# 1 基础知识

## 1.1 软件

软件就是计算机数据与指令的集合。软件 = 数据 + 指令 + 文档。

软件的目的是提高人与计算机交流的效率，方便人与机器交互。

## 1.2 常用命令

现在人机交互的方式分为图形化界面交互和命令式交互。

图形化界面简单直观，容易操作；使用命令的方式需要记住命令，操作麻烦。

但是在程序开发中，使用命令更加高效，有时使用一条命令就能完成若干的操作。我们也需要学些简单的命令。

在Windows中执行命令需要在CMD窗口中输入命令。打开控制台的方法：Windows + R键，输出cmd并回车。

常用的DOS命令：

> 盘符: ：进入指定的盘符。

> dir：列出当前目录下文件和文件夹

> md：创建目录

> rd：删除目录，不能删除非空的文件夹

> cd：进入指定目录，cd ..表示退回上一级，cd \表示退回到根目录。

> echo "hello java" > a.txt：将文本写入到指定文件，将会创建文件

> type a.txt：显示文件内容

> del：删除文件

> exit：退出控制台

> cls：清空屏幕

另外，在控制台中可以用：

\*和?通配符；按下tab进行命令补全；上下方向键显示历史命令。

# 2 Java简介

Java是SUN (Stanford University Network，斯坦福大学网络)公司于1995年推出的一门高级编程语言。Java原是印度尼西亚爪哇岛的英文名称，因盛产咖啡而闻名。Java最初目标在于家用电器等小型系统的编程，随着市场的变化，Java着眼于网络应用的开发。

1995年5月，JAVA语言诞生。

2004年9月，J2SE 1.5发布，这是Java发展史上的一个里程碑。为了表示该版本的重要性，J2SE 1.5更名为Java SE 5.0。

2005年，JAVA版本正式分为JAVA EE（企业级），JAVA SE（基础）和JAVA ME（嵌入式）。我们现在学习的是Java SE，以后要学Java EE。必须先学好Java SE基础才能进行Java EE或者Android应用的开发。

2006年12月，Sun发布了Java 6.0。

2009年，Oracle（甲骨文公司）收购了Sun公司。即现在Java归Oracle管理。

2011年7月，Oracle发布了Java 7。

2014年，甲骨文公司发布了Java 8正式版。

## 2.1 Java的跨平台特性

这里着重讲Java的跨平台特性，语音的其他特性就不讲了，只能在学习中体会。

跨平台就是软件的运行不依赖于具体的操作系统。在一个操作系统下开发的应用，放到另一个操作系统下依然可以运行。而Java就具有这样的特性，Java语言编写成的程序在不同的OS上都可以运行，“一次编写，处处运行”。

而C语言的跨平台性就不好，在Windows上编译生成的C语言程序，在Linux系统下是不能运行的。至少也要将源代码再编译生成不同的目标代码，况且，不同OS的系统调用API不同，就更加大了跨平台的难度。

Java的跨平台是通过JVM（Java Virtual Machine，Java虚拟机）来实现的。

JVM相当于是一个用软件模拟的计算机。Java源程序会编译成.class字节码文件。.class字节码文件由JVM解释运行。因此，只要为每种操作系统提供不同的JVM，就能实现跨平台。即JVM拿到的是同一个字节码文件，由JVM适应不同的OS。图解：



图2-1 JVM

# 3 搭建Java开发环境

开发Java需要安装JDK，即Java开发工具(Java Development Kit)。JDK是整个Java的核心，包括了Java运行环境、Java工具和Java基础类库。

