[01 内容介绍 2](#_Toc156053563)

[0001 Java后端技术体系 2](#_Toc156053564)

[0002 Java就业方向 3](#_Toc156053565)

[0003 Java开发场景 3](#_Toc156053566)

[0004 Java应用领域 4](#_Toc156053567)

[02 Java概述 5](#_Toc156053568)

[0006 程序举例 5](#_Toc156053569)

[0007 Java故事 5](#_Toc156053570)

[0008 Java特性 5](#_Toc156053571)

[0009 sublime 6](#_Toc156053572)

[0010 jdk介绍 6](#_Toc156053573)

[0011我的电脑 7](#_Toc156053574)

[0012 jdk安装 7](#_Toc156053575)

[0013 jdk配置 7](#_Toc156053576)

[0016 快速入门 8](#_Toc156053577)

[0017 运行机制 9](#_Toc156053578)

[0018 开发细节 9](#_Toc156053579)

[0019 学习方法 10](#_Toc156053580)

[0020 转义字符 11](#_Toc156053581)

[0021 易犯错误 12](#_Toc156053582)

[0022-0024 注释 12](#_Toc156053583)

[0025 代码规范 13](#_Toc156053584)

[0026-0030 DOS命令【了解即可】 13](#_Toc156053585)

[0031-0032 本章作业 15](#_Toc156053586)

[03 变量 16](#_Toc156053587)

[0034 变量原理 16](#_Toc156053588)

[0035 变量概念 16](#_Toc156053589)

[0036 变量入门 17](#_Toc156053590)

[0037 变量细节/注意事项 17](#_Toc156053591)

[0038 加号使用 17](#_Toc156053592)

[0039 数据类型 17](#_Toc156053593)

[0040 整型使用 18](#_Toc156053594)

[0041 整型细节 18](#_Toc156053595)

[0042 浮点数使用 19](#_Toc156053596)

[0043-0044 浮点数细节 19](#_Toc156053597)

[0045 Java文档 20](#_Toc156053598)

[0046 字符型使用 20](#_Toc156053599)

[0047 字符型细节 20](#_Toc156053600)

[0028 字符型本质 20](#_Toc156053601)

# 01 内容介绍

## 0001 Java后端技术体系

1. Java基础
2. 编程思想

Java概述——Java运行机制、JDK、

变量——数据类型、

运算符——二进制、位运算符、

控制结构——循环、

数组、排序和查找——数组、

面向对象编程——成员方法、成员方法传参机制、构造器、封装、继承、多态。

1. 编程能力

面向对象编程——类变量和类方法、接口、内部类

枚举和注解

Exception——异常体系图、异常处理

常用类——String、StringBuffer、StringBuilder

集合——集合框架体系、Collection（ArrayList、Vector、HashSet）、Map（HashMap）

泛型——自定义泛型

线程——线程使用、Synchronized、互斥锁

IO流——输入流（InputStream）、输入流（OutputStream）

1. 分析需求、代码实现能力

网络编程——TCP编程

反射——反射获取类的结构信息

Mysql——Mysql数据类型、CRUD==增删改查、函数、外连接、索引

JDBC和连接池——JDBC API、事务、连接池（Druid）、DAO增删改查-Basic DAO

正则表达式——三个常用类（Pattern、Matcher、PatternSyntaxException）、分类、捕获、反向引用、元字符

Java8、Java11新特性——8（lambda、函数式接口、steam API）、11（类型推断、集合增强API）

1. Java高级

Java多线程/高并发

数据结构和算法

设计模式（23种）

JVM

1. JavaWEB
2. 主流的框架和项目管理

SSM、项目管理、数据库

1. 分布式 微服务 并行架构
2. DevOps（开发运维一体化）、自动化部分管理项目、解决CI/CD
3. 大数据技术
4. 项目
5. 大厂高频面试题
6. 底层源码/内核研究
7. 编程基础的拓展

