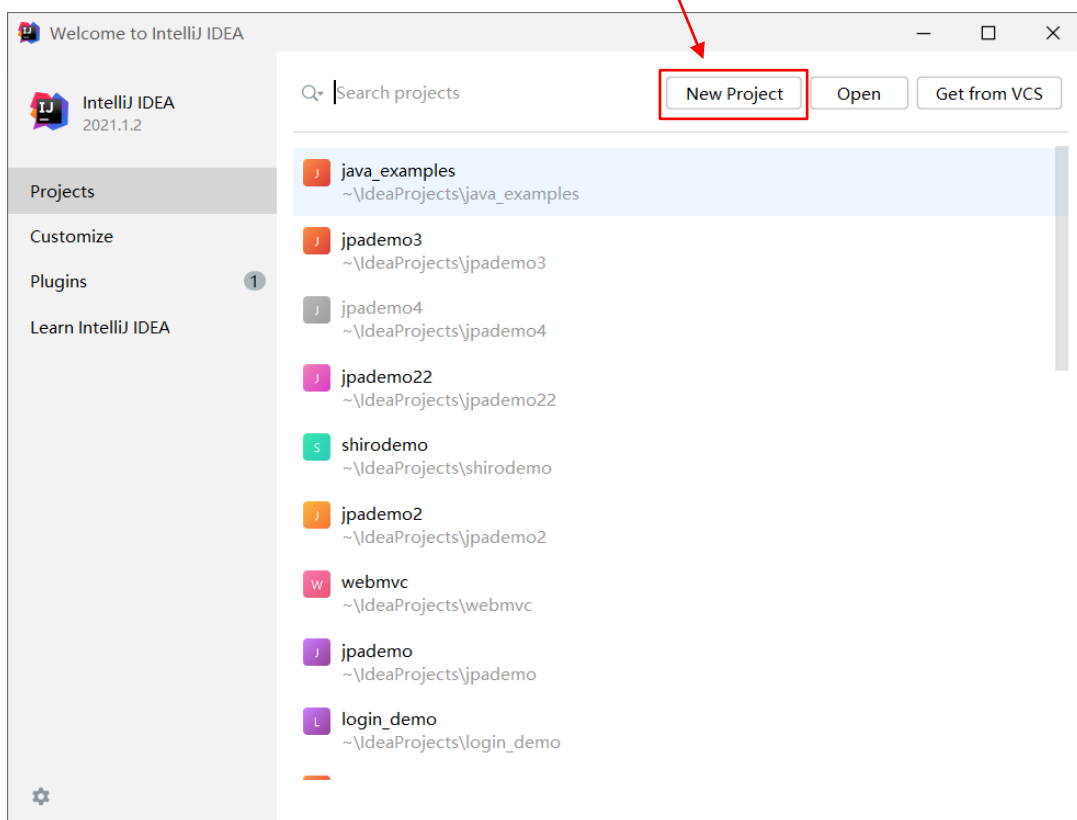
■ 破解版：

- <https://www.cnblogs.com/kkakura/p/13686904.html>

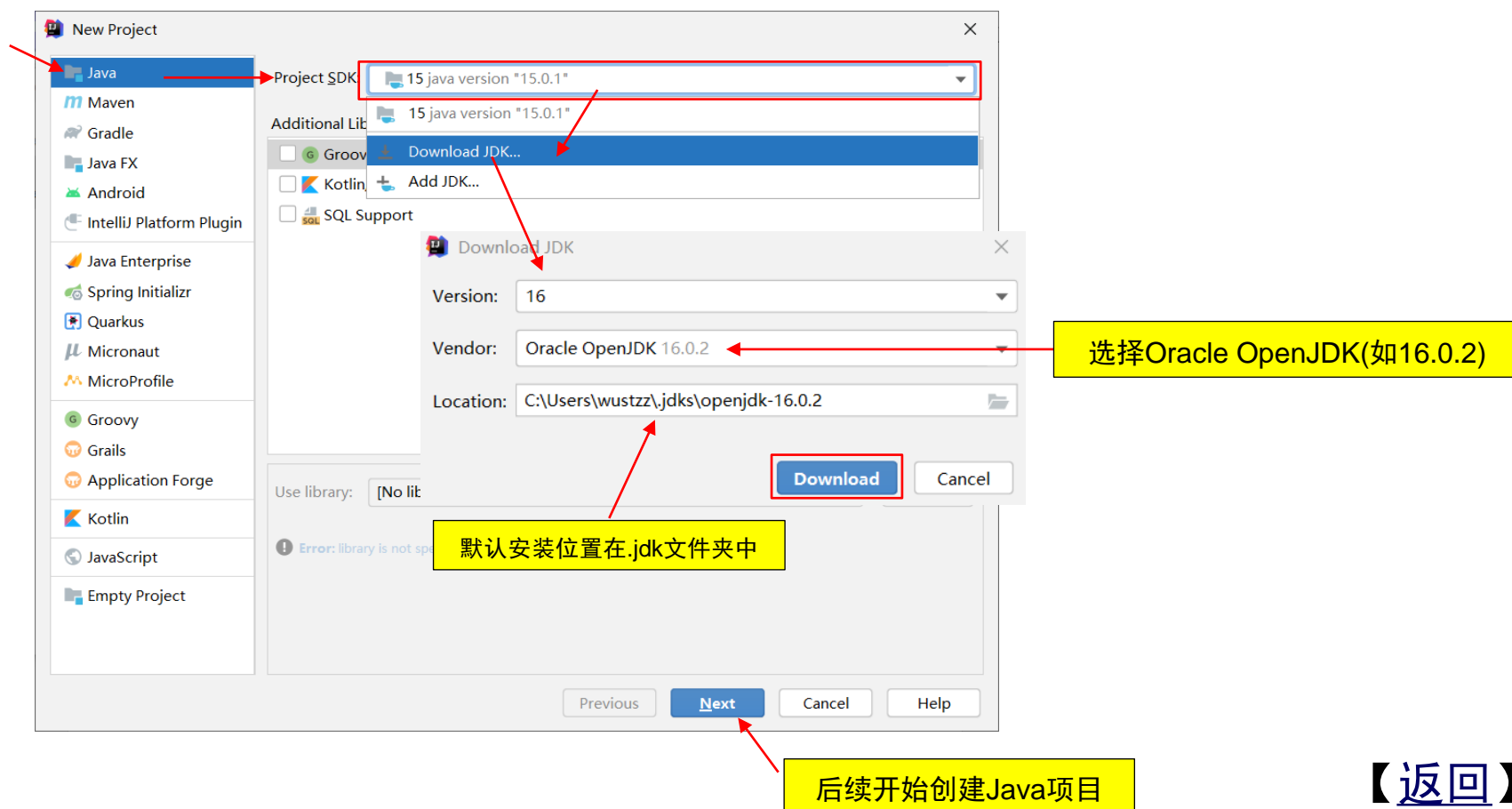
【[返回](#)】

2. 安装JDK

- 启动Idea → 点击 new project



- 接着选择Java → 在 Project SDK 中选择 Download JDK...
- 在Download JDK界面中选择相应的JDK版本进行下载(自动安装)



附：手工安装JDK（以JDK 1.8.0_202为例）

官网下载JDK：<https://www.oracle.com/java/technologies/javase/javase8-archive-downloads.html>

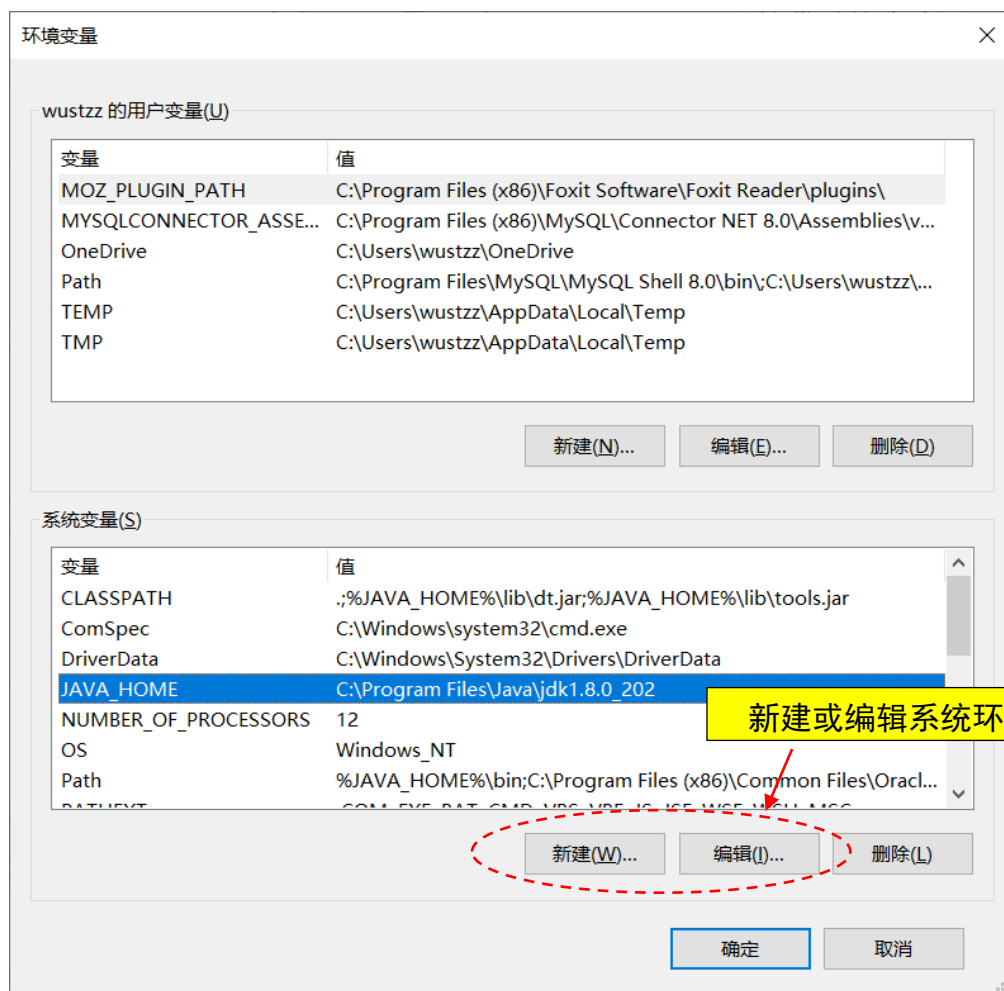
Solaris x64 (SVR4 package)	124.37 MB	jdk-8u202-solaris-x64.tar.Z
Solaris x64	85.38 MB	jdk-8u202-solaris-x64.tar.gz
Windows x86	201.64 MB	jdk-8u202-windows-i586.exe
Windows x64	211.58 MB	jdk-8u202-windows-x64.exe