## 3.1 安装JDK

首先在Oracle官网https://www.oracle.com找到JDK下载页面。

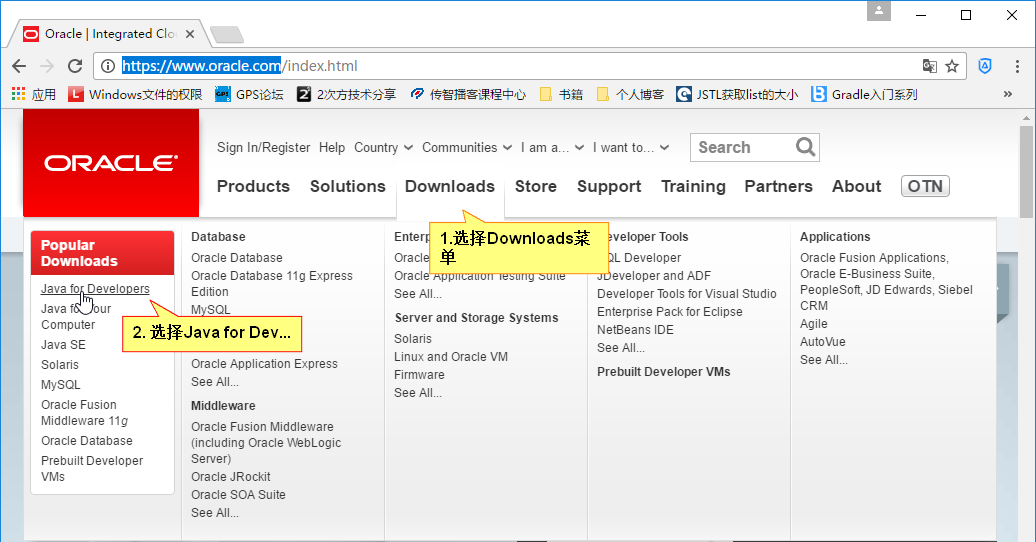


图3-1 Java for Developers

之后在Java SE的页面，点击下载的图标，出现下面界面，要按照图中所示操作，即接受协议再下载符合自己电脑的版本：



图3-2 下载JDK

下载完后，点击安装程序进行安装。安装时注意选择安装路径，路径中不要有空格和中文。安装时，把“公共JRE”这个组建去掉，不予安装，因为JDK中已经包含了JRE，不再需要额外的JRE。

说明：JRE就是Java运行时环境(Java Runtime Environment)。JRE用于支持运行Java程序，包括JVM和运行Java程序需要的核心类库。如果只是使用Java应用的用户，则只要安装公共JRE即可。

## 3.2 JDK安装目录介绍

bin目录：存放Java编译器、解释器等工具。

db目录：这里是JDK附带的一个轻量级数据库，名叫Derby。

include目录：存放调用系统资源的接口文件。

jre目录：存放Java运行环境文件。

lib目录：存放Java类库文件。

src.zip文件：是JDK Java类的源码。IDE会找到这个文件，以便在IDE中查看JDK源码。

## 3.3 配置环境变量

### 3.3.1 环境变量及其作用

环境变量用于存储操作系统级别的变量。环境变量分为系统变量和用户变量，系统变量可被系统中的所有用户共享使用；而用户变量只能被当前用户使用。因此每个用户都能配置自己的用户变量。用户也能设置系统变量，但是建议将自行配置的环境变量放在自己的用户变量中，这样比较清晰，互不影响。

通过实践表明环境变量能在系统中使用。首先查看下本机的环境变量：桌面 – 右键“此电脑” – 属性，出现下面界面，点击“高级系统设置”：



图3-3 计算机属性

在新界面中（默认“高级”选项卡），点击“环境变量”按钮：

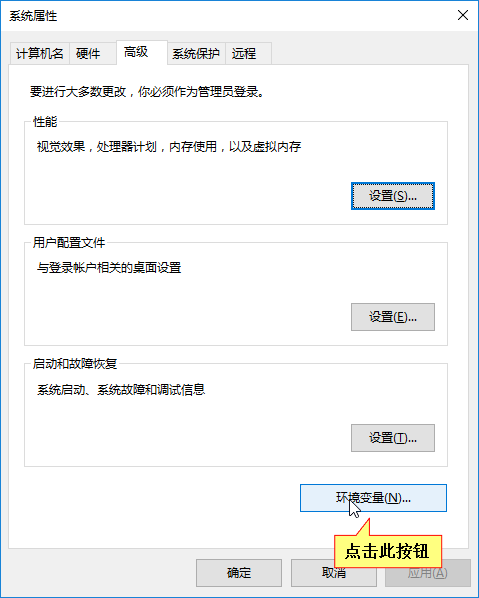


图3-4 高级

新界面显示的内容就是本机的用户变量和系统变量。

我们可以用这里的变量，在Windows中环境变量的变量名不区分大小写。比如用户变量里的“TEMP”表示本用户的临时目录，在文件管理器地址栏中输入%TEMP%回车就能进入此目录。%变量名%就表示引用这个变量。同样在控制台中执行echo %TEMP%也能显示此路径。