## 0002 Java就业方向

1. JavaEE软件工程师——电商、团购、众筹、sns、教育、金融、搜索
2. 大数据软件（应用、算法、数据分析和挖掘）工程师
3. Android软件工程师

## 0003 Java开发场景

1. SSM

Spring（轻量级的容器框架）

SpringMVC（分层的web开发框架）

Mybatis（持久化框架）

1. Android核心代码
2. 大数据Hadoop

## 0004 Java应用领域

1. 企业级应用
2. Android平台应用
3. 移动领域应用

# 02 Java概述

## 0006 程序举例

Def程序：计算机执行某些程序或解决某个问题所编写的一系列有序指令的集合。

## 0007 Java故事

1995 sun正式发布Java第一个版本。

2009 甲骨文公司收购sun

Java7 2011年发布

Java8 2014年发布，长期支持版本

Java11 2018年发布，长期支持版本

Java技术体系平台

1. Java SE标准版
2. Java EE企业版
3. Java ME小型版——移动终端上

## 0008 Java特性

Java重要特点

1. 面向对象
2. 健壮性——强类型机制、异常处理、垃圾的自动收集
3. 跨平台性——一个编译好的.class文件可以在多个系统中运行
4. 解释型语言（JavaScript、PHP、Java均位解释性语言）（C/C++均为编译性语言）

解释性语言：编译后的代码不能直接被机器执行，需要解释器来执行；

编译性语言：编译后的代码，可以直接被机器执行。

## 0009 sublime

常见的Java的开发工具

editplus、notepad++

sublime Text

IDEA

Eclipse

## 0010 jdk介绍

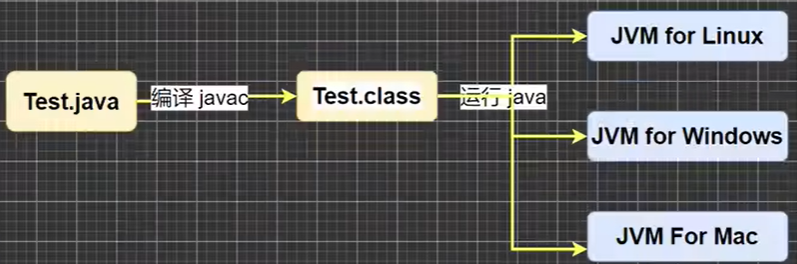
Java的运行机制和运行过程

Java语言具有跨平台性，因为有了JVM，同一个Java程序可以在多个操作系统中都可以执行。

安装了JDK后就拥有了JVM。

Java虚拟机==JVM==Java virtual machine

1. JVM==虚拟计算机，具有指令集并使用不同的存储区域。负责执行指令，管理数据、内存、寄存器，包含在JDK中。
2. 对不同的平台，有不同的虚拟机。
3. Java虚拟机机制屏蔽了底层运行平台的差别，实现了“一次编译、到处运行”。



JDK==java Development Kit==Java开发工具包

1. JDK = JRE + Java的开发工具【Java，javac，javadoc，javap等】
2. JDK = JVM + Java SE标准类库（java核心类库）+开发工具集。
3. JDK提供给Java开发人员使用，其中包含Java的开发工具，也包含了JRE。

