

第一章 JAVA 基础（一）

课程目标

- 1、JAVA 发展概述
- 2、JAVA 语言特点
- 3、开发环境搭建（下载、安装）
- 4、开发第一个 JAVA 程序
- 5、STS 开发第一个 JAVA 程序
- 6、JAVA 变量与数据类型

课程内容

1、JAVA 发展概述

1.1.2背景

Java是由Sun Microsystems公司推出的Java面向对象程序设计语言（以下简称Java语言）和Java平台的总称。Java

Java由James Gosling和同事们共同研发，并在1995年正式推出。Java最初被称为Oak，是1991年为消费类电子产品的嵌入式芯片而设计的。1995年更名为Java，并重新设计用于开发Internet应用程序。用Java实现的HotJava浏览器（支持Java applet）显示了Java的魅力：跨平台、动态Web、Internet计算。从此，Java被广泛接受并推动了Web的迅速发展，常用的浏览器均支持Javaapplet。另一方面，Java技术也不断更新。Java自面世后就非常流行，发展迅速，对C++语言形成有力冲击。在全球云计算和移动互联网的产业环境下，Java更具备了显著优势和广阔前景。2010年Oracle公司收购Sun Microsystems。

2、JAVA 语言特点

1. 面向对象

Java 是一种面向对象的语言，这里的对象是指封装数据及其操作方法的程序实体。

2. 平台可移植性

Java 程序具有与体系结构无关的特性，从而使 java 程序员可以方便的移植到网络中的不同计算机中，java 的类库中也实现了针对不同的接口，使这些类库也可以移植。

3. 分布性

Java 的分布性包括操纵分布和数据分布，其中操作分布是指在多个不同的主机上布置相关的操作，数据分布是指将数据分别存放在不同的主机上，这些主机是网络中的不同成员，java 可以拼接 url 对象访问网络对象，访问的方式与访问的本地系统相同。

4. 多线程

Java 具有多线程机制，这使得应用程序可以并行的执行，它的同步机制也保证了对共享数据的共享操作，而且线程具有优先级的机制，有助于分别使用不同线程的完成特定的行为，也提高了交互的实时响应能力。

5. 安全性

Java 程序代码要经过代码校验、指针校验等很多的测试步骤才能运行，所以未经过允许的 java 程序不可能出现损害系统平台的行为，而且使用 java 可以编写防病毒和防修改的系统。

6、工作方式

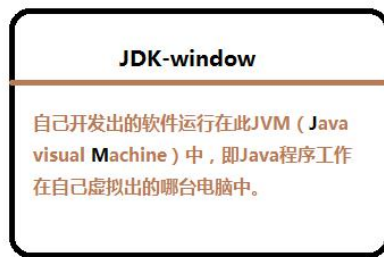
1. 编写源程序（用记事本编写源程序，以 .java 保存）

2. 编译源程序（使用 javac 命令编译文件，生成 .class 文件）

3. 运行（使用 java 命令运行 .class 文件，输出程序结果）

补充知识点：

Java 跨平台原理：（重要）



java跨平台原理：

① 因为java程序是运行在JVM这台虚拟的电脑中，所以与其所在的操作系统无关，哪么，这样就实现跨平台，也就是说与所在的操作系统无关。

3、开发环境搭建（下载、安装）

下载：

从 orace 公司的官网，下载自己需要的 JDK，下载时需注意：

- ① 与自己的操作系统匹配。
- ② 与操作系统的位数匹配（32 位与 64 位）
- ③ 选择版本号。

安装：

根据安装界面，选择“下一步”完成安装即可。

4、开发第一个 JAVA 程序（最重要）

准备工作：

安装完 JDK 环境后，要在 path 环境变量中添加你安装的 jdk 的路径（一直到 bin 目录，如：D:\Java\jdk1.8.0_131\bin）

注意事项：

- ① 在计算机的“系统属性”→“高级”→“环境变量”，在“系统变量”里面的 path 中添加你电脑上安装的 jdk 的目录路径。这样添加的 path 环境变量可以供任何用户使用，而在“Administrator 用户的环境变量”中定义的 path 只能代表该用户可以使用。

② 通过 cmd 窗口中查看到的 path 变量实际上是两部分并集（Administrator+系统变量）

③ 如何知道是否修改成功？

重新打开 cmd 窗口，输入 `java -version` 看到类似下面的界面代表配置成功：

```
C:\Users\Administrator.USER-20161129NL>java -version
java version "1.8.0_131"
Java(TM) SE Runtime Environment (build 1.8.0_131-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)
```

