# 第三章：词法结构

## 3.8 标识符

标识符没有长度限制，可以以字母开头或者\_或者美元符号开头或者以中文、日文、韩文等开头，不能以数字开头。如下红色的为标识符：

public class Person {

public String name;

public int age;

public void study() {

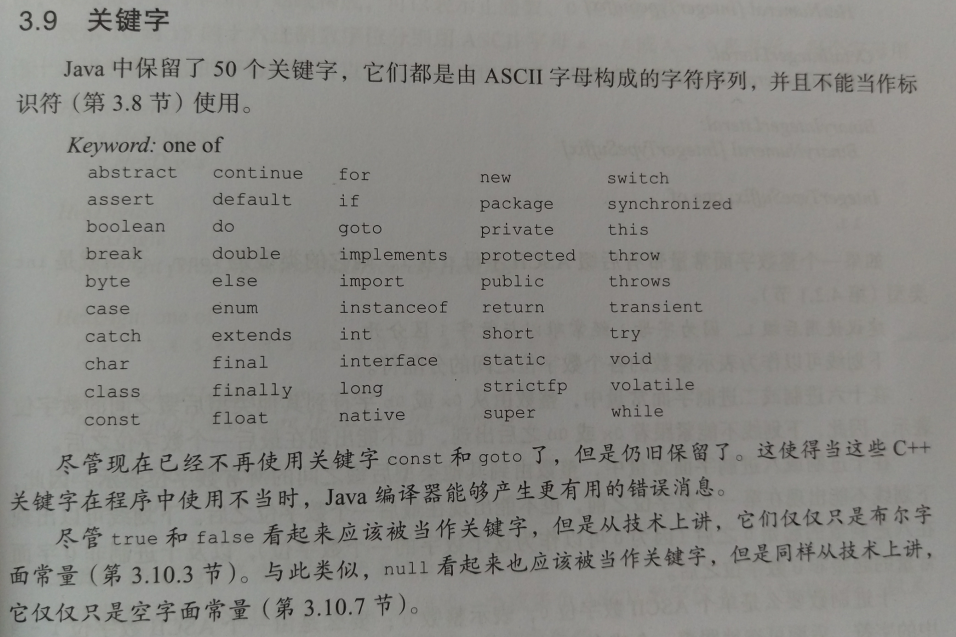
}

}

一个字符能不能用于标识符的首字母可以使用Character.isJavaIdentifierStart('戴')判断，返回true即为可以。

一个字符能不能用于标识符可以使用Character.isJavaIdentifierPart(5) 判断，返回true即为可以。

## 3.9 关键字



## 3.10字面常量

字面常量是类型为简单类型、String类型和空类型的值在源程序中的表示。

### 3.10.1 整数字面常量

整数字面常量可以表示十进制、十六进制、八进制、二进制。

如果一个整数字面常量带有后缀ASCII字面L或l，则它的类型是long，否则就是int类型。建议使用后缀L，因为字母l经常难以与数字1区分开。

下划线可以作为表示整数的各个数字位之间的分隔符。如：

int a = 123\_456;

System.out.println(a);

输出结果为：123456

十六进制前缀为0X或0x，二进制前缀为0B或0b，下划线不能紧跟着出现在0X或0B之后，也不能出现在数字的最后一位，如下：

long i = 0xFFL; 正确

long a = 0b0011\_1111\_0001L; 正确

long b = 0b0011\_1111\_0001\_L; 错误

long c = 0b\_0011\_1111\_0001L; 错误

八进制前缀0，下划线可以紧跟在这个前缀之后（因为这个0可以作为这个数字的一个数字位），但是不能出现在最后一个数字位，如下：

long a = 0\_17L; 正确

long b = 017\_L; 错误

十进制中的下划线不能出现在最前面和最后面。

十六进制中的10 ~ 15用a ~ f或A ~ F来表示。

八进制的数字位由0 ~ 7的数字组成。

注意：八进制永远由两位数字组成，单个的0表示十进制数。八进制的0为00，十六进制的0为0X0。

### 3.10.2 浮点数字面常量

浮点数字面常量包含几个部分：整数部分、十进制或十六进制小数点（用ASCII字符.表示）、小数部分、指数，以及类型后缀。

如果一个浮点数字面常量后带有后缀F或f，则它的类型是float，否则是double类型，而对于后者，也可以添加ASCII字母d或D作为后缀。

### 3.10.4 字符字面常量

字符字面常量是用由ASCII单绰号括起来的单个字母或单个转义序列表示的，其中的单引号是\u0027表示的字符。

字符字面常量只能表示UTF-16码元，也就是说它们的取值范围被限制在\u0000 ~ \uffff。字符字面常量示例如下：

‘t’

‘\t’

‘\\’