因此安装了JDK，就不用再单独安装JRE。

JRE==Java Runtime Environment==Java运行环境

1. JRE = JVM+ Java SE标准类库（java核心类库）【类】
2. 包含Java虚拟机和Java程序所需的核心类库等。

若想运行一个开发好的Java程序，计算机中只需安装JRE即可。

## 0011我的电脑

桌面右键——个性化——主题——桌面图标设置——计算机==我的电脑

## 0012 jdk安装



Java SE 8

Java SE 11

## 0013 jdk配置

配置环境变量path

1. dos命令行

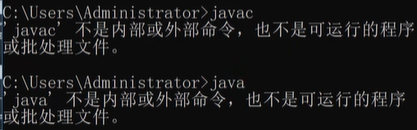
进入1：右键左下角图表——运行——cmd——确定

进入2：win+R——cmd——确定

1. 敲入javac或java，出现错误提示的原因：

当前执行的程序在当前目录下若不存在，系统会在系统中已有的一个名为path的环境变量指定的目录中查找，若仍未找到，则出现错误提示。

所以进入到jdk安装路径\bin目录下，执行javac，会看到javac参数提示信息。



1. 配置环境变量path的步骤
2. 我的电脑-属性-高级系统设置-环境变量
3. 增加JAVA\_HOME环境变量，指向jdk的安装目录。
4. 编辑path环境变量，增加%JAVA\_HOME%\bin
5. 打开DOS命令行，任意目录下敲入javac，若出现javac的参数信息，配置成功。

注：环境变量中，用户变量和系统变量的区别：系统变量的使用范围更广泛。

## 0016 快速入门

Sublime快速入门

1、文件另存为

2、调节字体大小

首选项——设置——调节font\_size中的数值——保存

Exercise：开发Hello.java程序，输入“hello world”

步骤：

1. 将Java代码编写到扩展名为Hello.java的文件中



1. 通过javac命令，对该java文件进行编译，生成.class文件。

编译过程：在Hello.java所在的目录中输入cmd，进入控制台，输入javac Hello.java

输入时，按Tab键可自动补齐.java文件的完整名称。



若java代码中含有中文：

【点击文件——以……编码保存——GBK或UTF-8——保存文件后，再编译。】

【仍然不行的话，在控制台中输入javac –encoding UTF-8 Hello.java】

控制台右键——属性——当前代码页：



1. 通过Java命令，对生成的class文件进行运行。

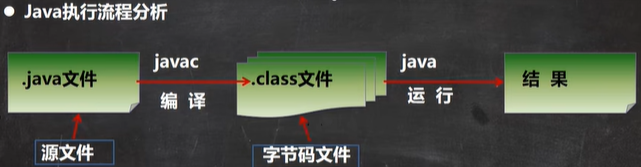
运行过程：在控制台中，输入java Hello

//不要加.class了，输入java Hello后，实际上运行的是Hello这个类。



## 0017 运行机制

Java执行流程分析



Def编译：

编译的语句：javac Hello.java

输入时，按Tab键可自动补齐.java文件的完整名称。

1. 有了java源文件，通过编译器，将其编译成JVM可识别的字节码文件（.class）
2. 在该源文件目录下，通过javac编译工具，对Hello.java文件进行编译。
3. 若程序没有错误，则没有任何提示，但在当前目录下会出现一个Hello.class文件，该文件称为字节码文件，是可以执行的java程序。

Def运行：

运行的语句：java Hello

1. 有了可执行的java程序（Hello.class字节码文件），通过运行工具java.exe对字节码文件进行执行。 本质就是.class装载到jvm机执行。

java程序开发注意事项：

修改过的Hello.java源文件，需要重新编译，生成新的.class文件后，再次执行方可生效。

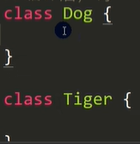
## 0018 开发细节

1. Java源文件以.java为扩展名，源文件的基本组成部分是类==class，如Hello类。
2. Java应用程序的执行入口是main()方法，它有固定的书写格式：

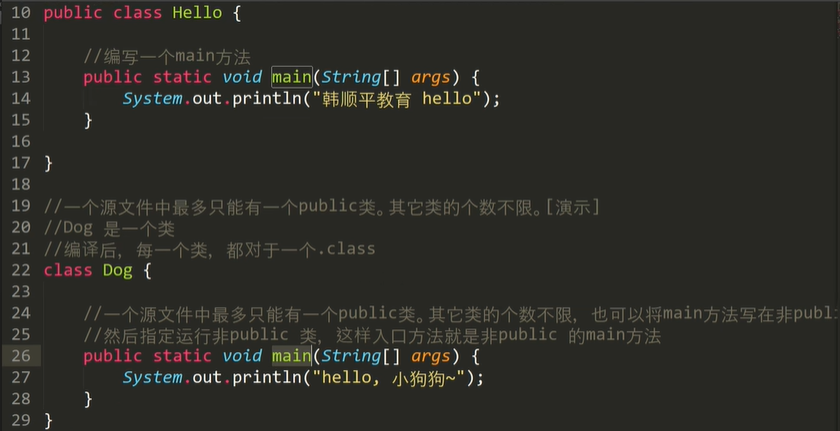
public static void main(String[] args) {……}

