

day01【前言、入门程序、常量、变量】

今日内容

- Java语言的发展历史
- Java开发环境搭建
- 编写HelloWorld程序
- 常量和变量

教学目标

- ☐ 能够计算二进制和十进制数之间的互转
- ☐ 能够使用常见的DOS命令
- ☐ 理解Java语言的跨平台实现原理
- ☐ 理解JDK/JRE/JVM的组成和作用
- ☐ 能够配置环境变量JAVA_HOME
- ☐ 能够编写HelloWorld程序编译并执行
- ☐ 能够辨识关键字
- ☐ 理解标识符的含义
- ☐ 能够定义出所有类型的常量
- ☐ 理解Java中的基本数据类型分类
- ☐ 能够定义8种基本数据集类型的变量

第一章 开发前言

1.1 Java语言概述

目标:

了解java语言的概述和java的应用

步骤:

- 什么是java语言
- java语言发展历史
- java语言能做什么

讲解:

什么是Java语言

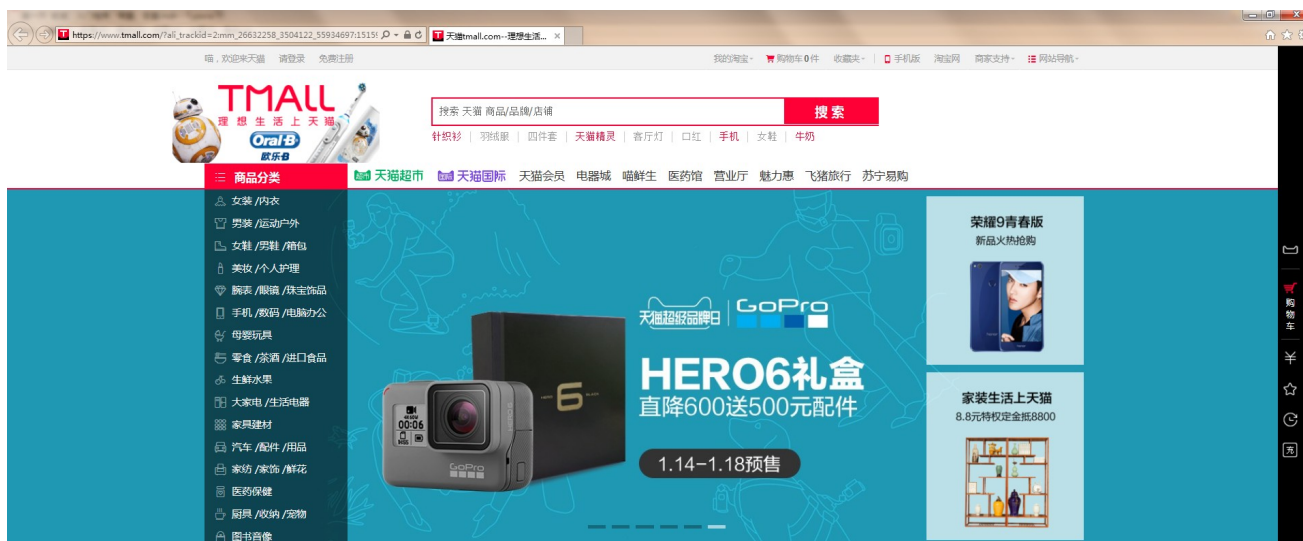
Java语言是美国Sun公司（Stanford University Network），在1995年推出的高级编程语言。所谓编程语言，是计算机的语言，人们可以使用编程语言对计算机下达命令，让计算机完成人们需要的功能。

Java语言发展历史

- 1995年Sun公司发布Java 1.0版本
- 1997年发布Java 1.1版本
- 1998年发布Java 1.2版本
- 2000年发布Java 1.3版本
- 2002年发布Java 1.4版本
- 2004年发布Java 5.0版本
- 2006年发布Java 6.0版本
- 2009年 甲骨文公司收购Sun公司，并于2011发布Java 7.0版本
- 2014年发布Java 8.0版本
- 2017年发布Java 9.0版本
- 2018年3月发布Java 10.0版本
-

Java语言能做什么

Java语言主要应用在互联网程序的开发领域。常见的互联网程序比如天猫、京东、物流系统、网银系统等，在服务器后台处理大数据的存储、查询、数据挖掘等方面，Java也有很多应用。



韵达 EXPRESS

390729 快件跟踪记录 我要投诉 在线客服

发件客户城市	揽收时间		途径中转城市
揽件	地址	姓名/电话	进港扫描
揽件业务员			卸车扫描
揽件分部			拆包扫描
揽件乡镇			装包扫描
揽件网点城市			装车扫描
			出港扫描

处理时间	快件监控记录
2017-12-21 09:28:13	到达: 江苏昆山市市区公司 已揽件
2017-12-21 21:19:08	到达: 江苏苏州分拨中心
2017-12-21 21:22:10	到达: 江苏苏州分拨中心 发往: 北京分拨中心
2017-12-22 17:51:42	到达: 北京分拨中心 上级站点: 江苏常熟市公司
2017-12-22 17:51:58	到达: 北京分拨中心 上级站点: 江苏常熟市公司
2017-12-22 18:30:51	到达: 北京分拨中心 发往: 北京
2017-12-23 05:13:30	到达: 北京分拨中心 发往: 北京市

ICBC 工银融e行

全部 首页 财富广场 惠生活 账户列表 安全 设置 我的网银

最爱 注册账户 转账 缴费 转账汇款 贵金属 本地特色 我的账户 融e购 工银e支付 信用卡 基金 理财 保险

我的网银 最近使用: 我的网银

等级: ★★★★★ 标签: 六星用户 支付活跃 客户年龄大于16年 财富通用户 转账汇款 中青年

预留验证信息: 95588 上次登录时间: 2018-01-14 12:43:08

我的积分: 6,242 我的信用报告 购物车: 0 优惠券: 0

我的卡包 您共有 3 张银行卡

ICBC 中国工商银行 明细 还款 开始还款 更多

ICBC 中国工商银行 明细 还款 开始还款 更多

ICBC 中国工商银行 明细 还款 开始还款 更多

小结:

- java是一门高级编程语言,可以使用java语言对计算机下达命令,让计算机完成人们需要的功能。
- Java语言主要应用在互联网程序的开发领域。常见的互联网程序比如天猫、京东、物流系统、网银系统等。

1.2 计算机基础知识

二进制

目标:

掌握十进制和二进制之间的转换

步骤:

- 二进制的概述
- 十进制如何转换为二进制
- 二进制如何转换为十进制

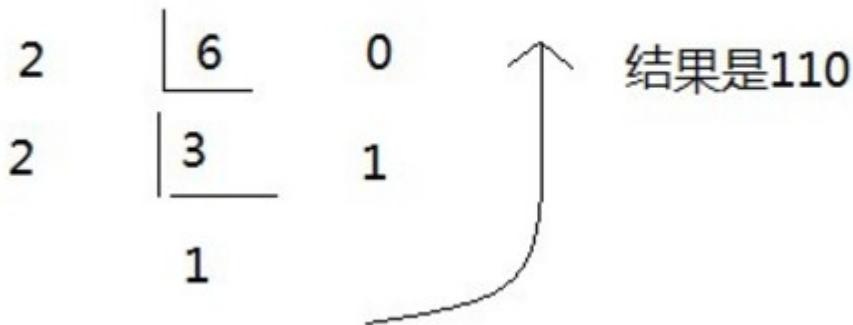
讲解:

计算机中的数据不同于人们生活中的数据，人们生活采用十进制数，而计算机中全部采用二进制数表示，它只包含0、1两个数，逢二进一， $1+1=10$ 。每一个0或者每一个1，叫做一个bit（比特）。

下面了解一下十进制和二进制数据之间的转换计算。

- **十进制数据转成二进制数据**：使用除以2倒取余数的方式

十进制数字6转成二进制,除以2获取余数



- **二进制数据转成十进制数据**：

- 把系数*基数的权次幂相加即可
 - 系数：就是每一位上的数据。
 - 基数：X进制，基数就是X。
 - 权：在右边，从0开始编号，对应位上的编号即为该位的权。
- 使用8421编码的方式 8421码是中国大陆的叫法，8421码是BCD代码中最常用的一种。在这种编码方式中每一位二值代码的1都是代表一个固定数值，把每一位的1代表的十进制数加起来，得到的结果就是它所代表的十进制数码。

二进制数据 1001011转成十进制

1	0	0	1	0	1	1
64	32	16	8	4	2	1

把有1位上的十进制求和
 $64+8+2+1=75$

小贴士：

二进制数系统中，每个0或1就是一个位，叫做bit（比特）。

小结:

1. 十进制数据转成二进制数据：使用除以2倒取余数的方式
2. 二进制数据转成十进制数据:
 1. 把系数*基数的权次幂相加即可
 2. 使用8421编码的方式

字节

目标:

了解字节的概述和单位换算

步骤:

- 字节的概述
- 单位换算

讲解:

字节是我们常见的计算机中最小存储单元。计算机存储任何的数据，都是以字节的形式存储，右键点击文件属性，我们可以查看文件的字节大小。

8个bit（二进制位）0000-0000表示为1个字节，写成**1 Byte**或者**1 B**。

位(bit):数字0或数字1,就是一位

字节(byte):8个位为一个字节,是计算机中的最小存储单位

- 1 B(字节) = 8 bit (位)
- 1 KB = 1024 B
- 1 MB = 1024 KB
- 1 GB = 1024 MB
- 1 TB = 1024 GB
-

小结:

字节是我们常见的计算机中最小存储单元

常用DOS命令

目标:

了解常用的DOS命令

步骤:

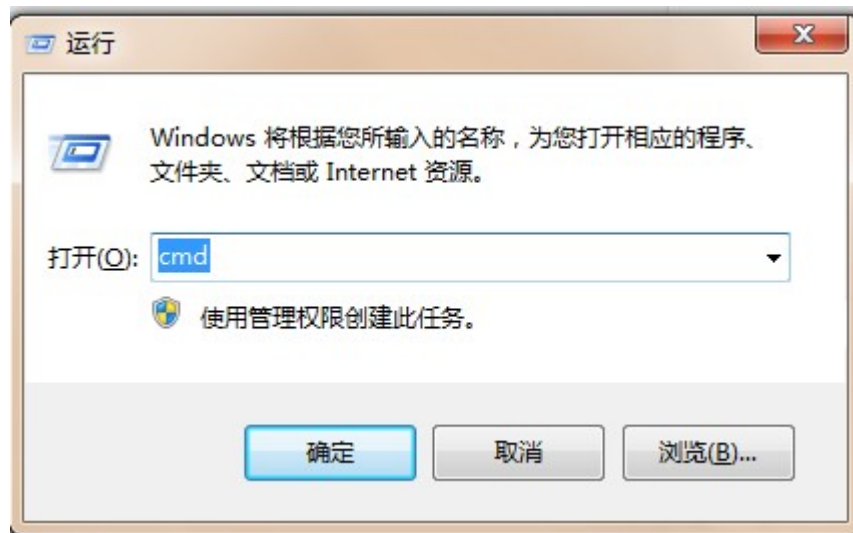
- 打开cmd命令行窗口
- 演示DOS命令

讲解:

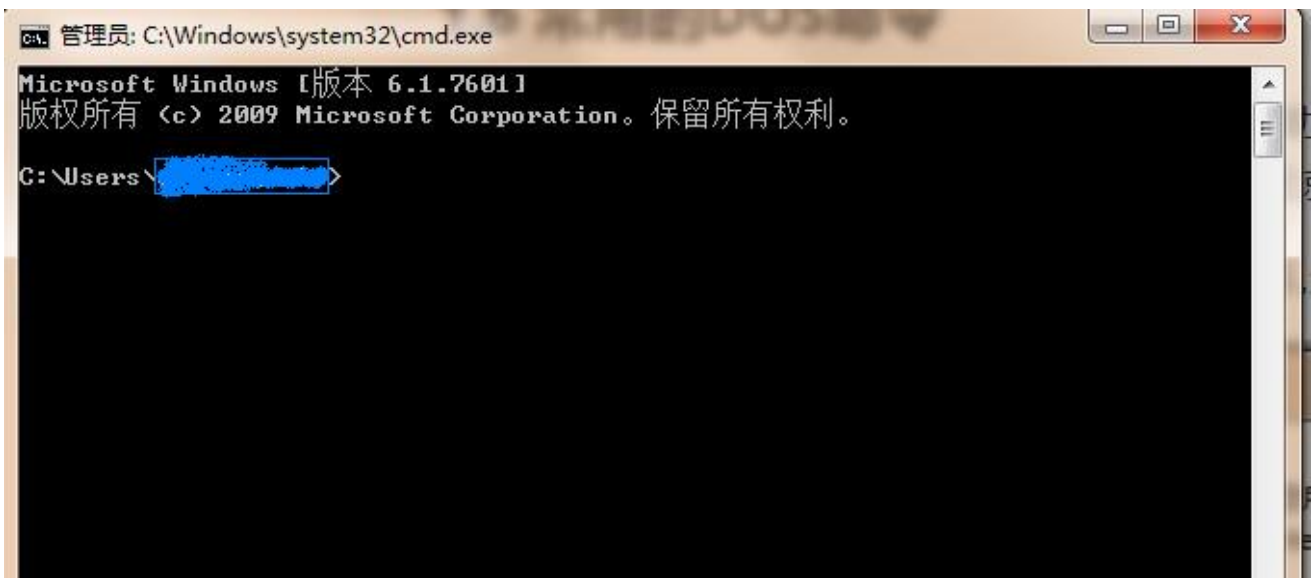
Java语言的初学者，学习一些DOS命令，会非常有帮助。DOS是一个早期的操作系统，现在已经被Windows系统取代，对于我们开发人员，目前需要在DOS中完成一些事情，因此就需要掌握一些必要的命令。

- **进入DOS操作窗口**

- 按下 **Windows+R** 键，打开运行窗口，输入 **cmd** 回车，进入到DOS的操作窗口。



- 打开DOS命令行后，看到一个路径 **C:\Users\...** 就表示我们现在操作的磁盘是C盘。



- **常用命令**

命令	操作符号
盘符切换命令	盘符名:
查看当前文件夹	dir
进入文件夹命令	cd 文件夹名
退出文件夹命令	cd ..
退出到磁盘根目录	cd \
清屏	cls

小结:

略

第二章 Java语言开发环境搭建

2.1 Java虚拟机——JVM

目标:

理解java程序跨平台的原理

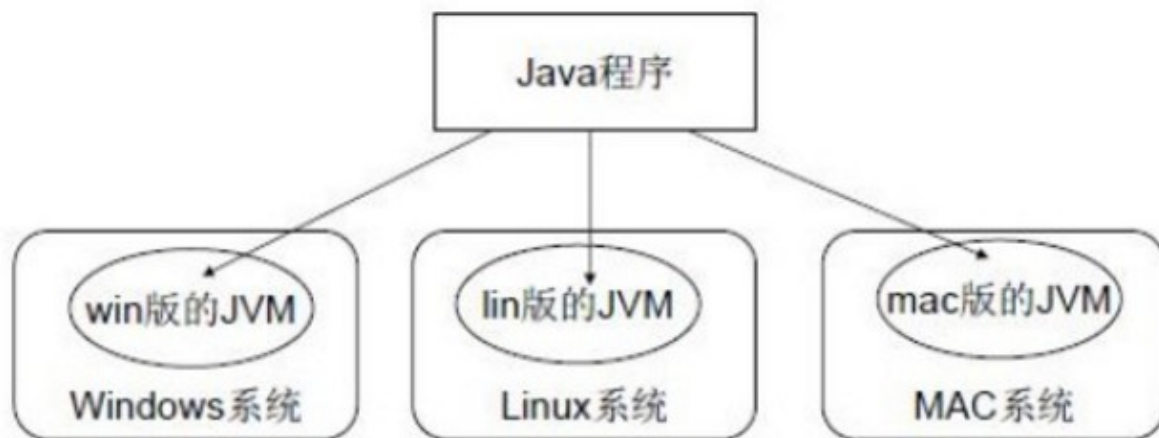
步骤:

- jvm的概述
- java跨平台的原理

讲解:

- **JVM**（Java Virtual Machine）：Java虚拟机，简称JVM，是运行所有Java程序的假想计算机，是Java程序的运行环境之一，也是Java 最具吸引力的特性之一。我们**编写的Java代码，都运行在JVM 之上**。
- **跨平台**：任何软件的运行，都必须运行在操作系统之上，而我们用Java编写的软件可以运行在任何的操作系统上，这个特性称为**Java语言的跨平台特性**。该特性是由JVM实现的，我们编写的程序运行在JVM上，而JVM

运行在操作系统上。



如图所示，Java的虚拟机本身是不具备跨平台功能的，每个操作系统下都有不同版本的虚拟机。

小结:

我们编写的java程序都是运行在JVM之上的,所以想要实现跨平台,我们只需要在运行的操作系统之上安装对应版本的虚拟机(JVM)即可实现跨平台.