开发第一个 java 程序：

第一步： 定义一个 java 源程序（后缀名为.java）

第二步： 使用 `javac` 命令将 java 源程序编译为.class 字节码文件。

第三步： 使用 `java` 命令执行这个字节码文件即可。



归纳为： 编译→执行

步骤一：

```
public class HelloWorld{
    public static void main(String[] args){
        System.out.println("hello,大家好！");
    }
}
```

HelloWorld.java源程序

步骤二：

 HelloWorld.class	2018/3/26 15:06	CLASS 文件	1 KB
 HelloWorld.java	2018/3/26 15:05	JAVA 文件	1 KB

将java源程序经过编译后生成的字节码文件。

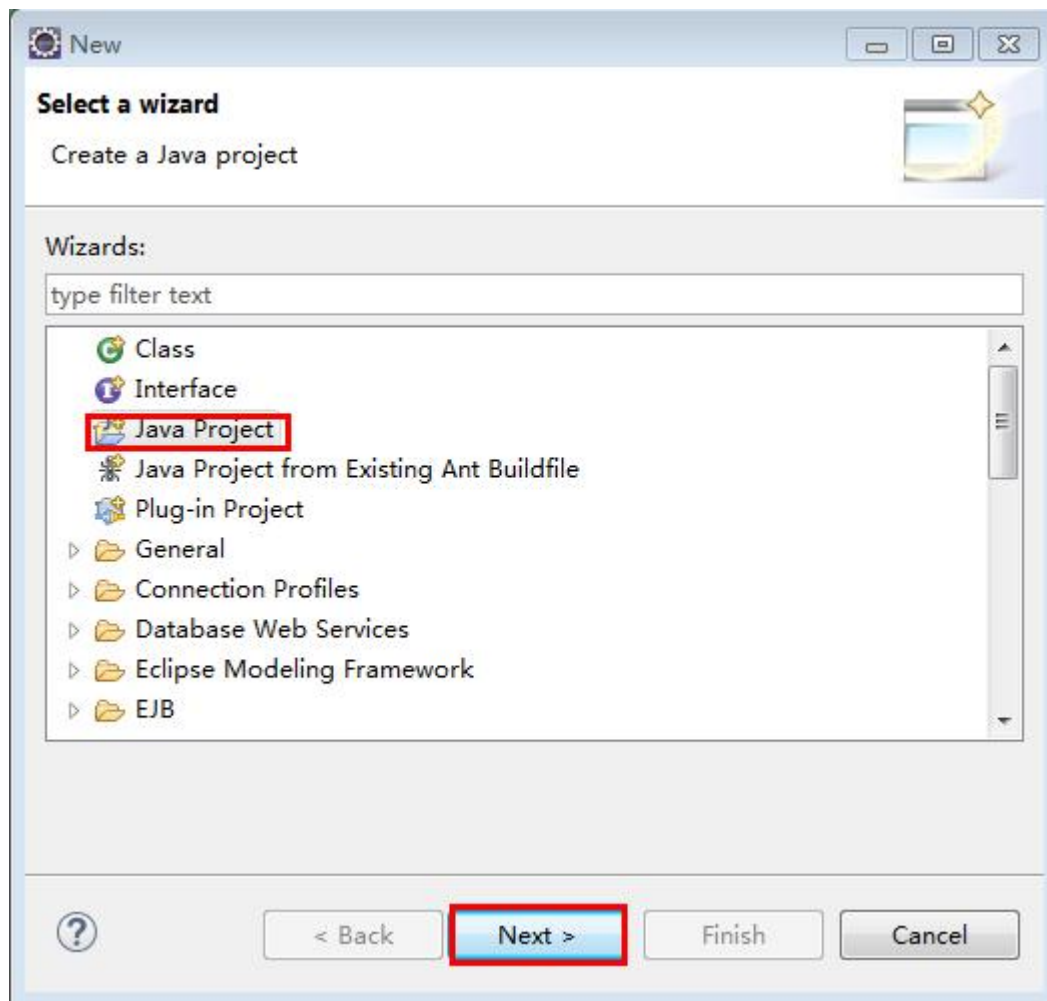
步骤三：

```
C:\java>java HelloWorld
hello,大家好!
```

