实验1 基础环境

1 实验目的

1、熟悉Java开发环境的配置和使用
2、熟悉开发和调试技巧
3、熟悉Java基本语法

2 实验环境

开发环境：JDK 8.0（或更高版本）
开发工具：Eclipse
设计工具：StarUML（或PlantUML等其他工具）

3 实验内容

3.1 安装JDK

（1）下载JDK。学习过程中既可以选择开源版本（OpenJDK），也可以选择Oracle的商业版本。

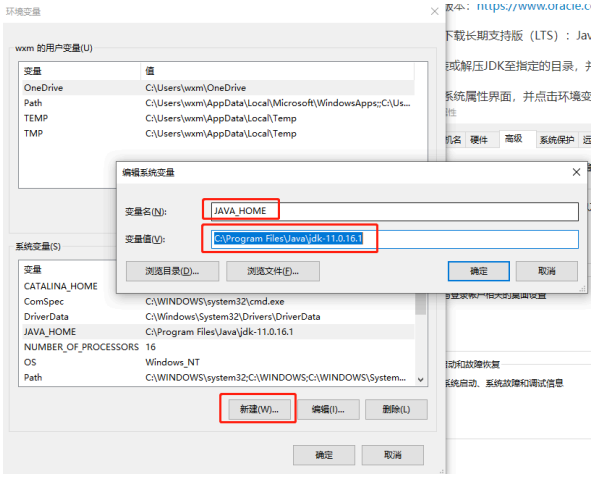
- OpenJDK地址：https://jdk.java.net/
- 商业版本：https://www.oracle.com/java/technologies/downloads/

这里推荐下载长期支持版（LTS）：Java SE 8、Java SE 11、Java SE 17

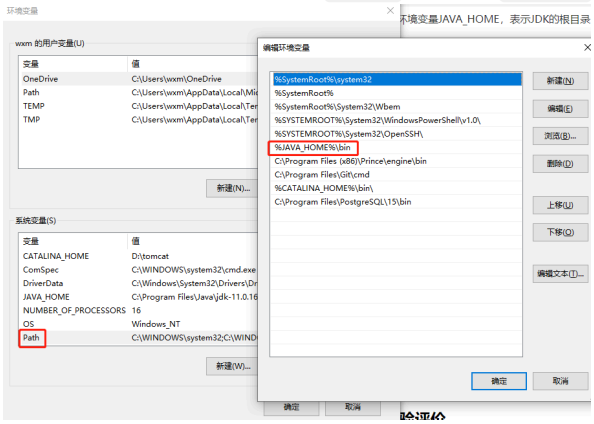
（2）安装或解压JDK至指定的目录，并配置`javac`和`java`两个命令的路径。

- 打开系统属性界面，并点击环境变量。

- 新建环境变量JAVA_HOME，表示JDK的根目录，以便简化其他变量配置

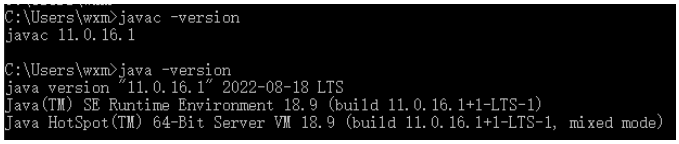


- 选择系统路径环境变量`Path`，添加`javac`和`java`两个命令的路径



（3）命令行检测环境。

- 查看编译命令版本：javac -version
- 查看运行命令版本：java -version



3.2 第一个Java程序

（1）创建文本文件`Test.java`用于编写源代码

（2）打开文本编辑器，输入以下代码：

```
public class Test{
  public static void main(String[] args){
    System.out.println("Hello");
  }
}
```