‘\u03a9’

‘\uFFFF’

‘\177’

‘TM’

‘\n’ 换行符

‘\r’ 回车符

### 3.10.5 字符串字面常量

示例如下：

“” 空字符串

“\”” 只包含一个 ” 的字符串

“This is a “ +

“two-line String” 这是一个由两个字符串字面常量构成的值为字符串的常量表达式

### 3.10.6 字符和字符串字面常量的转义序列

字符和字符串转义序列使得字符字面常量和字符串字面常量中可以不使用Unicode转义字符表示某些非图形字符，以及单引用、双引用和反斜杠字符。如下：

\b 退格BS，Unicode \u0008

\t 水平制表符HT，Unicode \u0009

\n 换行LF，Unicode \u000a

\f 换页FF，Unicode \u000c

\r 回车CR，Unicode \u000d

\” 双引号”，Unicode \u0022

\’ 单引号’ ，Unicode \u0027

\\ backslash \，Unicode \u005c

如果跟在转义序列中的反斜杠后面的字符不是ASCII字符b、t、n、f、r、”、’、\、0、1、2、3、4、5、6、或7，那么就会产生编译时错误。

### 3.10.7 空字面常量

空类型只有一个值，即空引用，用null表示。

## 3.11分隔符

分隔符总共有12个，它们都是ASCII字符（标点符），如下：

( ) ｛ ｝ [ ] ; , . ... @ ::

## 3.12操作符

操作符总共包含38个符号，它们都是由ASCII字符构成的。

= > < ! ~ ? : ->

== >= <= != && || ++ --

+ - \* / & | ^ % << >> >>>

+= -= \*= /= &= |= ^= %= <<= >>= >>>=

# 第4章 类型、值和变量

Java编码语言的类型可以分成两类：简单类型和引用类型。简单类型包括boolean类型和数字类型，其中数字类型包括整数类型byte、short、int、long和char以及浮点数类型float和double。引用类型包括类类型、接口类型和数组类型，另外还有一个特殊的空类型。对象是动态创建的类类型的实例或者是动态创建的数组，而引用类型的值是对对象的引用。所有对象，包括数组在内，都支付Object类的方法。字符串字面常量是用String对象表示的。

其实还有一种空类型，即表达式null的类型，它没有名字，所以不能声明一个变量是空类型，也不能将变量类型转换为空类型，空引用只能是空类型表达式的值。

可以忽略空类型，只需要认为空仅仅是一个特殊的字面常量，它可以具有任何引用类型。

## 4.2 简单类型和值

简单类型值互相之间不共享状态。数字类型包括整数类型和浮点数类型。

整数类型包括byte、short、int和long，它们的值分别是8位、16位、32位和64位有符号二进制补码表示的整数；char也是一种整数类型，它的值是16位无符号整数，表示UTF-16码元。

浮点数类型包括float和double，前者的值包括32位的IEEE 754浮点数，而后者的值包括64位IEEE 754浮点数。

boolean类型只有两个值：true和false。

### 4.2.1 整数类型和值

byte 类型的值是从-128到127的闭区间

short类型的值是从-32768玻32767的闭区间

int类型的值是从-2147483648到2147483647的闭区间

对于类型的值是从-9223372036854775808到922337203685477580的闭区间。

char类型的值是从’\u0000’到’\uffff’的闭区间，即从0到65535。

### 4.2.3 浮点数类型、格式和值

NaN值表示非数字，即Not a Number，NaN值用来表示某些无效操作的结果，例如0除0。

### 4.3 引用类型和值

Java有四种引用类型：类类型、接口类型、类型变量、数组类型

#### 4.3.1对象

对象是类的实例或数组。

引用值（经常直接简称“引用”）是指向这些对象的指针，并且有一个特殊的空引用，它不指向任何对象。

如果打印一个对象，则是打印这个对象的toString，如果拿一个String使用 + 链接一个对象，则是String + 对象.toString()，如果toString为null，则使用null。

每个对象都与一个监视器关联，它会被synchronized方法和synchronized语句用来对多线程并发访问对象状态进行控制。

#### 4.3.2 Object类

所有类和数组类型都继承了Object类的方法，具体包括：

* clone方法，用来创建对象的副本
* equals方法，定义了对象的比较
* finalize方法，在对象销毁前运行
* getClass方法，返回表示对象所属类的Class对象，被声明为synchronized的方法会在与该方法的类的Class对象相关联的监视器上同步。
* hashCode方法，它与equals方法一起，在诸如java.util.Hashmap这样的散列表中非常有用
* wait、notify、notifyAll方法，在使用线程的并发编程中会用到它们
* toString方法，返回对象的String表示。

#### 4.3.3 String类

String类的实例表示Unicode码位序列。

# 泛型