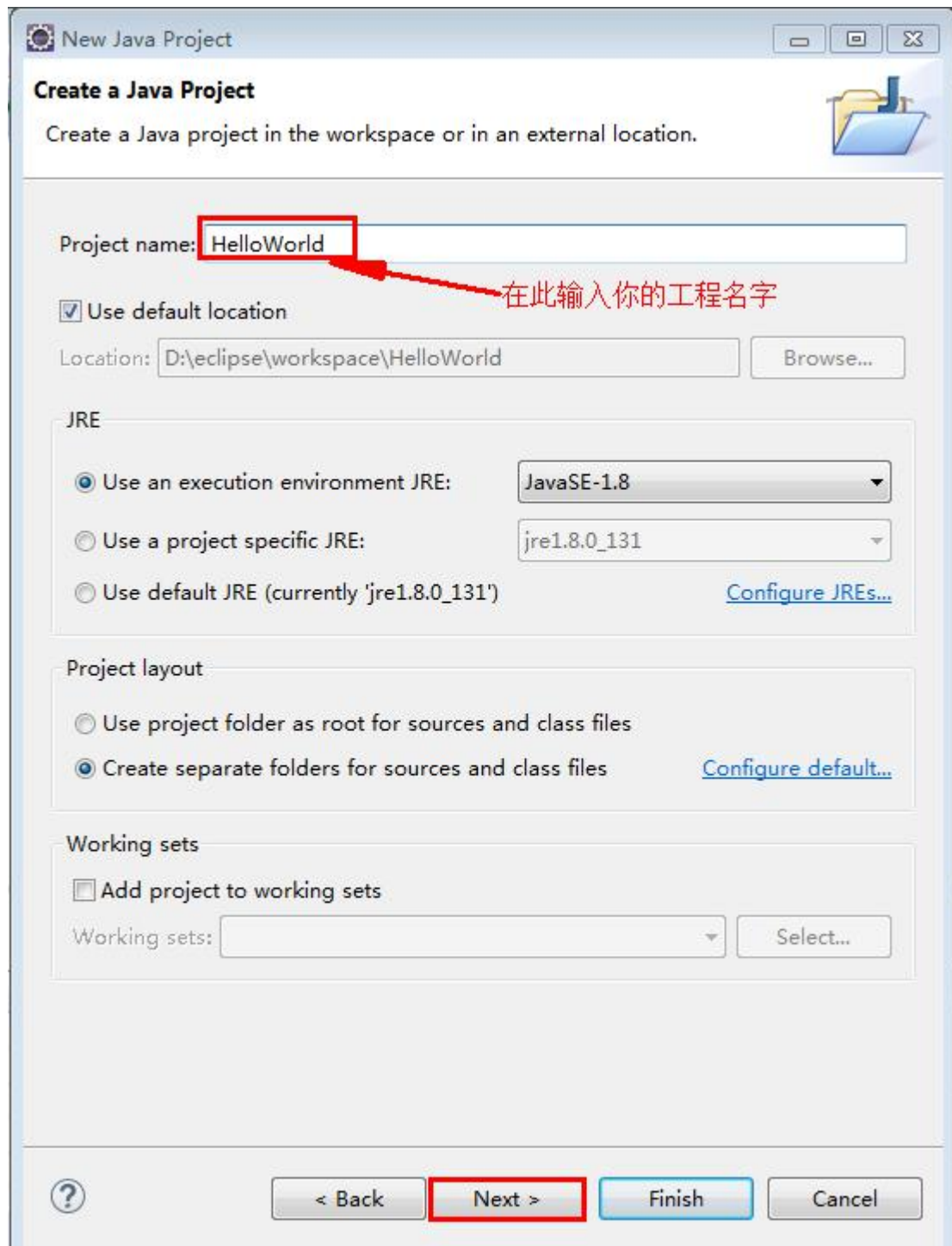
执行字节码文件

5、STS 开发第一个 JAVA 程序

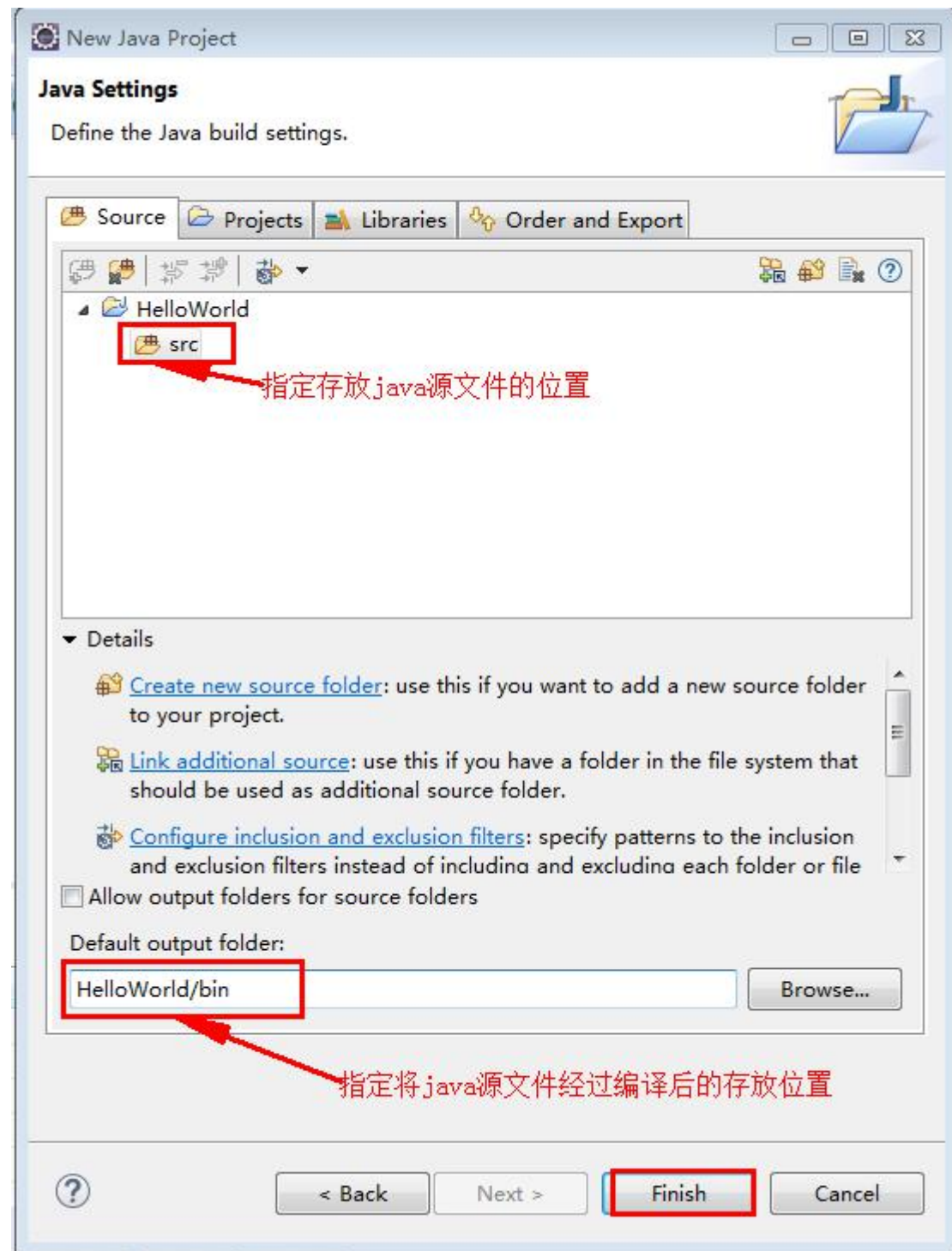
第一步：新建 java 工程（项目）：



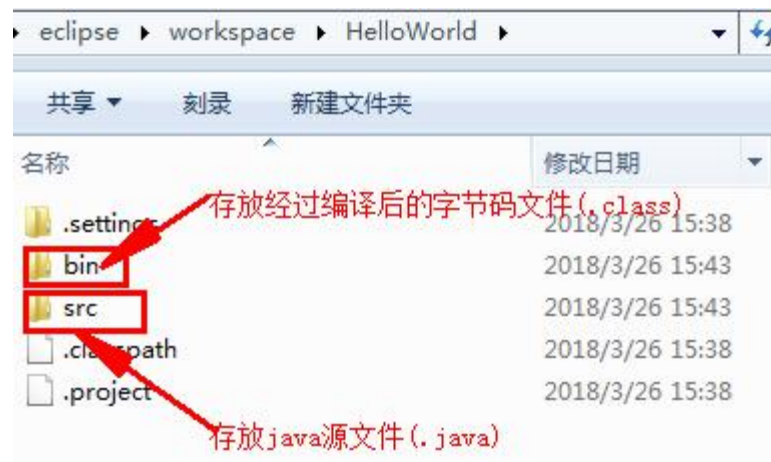
第二步：指定工程名称：



第三步：指定 java 源文件及编译后的字节码文件存放位置：



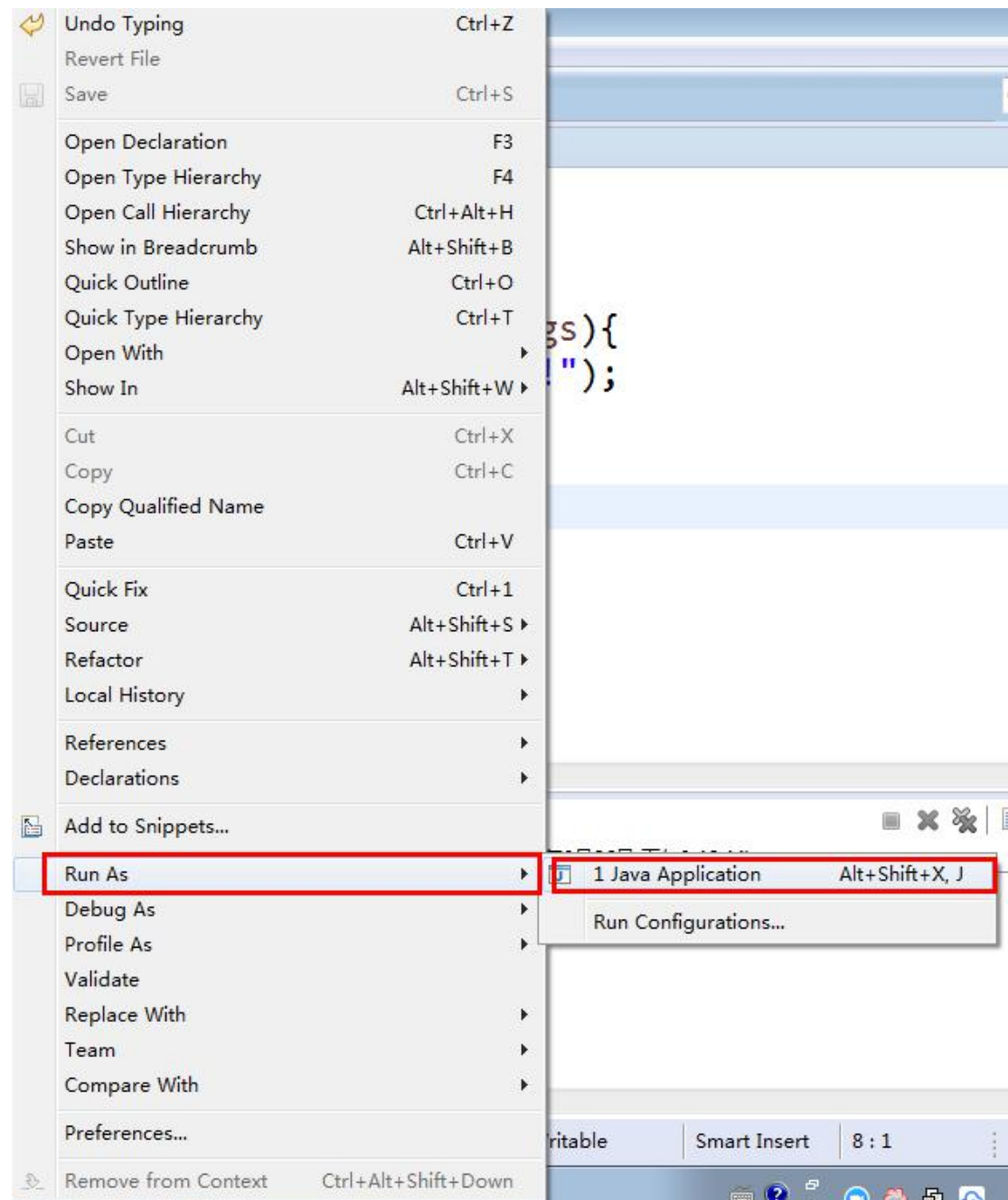
第四步：完成后的 java 工程的目录结构：



第五步：编写你的 java 源程序：

```
public class FirstApp {  
    public static void main(String[] args){  
        System.out.println("hello,world!");  
    }  
}
```


第六步：运行 java 源程序：



快捷键： **Alt + Shift + x, j**（先同时按下 Alt + Shift + X，释放后再按下 j 键）

6、JAVA 变量与数据类型

Java 变量： 变量就是内存中的一块临时存放数据的区域。

6.1)、Java 变量的命名规则：（法律、法规）

- ① java 的变量名必须是由**字母、数字、下划线、\$**这四种字符组成。
- ② Java 变量名的首字符不能是数字。
- ③ Java 变量名区分大小写，如:name 与 Name 代表两个不同的变量。
- ④ Java 变量名不能是系统保留字，如：main public static void class 等等。

6.2)、Java 变量的命名规范：（良好的编程习惯）

- ① 使用驼峰式命名法。
首个单词的每个一字母小写，其它每个单词的首字母大写。如：helloWorld...
- ② 尽量做到望文生义。
如：人名：name 年龄：age 住址：addr...