Java SE Runtime Environment 8u202
This software is licensed under the [Oracle Binary Code License Agreement for Java SE Platform Products](#)

Product / File Description	File Size	Download
----------------------------	-----------	----------

exe文件下载后直接运行安装(一路默认...)

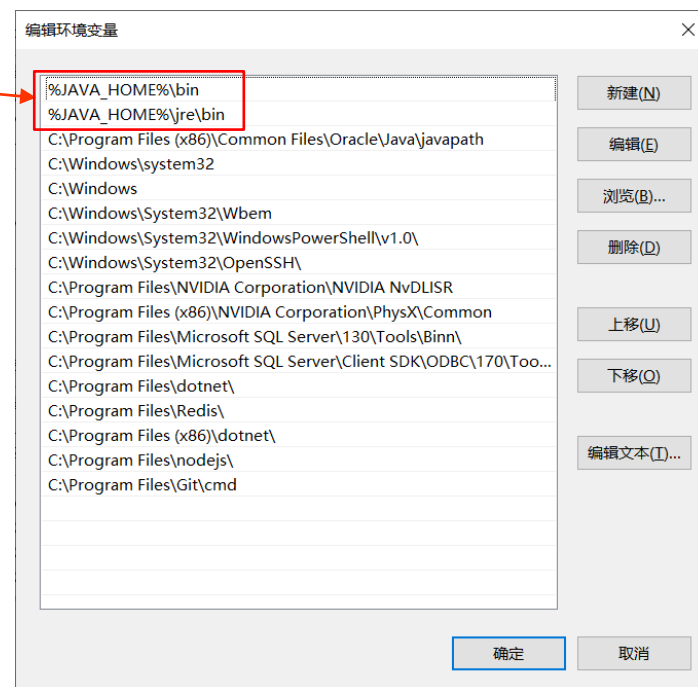
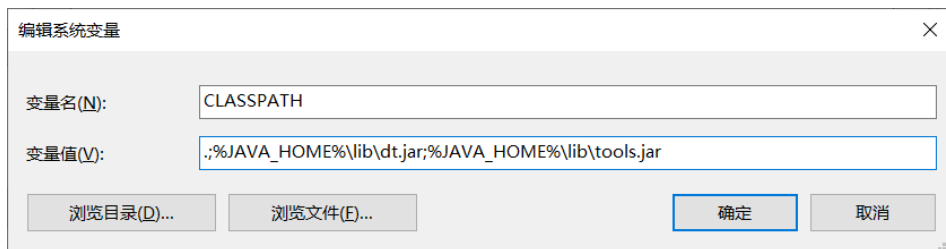
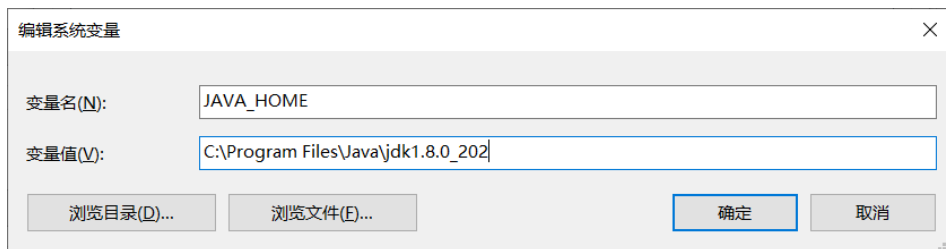
手工配置 Java 环境变量



新建或编辑系统环境变量(参数见下页)

Java8环境变量

- 新建：JAVA_HOME 值： C:\Program Files\Java\jdk1.8.0_202
- 新建：CLASSPATH 值： .;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar
- 编辑：PATH，添加新值： %JAVA_HOME%\bin 和 %JAVA_HOME%\jre\bin



检测Java环境是否配置成功

打开cmd命令窗口，输入：**java -version** 和 **where java**

```
命令提示符
Microsoft Windows [版本 10.0.18363.1198]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\wustzz>java -version
java version "1.8.0_202"
Java(TM) SE Runtime Environment (build 1.8.0_202-b08)
Java HotSpot(TM) 64-Bit Server VM (build 25.202-b08, mixed mode)

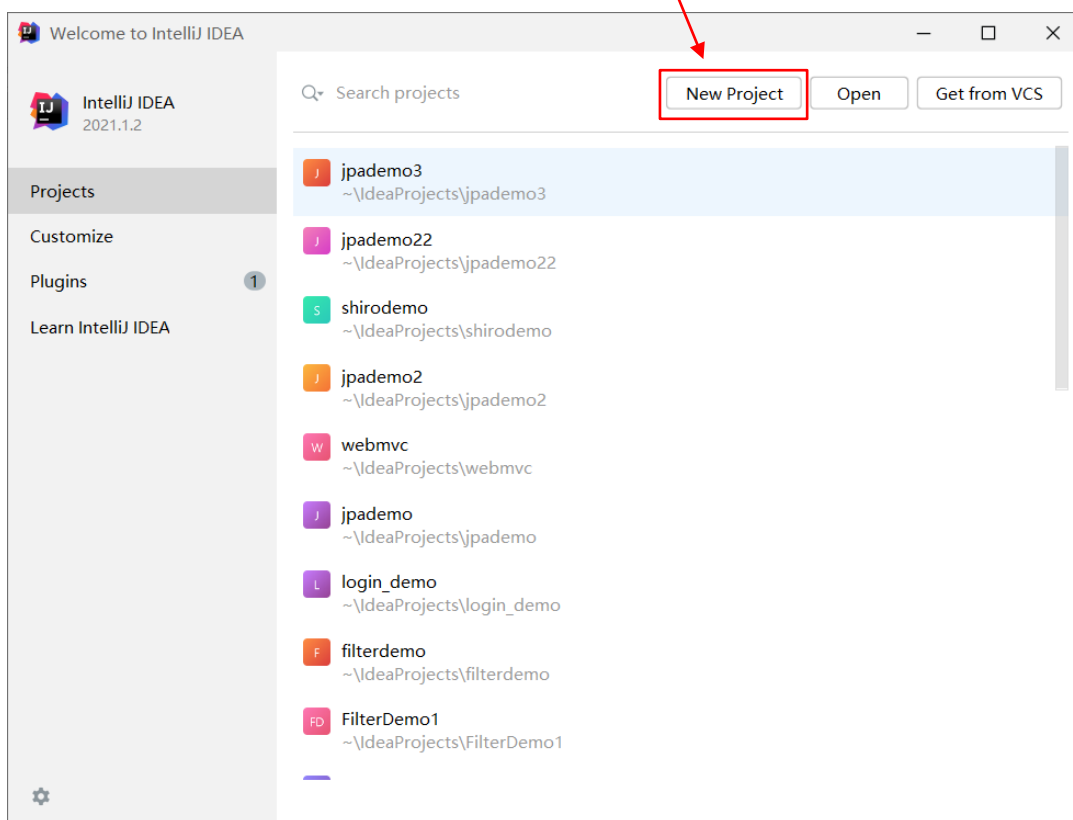
C:\Users\wustzz>where java
C:\Program Files\Java\jdk1.8.0_202\bin\java.exe
C:\Program Files\Java\jdk1.8.0_202\jre\bin\java.exe
C:\Program Files (x86)\Common Files\Oracle\Java\javapath\java.exe

C:\Users\wustzz>
```

