# 3.1 一个简单的java应用

* Java时区分大小写的
* 类名是以大写字母开头的名词。如果名字是由多个单词组成，每个单词的第一个字母都应该大写，称为大驼峰命名法
* 源代码的文件名必须和公共类的名字相同
* 运行以编译的程序时，java虚拟机将从指定的类中的main方法开始执行。main方法必须声明为public的。

# 3.2 注释

* 单行注释： //
* 多行注释： /\* \*/
* 文档注释

注意：多行注释不能进行嵌套，因为注释的内容中有可能包含\*/。

# 3.3数据类型

* 强类型语言
* 8个基本类型：

4种整型、2种浮点型、1个字符型、1个布尔型

注意：一个表示任意精度的对象big number（Big Decimal类）

## 3.3.1 整型

整型都是有符号类型的！byte、short、 int、 long

* byte: 1字节 ———— -27 ~ 27-1
* short: 2字节 ———— -215 ~ 215-1
* int: 4字节 ———— -231 ~ 231-1
* long: 8字节 ———— -263 ~ 263-1

java的整型的大小于平台无关，方便程序的移植。

* 3000l: L、l指明为long类型
* 0x\0X：十六进制
* 0：八进制
* 0B\0b: 二进制
* 1\_000\_000: 字面量加下划线，便于阅读

## 3.3.2 浮点类型

double类型的精度时float类型的2倍，所以叫双精度型。

* float: 4字节 ———— 大约±3.402 803 47E+38（有效位数6~7位）
* double: 8字节 —— 大约±1.797 693 134 862 515 70E+308（有效位数15位）
* float类型要加f\F，否则默认为double型
* 可以采用十六进制表示浮点数
* 特殊浮点数：
* 正无穷大
* 负无穷大
* NaN不是一个数字

看一个变量是不是数字，不能采用 x == Double.NaN，应该采用Double.isNaN(x)。

## 3.3.3 char字符

* 采用单引号
* \uxxxx :可以表示Unicode字符
* 转义字符在解析代码之前就进行转化

## 3.3.4 Unicode和char类型

Unicode之前，存在不同的编码机制，例如美国的ASCII，西欧的IOS 8859-1、俄罗斯的KOI-8、中国的GB 18030等。对于任意给定的代码值，在不同的编码方案下可能对应不同的字母；采用大字符集的编码长度有可能不同。

* 码点（code point）：是指一个编码表中某个字符对应的代码值。
* Unicode标准中，码点采用十六进制书写，并加上前缀U+
* Unicode的按码点分为17个代码级别
* UTF-16采用不同长度的编码表示所有的Unicode码点

Unicode只规定每个字符对应的数字量，而不规定具体的存储方案。

UTF-16: 将任何字符对应的数字都用两个字节表示，这种方案编码方便，但是对空间比较浪费。

UTF-8: UTF-8采用变字节来编码，根据数字的范围选择采用的字节的个数，可能是一个、也可能是两个或者三个。

* 举例：

汉字“汉”，

* 对应的Unicode数字是U+6C49
* UTF-16表示：01101100 01001001
* UTF-8:

0xxxxxxx的，当成1个字节来读

110xxxxx 10xxxxxx，当成2个字节来读

1110xxxx 10xxxxxx 10xxxxxx，当成3个字节来读

Unicode2后来增加到4字节，进而出现UTF-32（4字节）、UTF-8（1~8字节）、UTF-16（2字节或4字节）。

## 3.3.5 boolean类型

* true 和false
* boolean型和整数类型之间不能进行转化

if（x=0）不能通过编译，表达式的值是整数，不能转化为boolean。再C\C++则允许这种表达。

# 3.4 变量

* Java中，每个变量都有一个类型
* 变量名必须以字母开头，后面可接数字、\_、$
* 不同的语言中对字母的定义可能不太一样
* 变量是大小写敏感的

## 3.4.1 变量的初始化

* 声明一个变量后，必须使用复制语句对变量进行显式的初始化，不能使用未经初始化的变量。
* 变量的声明应尽量靠近第一次使用它的地方

## 3.4.2 常量

* final: 关键字来修饰常量，只能赋值一次，赋值后不能再修改
* static final: 类常量可以被类的方法使用
* public static final:的类常量还可以被其他类使用