6.3) 如何在 java 中定义变量：

```
public static void main(String[] args) {  
    //声明变量的语法：  
    //→ 变量类型 变量名；  
    //1、先声明变量再为变量赋值（适合于动态 赋值）  
    int age;  
    age = 20;  
    //2、声明变量的同时为其赋值  
    String name = "张三";  
    System.out.println(age);  
    System.out.println(name);  
}
```

6.4) java 的 8 种基本数据类型：

整型：(四种)

byte (字节型，占用一个字节)：所占空间： $2^8 = 256$,因其有负数：-128-127

short （短整型，占用两个字节)：所占空间： $2^{16} = 65536$,因其有负数：-32768-32767

int （整型，占有四个字节)：所占空间： 2^{32} ,因其有负数，-2147483648-2147483647

Long （长整型，占有八个字节)：所占空间： 2^{64} ，因其有负数，

-9223372036854775808 --9223372036854775807

浮点型（两种）：

float: 单精度浮点型，占有四个字节，精度为 7 位有效数字。

double: 双精度浮点型，占有八个字节，精度为 15 位有效数字。

布尔型(boolean):

只能取两个值：true 或 false, 占 1 个字节。

字符型(char):

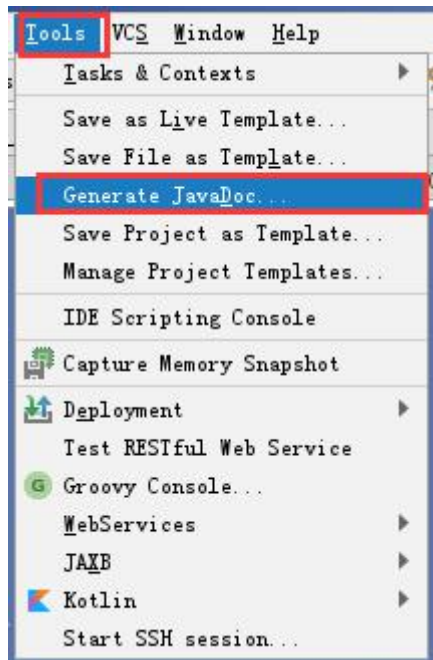
占有两个字节，可以存放一个汉字。

补充知识点：（idea 中）

1、定义一个 javadoc 文档：

```
/**
 * @author: Feng. Wang
 * @Company: Zelin. ShenZhen
 * @Description: java 注释
 * @Date: Create in 2019/4/22 16:48
 * @version : 1.0.0
 */
public class Lesson1_04 {
    public static void main(String[] args) {
        System.out.println("hello,这是单行注释"); // 1. 这里代表的单行注释
        /*
        2. 这里代表的是多行注释
        int a = 10;
        a = 5;
        int b = 55;
        */
    }
    /**
     * @param age 要显示的年龄
     * @return 显示的个人信息
     */
    public String sayHello(int age) {
        return "年龄: " + age;
    }
}
```

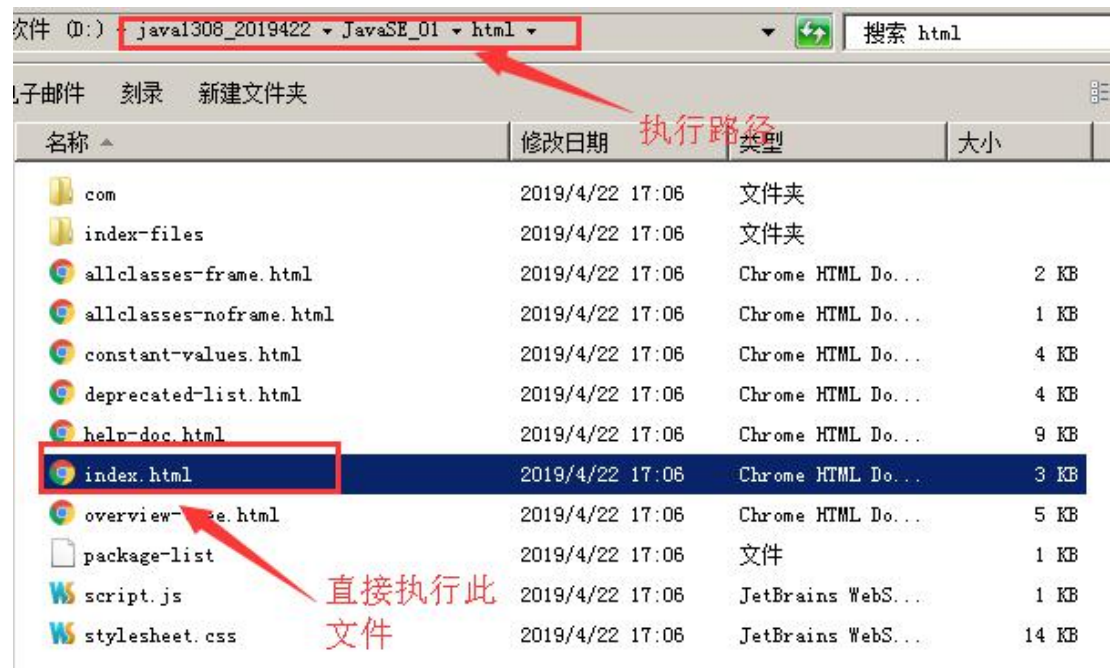
2、在 idea 中对文档注释进行输出：



3、在上面打开的工具中对输出进行设置:



4、输出后的目录结构如下：



6.5) java 数据类型使用：

整型数据类型：

```
// 整型数据类型的使用
public static void main(String[] args) {
    // byte 类型使用：（字节型）
    byte a = 120;
    System.out.println(a);
    // 得到 byte 类型的最大值
    byte maxByte = Byte.MAX_VALUE;
    // 得到 byte 类型的最小值
    byte minByte = Byte.MIN_VALUE;
    System.out.println("maxByte:" + maxByte);
    System.out.println("minByte:" + minByte);

    // 短整型
    System.out.println("-----");
    // 查看 short 的最大值
    short maxShort = Short.MAX_VALUE;
    // 查看 short 的最小值
    short minShort = Short.MIN_VALUE;
    System.out.println("maxShort : " + maxShort + ", minShort: " + minShort);
    // short b = 32768; // 错误：原因是超过了 short 的范围
    // short b = -32769; // 错误：原因是超过了 short 的范围

    System.out.println("-----");
}
```



```
// 整型：
// 查看整型的最大值与最小值范围：
int maxInt = Integer.MAX_VALUE;           // 整型的最大值
int minInt = Integer.MIN_VALUE;           // 整型的最小值
System.out.println("maxInt:" + maxInt + ",minInt:" + minInt);

System.out.println("-----");
// 长整型：
// 查看长整型的最大值与最小值范围：
long maxLong = Long.MAX_VALUE;             // 长整型的最大值
long minLong = Long.MIN_VALUE;             // 长整型的最小值
System.out.println("maxLong:" + maxLong + ",minLong:" + minLong);

}
```

浮点类型：

```
// 浮点数据类型使用
public static void main(String[] args) {
    // java中默认的小数类型为double类型
    double d = 3.144234254234234234;
    System.out.println("d = " + d);
    // 声明float类型时，注意要在其后加上f(重点)
    float f = 3.144234254234234234f;
    System.out.println("f = " + f);
    // float类型：
    float floatMax = Float.MAX_VALUE;       // 最大值
    float floatMin = Float.MIN_VALUE;       // 最小值
    System.out.println("floatMax:" + floatMax + ",floatMin:" + floatMin);
    System.out.println("-----");
    // double类型：
    double doubleMax = Double.MAX_VALUE;    // 最大值
    double doubleMin = Double.MIN_VALUE;    // 最小值
    System.out.println("doubleMax:" + doubleMax + ",doubleMin:" + doubleMin);
}
```

字符类型与布尔类型：

```
// 布尔类型与字符型
public static void main(String[] args) {
    // 布尔类型：(只能取true或false)
    boolean b = true;
    System.out.println("b = " + b);
    // 字符型：(一定用单引号表示，不能用双引号，占两个字节)
    char a = '人';
    System.out.println(a);

}
```


4、运算符与表达式

4.1) 算术运算符

4.1.1) 双目运算符：(+ - * / %)

```
//算术运算符用法
public static void main(String[] args) {
    //1. 双目运算符(+ - * / %) : 由两个运算数
    int a = 10;
    int b = 3;
    System.out.println("a + b = " + (a + b));
    System.out.println("a - b = " + (a - b));
    System.out.println("a * b = " + (a * b));
    System.out.println("a / b = " + (a / b));
    System.out.println("a % b = " + (a % b));
}
```

运行结果如下：

```
a + b = 13
a - b = 7
a * b = 30
a / b = 3
a % b = 1
```

4.1.2) 单目运算符：(++ --) (重难点)

```
//单目运算符的用法(++ --)
public static void main(String[] args) {
    int a = 10;
    int b = 5;
    //++运算符
    //情况一：前加
    //1.1) 单独使用
    ++a; //相当于: a = a + 1;
    System.out.println("a = " + a);
    //1.2) 与表达式一起使用
    int c = ++a; //相当于: ① a = a + 1; (先自增再赋值) ② c = a;
    System.out.println("a = " + a + ", c = " + c);
    //情况二：后加
    //1.1) 单独使用
    a++; //相当于: a = a + 1;
    System.out.println("a = " + a);
    //1.2) 与表达式一起使用
    c = a++; //相当于: ① c = a; ② a = a + 1; (先赋值再自增)
    System.out.println("a = " + a + ", c = " + c);
}
```

```
//--运算符
//情况一：前减
//2.1)单独使用时
--a;           //相当于：a = a - 1;
System.out.println("a = " + a);
//2.2)与表达式一起使用时
c = --a;       //相当于：① a = a - 1; ② c = a; (先自减再赋值)
System.out.println("a = " + a + ",c = " + c);

//情况二：后减
//2.1)单独使用时
a--;           //相当于：a = a - 1;
System.out.println("a = " + a);
//2.2)与表达式一起使用时
c = a--;       //相当于：① c = a; ② a = a - 1; (先赋值再自减)
System.out.println("a = " + a + ",c = " + c);
```