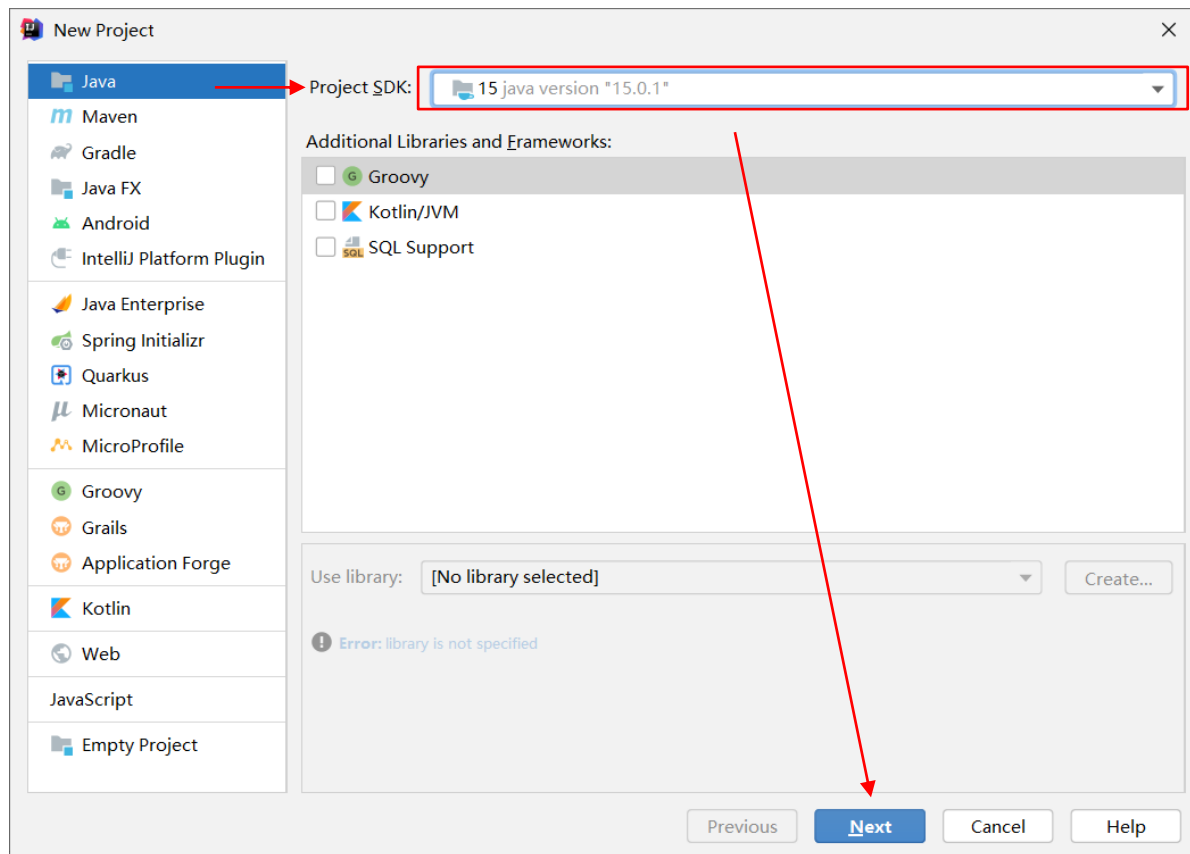
[【返回】](#)

1.3 Hello Java

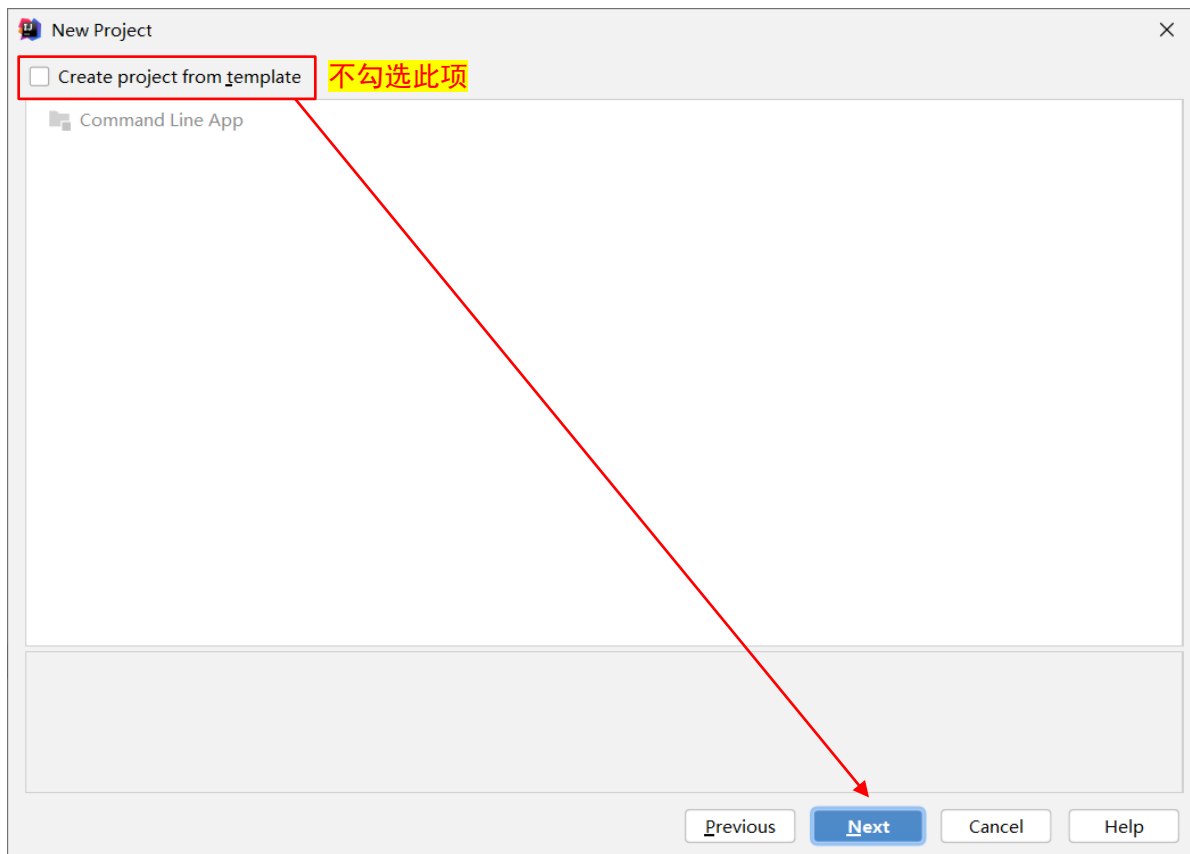
- 启动Idea → 点击 new project



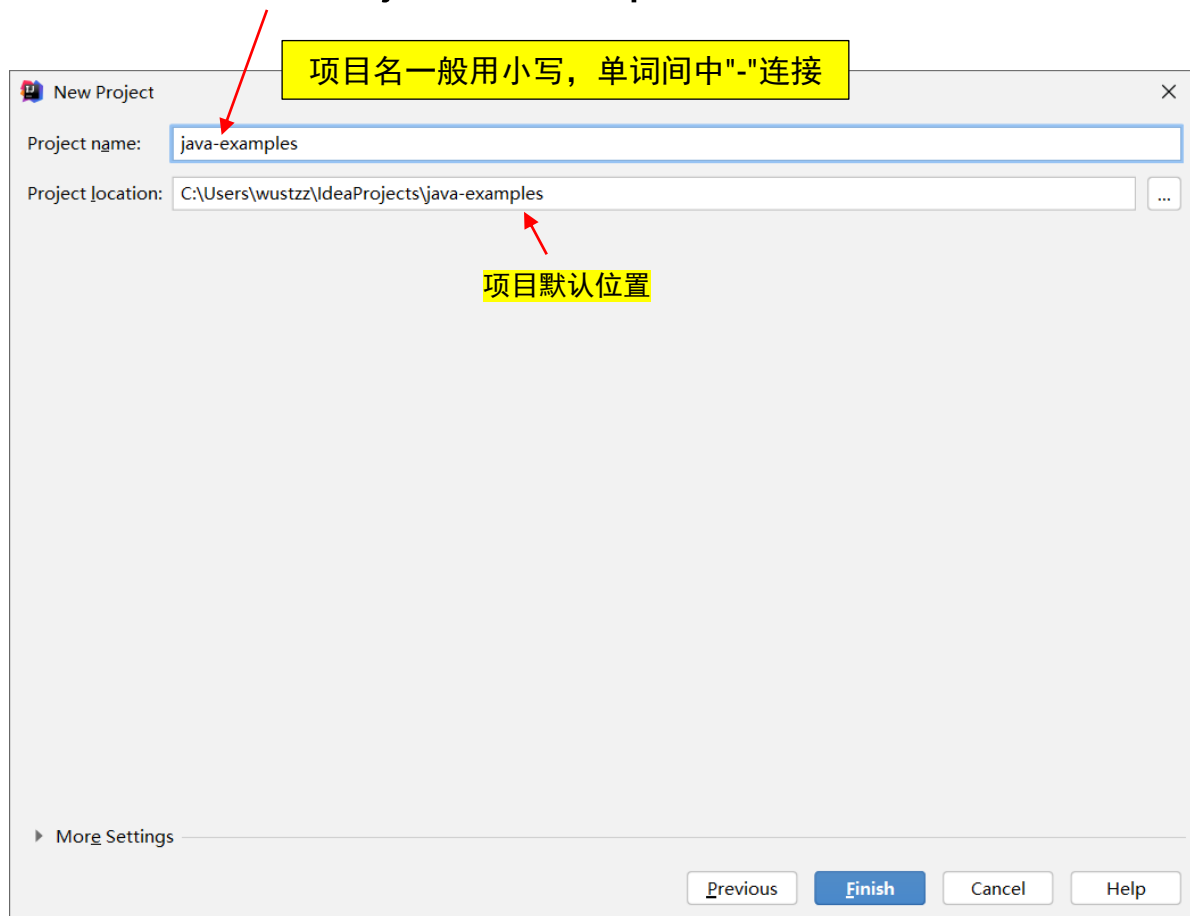
- 选择Java → 在 Project SDK栏选择已经安装的JDK版本 → Next