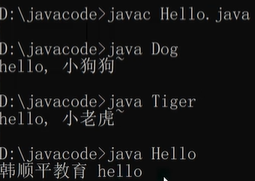
1. Java语言区分大小写。
2. Java方法由一条条语句构成，每个语句以；结束
3. 大括号成对出现。
4. 一个源文件中最多只能有一个public类，其他类的个数不限。

【编译之后，每一个类都会对应生成一个.class（无论是否是public类）】

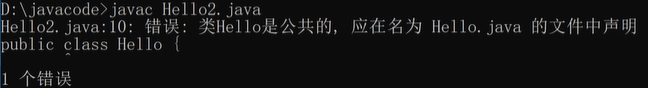
 

也可以将main方法写在非public类中，然后指定运行非public类，这样入口方法就是非public的main方法。





1. 若源文件中包含一个public类，则文件名必须按照该类名命名。



## 0019 学习方法

1. 看看需求能否使用传统技术解决
2. 引出新技术和知识点
3. 学习新技术或知识点的基本原理和基本语法（先不考虑细节）
4. 快速入门（基本程序，crud）
5. 开始研究技术的注意事项、使用细节、使用规范、如何优化。

## 0020 转义字符

1. \t 一个制表位，用于对齐
2. \n 换行符
3. \\ 一个\
4. \” 一个”
5. \’ 一个‘
6. \r 一个回车

回车的解读：

Eg：System.out.println(“韩顺平教育\r北京”);

Step1：输出韩顺平教育

Step2：\r 表示回车，光标从“育“的后面回到”韩“的前面。

Step3：把“韩顺“用”北京“代替

最终结果：输出了“北京平教育“

Eg：System.out.println(“韩顺平教育\r\n北京”);

