

# 1. Java概述

---

## 1.1 Java语言发展史（了解）

语言：人与人交流沟通的表达方式

计算机语言：人与计算机之间进行信息交流沟通的一种特殊语言

Java语言是美国Sun公司（Stanford University Network）在1995年推出的计算机语言

Java之父：詹姆斯·高斯林（James Gosling）

2009年，Sun公司被甲骨文公司收购，所以我们现在访问oracle官网即可：<https://www.oracle.com>

当前，我们课程使用的JDK版本：11.0

## 1.2 Java语言跨平台原理（理解）

Java程序并非是直接运行的，Java编译器将Java源程序编译成与平台无关的字节码文件(class文件)，然后由Java虚拟机（JVM）对字节码文件解释执行。所以在不同的操作系统下，只需安装不同的Java虚拟机即可实现java程序的跨平台。

## 1.3 JRE和JDK（记忆）

JVM（Java Virtual Machine），Java虚拟机

JRE（Java Runtime Environment），Java运行环境，包含了JVM和Java的核心类库（Java API）

JDK（Java Development Kit）称为Java开发工具，包含了JRE和开发工具

总结：我们只需安装JDK即可，它包含了java的运行环境和虚拟机。

## 1.4 JDK的下载和安装（应用）

### 1.4.1 下载

通过官方网站获取JDK

<http://www.oracle.com>

**注意：**针对不同的操作系统，需要下载对应版本的JDK。

具体下载步骤请参见《JDK下载及安装说明文档》

### 1.4.2 安装

傻瓜式安装，下一步即可。但默认的安装路径是在C:\Program Files下，为方便统一管理建议修改安装路径，将与开发相关的软件都安装到一个目录下，例如：E:\develop。

**注意：**安装路径不要包含中文或者空格等特殊字符（使用纯英文目录）。

具体安装步骤请参见《JDK下载及安装说明文档》

### 1.4.3 JDK的安装目录介绍

目录名称	说明
bin	该路径下存放了JDK的各种工具命令。javac和java就放在这个目录。
conf	该路径下存放了JDK的相关配置文件。
include	该路径下存放了一些平台特定的头文件。
jmods	该路径下存放了JDK的各种模块。
legal	该路径下存放了JDK各模块的授权文档。
lib	该路径下存放了JDK工具的一些补充JAR包。

## 2. 第一个演示程序

### 2.1 常用DOS命令（应用）

在接触集成开发环境之前，我们需要使用命令行窗口对java程序进行编译和运行，所以需要知道一些常用DOS命令。

1、打开命令行窗口的方式：win + r打开运行窗口，输入cmd，回车。

2、常用命令及其作用

操作	说明
盘符名称:	盘符切换。E:回车，表示切换到E盘。
dir	查看当前路径下的内容。
cd 目录	进入单级目录。cd itheima
cd ..	回退到上一级目录。
cd 目录1\目录2...	进入多级目录。cd itheima\javaSE
cd \	回退到盘符目录。
cls	清屏。
exit	退出命令提示符窗口。

### 2.2 Path环境变量的配置（应用）

#### 2.2.1 为什么配置环境变量

开发Java程序，需要使用JDK提供的开发工具（比如javac.exe、java.exe等命令），而这些工具在JDK的安装目录的bin目录下，如果不配置环境变量，那么这些命令只可以在该目录下执行。我们不可能把所有的java文件都放到JDK的bin目录下，所以配置环境变量的作用就是可以使bin目录下的java相关命令可以在任意目录下使用。

#### 2.2.2 配置环境变量步骤

具体配置步骤请参见《Java环境变量配置说明》文档。

## 2.3 HelloWorld案例（应用）

HelloWorld案例是指在计算机屏幕上输出“HelloWorld”这行文字。各种计算机语言都习惯使用该案例作为第一个演示案例。

### 2.3.1 Java程序开发运行流程

开发Java程序，需要三个步骤：编写程序，编译程序，运行程序。

### 2.3.2 HelloWorld案例的编写

- 1、新建文本文档文件，修改名称为HelloWorld.java。
- 2、用记事本打开HelloWorld.java文件，输写程序内容。

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

### 2.3.3 HelloWorld案例的编译和运行

存文件，打开命令行窗口，将目录切换至java文件所在目录，编译java文件生成class文件，运行class文件。

编译：javac 文件名.java

范例：javac HelloWorld.java

执行：java 类名

范例：java HelloWorld

## 2.4 HelloWorld案例常见问题（理解）

### 2.4.1 BUG

在电脑系统或程序中，隐藏着的一些未被发现的缺陷或问题统称为bug（漏洞）。

### 2.4.2 BUG的解决

- 1、具备识别BUG的能力：多看
- 2、具备分析BUG的能力：多思考，多查资料
- 3、具备解决BUG的能力：多尝试，多总结

### 2.4.3 HelloWorld案例常见问题

- 1、非法字符问题。Java中的符号都是英文格式的。
- 2、大小写问题。Java语言对大小写敏感（区分大小写）。
- 3、在系统中显示文件的扩展名，避免出现HelloWorld.java.txt文件。

- 4、编译命令后的java文件名需要带文件后缀.java
- 5、运行命令后的class文件名（类名）不带文件后缀.class
- ...