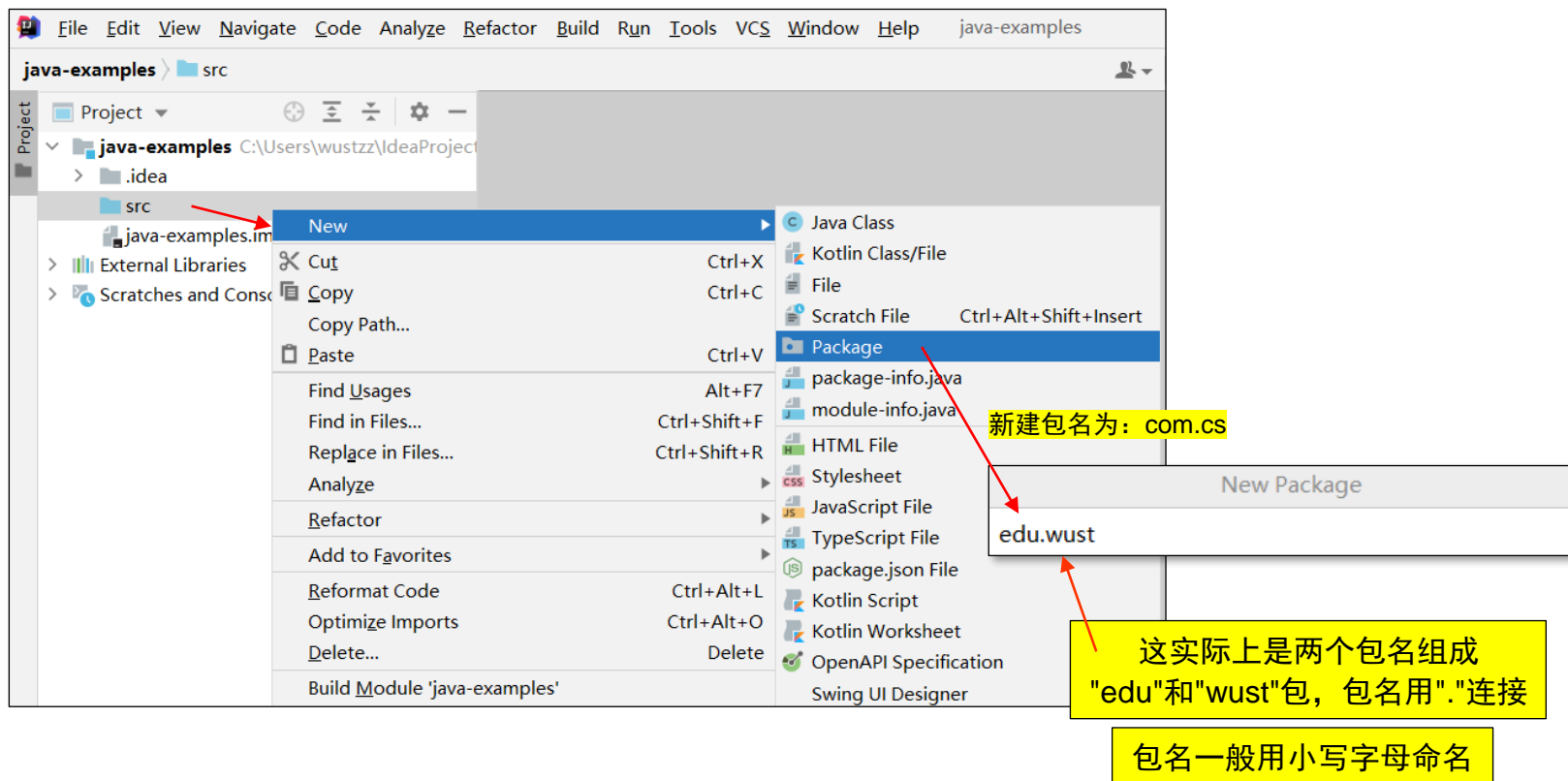
- 默认不勾选 Create project from template → Next



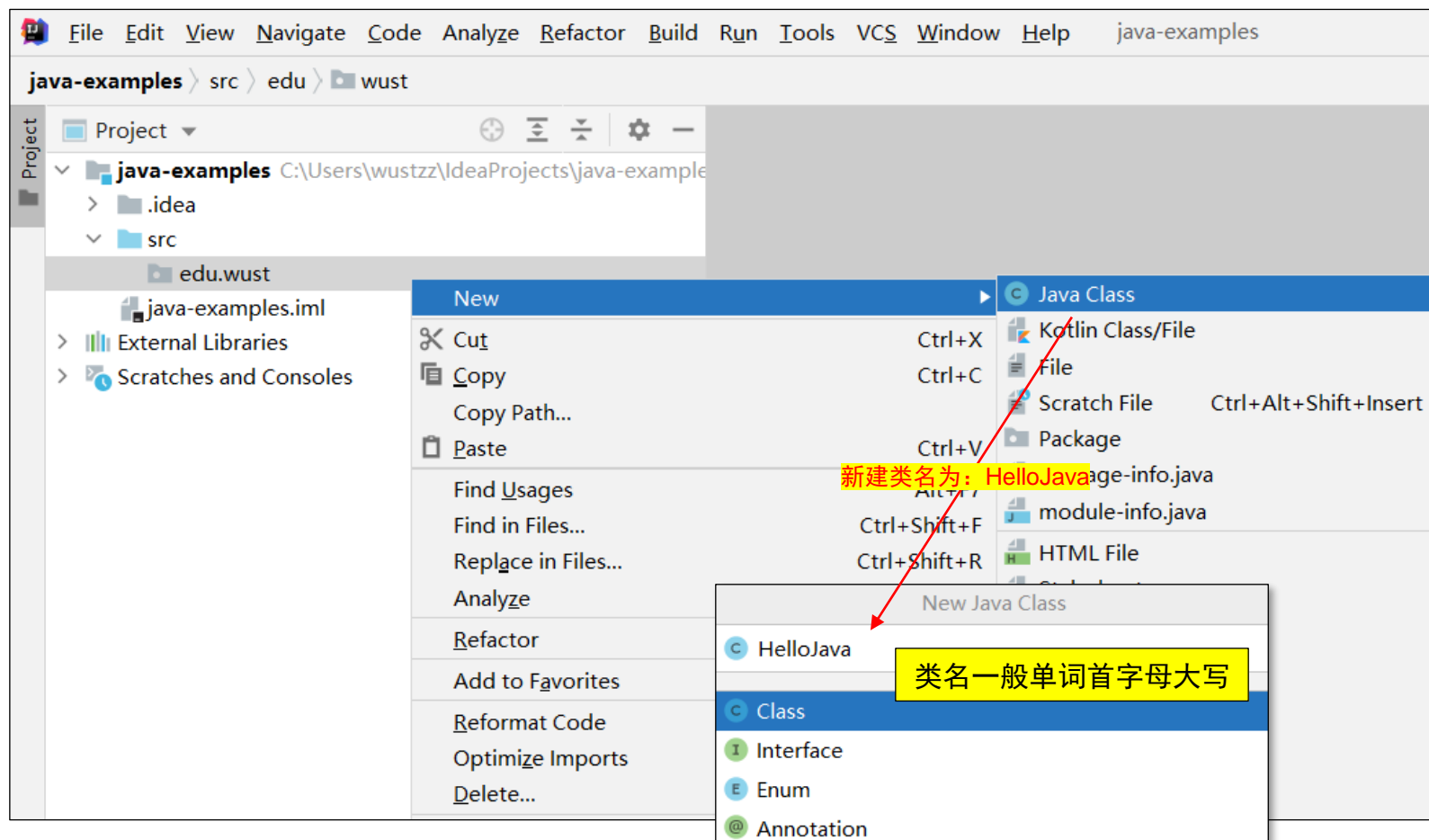
■ 设置项目名称：java-examples → Finish



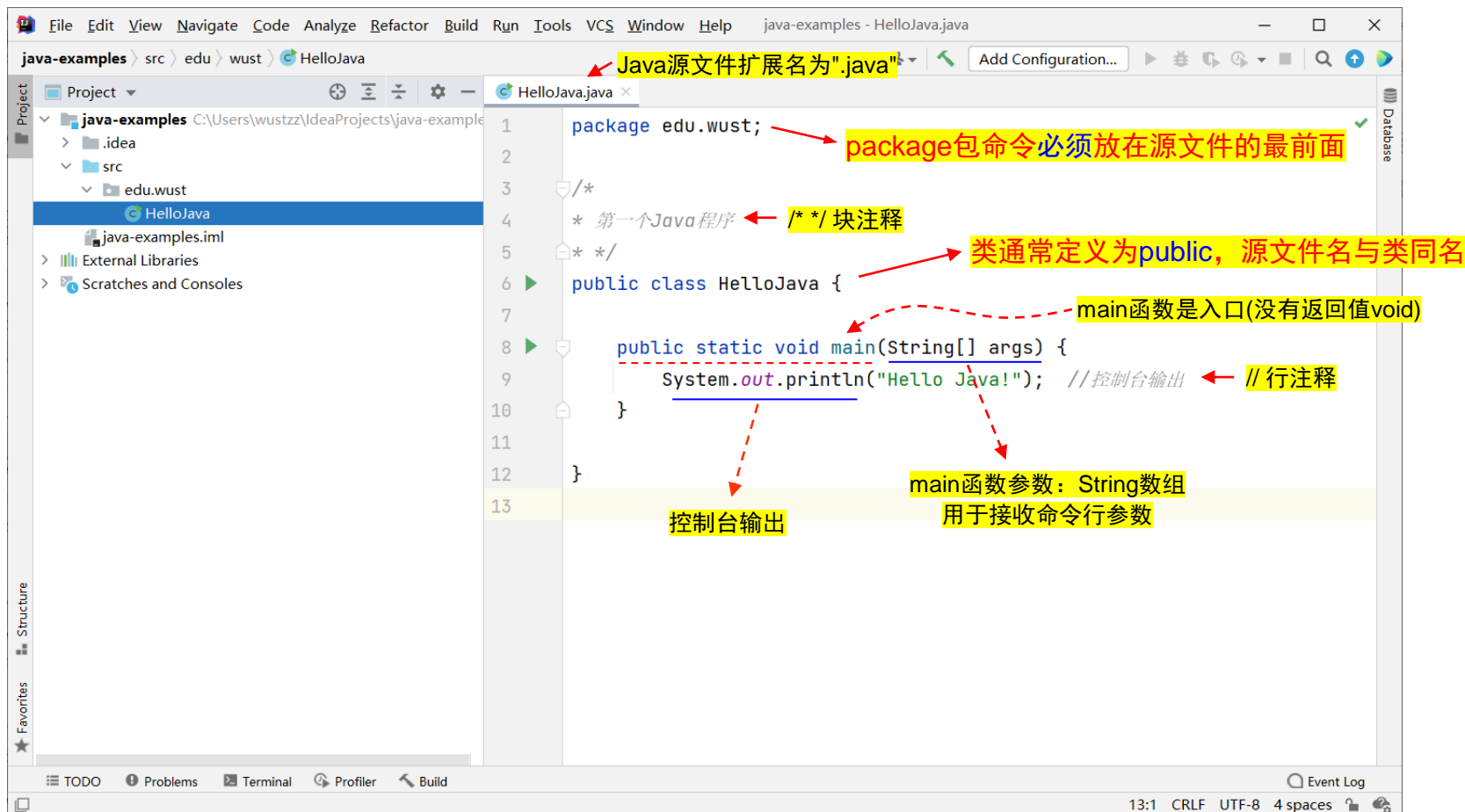
- src 文件夹右键 → New → Package(包) → 新建包名为: com.cs