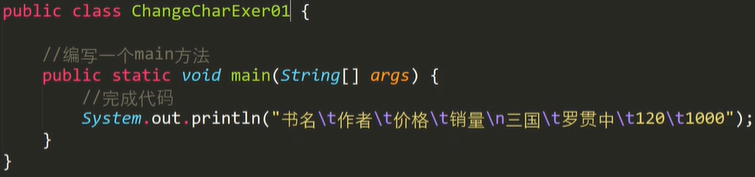
输出：

韩顺平教育

北京

Exercise：





## 0021 易犯错误

1. 找不到文件

文件名写错 或 路径错误

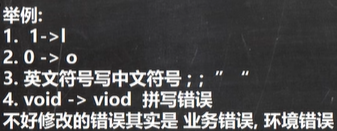
1. 主类名和文件名不一致

声明为public的主类应与文件名一致，否则编译失败。

1. 缺少分号

编译失败，注意错误出现的行数，再到源代码中指定位置改错。

常见语法错误举例：



## 0022-0024 注释

Def注释：

用于注解说明解释程序的文字。提高代码可读性。

Java的注释类型：

（1）单行注释： //注释内容

（2）多行注释： /\* 注释内容\*/

注：被注释的文字不会被JVM解释执行； 多行注释中不能再嵌套多行注释。

（3）文档注释：注释内容可以被JDK提供的工具javadoc所解析，生成一套以网页文件形式体现的该程序的说明文档，一般写在类中。

文档注释的基本格式：



如何生成对应的文档注释：

Javadoc –d 文件夹名 –xx –yy Demo3.java

注释快捷键：ctrl + /

取消注释：ctrl + /



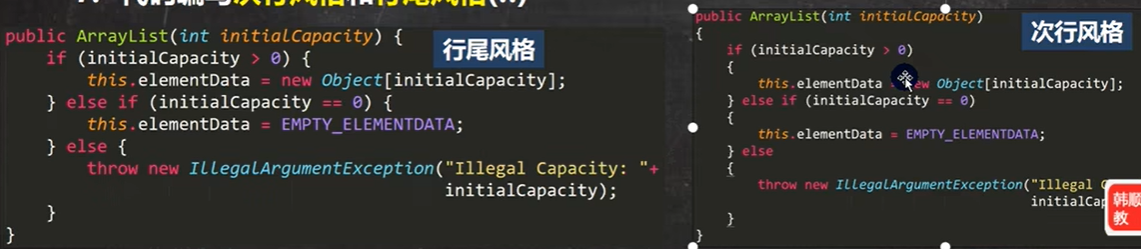
javadoc标签

## 0025 代码规范

1. 类、方法的注释，要以javadoc的方式来写。
2. 非javadoc的注释，往往给代码的维护者看，注重告诉读者为什么这么写，如何修改、注意什么问题等。
3. 选中，然后使用tab操作，实现缩进，默认整体向右移动，

使用shift+tab整体向左移动。

1. 运算符和等号两边习惯性各加一个空格，eg：2 + 4 \* 5 + 345 – 89
2. 源文件使用utf-8编码
3. 行宽度不超过80字符
4. 代码编写 行尾风格 和 次行风格。



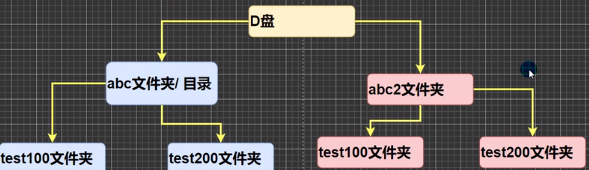
## 0026-0030 DOS命令【了解即可】

Def DOS == Disk Operating System == 磁盘操作系统：

1. 相对路径、绝对路径

相对路径：从当前目录开始定位，形成的一个路径。

绝对路径：从顶级目录开始定位，形成的一个路径。



Eg：当前路径为test100，访问test200中的文件hello.txt：

相对路径：..\..\abc2\test200\hello.txt

绝对路径：d:\ abc2\test200\hello.txt

1. DOS系统：接收指令-》解析指令-》执行指令
2. 常见的DOS命令

md d:\\temp 创建

rd d:\\temp 删除

dir d:\abc2\test2 查看当前目录有什么

cd = change directory

cd /D c: 切换到c盘

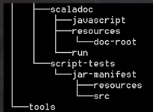
cd d:abc2\test2 切换到当前磁盘的其他目录下（使用绝对路径）

cd ..\..\abc2\test2 切换到当前磁盘的其他目录下（使用相对路径）

cd .. 切换到上一级

cd \ 切换到跟目录

tree d:\abc 查看指定目录下所有的子级目录



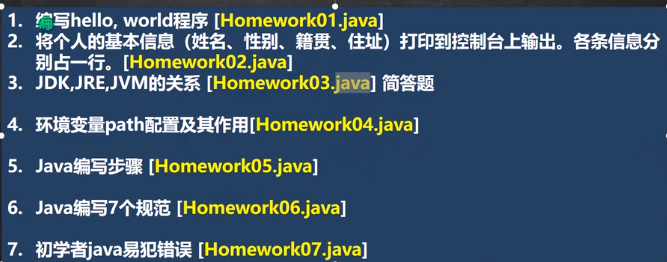
cls 清屏

exit 退出DOS

注：其他的相对少见的几个指令

md, rd, copy[拷贝文件], del, echo[输入内容到文件], type, move[剪切]