## 2.5 Notepad++软件的安装和使用（应用）

### 2.5.1 什么要使用Notepad++软件

Notepad++功能比windows中的自带记事本功能强大，除了可以用来制作一般的纯文字说明文件，也十分适合编写计算机程序代码。Notepad++有行号，能够快速定位问题位置，还有语法高亮度显示、代码折叠等功能。而且它是免费的。

### 2.5.2 Notepad++软件安装

安装：傻瓜式安装，一直下一步即可。建议也安装到统一的发展软件目录下，比如E:\develop。

具体安装步骤请参见《Nodepad++软件安装及配置说明》文档。

### 2.5.3 Notepad++软件配置

安装完毕之后，为了使用方便，做一个简单的配置：修改默认语言和编码。

具体配置的说明请参见《Nodepad++软件安装及配置说明》文档。

## 3. java基础语法

### 3.1 注释（理解）

注释是对代码的解释和说明文字，可以提高程序的可读性，因此在程序中添加必要的注释文字十分重要。Java中的注释分为三种：

单行注释。单行注释的格式是使用//，从//开始至本行结尾的文字将作为注释文字。

```
// 这是单行注释文字
```

多行注释。多行注释的格式是使用/\* 和 \*/将一段较长的注释括起来。

```
/*
这是多行注释文字
这是多行注释文字
这是多行注释文字
*/
注意：多行注释不能嵌套使用。
```

文档注释。文档注释以/\*\* 开始，以 \*/ 结束。（以后讲）

### 3.2 关键字（理解）

关键字是指被java语言赋予了特殊含义的单词。

关键字的特点：

关键字的字母全部小写。

常用的代码编辑器对关键字都有高亮显示，比如现在我们能看到的public、class、static等。

### 3.3 常量（应用）

常量：在程序运行过程中，其值不可以发生改变的量。

Java中的常量分类：

字符串常量 用双引号括起来的多个字符（可以包含0个、一个或多个），例如"a"、"abc"、"中国"等

整数常量 整数，例如：-10、0、88等

小数常量 小数，例如：-5.5、1.0、88.88等

字符常量 用单引号括起来的一个字符，例如：'a'、'5'、'B'、'中'等

布尔常量 布尔值，表示真假，只有两个值true和false

空常量 一个特殊的值，空值，值为null

除空常量外，其他常量均可使用输出语句直接输出。

```
public class Demo {  
    public static void main(String[] args) {  
        System.out.println(10); // 输出一个整数  
        System.out.println(5.5); // 输出一个小数  
        System.out.println('a'); // 输出一个字符  
        System.out.println(true); // 输出boolean值true  
        System.out.println("欢迎来到黑马程序员"); // 输出字符串  
    }  
}
```

### 3.4 数据类型（记忆、应用）

#### 3.4.1 计算机存储单元

我们知道计算机是可以用来存储数据的，但是无论是内存还是硬盘，计算机存储设备的最小信息单元叫“位（bit）”，我们又称之为“比特位”，通常用小写的字母“b”表示。而计算机中最基本的存储单元叫“字节（byte）”，

通常用大写字母“B”表示，字节是由连续的8个位组成。

除了字节外还有一些常用的存储单位，其换算单位如下：

1B（字节）= 8bit

1KB = 1024B

1MB = 1024KB

1GB = 1024MB

1TB = 1024GB

### 3.4.2 Java中的数据类型

Java是一个强类型语言，Java中的数据必须明确数据类型。在Java中的数据类型包括基本数据类型和引用数据类型两种。

Java中的基本数据类型：

数据类型	关键字	内存占用	取值范围
整数类型	byte	1	-128~127
	short	2	-32768~32767
	int(默认)	4	-2的31次方到2的31次方-1
	long	8	-2的63次方到2的63次方-1
浮点类型	float	4	负数：-3.402823E+38到-1.401298E-45 正数：1.401298E-45到3.402823E+38
	double(默认)	8	负数：-1.797693E+308到-4.9000000E-324 正数：4.9000000E-324 到1.797693E+308
字符类型	char	2	0-65535
布尔类型	boolean	1	true , false