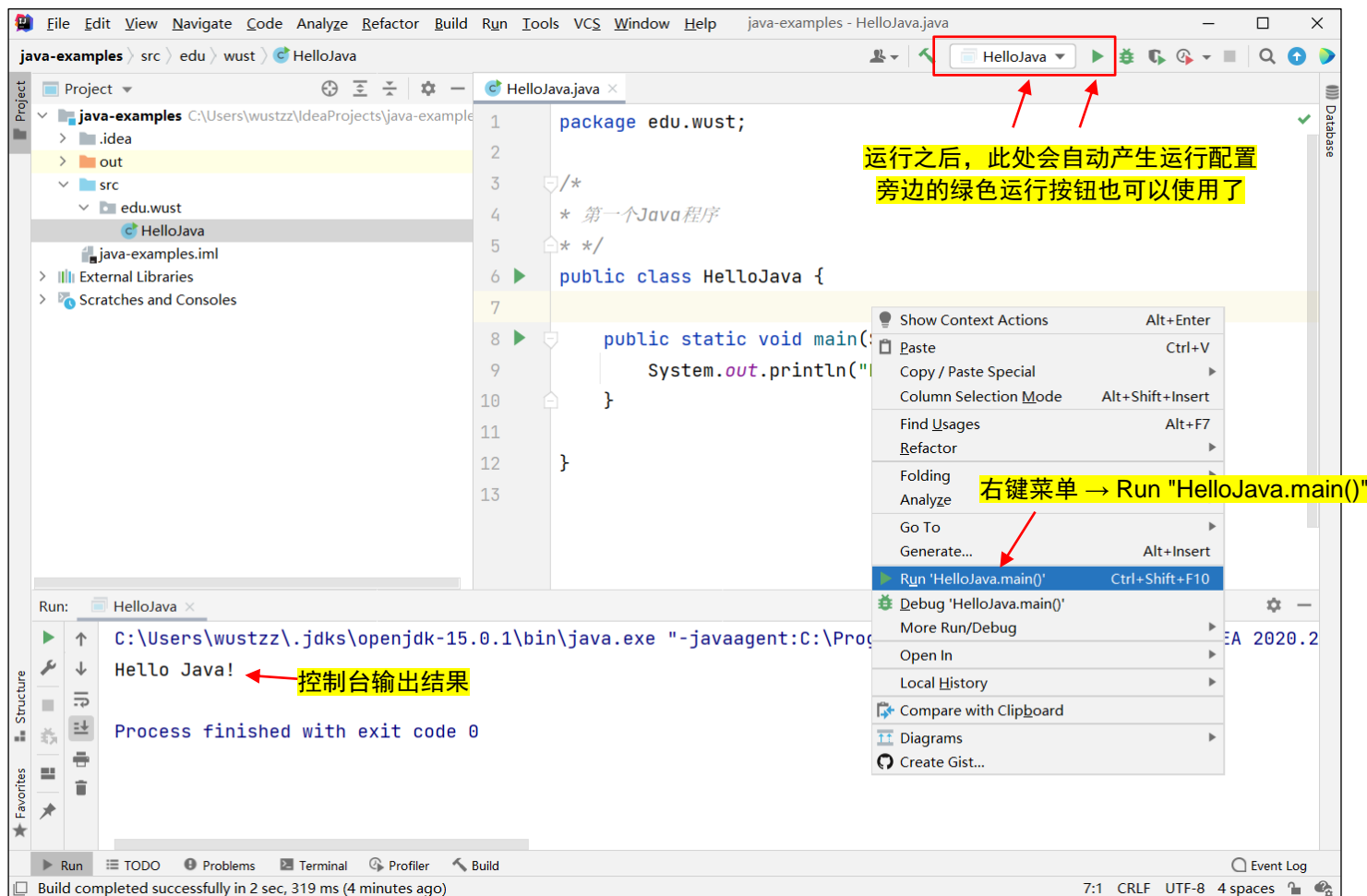
- edu.wust 包右键 → New → Class(类) → 新建类名为: HelloJava



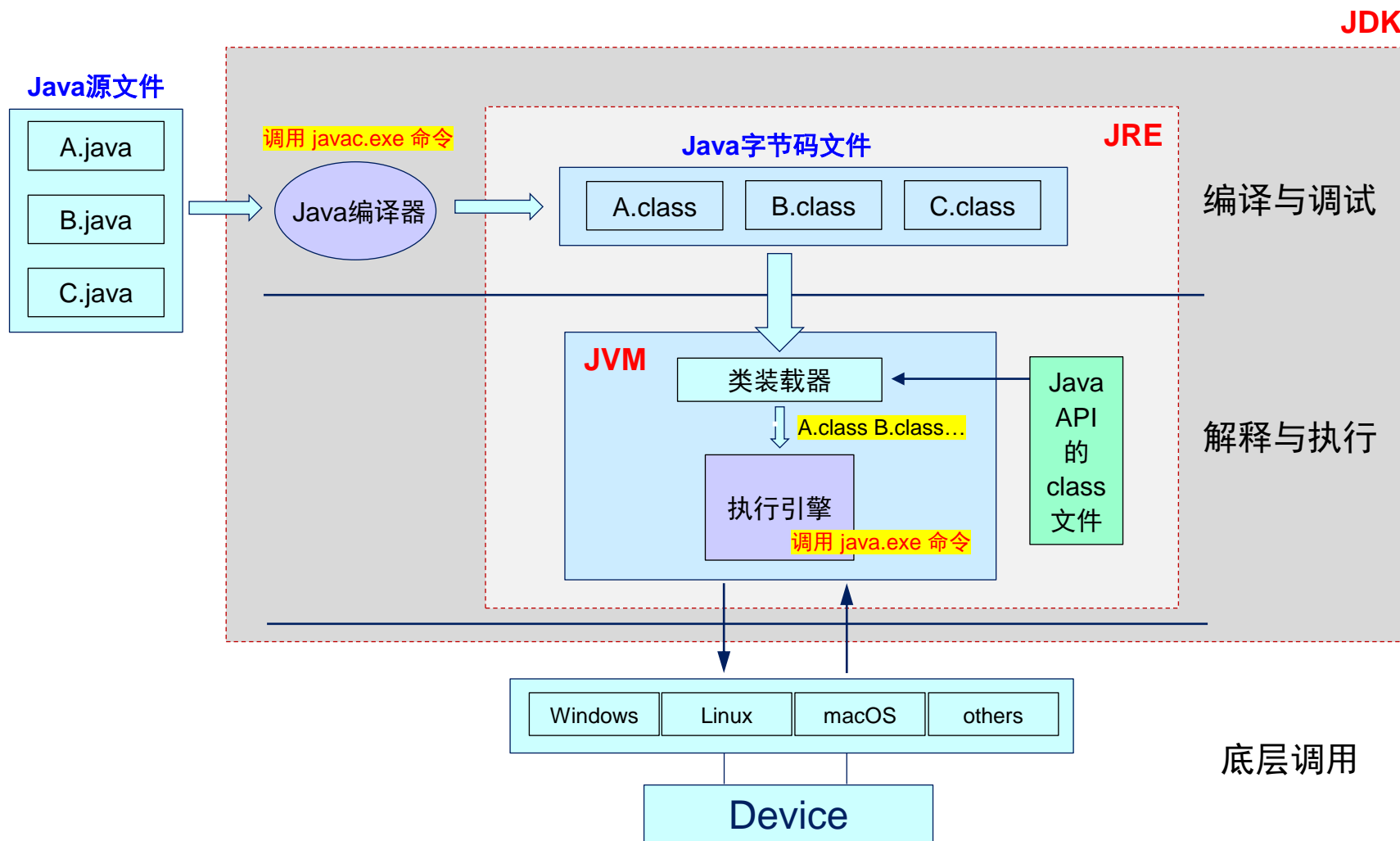
HelloJava代码



程序运行 -- 初次运行时



1.4 Java运行机制



重要概念

■ JVM: Java虚拟机 (Java Virtual Machine)