## 0031-0032 本章作业

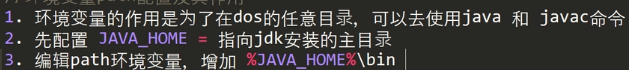


我的问题：总是生成不了.class文件。。。

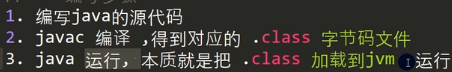
1. JDK、JRE、JVM的关系



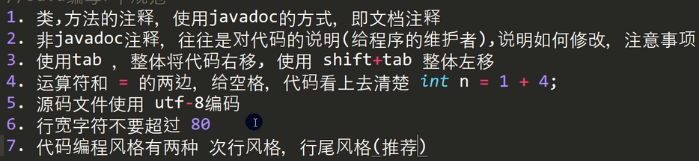
1. 环境变量path配置及其作用



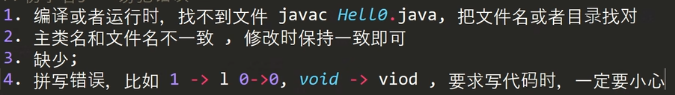
1. Java编写步骤



1. Java编写的7个规范



1. 初学者Java易犯错误



# 03 变量

本章重点内容：

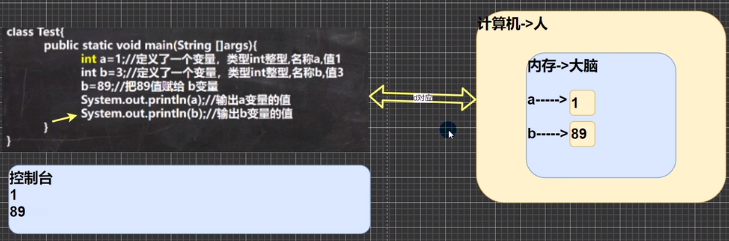
（1）变量介绍 （2）+号的使用 （3）数据类型 （4）编码 （5）数据类型转换

## 0034 变量原理

为什么需要变量

变量是程序的基本组程单位。

变量有3个基本要素：类型、名称、值。



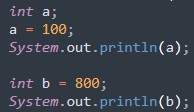
## 0035 变量概念

Def变量

变量==内存中一个数据存储空间的表示。

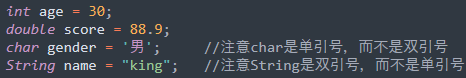
变量使用的基本步骤：

1. 声明变量
2. 赋值
3. 使用



## 0036 变量入门

变量类型



## 0037 变量细节/注意事项

变量使用的注意事项

* 1. 变量表示内存中的一个存储区域【不同的变量，类型不同，占用的空间大小不同】int——4B，double——8B
  2. 该区域有自己的名称【变量名】和类型【数据类型】
  3. 变量必须先声明，后使用，即有顺序
  4. 该区域的数据可以在同一类型范围内不断变化
  5. 变量在同一个作用域内不能重名
  6. 变量 = 变量名 + 值 + 数据类型【变量三要素】

## 0038 加号使用

1. 当左右两边都是数值类型时，做加法运算
2. 当左右两边有一方为字符串，做拼接运算
3. 注意：加减法运算的运算顺序是从左到右的

System.out.println(100+98); 198

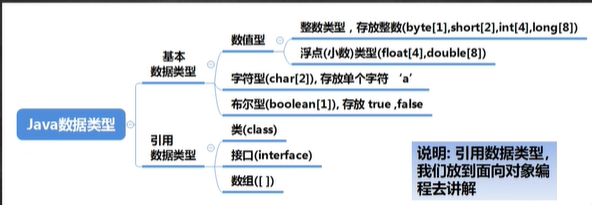
System.out.println(“100“+98); “100”代表字符串100，输出结果为10098