说明：

e+38表示是乘以10的38次方，同样，e-45表示乘以10的负45次方。

在java中整数默认是int类型，浮点数默认是double类型。

## 3.5 变量（应用）

### 3.5.1 变量的定义

变量：在程序运行过程中，其值可以发生改变的量。

从本质上讲，变量是内存中的一小块区域，其值可以在一定范围内变化。

变量的定义格式：

```
数据类型 变量名 = 初始化值; // 声明变量并赋值
int age = 18;
System.out.println(age);
```

或者

```
// 先声明，后赋值（使用前赋值即可）
数据类型 变量名；
变量名 = 初始化值；
double money；
money = 55.5；
System.out.println(money);
```

还可以在同一行定义多个同一种数据类型的变量，中间使用逗号隔开。但不建议使用这种方式，降低程序的可读性。

```
int a = 10, b = 20; // 定义int类型的变量a和b，中间使用逗号隔开
System.out.println(a);
System.out.println(b);

int c,d; // 声明int类型的变量c和d，中间使用逗号隔开
c = 30;
d = 40;
System.out.println(c);
System.out.println(d);
```

变量的使用：通过变量名访问即可。

### 3.5.2 使用变量时的注意事项

1. 在同一对花括号中，变量名不能重复。
2. 变量在使用之前，必须初始化（赋值）。
3. 定义long类型的变量时，需要在整数的后面加L（大小写均可，建议大写）。因为整数默认是int类型，整数太大可能超出int范围。
4. 定义float类型的变量时，需要在小数的后面加F（大小写均可，建议大写）。因为浮点数的默认类型是double，double的取值范围是大于float的，类型不兼容。

## 3.6 标识符（记忆、理解）

标识符是用户编程时使用的名字，用于给类、方法、变量、常量等命名。

Java中标识符的组成规则：

由字母、数字、下划线“\_”、美元符号“\$”组成，第一个字符不能是数字。

不能使用java中的关键字作为标识符。

标识符对大小写敏感（区分大小写）。

Java中标识符的命名约定：

小驼峰式命名：变量名、方法名

首字母小写，从第二个单词开始每个单词的首字母大写。

大驼峰式命名：类名

每个单词的首字母都大写。

另外，标识符的命名最好可以做到见名知意

例如：username、studentNumber等。

### 3.7 类型转换（理解）

在Java中，一些数据类型之间是可以相互转换的。分为两种情况：自动类型转换和强制类型转换。

自动类型转换：

把一个表示数据范围小的数值或者变量赋值给另一个表示数据范围大的变量。这种转换方式是自动的，直接书写即可。例如：

```
double num = 10; // 将int类型的10直接赋值给double类型
System.out.println(num); // 输出10.0
```

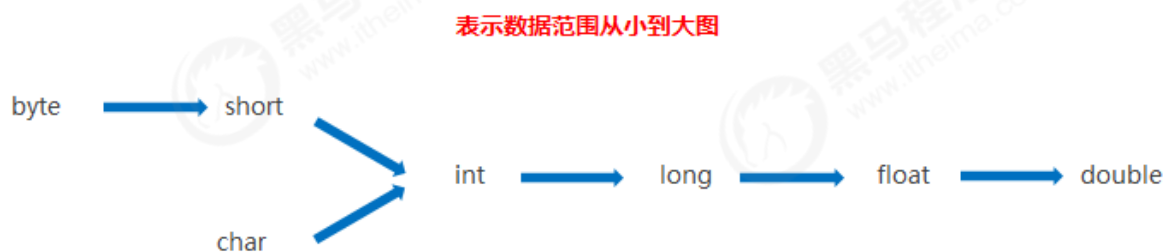
强制类型转换：

把一个表示数据范围大的数值或者变量赋值给另一个表示数据范围小的变量。

强制类型转换格式：目标数据类型 变量名 = (目标数据类型)值或者变量;

例如：

```
double num1 = 5.5;
int num2 = (int) num1; // 将double类型的num1强制转换为int类型
System.out.println(num2); // 输出5（小数位直接舍弃）
```



说明：

1. char类型的数据转换为int类型是按照码表中对应的int值进行计算的。比如在ASCII码表中，'a'对应97。

```
int a = 'a';
System.out.println(a); // 将输出97
```

2. 整数默认是int类型，byte、short和char类型数据参与运算均会自动转换为int类型。

```
byte b1 = 10;
byte b2 = 20;
byte b3 = b1 + b2;
// 第三行代码会报错，b1和b2会自动转换为int类型，计算结果为int，int赋值给byte需要强制类型转换。
// 修改为：
int num = b1 + b2;
// 或者：
byte b3 = (byte) (b1 + b2);
```



1. boolean类型不能与其他基本数据类型相互转换。