2.2 JRE 和 JDK

目标:

理解JVM和JRE以及JDK之间的关系

步骤:

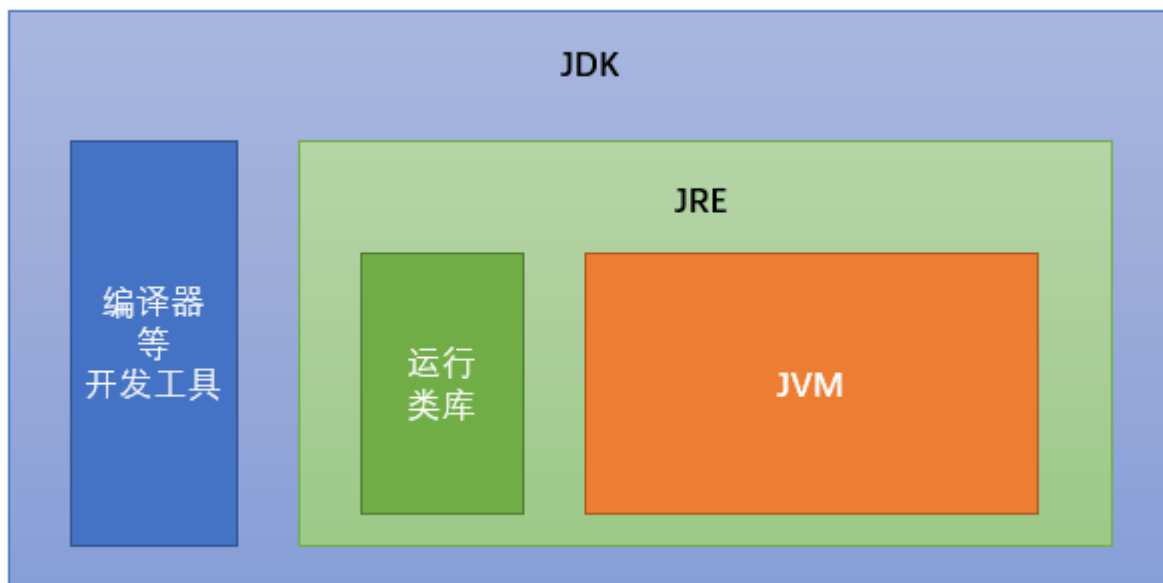
- JRE的概念
- JDK的概念

讲解:

- **JRE** (Java Runtime Environment) : 是Java程序的运行时环境, 包含 **JVM** 和运行时所需要的 **核心类库**。
- **JDK** (Java Development Kit) : 是Java程序开发工具包, 包含 **JRE** 和开发人员使用的工具。

我们想要运行一个已有的Java程序, 那么只需安装 **JRE** 即可。

我们想要开发一个全新的Java程序, 那么必须安装 **JDK**, 其内部包含 **JRE**。



小贴士：

三者关系：JDK > JRE > JVM

小结:

JDK > JRE > JVM

2.3 JDK9安装图解

目标:

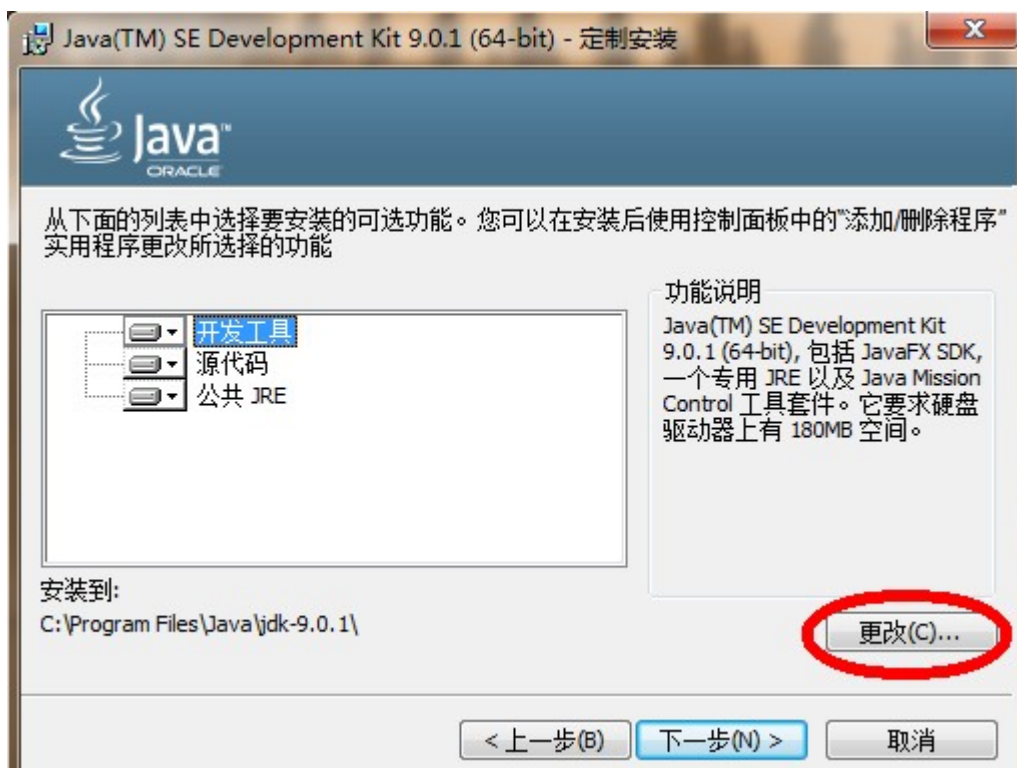
我们需要编写和运行java程序,就需要安装jdk

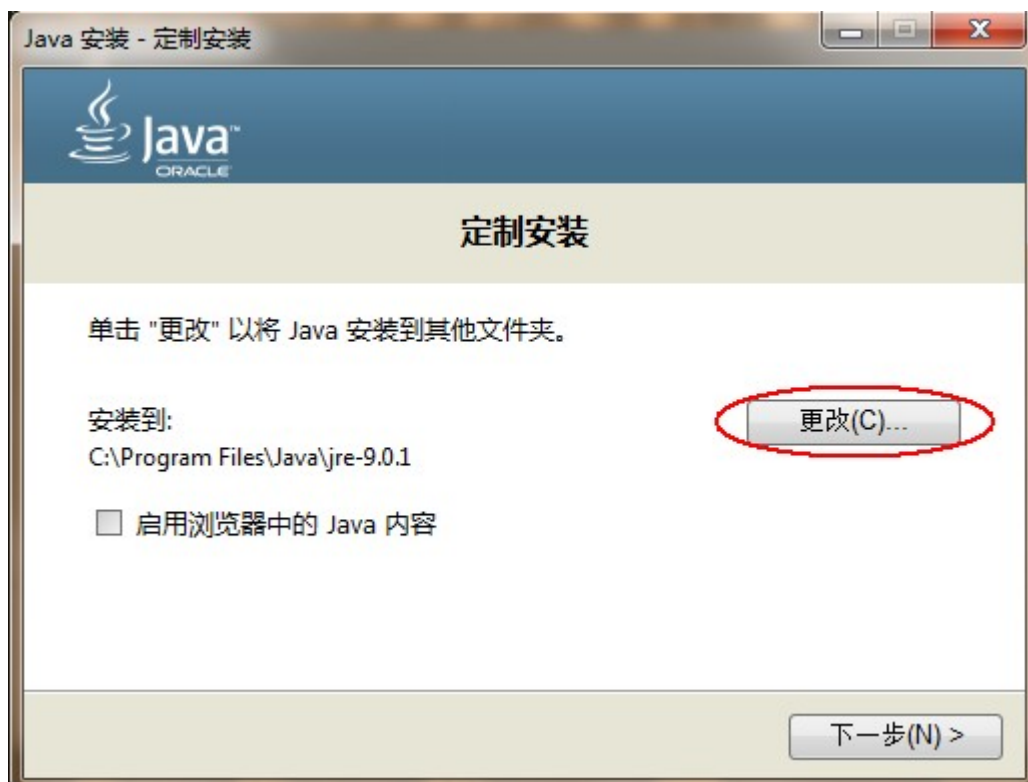
步骤:

按照jdk9

讲解:

jdk1





小贴士：

安装路径中，建议不要包含中文和空格。

小结:

注意:安装开发软件,路径最好不要出现中文路径

2.4 JAVA_HOME环境变量的配置

目标:

通过刚刚jdk的安装,我们已经可以使用jdk中提供的工具了,但是每次都需要进入到jdk 的bin目录下,否则使用不了,这个过程会比较麻烦,所以为了开发方便,我们想在任意的目录下都可以使用JDK的开发工具,则必须要配置环境变量,配置环境变量的意义在于告诉操作系统,我们使用的JDK开发工具在哪个目录下。

步骤:

- 配置环境变量

讲解:

配置环境变量作用

开发Java程序,需要使用JDK中提供的工具,工具在JDK9安装目录的 `bin` 目录下。



在DOS命令行下使用这些工具,就要先进入到JDK的bin目录下,这个过程就会非常的麻烦。

```
管理员: C:\Windows\system32\cmd.exe

C:\Users\>cd \

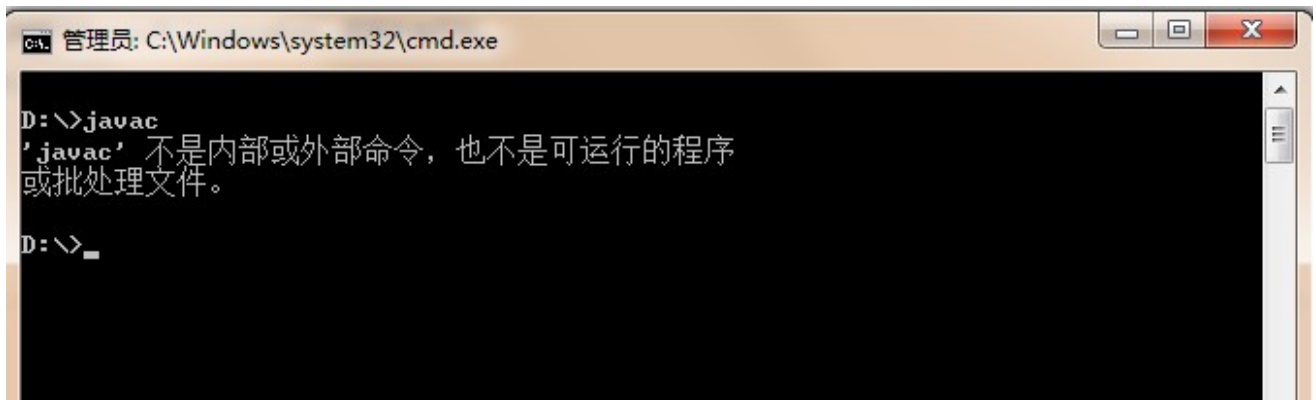
C:\>cd java9

C:\Java9>cd jdk-9.0.1

C:\Java9\jdk-9.0.1>cd bin

C:\Java9\jdk-9.0.1\bin>javac
用法: javac <options> <source files>
其中, 可能的选项包括:
    @<filename>
    -Akey[=value]
    --add-modules <模块>[,<模块>]*
从文件读取选项和文件名
传递给注释处理程序的选项
```

不进入JDK的 `bin` 目录, 这些工具就不能使用, 会报错。



为了开发方便,我们想在任意的目录下都可以使用JDK的开发工具,则必须要配置环境变量,配置环境变量的意义在于告诉操作系统,我们使用的JDK开发工具在哪个目录下。

配置环境变量步骤

Windows 7,8版本

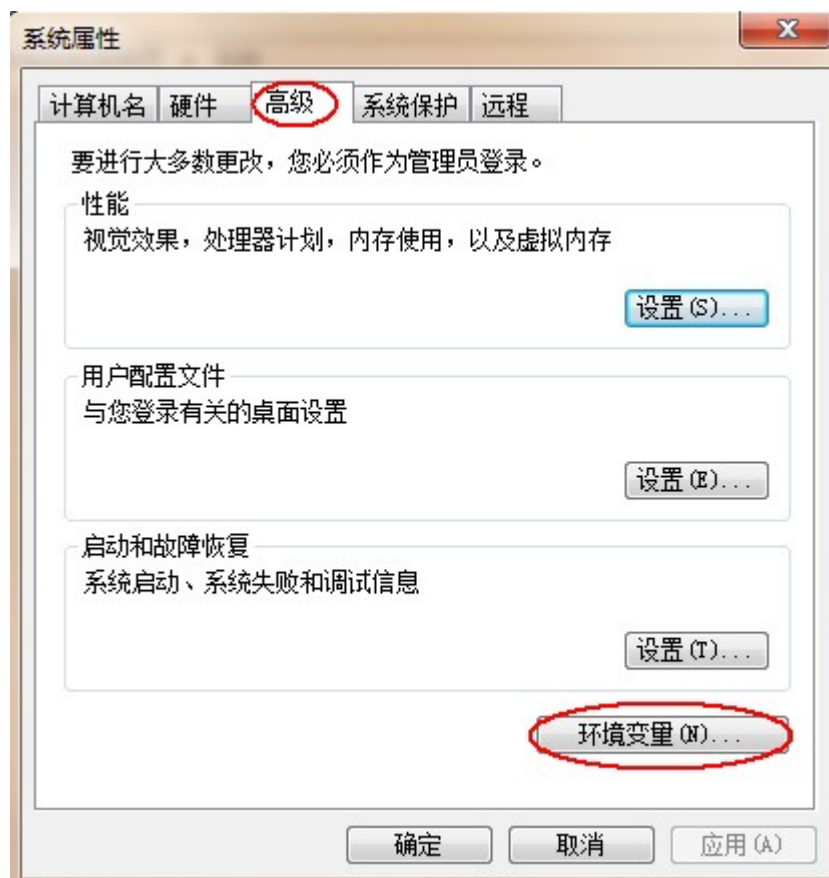
1. 计算机鼠标右键,选择 属性



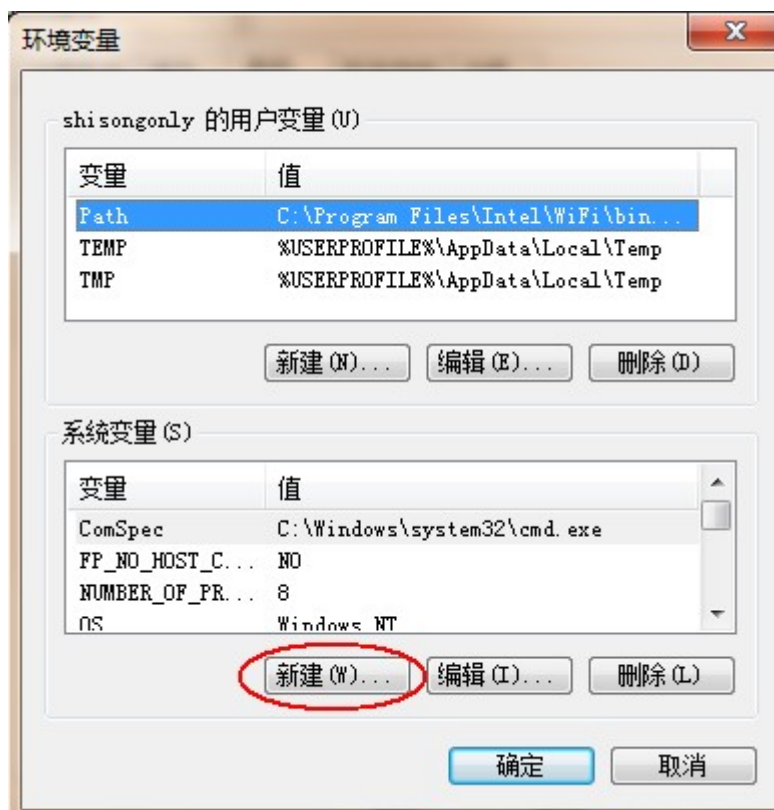
2. 选择 高级系统设置



3. 高级 选项卡, 点击 环境变量



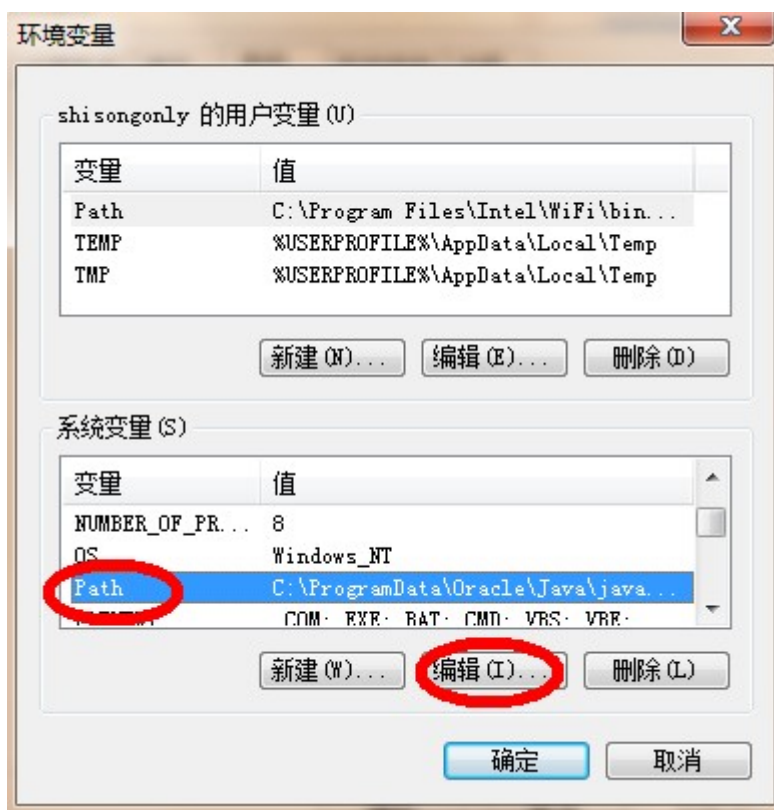
4. 点击 **新建**，创建新的环境变量



5. 变量名输入 **JAVA_HOME**，变量值输入JDK9的安装目录 **C:\Java9\jdk-9.0.1**



6. 选中 Path 环境变量，双击 或者 点击编辑



小贴士:

Path中 C:\ProgramData\Oracle\Java...，该路径是安装后默认配置的路径，可以删除。

7. 在变量值的最前面，键入 %JAVA_HOME%\bin; 分号必须要写，必须是英文格式。



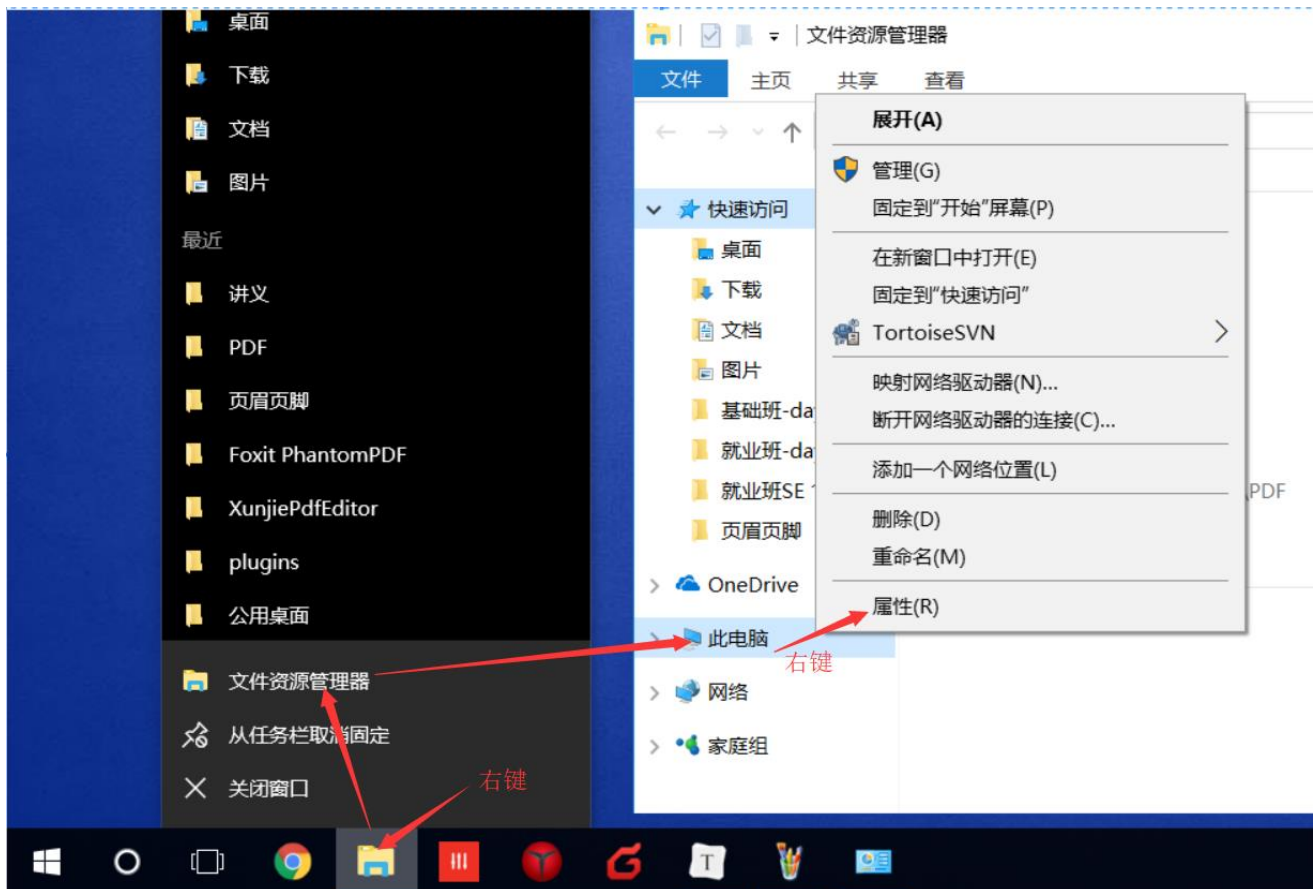
8. 环境变量配置完成，重新开启DOS命令行，在任意目录下输入 javac 命令，运行成功。


```
管理员: C:\Windows\system32\cmd.exe

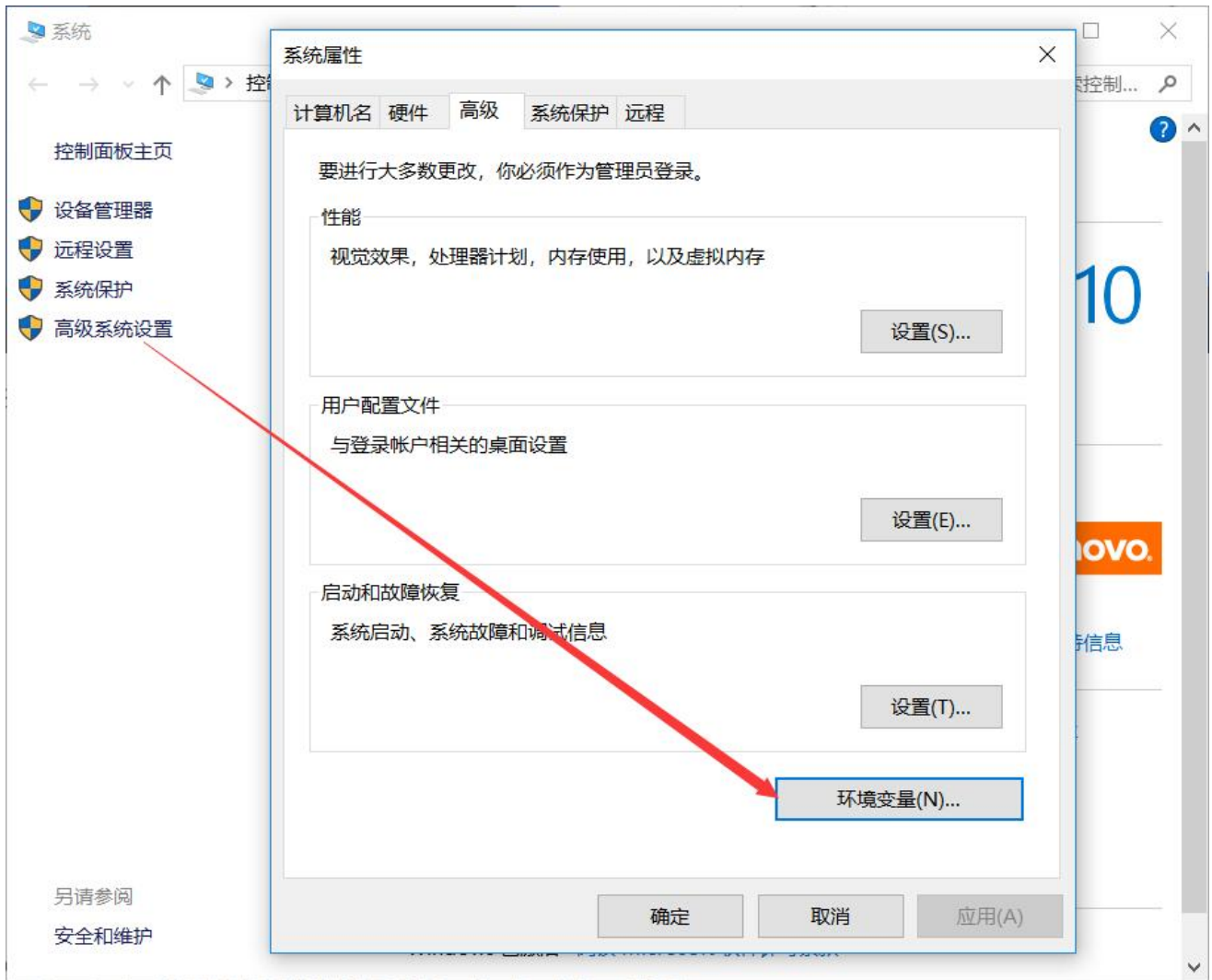
D:\>javac
用法: javac <options> <source files>
其中, 可能的选项包括:
@<filename>                从文件读取选项和文件名
                             传递给注释处理程序的选项
-akey[=value]
--add-modules <模块>[.<模块>]*
    除了初始模块之外要解析的根模块; 如果 <module>
    为 ALL-MODULE-PATH, 则为模块路径中的所有模块。
--boot-class-path <path>, -bootclasspath <path>
    覆盖引导类文件的位置
--class-path <path>, -classpath <path>, -cp <path>
    指定查找用户类文件和注释处理程序的位置
-d <directory>
    指定放置生成的类文件的位置
-deprecation
    输出使用已过时的 API 的源位置
```