System.out.println(100+3+“hello“); 103hello

System.out.println(“hello“+100+3); hello1003

## 0039 数据类型

每种数据都定义了明确的数据类型，在内存中分配了不同大小的内存空间【字节】



1. Java数据类型基本分为2个大类：数据类型、引用类型
2. 基本数据类型有8种：数值型【byte、short、int、long、float、double】、char、boolean

PS：在C语言中，char占1个字节，在Java中，char占2个字节。

1. 引用类型【类，接口，数组】

PS：String不是基本数据类型，而是一个类。

## 0040 整型使用

整数类型：用于存放整数值的，eg：12，30，3345……

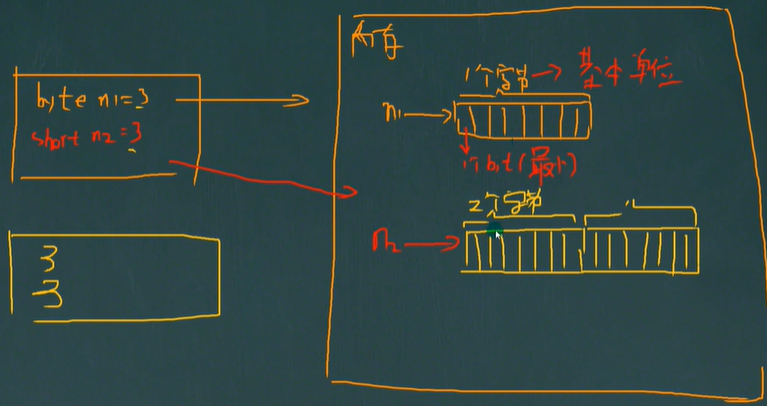


PS：在够用的情况下，最好选用小的数据类型。

## 0041 整型细节

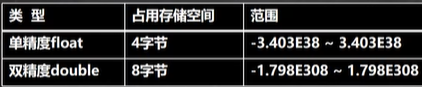
1. Java各整数类型有固定的范围和字段长度，不受具体OS的影响，以保证java程序的可移植性。
2. Java的整型常量默认为int型，声明long型常量须后加’l’或者’L’
3. java程序中变量常常声明为int型，除非不足以表示大数，才使用long。
4. bit：计算机中的最小存储单位。byte：计算机中基本存储单元。 1byte = 8bit

eg：byte n1 = 3; short n2 = 3;



## 0042 浮点数使用

浮点类型：用于表示一个小数，比如123.4，7.8，0.12等



说明：

1. 关于浮点数在机器中存放形式的简单说明：浮点数 = 符号位+指数位+尾数位
2. 尾数部分可能丢失，造成精度损失（小数都是近似值）

## 0043-0044 浮点数细节

1. 与整数类型类似，Java浮点类型也有固定的范围和字段长度，不受具体OS的影响。

【float为4个字节，double为8个字节】

1. Java的浮点型常量默认为double型，声明float型常量，须后加’f’或者’F’

float num1 = 1.1; //错误；从double转换到float可能会有损失

float num2 = 1.1F; //正确

double num3 = 1.1; //正确

double num4 = 1.1F; //float可以强制转化为double

1. 浮点型常量有2种表示形式：

（1）十进制数形式：5.12 512.0f .512 ——必须有小数点

double num5 = .123; 正确

（2）科学计数法形式：

5.12e2（==5.12\*10^2==512.0） 5.12E-2（==5.12\*10^-2==0.0512）

1. 通常情况下，应该使用double类型，因为它比float性更精确。

double num9 = 2.1234567851; //输出2.1234567851

float num10 = 2.1234567851F; //输出2.1234567

1. 浮点数使用陷阱！！！：2.7 和 8.1 / 3比较

double num11 = 2.7; //输出2.7

double num12 = 8.1/3; //输出2.6999999999999997

当我们对运算结果是小数的进行相等判断时，要小心。

应该是以2个数的差值的绝对值，在某个精度范围内进行判断。

if (num11 == num12) //不对

if(Math.abs(num11 – num12) < 0.000001) //对

细节：如果是直接查询得到的小数，或者直接赋值，则可以直接判断相等。

## 0045 Java文档

Java API文档：

1. API == Application Programming Interface == 应用程序变成接口：是Java提供的基本编程接口（java提供的类还有相关的方法）。

中文在线文档：<https://www.matools.com——>在线API

1. Java语言提供了大量的基础类，因此Oracle公司也为这些基础类提供了相应的API文档，用于告诉开发者如何使用这些类，以及这些类里包含的方法。
2. Java类的组织形式

## 0046 字符型使用

## 0047 字符型细节

## 0028 字符型本质