

# Java基础

## ✧ 1. 背景知识

---

### 创始人

- 詹姆斯·高斯林

### 历史

- 1995年Sun公司
- 2009年Oracle公司

### 应用

- 桌面应用开发
- 企业级应用开发
- 移动应用开发
- 服务器系统
- 大数据开发
- 游戏开发

### Java技术体系

- Java SE：标准版
- JavaEE：企业版
- JavaME：小型版

## 问题

## \* 2. 快速入门

### JDK下载安装

- 官网: <https://www.oracle.com/java/technologies/>
- 环境变量配置
  - 方法一: 在系统环境变量的Path里面配置JDK安装的BIN的完整目录  
如: D:abc\jdk-17.0.32\bin
  - 方法二: 系统环境变量里先配置JAVA\_HOME  
变量名为: `JAVA_HOME`  
变量值为: `D:abc\jdk-17.0.32`  
然后再Path里面配置 `%JAVA_HOME%\bin`

### cmd常见命令

- JDK查看编译工具和运行工具版本号的命令: `javac -version` 和 `java -version`

```
1 E: //切换到E盘
2 cd [目录] //进入指定的目录
3 cd .. //退回到上一级目录
4 cd / //退回到根目录
5 dir //显示当前目录下所有的内容
6 cls //清空屏幕
```

### Java程序编程的3个步骤

编写代码 → 编译代码 (javac) → 运行代码 (java)

文件名称要和类名称一致

## JDK的组成

- ① JVM: Java虚拟机, 真正运行Java程序的地方
- ② 核心类库: Java自己写好的程序, 给程序员自己的程序进行调用的
- ③ JRE: Java的运行环境
- ④ JDK: Java开发工具包 (上面的所有)

## javadoc命令

- 文档注释是一种特殊的注释格式, 用于为 Java 程序中的类、方法、字段等元素提供文档说明。文档注释以 `/**` 开始, 以 `*/` 结束, 可以包含多行描述性文本和标记
- 文档注释是一种标准的注释格式, 在使用工具生成 API 文档时可以被提取出来, 用于生成详细的程序文档。它们提供了对代码的解释、使用示例、参数说明、返回值说明等重要信息, 帮助其他开发者理解和使用代码

标签	作用
<code>@author</code>	标识一个类的作者
<code>@param</code>	方法的参数
<code>@return</code>	标明返回值类型, 一般用于方法注释, 不能出现在构造方法中
<code>{@value}</code>	显示常量的值, 该常量必须是 <code>static</code> 属性
<code>@since</code>	版本号, 标明从哪个版本起开始有这个函数
<code>@version</code>	指定类的版本
<code>@exception</code>	可能抛出异常的说明, 一般用于方法注释
<code>@throws</code>	也是可能抛出异常的说明
<code>@serial</code>	说明一个序列化属性
<code>@serialData</code>	说明通过 <code>writeObject()</code> 和 <code>writeExternal()</code> 方法写的的数据
<code>@serialField</code>	说明一个 <code>ObjectStreamField</code> 组件

- javadoc 命令语法格式如下:

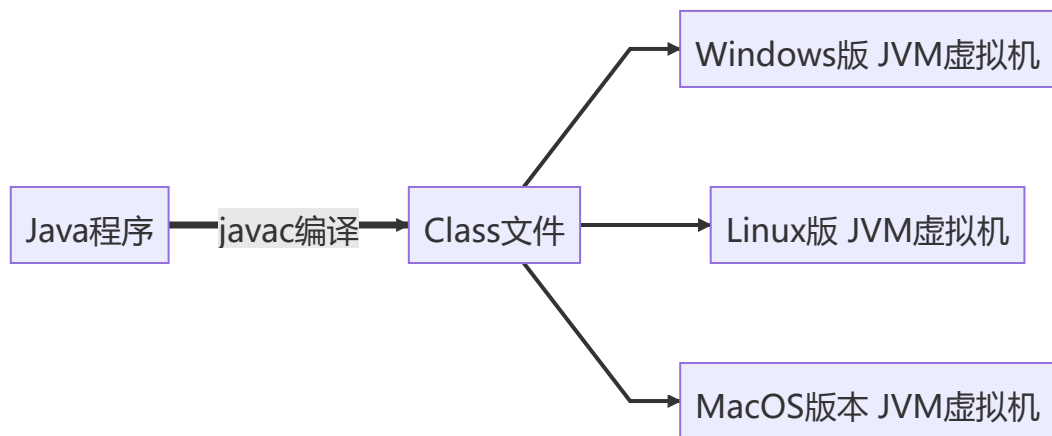
```
1 javadoc [options] [packagenames][sourcefiles]
```

格式说明：

- (1) options 表示 javadoc 命令的选项；
- (2) packagenames 表示包名；
- (3) sourcefiles 表示源文件名。