另外，还有一个用户变量叫USERPROFILE，这个变量在界面中隐藏了。这个变量指向用户的主目录，用这个有时很方便。

### 3.3.2 Path环境变量

前面说JDK下的bin目录是一些工具，我们需要使用。比如进入该目录下打开控制台，执行java命令会有一系列使用命令的提示。但是如果不在此目录下，该命令就无效：

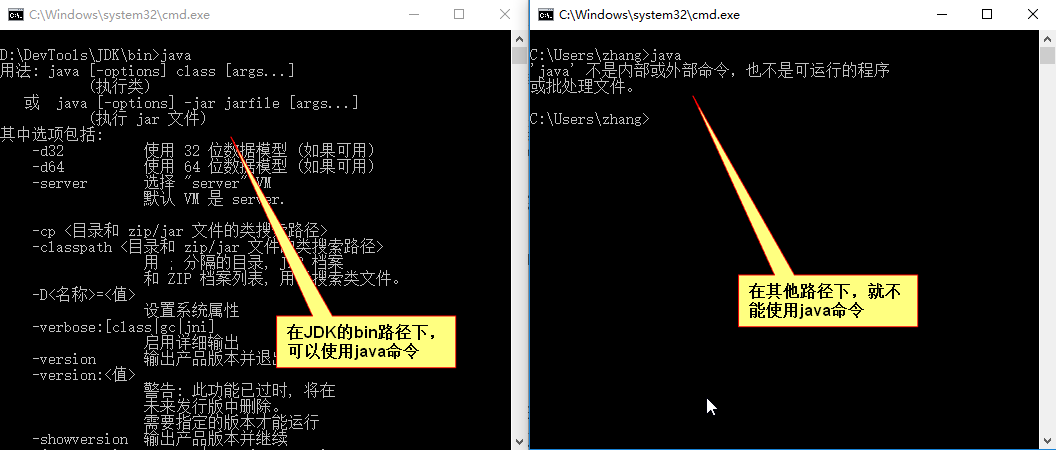


图3-5 比较

但是，一些系统命令，无论在什么路径下都能使用，比如calc（打开计算器）、diskpart命令等。

为了在其他路径下也能使用JDK中的命令，需要配置Path环境变量。

PATH变量是一个特殊变量，一旦把一个路径添加到Path环境变量中，那么该目录下的命令就能在控制台的任意路径下使用。系统命令就是在系统Path命令中设置了才能在任何地方使用的。

我们可以在系统变量的Path变量中添加上JDK\bin目录，但是前面说过，我们一般把自行配置的变量配置到用户变量中。

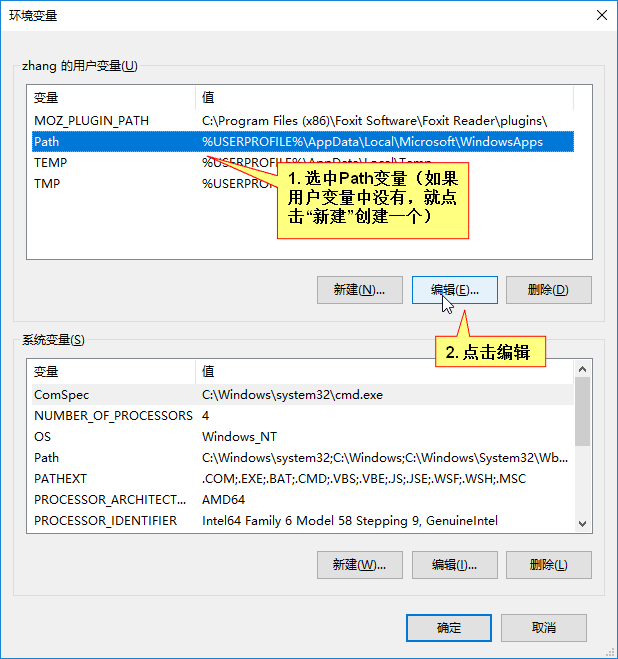


图3-6 环境变量

在原有路径之后添加自己的路径，注意路径之间用英文半角分号隔开！

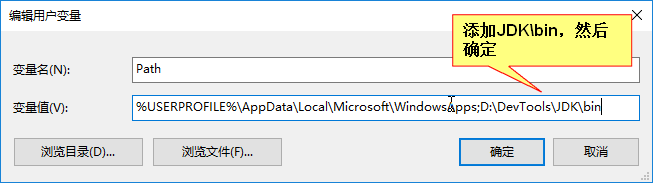


图3-7 添加路径

然后点击确定，并且下层界面也要依次点确定。

之后，需要重新启动控制台，这时在任意路径下都能使用java命令了。

但是一般标准的做法是在系统变量中增加一个变量，名字叫“JAVA\_HOME”，这个就是JDK的根目录，然后Path变量引用这个JAVA\_HOME变量。这样的设置有利于以后使用大多数IDE软件，因为很多软件会先检测JAVA\_HOME这个变量。

（1）首先在“环境变量”界面新建一个用户变量，名叫“JAVA\_HOME”，值是JDK根目录。（以后的很多工具也最好配置工具的根目录，都是以\_HOME结尾。标准做法。）

（2）然后添加Path环境变量（之前直接设置的JDK\bin可以删除了），值为“%JAVA\_HOME%\bin”，就是指向JDK的bin目录。这也说明了变量间可以相互引用，还用通过 “%变量名%”引用。

（3）点击确定，然后重启控制台，即可使用java中命令了。

还需要了解一下命令搜寻的顺序：

（1）首先在当前目录下寻找命令，如果找不到就寻找环境变量中的路径。

（2）先寻找用户Path变量中的路径，若找不到就寻找系统Path变量中的路径。

### 3.3.3 设置临时环境变量

用set命令可以查看或设置临时变量，用法如下：

**Set 查看所有环境变量信息。会整合用户变量和系统变量，比如用户变量和系统变量都有path变量。**

**Set 变量名 查看具体一个环境变量的值。**

**Set 变量名= 清空一个环境变量的值。**

**Set 变量名=具体值 给指定变量定义具体值。**

若想在原有环境变量值基础上添加新值，则通过%变量名%引用原有值，然后加上新值再设置给该变量即可。

例如给path环境变量加入新值

**Set path=新值;%path%**

注意：用set命令配置的环境变量只在当前控制台窗口有效，一旦窗口关闭，该临时的配置就消失了，还会变为自行设置的值，只是临时起作用。

# 4 Java程序入门