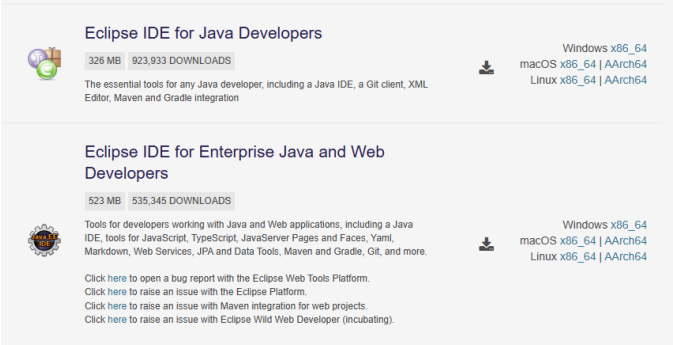
（3）在命令行进入源代码所在目录，用`javac Test.java`命令编译。编译后查看当前目录下是否生成字节码文件`Test.class`。

（4）使用`java Test`命令运行程序

3.3 使用集成开发环境

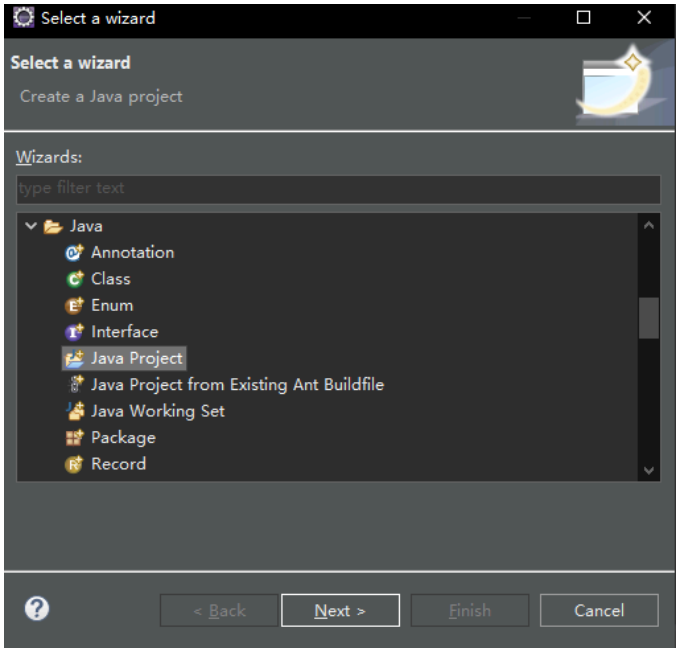
主流的Java集成开发环境包括`Intellij IDEA`、`Eclipse`、`NetBeans`等，本实验主要介绍`Eclipse`。