- JVM是一个虚拟的计算机，它有一套完整的体系架构，包括处理器、堆栈、寄存器等
- JVM可以理解成一个以字节码(class文件)为机器指令的CPU
- 对于不同的运行平台，有不同的虚拟机
- JVM屏蔽了底层运行平台的差别，实现了"Write Once, Run Anywhere"

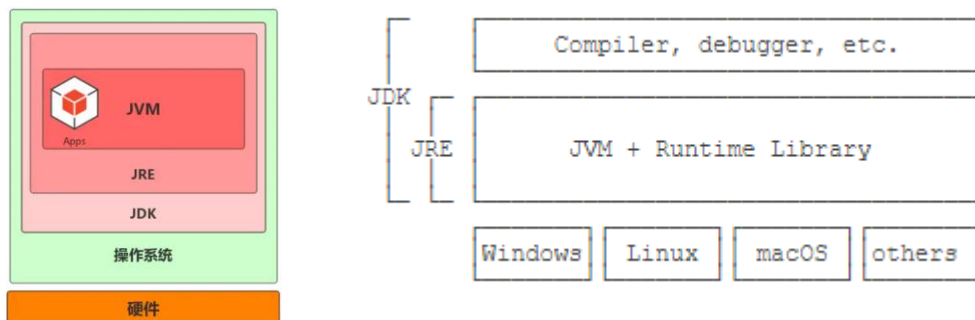
→ JVM的指令序列

■ JRE: Java运行时 (Java Runtime Environment)

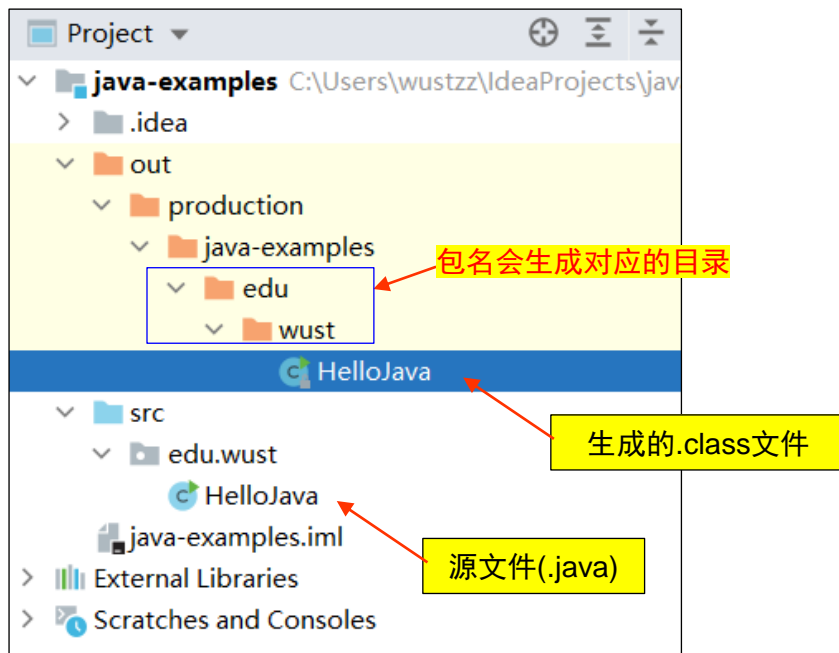
- JRE为Java提供了运行环境，其中重要的一环就是通过JVM将字节码解释成可执行的机器码。
- JRE由JVM, Java运行时类库, 动态链接库等组成

■ JDK: Java开发工具集 (Java Development ToolKit)

- JDK包含了Java运行环境(JRE)和开发工具(编译器, 调试器, javadoc等)
- 我们就是依靠JDK来开发和运行Java程序的



生成的.class文件



了解：手工命令行形式的编译与运行

■ 主要命令行

■ 编译程序：javac HelloJava.java

编译时文件名大小写无所谓

(要带扩展名,编译成功会生成同名.class文件)

字节码文件

提醒：去掉中文注解

■ 运行程序：java edu.wust>HelloJava

需要手工创建文件夹(edu/wust)并复制.class文件

(运行.class时,不加扩展名且区分大小写)

```
命令提示符
Microsoft Windows [版本 10.0.18363.1198]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\wustzz>
D:\>C:\Users\wustzz\.jdk\openjdk-15.0.1\bin\javac HelloJava.java
D:\>C:\Users\wustzz\.jdk\openjdk-15.0.1\bin\java edu.wust>HelloJava
Hello Java!
D:\>
```

切换到D盘根目录(假设Java源文件放在D盘根目录下)

使用Idea安装的JDK来编译源文件
(javac命令前面加上目录位置)

运行结果

使用java命令运行.class文件
(java命令前面加上目录位置)

【返回】

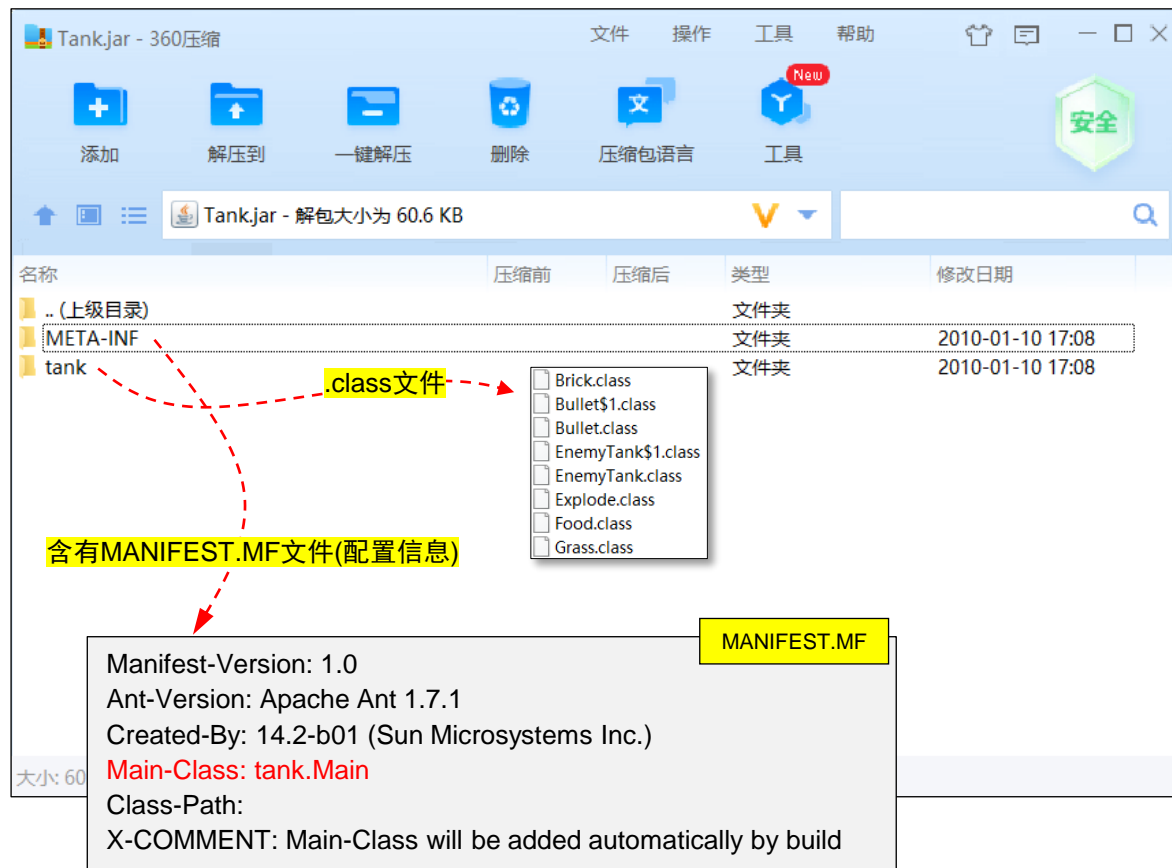
1.5 打包Java程序

- 在开发过程中，开发人员知道：哪个class是主程序，使用哪个class可以直接运行程序
- 但对普通用户来说：只希望双击哪个程序就能运行得到结果
- 解决：Java使用 **jar (java archive)** 文件来发布和运行
 - jar文件是一种按Java格式压缩的类包，是Java文件封装的最小单元
 - 如果用命令行运行jar文件：**java -jar test.jar**