## Java跨平台原理

跨平台性的原理是因为在不同版本的操作系统中安装有不同版本的Java虚拟机，Java程序的运行只依赖于Java虚拟机，和操作系统并没有直接关系。从而做到一处编译，处处运行。



## ✧ 3. Java基础语法

### 注释

```
1  1.单行注释：
2      // 后面跟解释文字
3  2.多行注释
4      /*
5          这里写注释文字
6          可以写多行
7      */
8  3.文档注释
9      /**
10         这里写文档注释
11         也可以写多行，文档注释可以利用JDK的工具生成帮助文档
12     */
```

## 字面量

常用数据	生活中的写法	程序中的写法	说明
整数	666, -88	666, -88,0	写法一致
小数	13.14, -5.21	13.14, -5.21	写法一致
字符	A,0,我	'A','0', '我'	程序中必须使用单引号，有且仅能一个字符
字符串	开发喵	"开发喵","0"	程序中必须使用双引号，内容可有可无
布尔值	真, 假	true>false	只有两个值，true表示真，false表示假
空值		值是null	一个特殊的值，空值

## 变量

数据类型    变量名称    =    初始值;

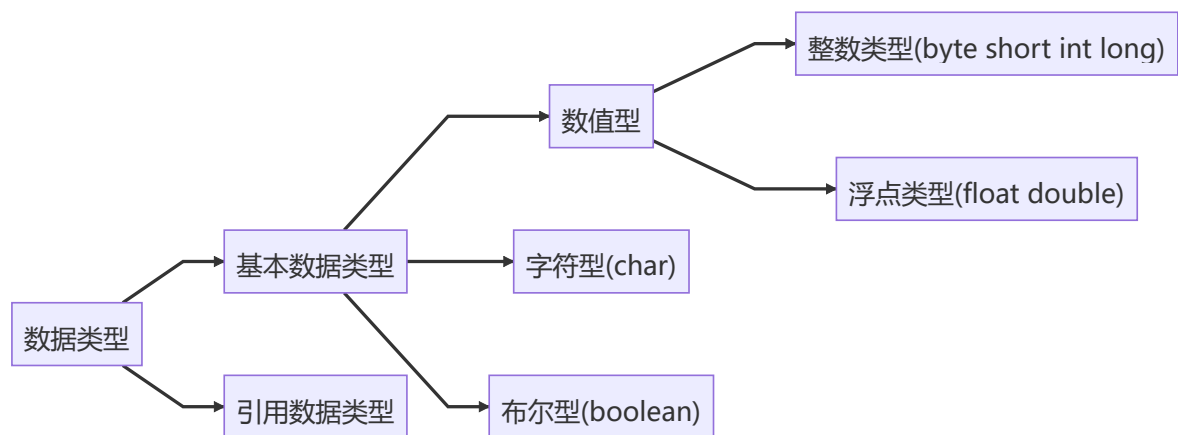
## 标识符

- 标识符由字母（A~Z 和 a~z）、数字（0~9）、下划线（\_）、美元符号（\$）以及部分Unicode字符集（各符号之间没有空格）组成。
- 标识符的首字母以字母、下划线或美元符号开头，后面可以是任何字母、数字、美元符号或下划线，但不能以数字开头。
- 标识符的命名不能是关键字、布尔值（true、false）和null。
- 标识符区分大小写，没有长度限制。

## 关键字

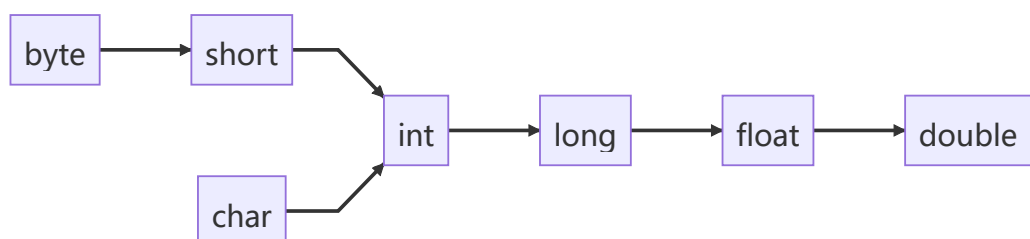
<b>abstract</b>	<b>default</b>	<b>if</b>	<b>protected</b>	<b>throws</b>
<b>assert</b>	<b>do</b>	<b>implements</b>	<b>public</b>	<b>transient</b>
<b>boolean</b>	<b>double</b>	<b>import</b>	<b>return</b>	<b>try</b>
<b>break</b>	<b>else</b>	<b>instanceof</b>	<b>strictfp</b>	<b>void</b>
<b>byte</b>	<b>enum</b>	<b>int</b>	<b>short</b>	<b>volatile</b>
<b>case</b>	<b>extends</b>	<b>interface</b>	<b>static</b>	<b>while</b>
<b>catch</b>	<b>final</b>	<b>long</b>	<b>super</b>	_
<b>char</b>	<b>finally</b>	<b>native</b>	<b>switch</b>	
<b>class</b>	<b>float</b>	<b>new</b>	<b>synchronized</b>	
<b>const</b>	<b>for</b>	<b>package</b>	<b>this</b>	
<b>continue</b>	<b>goto</b>	<b>private</b>	<b>throw</b>	