（1）下载合适的Eclipse版本。面向Java开发的版本包括标准版和Java EE版，后者增加了Web开发的插件，因此，普通学习阶段下载标准版即可。

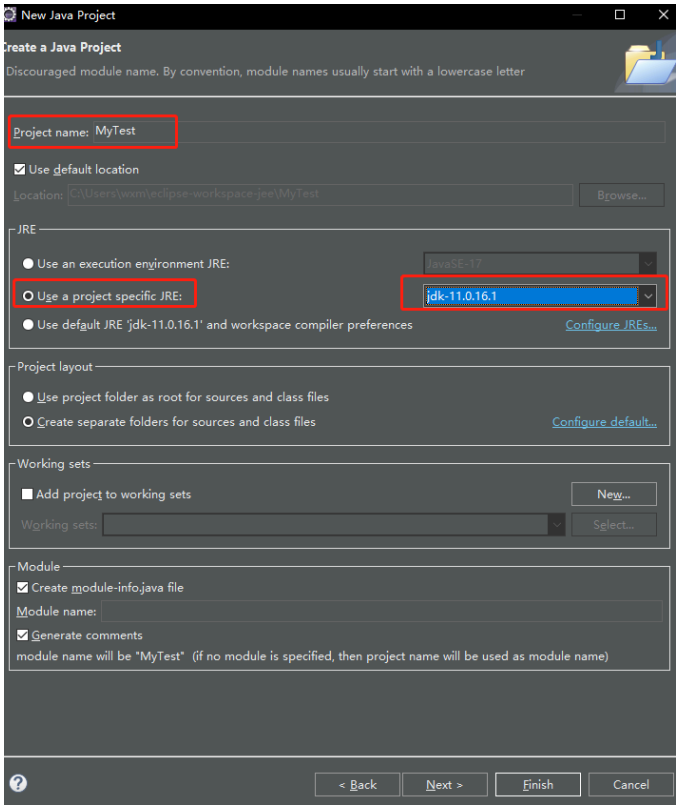


（2）本地安装好Java环境后，建议直接下载压缩包版本的Eclipse，解压后即可运行。

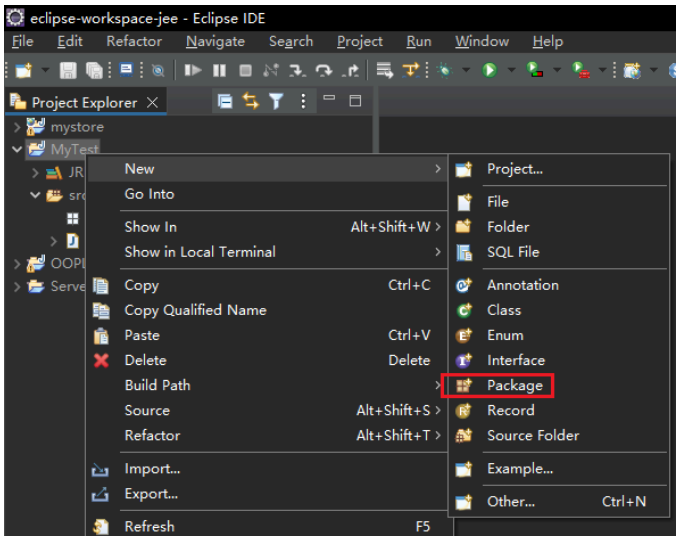
（3）打开Eclipse，选择新建Java项目`Java Project`。



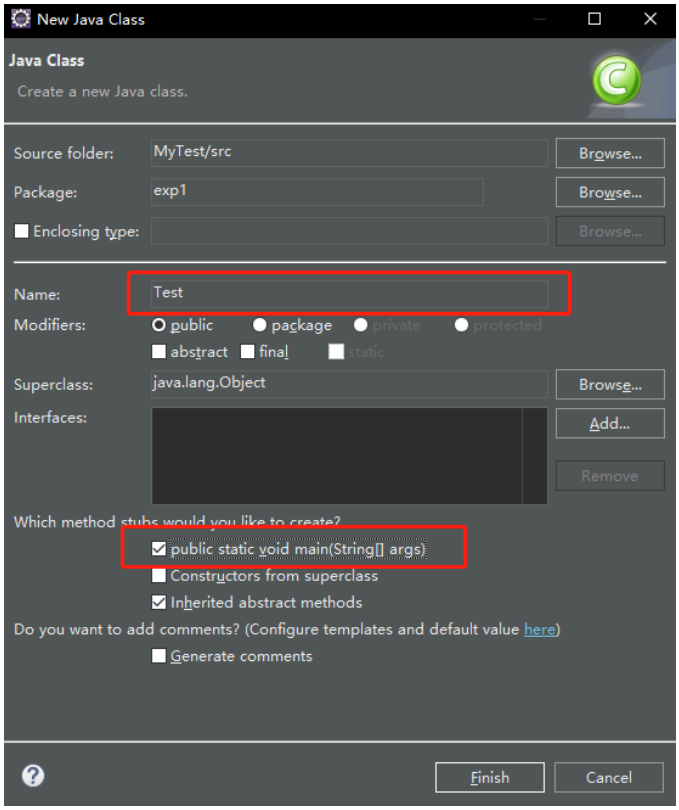
（4）编辑项目信息，填写项目名称，选择JRE运行环境，这里推荐自己安装的环境。



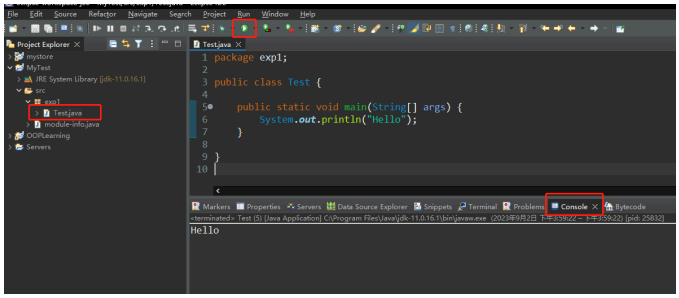
（5）在项目下新建一个包，可以将整个学期实验看作一个项目，并针对当前实验课的代码新建一个包。



（6）在包下面新建一个类



（7）编写源代码，编译运行。代码编写保存后会自动编译，因此代码编写完成保存后只需要点击运行按钮。查看标准输出的结果一般在Console栏查看。



3.4 调试程序

为确保程序逻辑的正确性，一般需要对程序进行调试，简单的调试就是根据程序执行过程中输出一些中间结果来判断程序逻辑，IDE也提供了Debug功能进行更高效的调试。以输出`10`次字符串为例：

```
public class Test {
  public static void main(String[] args) {
    int i;
    for(i = 0; i <= 10; i++) {