不是exe文件哦

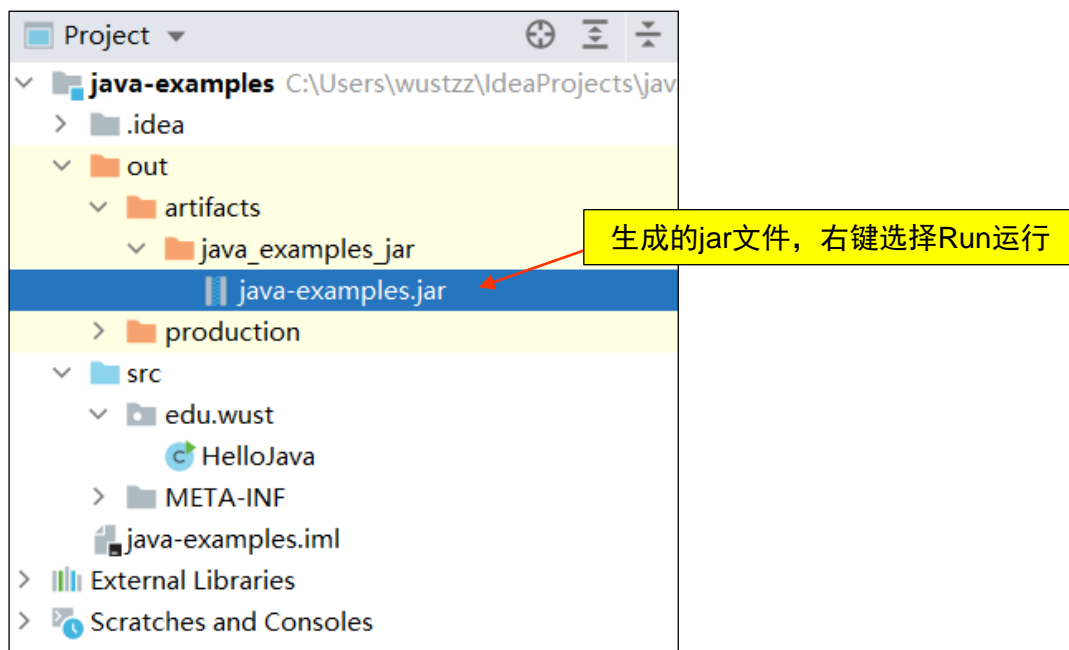
【坦克大战jar示例】

Tank.jar文件示例



打包操作

- 打包过程详见：<https://www.cnblogs.com/july7/p/11473751.html>



[【返回】](#)

1.6 程序示例

- 控制台输入输出程序
- 多个类的程序
- Java程序调试

【返回】

1. 控制台输入输出程序 -- 输入一个数据

HelloJava.java

```
package edu.wustz;  
  
import java.util.Scanner; //导入Scanner类
```

```
public class HelloJava {
```

```
    final static double VERSION = 1.0; //定义一个类成员 (符号常量)
```

```
    public static void main(String[] args) {
```

```
        System.out.print("请输入圆的半径=");
```

System.in: 标准输入流对象, 即键盘

```
        Scanner sc = new Scanner(System.in); //创建Scanner对象 (扫描键盘缓冲区)
```

```
        double r = sc.nextDouble(); //从键盘读取一个double数据, 回车结束
```

Scanner会自动转换数据类型

```
        System.out.println("计算圆面积=" + String.format("%.2f", Math.PI*r*r));
```

```
        System.out.println("版本号=" + VERSION);
```

String.format用于格式化数据

```
    }
```

```
}
```

请输入圆的半径r=10
计算圆面积=314.16
版本号=1.0

- 其他类型数据的读取: nextInt() ★ 、 nextByte()、nextFloat()、nextLong()
- nextLine(): 读取字符串数据(以回车结束) ★
- nextBoolean(): 读取boolean数据, 输入true或false

控制台输入输出程序 -- 一次输入多个数据

HelloJava.java

```
package edu.wust;  
  
import java.util.Scanner; //导入Scanner类  
  
public class HelloJava {  
    public static void main(String[] args) {  
        System.out.println("请输入若干数，#结束输入");  
        double sum = 0;    int count = 0;  
        Scanner sc = new Scanner( System.in );  
        while ( sc.hasNextDouble() ) {  
            sum += sc.nextDouble();  
            count++;  
        }  
        System.out.println( String.format("%d个数的平均值=%.2f", count,sum/count) );  
    }  
}
```

← 推荐用回车分割数据
(空格也行)

hasNextDouble(): 判断是否还有下一个double数据

请输入若干数，#结束输入

1.2

3.4

5

#

3个数的平均值=3.20

【返回】

2. 多个类的程序

涉及访问权限问题

- Java程序都是以类来组织，类名前可加 `public` 也可不加 (通常加`public`)
- 一个源文件可有多类
- 但一个源文件中只能含有一个标记为`public`的类
- 加了`public`的类其源文件名(`***.java`)必须跟这个类的名字相同
- `main`函数通常放在`public`类中
- 通常推荐一个类单独保存为一个文件
- 每个类编译完成后会生成各自的字节码(`.class`)文件

示例1：多个类写在同一个源文件

```
package edu.wust;
```

```
class A {
```

```
    private int value; //成员变量
```

```
    public A() {        //默认构造函数
```

```
}
```

```
    public A(int value) { //构造函数(重载)
```

```
        this.value = value;
```

```
}
```

```
    public void foo() {    //自定义方法
```

```
        System.out.println("value=" + value);
```

```
}
```

```
@Override
```

```
    public String toString() { //重写toString方法
```

```
        return "[value=" + value + "];
```

```
}
```

```
}
```

一个源文件只能有一个public类

```
public class HelloJava {
```

```
    public static void main(String[] args) {
```

```
        A a1 = new A();
```

```
        a1.foo();    // value=0
```

```
        A a2 = new A(100);
```

```
        System.out.println(a2); //自动调用a2.toString()
```

```
}
```

```
}
```

源文件名必须跟public类同名

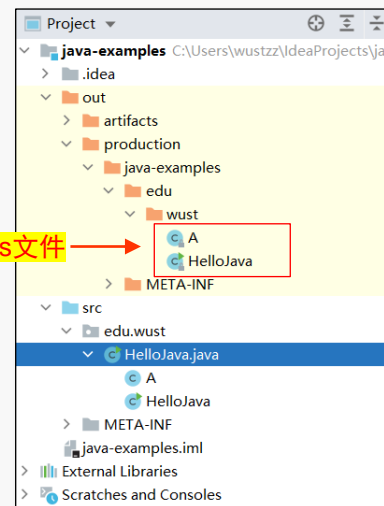
HelloJava.java

main函数通常放在public类中

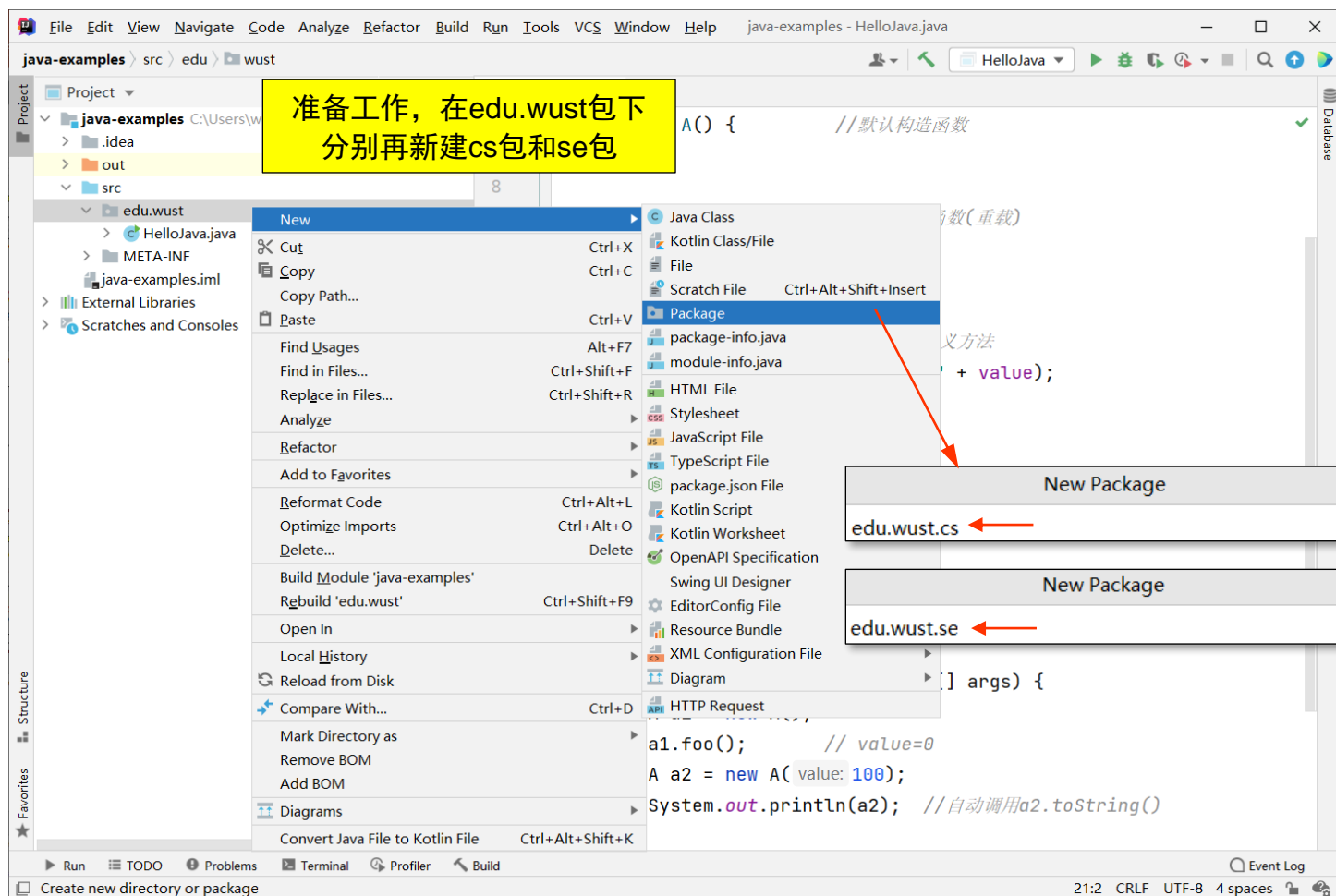
运行结果

```
value=0  
[value=100]
```

每个类都会生成.class文件



■ 示例2：一个类一个源文件（推荐）



java-examples > src > edu > wust > se > Main

Project: java-examples C:\Users\wustzz\IdeaProjects\java-examples

src > edu.wust > cs > A

src > se > Main

src > HelloJava.java

src > META-INF

src > java-examples.iml

src > External Libraries

src > Scratches and Consoles

A.java

```
1 package edu.wust.cs;
2
3 public class A {
4     private int value;
5
6     public A() {
7
8     }
9
10    public A(int value) { this.value = value; }
11
12    public void foo() { System.out.println("value=" + value); }
```