## 数据类型



## 数据类型转换

- 自动类型转换（扩展原始转换）



多种数据类型参与运算，其结果以大的数据类型为准

**i** `byte`, `short`, `char` 三种类型数据在和其他类型数据运算时，都会转换为 `int` 类型再运算

- 强制类型转换（缩小原始转换）

目标数据类型 变量名 = (目标数据类型)被转换的数据;

```
1 int a = 10;
2 byte b = (byte)a;
```

```
1 public class TypeConversionDemo3 {
2     public static void main(String[] args) {
3         // 目标: 掌握强制类型转换。
4         int a = 20;
5         byte b = (byte) a;
6         System.out.println(a);
7         System.out.println(b);
8
9         int i = 1500;
10        byte j = (byte) i;
11        System.out.println(j);
12
13        double d = 99.5;
14        int m = (int) d; // 强制类型转换
15        System.out.println(m); // 丢掉小数部分, 保留整数部分
16    }
17 }
```

## 字符在计算机中的存储原理

ASCII编码表中字符编码的规律:

- i** ① 1.字符0对应48，后面的1,2,3,4...9 对应的十进制整数依次往后顺延
- ② 2.字符a对应97，后面的b,c,d,e...z 对应的十进制整数依次往后顺延
- ③ 3.字符A对应65，后面的B,C,D,E...Z 对应的十进制整数依次往后顺延

ASCII 字符代码表 一

高四位   低四位	ASCII非打印控制字符										ASCII 打印字符																
	0000					0001					0010	0011	0100	0101	0110	0111											
	0					1					2	3	4	5	6	7											
	+进制	字符	ctrl	代码	字符解释	+进制	字符	ctrl	代码	字符解释	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符	ctrl
0000	0	0	BLANK NULL	空	16	▶	^P	DLE	数据链路转意	32		48	0	64	@	80	P	96	`	112	p						
0001	1	1	☺	^A SOH 头标开始	17	◀	^Q	DC1	设备控制 1	33	!	49	1	65	A	81	Q	97	a	113	q						
0010	2	2	☹	^B STX 正文开始	18	↕	^R	DC2	设备控制 2	34	"	50	2	66	B	82	R	98	b	114	r						
0011	3	3	♥	^C ETX 正文结束	19	!!	^S	DC3	设备控制 3	35	#	51	3	67	C	83	S	99	c	115	s						
0100	4	4	♦	^D EOT 传输结束	20	↑	^T	DC4	设备控制 4	36	\$	52	4	68	D	84	T	100	d	116	t						
0101	5	5	♣	^E ENQ 查询	21	⌘	^U	NAK	反确认	37	%	53	5	69	E	85	U	101	e	117	u						
0110	6	6	♠	^F ACK 确认	22	■	^V	SYN	同步空闲	38	&	54	6	70	F	86	V	102	f	118	v						
0111	7	7	●	^G BEL 震铃	23	↑↓	^W	ETB	传输块结束	39	'	55	7	71	G	87	w	103	g	119	w						
1000	8	8	◼	^H BS 退格	24	↑	^X	CAN	取消	40	(	56	8	72	H	88	X	104	h	120	x						
1001	9	9	○	^I TAB 水平制表符	25	↓	^Y	EM	媒体结束	41	)	57	9	73	I	89	Y	105	i	121	y						
1010	A	10	◻	^J LF 换行/新行	26	→	^Z	SUB	替换	42	*	58	:	74	J	90	Z	106	j	122	z						
1011	B	11	◻	^K VT 垂直制表符	27	←	^[	ESC	转意	43	+	59	;	75	K	91	[	107	k	123	{						
1100	C	12	♀	^L FF 换页/新页	28	└─	^_	FS	文件分隔符	44	,	60	<	76	L	92	\	108	l	124							
1101	D	13	♪	^M CR 回车	29	↔	^]	GS	组分隔符	45	-	61	=	77	M	93	]	109	m	125	}						
1110	E	14	🎵	^N SO 移出	30	▲	^_	RS	记录分隔符	46	.	62	>	78	N	94	^	110	n	126	~						
1111	F	15	☼	^O SI 移入	31	▼	^-	US	单元分隔符	47	/	63	?	79	O	95	_	111	o	127	Δ						Back Space