      System.out.println("Java");
    }
  }
}
```

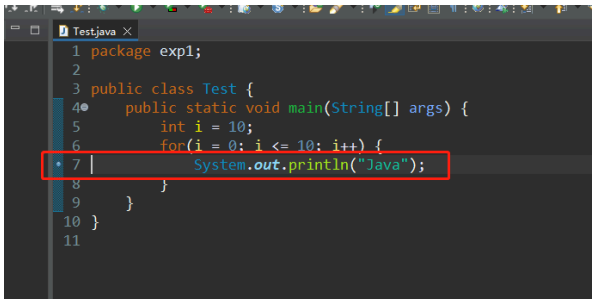
结果我们发现输出了11次字符串，需要进行调试。

（1）使用`System.out`输出中间结果。按如下代码，将`i`每次迭代过程中的值打印出来，发现输出了`0~10`。因此，问题定位到i的判断条件应该为`i<10`。

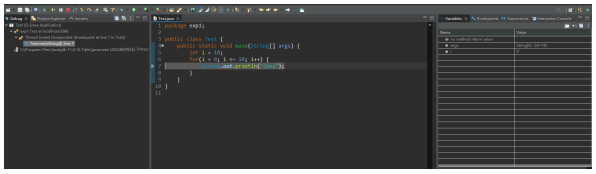
```
public class Test {
  public static void main(String[] args) {
    int i;
    for(i = 0; i <= 10; i++) {
      System.out.println("Java");
      System.out.println(i);
    }
  }
}
```

（2）使用Debug调试程序。

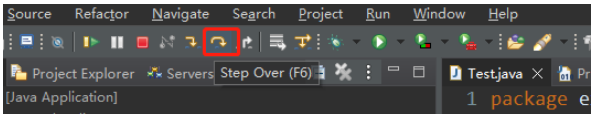
- 选择要观察的代码执行点，设置断点：双击目标行的最左边栏



- 点击`Debug Test`按钮，切换到Debug界面，可以查看到当前运行的函数、变量等信息

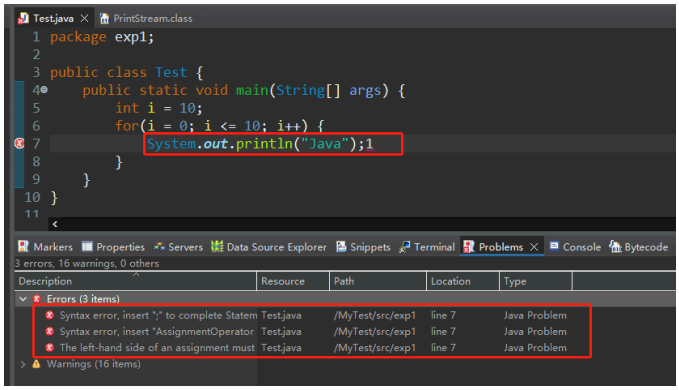


- 点击`Step Over`进行单步执行，并观察变量的变化情况来判断程序逻辑。



- 如何需要观察函数执行过程中涉及的子函数执行情况，则可以尝试`Step into`和`Step out`命令。
- 尝试通过右上角按钮切换`Debug`视图和普通视图