Main.java

```
1 package edu.wust.se;
2
3 import edu.wust.cs.A;
4
5 public class Main {
6     public static void main(String[] args) {
7         A a1 = new A();
8         a1.foo();
9         A a2 = new A(100);
10        System.out.println(a2);
11    }
```

Annotations:

- edu.wust.cs包
- 新建的类默认都是public
- // 成员变量
- // 默认构造函数
- edu.wust.se包
- 重要指令: import 包名.类; (导入想用的public类) 由于A类和Main类分属不同包, 要使用就必须导入

Summary:

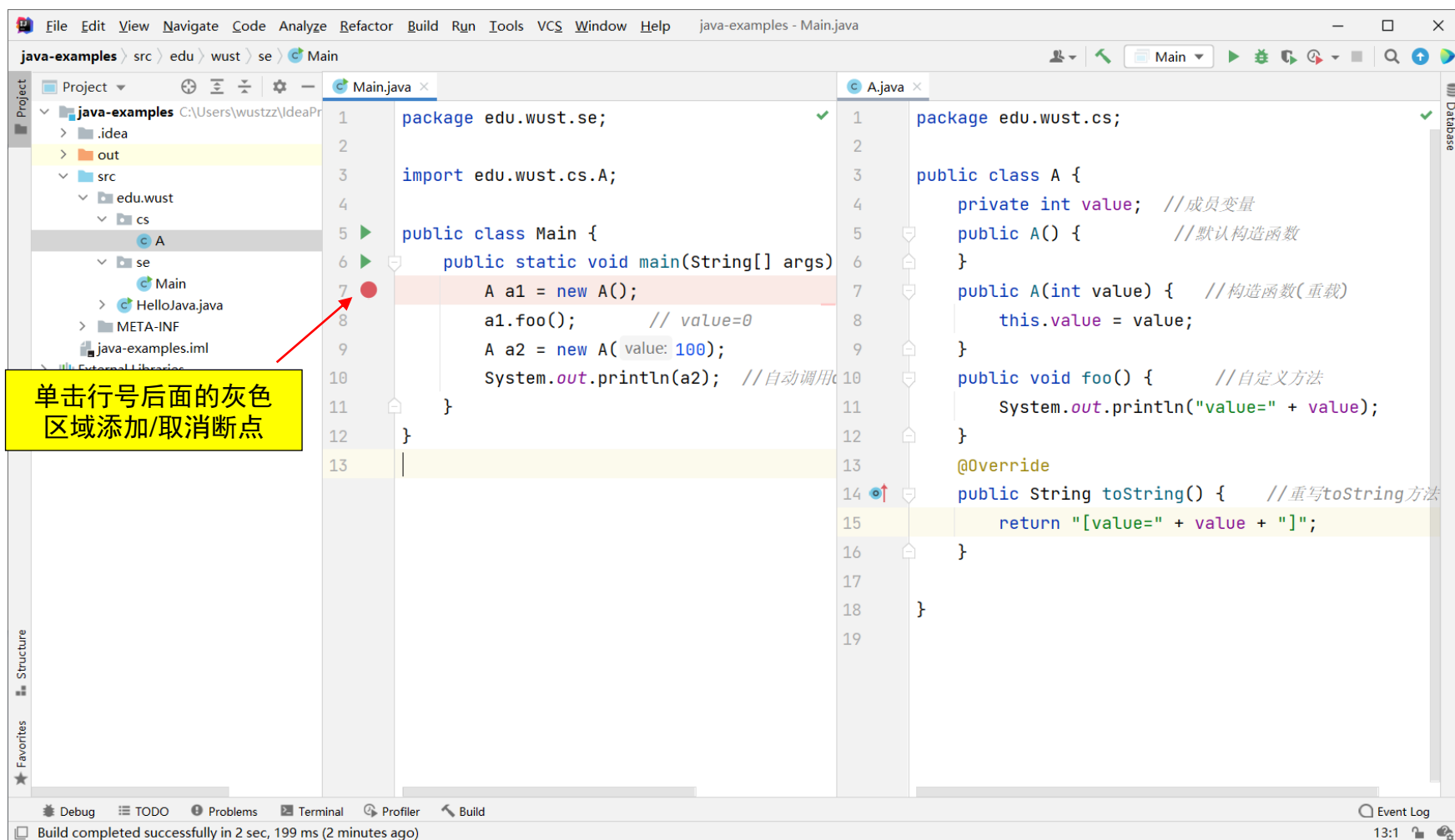
- 一个类一个源文件
- 在cs包中新建A类
- 在se包中新建Main类
- 代码同前例

Run | TODO | Problems | Terminal | Profiler | Build

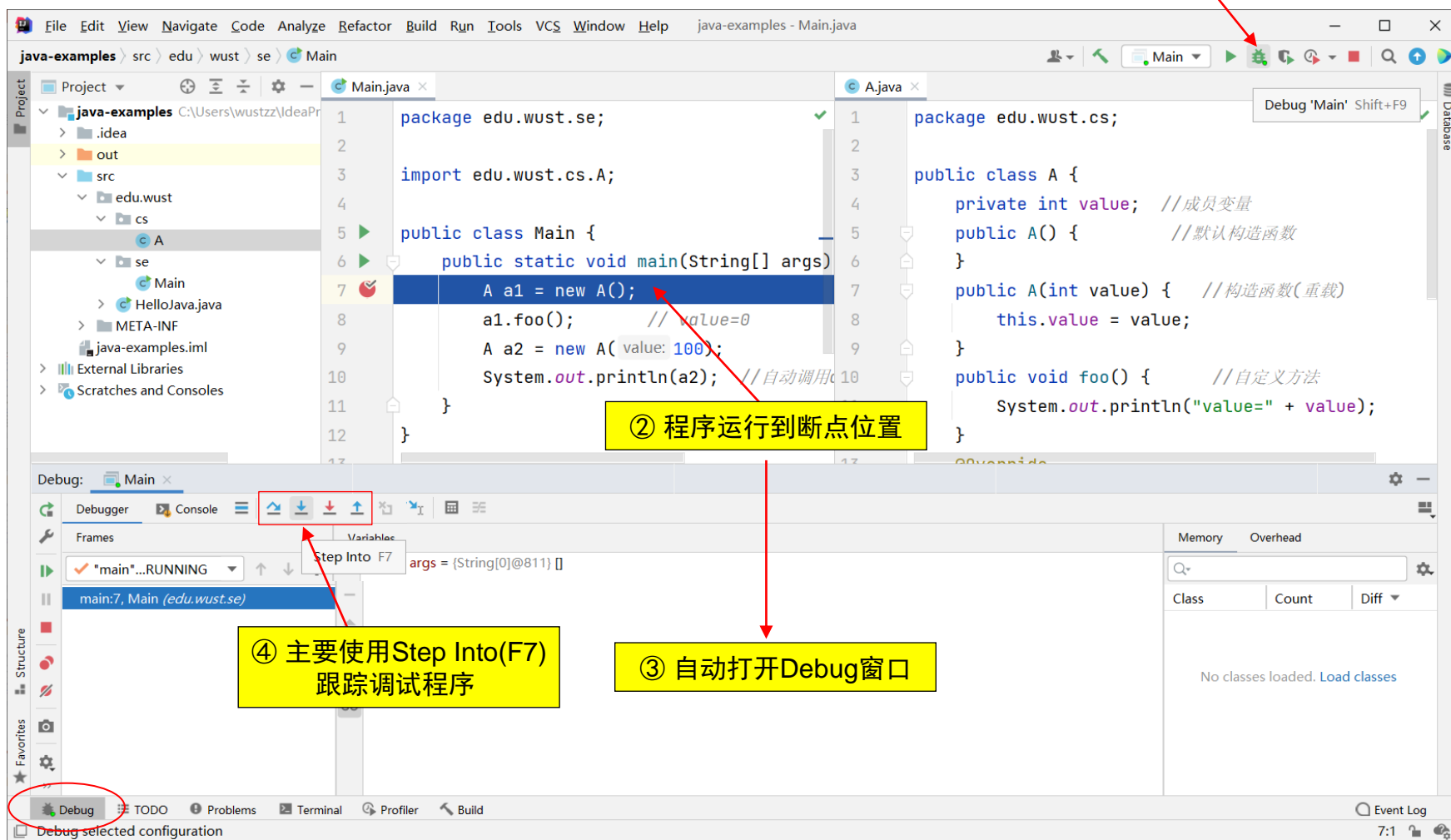
Build completed successfully in 1 sec, 215 ms (21 minutes ago)

3:22 CRLF UTF-8 4 spaces

3. Java程序调试



程序调试示意



附录：Idea常用快捷键

- 代码对齐：Ctrl+Shift+L
- 代码提示：Alt+/
- Ctrl+D：复制行
- Ctrl+X：剪切行
- Ctrl+/ 或 Ctrl+Shift+/：注释（//或者/**/）。
- Ctrl+F：查找
- Ctrl+R：替换
- Ctrl+Z：撤销
- Ctrl+单击：查看源码定义
- Ctrl+H：显示类结构图（类的继承层次）。

【完】