Lab1 Hello World

- 安装JDK
- 安装编辑器
- hello world

吴昊伦

51265901074@stu.ecnu.edu.cn

安装JDK-下载

什么是JDK?

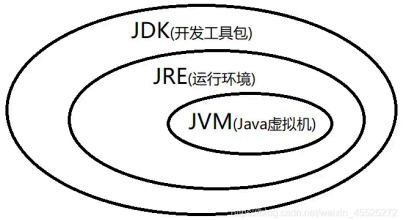
我们先来看看几个相关的概念

1. JDK(Java Development Kit) 是 Java 语言的软件开发工具包(SDK)。
2. JRE（Java Runtime Environment，Java运行环境），包含JVM标准实现及Java核心类库。JRE是Java运行环境，并不是一个开发环境，所以没有包含任何开发工具（如编译器和调试器）
3. JVM是Java Virtual Machine（Java虚拟机）的缩写，JVM是一种用于计算设备的规范，它是一个虚构出来的计算机，是通过在实际的计算机上仿真模拟各种计算机功能来实现的。它的主要功能是把自己的指令集（字节码）映射到本读的CPU指令集和OS的系统调用。

安装JDK-下载

JDK，JRE，JVM三者有什么关系?

在jdk的安装目录下有一个jre目录，里面有两个文件夹bin和lib，在这里可以认为bin里的就是JVM，lib中则是JVM工作所需要的类库。



安装JDK-下载

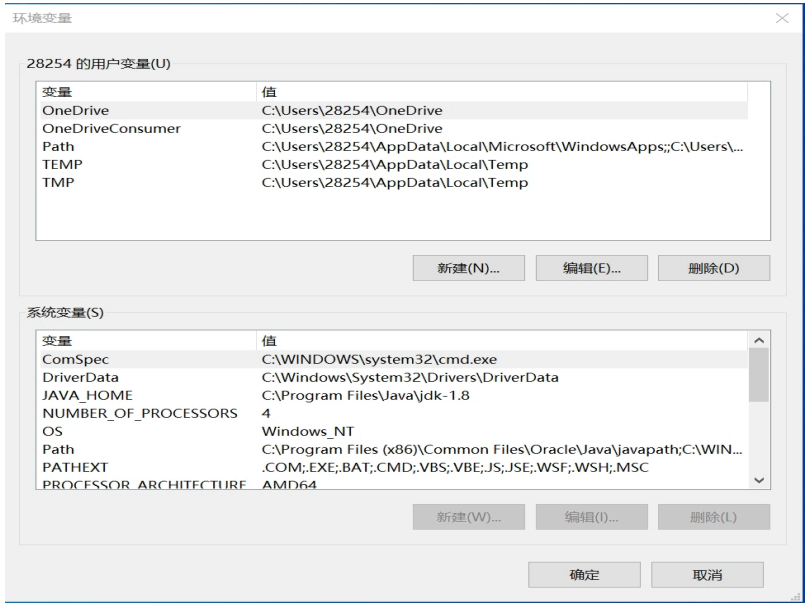
1. 我们选择JDK8版本，根据自己的系统选择相应版本的进行下载即可：

2. 这里可能会遇到的问题是需要你输入Oracle的账号密码，不想自己注册的话在 这里 选择可用的账号密码即可。

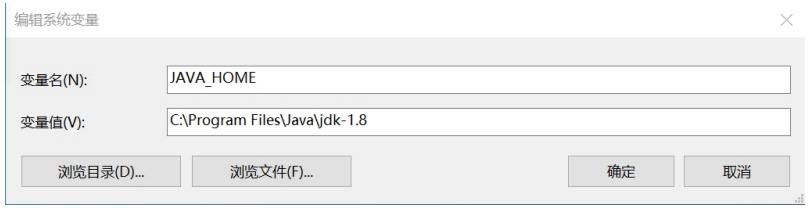
3. 进入安装界面，记住这个安装路径(**划重点！后面配置环境要用到！**).

安装JDK-配置环境变量

1. 在搜索栏中输入path进入环境变量的编辑界面.

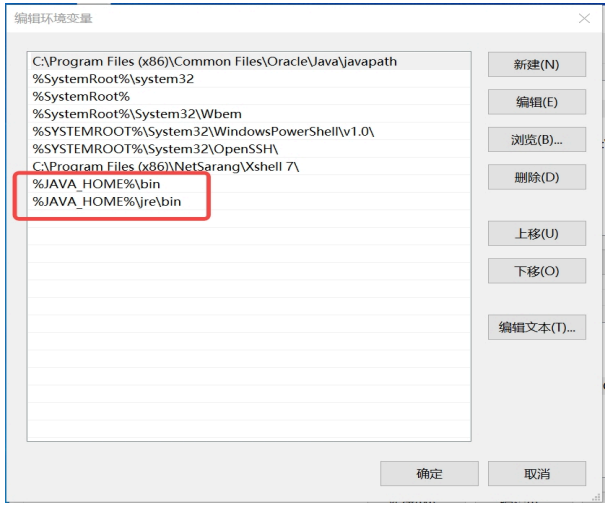


2. 在系统变量中新建一个JAVA_HOME变量，变量值填写**上一步设置的安装路径。**



安装JDK-配置环境变量

3. 然后在系统变量中找到`Path`变量进行编辑，分别添加`%JAVA_HOME%\bin`和`%JAVA_HOME%\jre\bin`.