运行结果如下：

```
a = 11
a = 12,c = 12
a = 13
a = 14,c = 13
a = 13
a = 12,c = 12
a = 11
a = 10,c = 11
```

4.2) 关系运算符 (> >= < <= == !=)

```
//关系运算符(> >= < <= == !=),结果为布尔值(true false)
//适用于：①表示判断的地方，如条件结构或循环结构
//② 一般与逻辑运算表达式结合使用
public static void main(String[] args) {
    int a = 10;
    int b = 5;
    boolean c = a > b;           //关系运算符返回的是一个布尔值
    System.out.println("c = " + c);
    c = a == b;
    System.out.println("c = " + c);
    int d = a + b;
    boolean d1 = d < a - b;
    System.out.println("d1 = " + d1);
}
```

运行结果：

```
c = true
c = false
d1 = false
```

4.3)逻辑运算符:

```
/**
 * 逻辑运算符( && || !):组合多个关系运算表达式,返回的是一个布尔值
 * @author Administrator
 *
 */
public class Lesson1_07 {

    public static void main(String[] args) {

        int a = 10;
        int b = 4;
        int c = 5;
        int d = 8;

        //1、逻辑与(&&)
        //1.1)基本用法:
        //原理: 当多个表达式参与运算时, 只有所有的表达式都成立时, 整个表达式才返回 true, 否则, 返回 false
        boolean bb = a > b && c > d && b < d;
        System.out.println("bb = " + bb);
        //1.2)短路用法:
        //原理: 当多个表达式参与运算时, 只要有一个表达式返回 false, 整个表达式的结果就为 false, 其后的表达式不再执行
        boolean cc = a++ > --b && c++ > d-- && --b < d++;
        System.out.println("a = " + a + ",b = " + b + ",c = " + c + ",d = " + d + ",cc = " + cc);

        //2、逻辑或(||)
        //2.1)基本用法:
        //原理: 当多个表达式参与运算时, 只要有一个表达式为 true, 整个表达式就为 true, 只有所有的都为 false, 整个表达式才会为 false
        boolean dd = a > b || c > d || b < d;
        System.out.println("dd = " + dd);
        //2.2)短路用法:
        //原理: 当多个表达式参与运算时, 只要有一个表达式返回 true, 整个表达式就为 true, 其后的表达式不再执行
        boolean ee = a++ < --b || ++c >= d-- || a-- > d++;
        System.out.println("a = " + a + ",b = " + b + ",c = " + c + ",d = " + d + ",ee = " + ee);

        //3. 逻辑非(!)
        ee = !ee;
        System.out.println("ee = " + ee);
    }
}
```

```
    }  
}
```

运行效果如下：

```
bb = false  
a = 11,b = 3,c = 6,d = 7,cc = false  
dd = true  
a = 12,b = 2,c = 7,d = 6,ee = true  
ee = false
```

4.4)赋值运算符：

```
/**  
 * 赋值运算符：(= += -= *= /= %=  
 * @author Administrator  
 *  
 */  
public class Lesson1_08 {  
  
    public static void main(String[] args) {  
        //1.简单赋值运算符(=)  
        int a = 10;  
        //2.复合赋值运算符(+= -= *= /= %=  
        int b = 5;  
        b += 2;    //相当于: b = b + 2;  
        System.out.println("b = " + b);  
        b -= 1;    //相当于: b = b - 1;  
        System.out.println("b = " + b);  
        b *= 5;    //相当于: b = b * 5;  
        System.out.println("b = " + b);  
        b /= 2;    //相当于: b = b / 2;  
        System.out.println("b = " + b);  
        b %= 3;    //相当于: b = b % 3;  
        System.out.println("b = " + b);  
    }  
}
```

运行效果如下：

```
b = 7  
b = 6  
b = 30  
b = 15  
b = 0
```

4.5)运算符的优先级：

运算符的优先级：

① ()

② ++ -- !

算术运算符

③ * / %

④ + -

关系运算符

⑤ > >= < <= == !=

逻辑运算符

⑥ &&

⑦ ||

赋值运算符

⑧ = += -= *= /= %=