Windows 10 版本

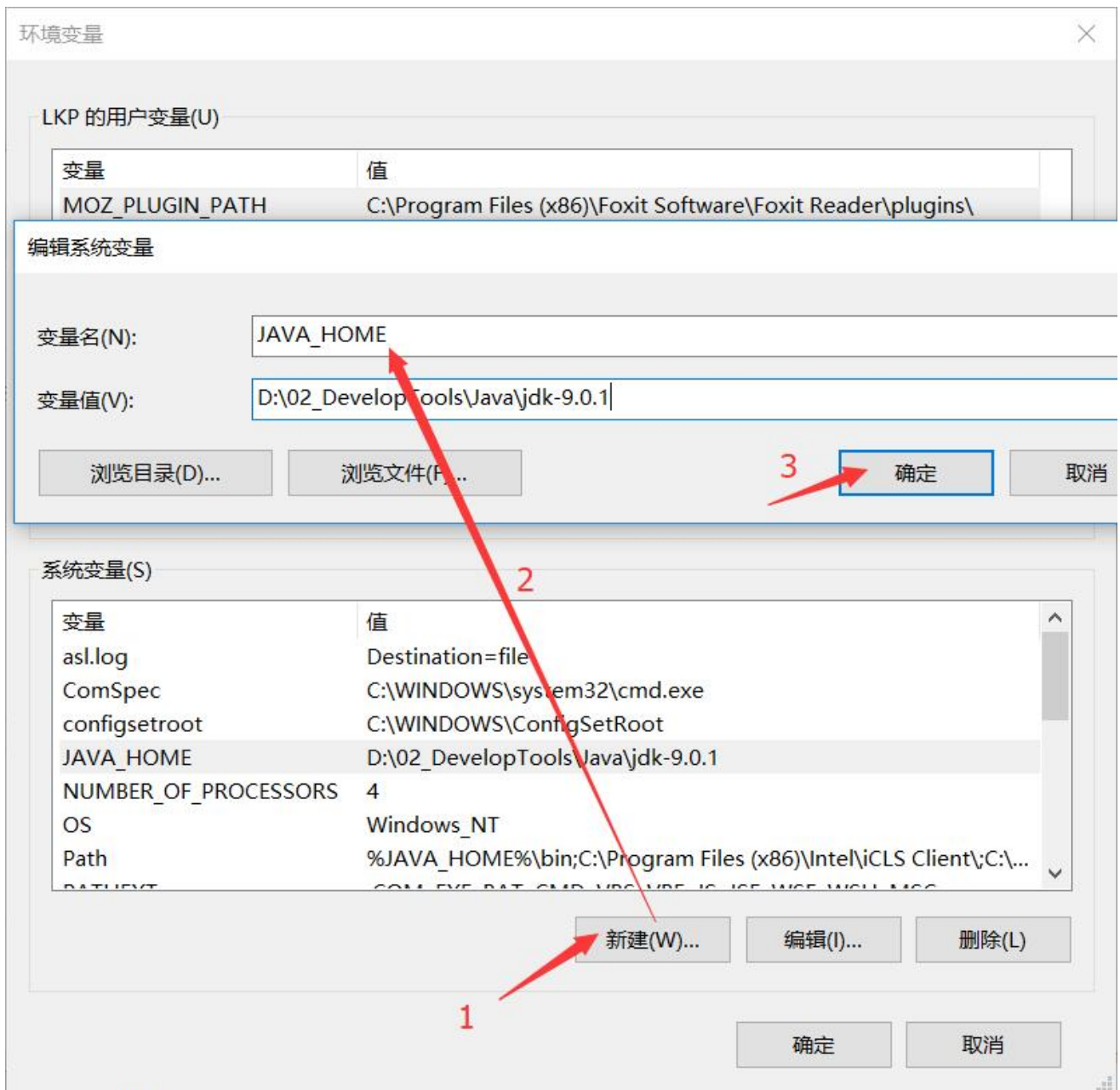
1. 文件资源管理器 --> 此电脑鼠标右键 --> 选择 属性



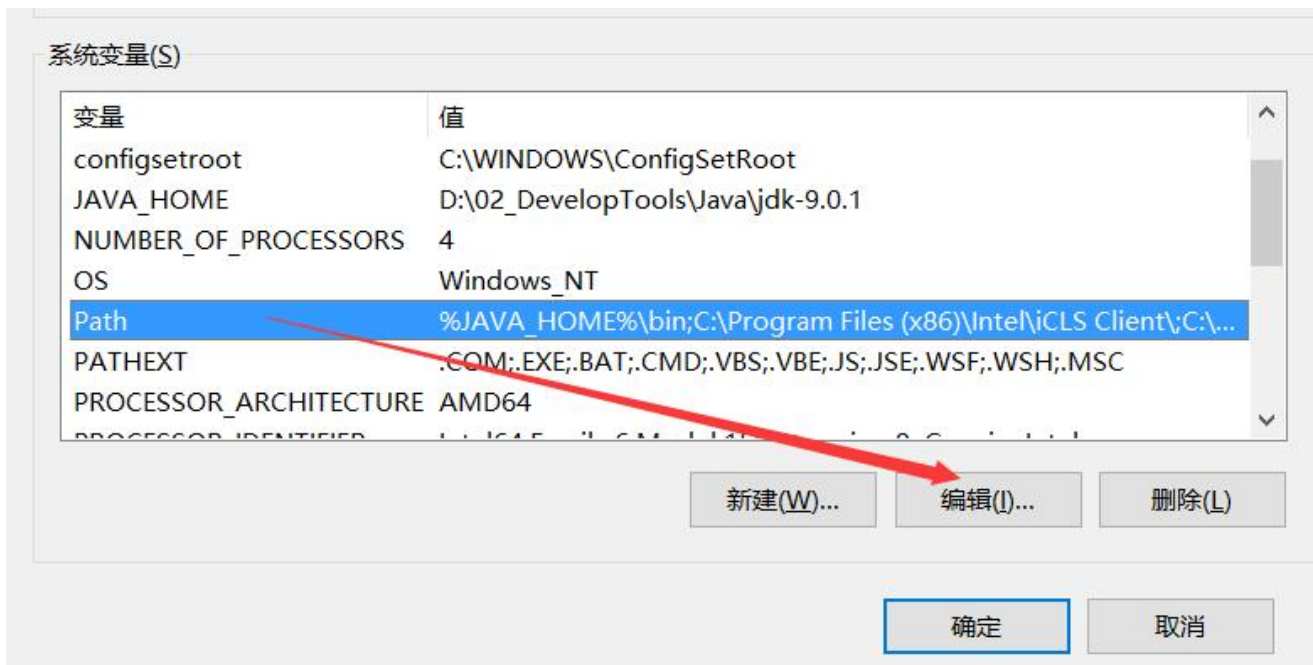
2. 选择 高级系统设置 --> 选择 环境变量



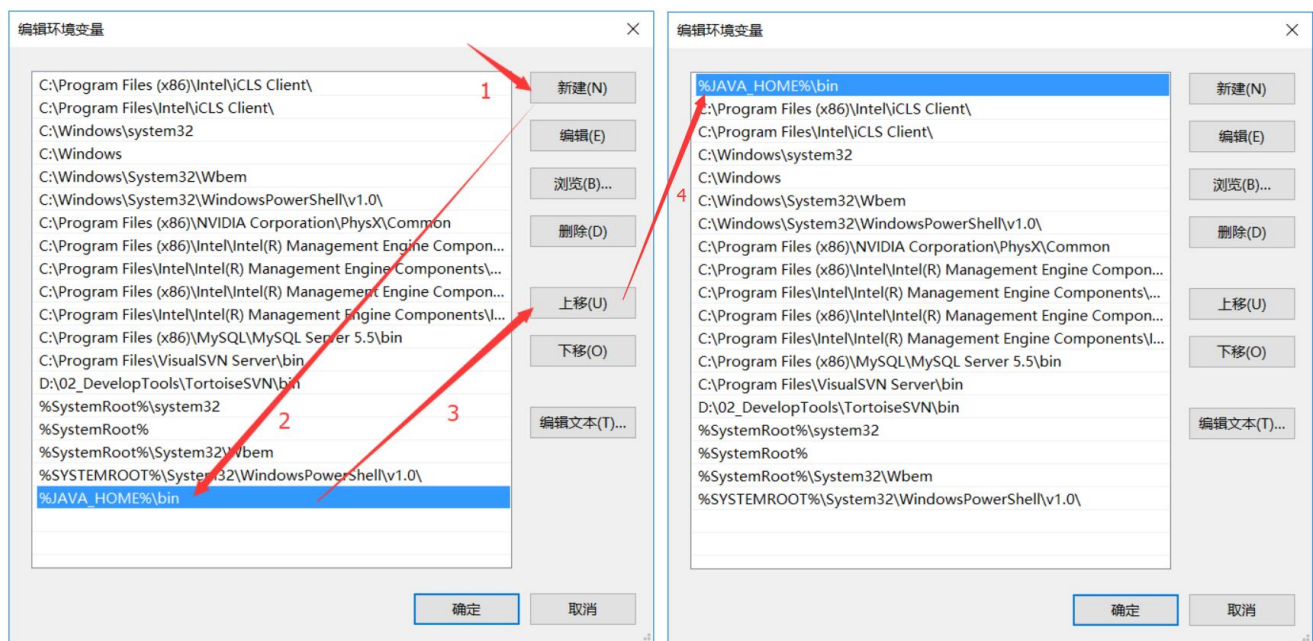
3. 点击下方系统变量的 **新建**，创建新的环境变量，变量名输入 `JAVA_HOME`，变量值输入JDK9的安装目录 `D:\02_DevelopTools\Java\jdk-9.0.1`



4. 选中 Path 环境变量，双击 或者 点击编辑



5. 点击 **新建**，键入 `%JAVA_HOME%\bin`，必须是英文格式。选中该行，上移到最上方，点击确定。



6. 环境变量配置完成，重新开启DOS命令行，在任意目录下输入 `javac` 命令。

小结:

略

第三章 HelloWorld入门程序

1.开发第一程序

目标:

开发环境已经配置好了,接下来可以编写人生中第一个程序了.

步骤:

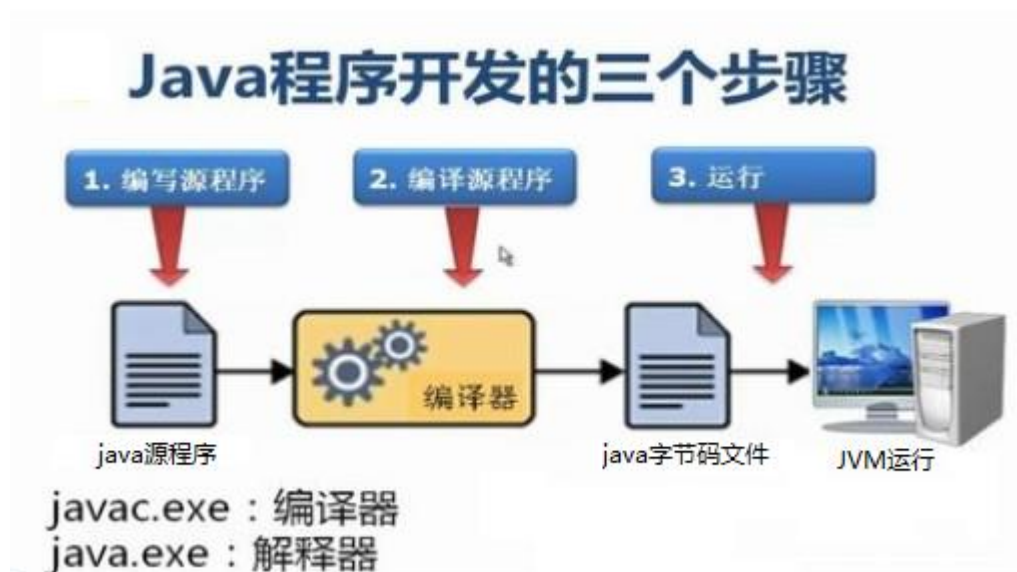
- 编写程序
- 编译程序
- 运行程序

讲解:

3.1 程序开发步骤说明

开发环境已经搭建完毕,可以开发我们第一个Java程序了。

Java程序开发三步骤：**编写、编译、运行。**



3.2 编写Java源程序

1. 在 `D:\day01` 目录下新建文本文件，完整的文件名修改为 `HelloWorld.java`，其中文件名为 `HelloWorld`，后缀名必须为 `.java`。

2. 用记事本打开

使用EditPlus记事本软件。

3. 在文件中键入文本并保存，代码如下：

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

文件名必须是 `HelloWorld`，保证文件名和类的名字是一致的，注意大小写。

每个字母和符号必须与示例代码一模一样。

标点符号和字母都要在英文半角下输入