（3）通过`Problems`栏观察编译错误和警告。

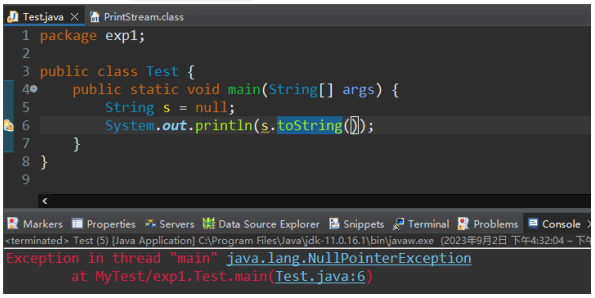


（4）通过`Console`栏观察结果或运行错误。

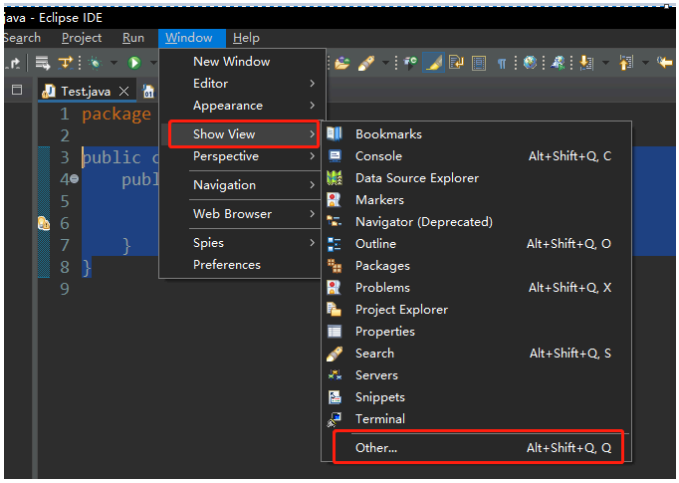
- 先编写一段可能发生运行错误的代码：

```
public class Test {
  public static void main(String[] args) {
    String s = null;
    System.out.println(s.toString());
  }
}
```