4. 单击确定保存所有更改,在命令行中输入`java -version`，如果出现以下类似的信息说明JDK安装成功!

```
C:\Users\28254>java -version
java version "1.8.0_391"
Java(TM) SE Runtime Environment (build 1.8.0_391-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.391-b13, mixed mode)
```

Wait a Second...

我们刚刚干了啥？为什么要配这些变量？这些变量是干嘛的？

（遇事不决GPT！）

1. `JAVA_HOME`环境变量

- **作用**：`JAVA_HOME`变量指向JDK的安装目录。这个变量被一些Java应用和开发工具用来确定JDK的安装位置，以便正确运行Java应用和工具。
- **原因**：某些软件、脚本和工具链（如Apache Tomcat、Gradle、Maven等）需要使用`JAVA_HOME`来寻找Java编译器（`javac`）、Java运行时环境（`java`）等工具。

2. 更新`PATH`环境变量

- **作用**：将JDK的`bin`目录添加到`PATH`环境变量中，使得可以从命令行或任何地方运行Java编译器（`javac`）、Java虚拟机（`java`）以及其他Java命令行工具，而无需指定它们的完整路径。
- **原因**：这样配置后，用户可以在任何目录下通过命令行执行Java命令，极大地提高了开发和运行Java程序的便利性。

3. 为什么这么做

- **便利性**：配置环境变量后，开发者无需每次编译或运行Java程序时都输入完整的JDK路径，简化了开发流程。
- **兼容性**：一些基于Java的应用和开发工具需要通过这些环境变量来定位Java安装路径，以确保它们能正确执行。
- **灵活性**：通过环境变量，可以在系统中安装多个版本的JDK，并且可以通过修改`JAVA_HOME`变量来切换不同版本的JDK，以满足不同项目的需求。

编辑器的安装（自行完成）

1. 任何文本编辑器都可以用来写代码，选择太多了，这里推荐更轻量化的**Sublime Text**。（傻瓜式安装，自行百度即可）
2. 对本课程来说，**Sublime Text**已经足够完成大部分的实验，但对于更大的项目和更专业的开发场景来说，安装一款IDE就十分有必要了。对于Java语言，常用的IDE主要有**Intellij IDEA（更推荐）和Eclipse**。
3. 社区版本的`IDEA`已经足够覆盖大部分的日常开发需求，旗舰版有30天的免费使用时长，但是通过ECNU的学生邮箱进行学生认证可以持续免费使用。

Hello World

编写代码

1. 打开我们的Sublime，新建一个文件，并写入以下内容：

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

2. 将文件保存为HelloWorld.java,**注意类名必须和文件名一致，这是Java文件命名的基本规则之一**。(好奇的可以试一下不一致会怎么样)

Hello World

运行代码

1. win+R后输入cmd打开命令行，然后进入HelloWorld.java的保存路径：

```
$ cd 你的HelloWorld.java的保存路径
```

2. 执行：

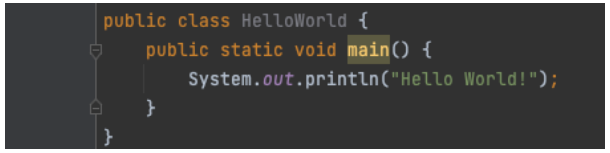
```
$ javac HelloWorld.java
$ java HelloWorld
```

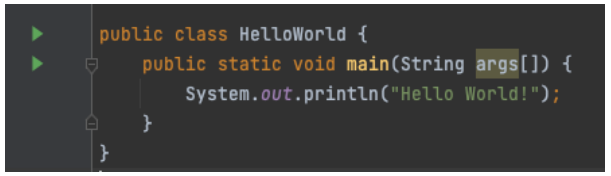
3. Done!

Hello World

一些细节

1. 回忆一下C语言中helloworld的写法，是不是发现Java中似乎多了一个奇怪的参数`String args[]`,去掉会怎么样?





通过对比以上两张图，我们发现去掉`String args[]`后，main函数变得不可执行了。**现在可以自己试试看去掉`public`或者`static`又会发生什么。**

实际上这是Java中固定的主函数的写法，去掉任何一个都会导致主函数无法正确调用，main函数也是JVM实例运行的起点。（想详细了解main函数的同学可以看这里）

Hello World

3. String args[]实际上可以读取我们从命令行提供的参数，比如以上的程序可以改写为：

```
public class HelloWorld {
    public static void main(String args[]) {
        System.out.println(args[0]);
    }
}
```

通过执行执行以下的命令发现此时的输出已经变为了`12345`。

```
$ javac HelloWorld.java
$ java HelloWorld 12345
```

Hello World

4. `javac`和`java`两条命令究竟在干嘛？在执行完`javac`后，有没有发现多了什么文件？

   `javac`实际上完成的是编译工作，它会输出一个`class`文件，我们用`Sublime`打开`HelloWorld.class`

   发现它实际是一串字节码（bytecode），这些字节码就是Java虚拟机（JVM，刚刚提到过的bin文件里的东西）可以解释执行的东西。