第一个 `HelloWorld` 源程序就编写完成了，但是这个文件是程序员编写的，JVM是看不懂的，也就不能运行，因此我们必须将编写好的 `Java源文件` 编译成JVM可以看懂的 `字节码文件`，也就是 `.class` 文件。

3.3 编译Java源文件

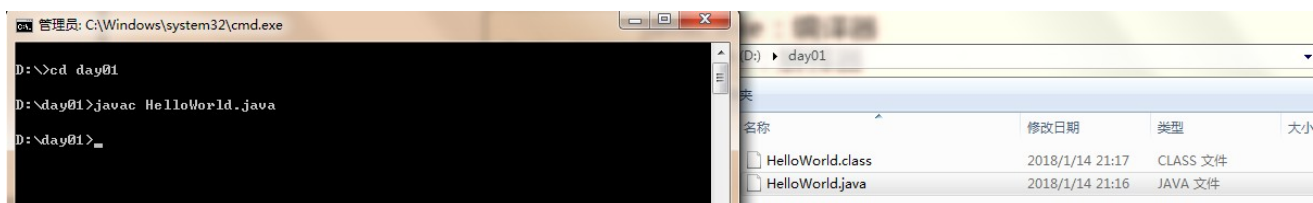
在DOS命令行中，**进入Java源文件的目录**，使用 `javac` 命令进行编译。

命令：

```
javac Java源文件名.后缀名
```

举例：

```
javac HelloWorld.java
```



编译成功后，命令行没有任何提示。打开 `D:\day01` 目录，发现产生了一个新的文件 `HelloWorld.class`，该文件就是编译后的文件，是Java的可运行文件，称为**字节码文件**，有了字节码文件，就可以运行程序了。

Java源文件的编译工具 `javac.exe`，在JDK安装目录的bin目录下。但是由于配置了环境变量，可以在任意目录下使用。

3.4 运行Java程序

在DOS命令行中，**进入Java源文件的目录**，使用 `java` 命令进行运行。

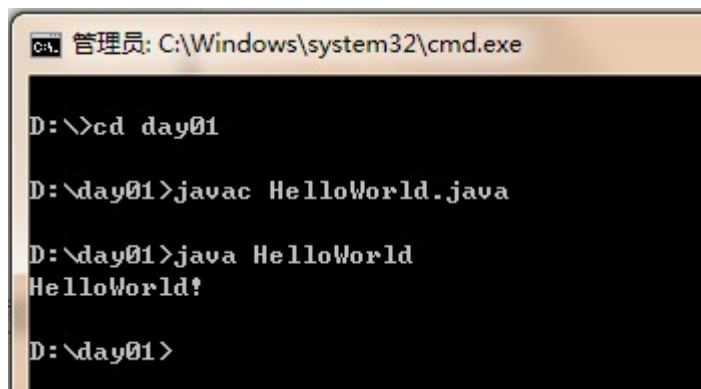
命令：

```
java 类名字
```

举例：

```
java HelloWorld
```

java HelloWorld 不要写 不要写 不要写 .class



Java程序 `.class文件` 的运行工具 `java.exe`，在JDK安装目录的bin目录下。但是由于配置了环境变量，可以再任意目录下使用。

小结:

- 编写源程序
 - 创建一个后缀为.java的文件
 - 打开该文件
 - 书写源代码
- 编译

- 在.java文件所在的路径打开终端
 - 使用编译工具进行编译 `javac 文件名.java`
 - 编译之后会产生一个.class文件 (字节码文件)
- 运行程序
 - 使用运行工具 `java 类名`
 - 注意类名后面不需要加后缀

2.开发第一个程序的注意事项:

目标:

编写第一个开发程序的注意事项

步骤:

演示注意事项

讲解:

注意事项:

- 文件后缀必须是.java文件
- 文件名必须是 `HelloWorld` , 保证文件名和类的名字是一致的, 注意大小写
- 标点符号和字母都要在英文半角下输入
- 运行命令为: `java 类名`, 千万别加.class文件
- 如果修改了源代码, 必须重新编译再运行

小结:

略

3.入门程序说明

目标:

刚刚已经书写了第一个java程序,接下来我们来解释一下这个程序,首先先来说一下编译和运行,以及main方法

步骤:

- 编译和运行
- main方法

讲解:

编译和运行是两回事

- **编译**: 是指将我们编写的Java源文件翻译成JVM认识的 `.class` 文件, 在这个过程中, `javac` 编译器会检查我们所写的程序是否有错误, 有错误就会提示出来, 如果没有错误就会编译成功。
- **运行**: 是指将 `.class` 文件交给JVM去运行, 此时JVM就会去执行我们编写的程序了。

关于main方法

- **main方法**：称为主方法。写法是**固定格式**不可以更改。main方法是程序的入口点或起始点，无论我们编写多少程序，JVM在运行的时候，都会从main方法这里开始执行。

4. 添加注释comment

目标:

为了让大家养成良好的编程习惯,以及让别人和自己能够快速理解我们自己编写的程序,我们一般会对代码添加注释,所以接下来要来学习一下注释

步骤:

- 注释的概念
- 注释的分类

讲解:

- **注释**：就是对代码的解释和说明。其目的是让人们能够更加轻松地了解代码。为代码添加注释，是十分必须的，它不影响程序的编译和运行。
- Java中有单行注释和多行注释
 - 单行注释以 `//` 开头，以 换行 结束
 - 多行注释以 `/*` 开头，以 `*/` 结束

小结:

略

5. 关键字keywords

目标:

在HelloWorld案例中我们书写了public class等单词,其实这些就是关键字,接下来我们就得了解关键字的概述,以及常见的关键字

步骤:

- 关键字的概述
- HelloWorld案例中出现的关键字

讲解:

- **关键字**：是指在程序中，Java已经定义好的单词，具有特殊含义。
 - HelloWorld案例中，出现的关键字有 `public`、`class`、`static`、`void` 等，这些单词已经被Java定义好，全部都是小写字母，在 EditPlus 中颜色特殊。
 - 关键字比较多，不能死记硬背，学到哪里记到哪里即可。

小结:

- 关键字是具有特殊含义的单词
- HelloWorld案例中出现的关键字: `public`、`class`、`static`、`void` 等

6. 标识符

目标:

在HelloWorld案例中,我们书写的类名就是标识符,接下来我们就来学习一下标识符的概述和命名规则以及规范

步骤:

- 标识符的概念
- 标识符的命名规则
- 标识符的命名规范

讲解:

- **标识符**: 是指在程序中, 我们自己定义内容。比如类的名字、方法的名字和变量的名字等等, 都是标识符。
 - HelloWorld案例中, 出现的标识符有类名字 `HelloWorld`。
- **命名规则**: **硬性要求**
 - 标识符可以包含 英文字母26个(区分大小写)、`0-9`数字、`$` (美元符号) 和 `_` (下划线) 等。
 - 标识符不能以数字开头。
 - 标识符不能是关键字。
- **命名规范**: **软性建议**
 - 类名规范: 首字母大写, 后面每个单词首字母大写 (大驼峰式)。
 - 方法名规范: 首字母小写, 后面每个单词首字母大写 (小驼峰式)。helloWorld
 - 变量名规范: 首字母小写, 后面每个单词首字母大写 (小驼峰式)。

小结:

略

第四章 常量

目标:

了解常量的概念和分类

步骤:

- 常量的概述
- 常量的分类
- 常量的练习

讲解:

4.1 概述

常量：是指在Java程序中固定不变的数据。

4.2 分类

类型	含义	数据举例
整数常量	所有的整数	0 , 1 , 567 , -9
小数常量	所有的小数	0.0 , -0.1 , 2.55
字符常量	单引号引起来,只能写一个字符,必须有内容	'a' , '(空格)' , '好'
字符串常量	双引号引起来,可以写多个字符,也可以不写	"A" , "Hello" , "你好" , ""
布尔常量	只有两个值（ 流程控制中讲解 ）	true , false
空常量	只有一个值（ 引用数据类型中讲解 ）	null

4.3 练习

需求：输出各种类型的常量。

```
public class ConstantDemo {
    public static void main(String[] args){
        // 输出整数常量
        System.out.println(123);
        // 输出小数常量
        System.out.println(0.125);
        // 输出字符常量
        System.out.println('A');
        // 输出布尔常量
        System.out.println(true);
        // 输出字符串常量
        System.out.println("你好Java");
    }
}
```

第五章 变量和数据类型

5.1 数据类型

目标:

掌握数据类型的分类

步骤:

- 数据类型的分类

- 基本数据类型

讲解:

数据类型分类

Java的数据类型分为两大类：

- 基本数据类型：包括 整数、浮点数、字符、布尔。
- 引用数据类型：包括 类、数组、接口、Lambda。字符串引用数据类型

基本数据类型

四类八种基本数据类型：

数据类型	关键字	内存占用	取值范围
字节型	byte	1个字节	-128 至 127 (2^7-1)
短整型	short	2个字节	-32768 至 32767($2^{15}-1$)
整型	int (默认)	4个字节	-2^{31} 至 $2^{31}-1$
长整型	long	8个字节	-2^{63} 至 $2^{63}-1$
单精度浮点数	float	4个字节	1.4013E-45 至 3.4028E+38
双精度浮点数	double (默认)	8个字节	4.9E-324 至 1.7977E+308
字符型	char	2个字节	0 至 $2^{16}-1$
布尔类型	boolean	1个字节	true , false

注意事项:

- 1.字符串类型是引用数据类型,不是基本数据类型
- 2.Java中的默认类型：整数类型是 `int`、浮点类型是 `double`。
- 3.小数是一个无限趋近的数,并非精确值
- 4.数据范围不一定是字节越大范围越大

小结:

- 数据类型: 基本数据类型,引用数据类型
- 基本数据类型: 4类8种

5.2 变量概述

目标:

变量的概述

步骤:

- 变量的概述

讲解:

- **变量**：常量是固定不变的数据，那么在程序中可以变化的量称为变量。

数学中，可以使用字母代替数字运算,例如 $x=1+5$ 或者 $6=x+5$ 。

程序中，可以使用字母保存数字的方式进行运算，提高计算能力，可以解决更多的问题。比如x保存5，x也可以保存6，这样x保存的数据是可以改变的，也就是我们所讲解的变量。

Java中要求一个变量每次只能保存一个数据，必须要明确保存的数据类型。

小结:

- 注意:
 - java中的变量只能保存一个数据
 - java中的变量必须明确保存的数据类型

5.3 变量的定义

目标:

掌握变量定义的格式

步骤:

- 变量的定义格式
- 变量的练习

讲解:

变量定义的格式包括三个要素：`数据类型`、`变量名`、`数据值`。

格式

格式一声明的同时初始化：`数据类型 变量名 = 数据值;`

格式二先声明后初始化：

`数据类型 变量名;`
`变量名 = 值;`

练习

定义所有基本数据类型的变量，代码如下：

```
public class Variable {  
    public static void main(String[] args){  
        // 定义字节型变量  
        byte b = 100;  
        System.out.println(b);  
        // 定义短整型变量  
        short s = 1000;  
    }  
}
```

```

        System.out.println(s);
        // 定义整型变量
        int i = 123456;
        System.out.println(i);
        // 定义长整型变量
        long l = 12345678900L;
        System.out.println(l);
        // 定义单精度浮点型变量
        float f = 5.5F;
        System.out.println(f);
        // 定义双精度浮点型变量
        double d = 8.5;
        System.out.println(d);
        // 定义布尔型变量
        boolean bool = false;
        System.out.println(bool);
        // 定义字符型变量
        char c = 'A';
        System.out.println(c);
    }
}

```

long类型：建议数据后加L表示。

float类型：建议数据后加F表示。

小结:

- 定义变量的格式:数据类型 变量名 = 数据值;

5.4 注意事项

目标:

使用变量的注意事项

步骤:

- 演示使用变量的注意事项

讲解:

- 变量名称：在同一个大括号范围内，变量的名字不可以相同。
- 变量赋值：定义的变量，不赋值不能使用。
- 给float和long类型的变量赋值,记得加F和L
- 超过了变量的作用域,就不能再使用变量了 作用域: 从定义变量的位置开始,一直到直接所属的大括号结束为止
- 同一条语句可以同时定义多个同类型的变量
- 如果给byte和short类型的变量赋值,记得不要超过该类型所表示的范围

小结:

略