- 运行并在`Console`栏观察

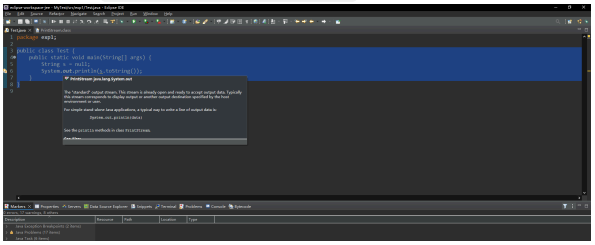


（5）一些试图或界面被关闭后可以通过`Show View`选项重新调出。

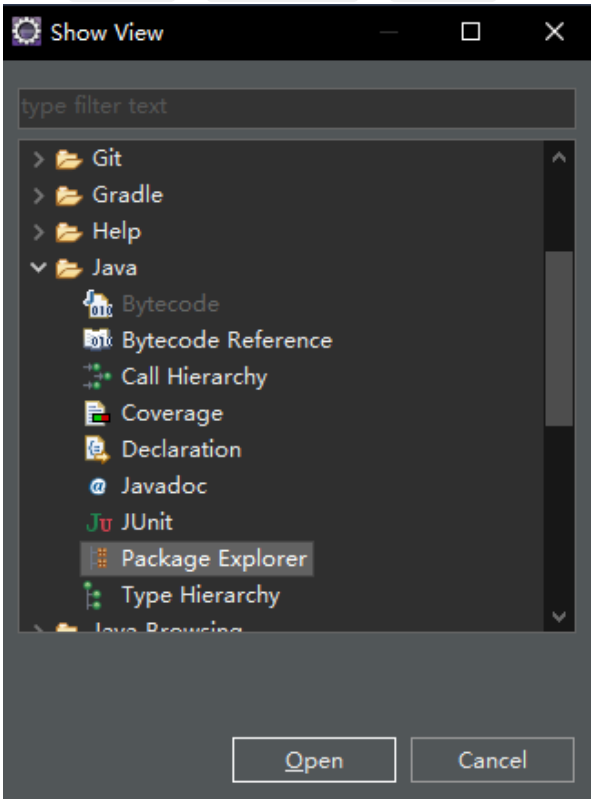


尝试关闭项目管理视图，再通过`Show View`打开。

- 先关闭`Project Explorer`，如下所示



- 通过`Window`->`Show View`->`Others`，在Java相关的视图集合下面找到目标项：



- 打开并拖动还原视图。

3.5 熟悉基础语法

（1）计算`1+1/2+1/3+...+1/n`。

- 编写代码实现计算过程，输入不同的参数观察结果正确性。

```
public class Test {
  public static void main(String[] args) {
    System.out.println(calc(5));
  }
  public static double calc(int n) {
    double value = 0.0;
    for(int i = 1; i <= n; i++) {
      value += 1.0 / i;
    }
    return value;
  }
}
```

- 采用`Debug`模式调试程序，观察`value`的变化情况。
- 思考哪些语句发生了类型转换。

（2）输入字符串，解析计算式并计算。

- 调试并运行以下代码。

```java
public class Test {
  public static void main(String[] args) {
    try (Scanner input = new Scanner(System.in)) {
      while(true) {
        try{
          System.out.print("输入计算式，操作数与操作符用空格隔开：");
          if(input.hasNextLine()) {
            String s = input.nextLine();
            System.out.println("计算结果：" + calc(s));
          }
        }catch(Exception e) {
          System.out.println("异常类型：" + e.getMessage());
        }
      }
    }
  }

  public static double calc(String calcStr) throws Exception {
    String[] items = calcStr.split(" ");
    if(items.length != 3)
      throw new Exception("计算式不正确");
    double x;
    double y;
    try {
      x = Double.parseDouble(items[0]);
      y = Double.parseDouble(items[2]);
    }catch(Exception e) {
      throw new Exception("操作数格式错误");
    }
    switch(items[1]) {
      case "+":return x + y;
      case "-":return x - y;
      case "*":return x * y;
      case "/":return x / y;
      default: throw new Exception("未知的计算符");
    }
  }
}
```

- 输入符合规范的计算式，观察结果是否正确。

```
输入计算式，操作数与操作符用空格隔开：1 + 1
计算结果：2.0
输入计算式，操作数与操作符用空格隔开：3 - 2
计算结果：1.0
输入计算式，操作数与操作符用空格隔开：6 * 9
计算结果：54.0
输入计算式，操作数与操作符用空格隔开：9 / 5
计算结果：1.8
```

- 输入不符合规范的计算式，观察程序对输入错误的处理情况。

```
输入计算式，操作数与操作符用空格隔开：1 +
异常类型：计算式不正确
输入计算式，操作数与操作符用空格隔开：1 + i
异常类型：操作数格式错误
输入计算式，操作数与操作符用空格隔开：1 add 7
异常类型：未知的计算符
```

- 分析代码，熟悉标准输入输出的使用。
- 分析代码，熟悉循环和选择等控制语句的使用。
- 思考如何构建一个持续运行的交互式计算器。

4 实验要求

4.1 实验评价

1、必须完成实验所有步骤，实验中的程序全部能正确运行。
2、遇到问题时能及时与指导老师沟通并解决问题。

4.2 实验报告

本次实验以验证为主，不需要提交实验报告。

5 实验教学录屏

仅供预习和复习参考，实验内容和要求以正式课堂为准。
https://www.bilibili.com/video/BV1J14y1C7ur/