学习内容参考网址：

菜鸟编程:<http://www.runoob.com/java/java-tutorial.html>

第一个java程序：源代码和字节码

**javac** 后面跟着的是java文件的文件名，例如 HelloWorld.java。该命令用于将 java 源文件编译为 class 字节码文件，如： **javac HelloWorld.java**。

运行javac命令后，如果成功编译没有错误的话，会出现一个HelloWorld.class的文件。

**java** 后面跟着的是java文件中的类名,例如 HelloWorld 就是类名，如: java HelloWorld。

**注意**：java命令后面不要加.class。

Java分为三个体系：j2se j2ee和j2me

* JavaSE（J2SE）（Java2 Platform Standard Edition，java平台标准版）

标准版面向普通的用户

* JavaEE(J2EE)(Java 2 Platform,Enterprise Edition，java平台企业版)

面向企业

* JavaME(J2ME)(Java 2 Platform Micro Edition，java平台微型版)。

做嵌入式可开发

Java语言是体系结构中立的：编译为字节码格式（后缀为class的文件）

Java编译器是用Java实现的，Java的运行环境是用ANSI C实现的。

JRE—java运行时环境（[Java Runtime Environment](http://www.baidu.com/link?url=vQRYj3IljdhCjFdIiSrebJUiCvDl-baIO-_H69Un2qoC1hpy9AyMINrfWj4LMFGZfGF1X0s4vY3MedCaAWlyJ_)）

一般下载时jdk自带了jre

Java可以看做一系列对象的集合

Java中一切皆对象，OO

Java大小写敏感，类名大写字母开头，所有的方法名首字母小写；源文件名必须和公共类名相同

主方法入口：所有的Java 程序由public static void main(String []args)（格式是固定的）方法开始执行

Jvm在java程序运行时自动调用该方法

Java修饰符：

1、访问控制修饰符：default（默认就是包的） public protected private

2、非访问控制符：final abstract strictfp

Java枚举：使用枚举主要为了减少程序中的bug，java的枚举是一种特殊的类

Java的继承关键字：extends,与php相同

Final关键字：表示一个变量在初始化后就不在改变了

Finaly:try—catch—finally的分支语句，为了程序的健壮性和完整性。无论有没有异常发生都执行代码。

Implements关键字：继承接口，与php的相同。

Package：一系列相关类组成一个包。

Super关键字：表示基类，c#中是base,php中是parent

**注释：**

与c#、php相同

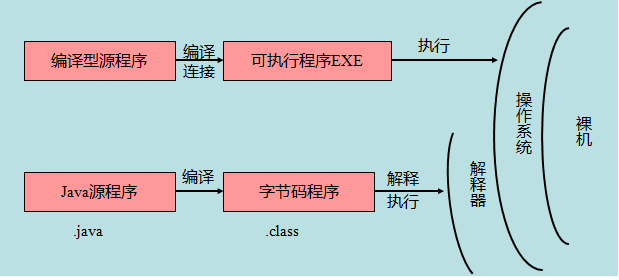
单行注释：//

多行注释:/\* \*/

Java中的父类称为**超类**。

接口：在继承中扮演着很重要的角色，作为对象之间的通信协议。其实接口就是定义了一种规范。

Java源程序与编译型运行的区别：



类可以看做java中的模板。成员变量：对象可以访问，类变量：只能由类来访问，对象也可以访问，因为存在对象则类必存在。

构造方法：一个类必须有一个构造方法，否则java编译器会提供一个默认的构造方法。构造函数必须与类同名，且没有返回类型（实际有返回类类型，就是当前类的实例）；如果提供了构造方法，则编译器不会无惨构造方法

源文件声明规则：

1. 一个源文件只能有一个public类
2. 一个源文件可以有多个非public类
3. 源文件的名称需要和public的类相同
4. 如果一个类定义在某个包内，那么package语句应该在源文件的首行
5. Import语句应该放在package和类定义之间

Java的特殊的类：匿名类（一般只是用一次的类定义为匿名类）和内部类

Java包：包主要用来对类和接口进行分类

Import语句：如果给出一个完整的限定名，包括包名、类名，那么java编译器就可以很容易的定位到源代码或者类。

可以在javac后面继续添加文件来编译具有关系的文件

Javac Test.java Employee.java

当创建变量的时候需要到内存中申请空间，内存管理系统根据变量的类型来为变量分配空间。

Java的两大数据类型：

1. 内置数据类型—值类型
2. 引用数据类型—引用类型

内置数据类型：

byte -128 ~ 127默认值为0

short int long float double Boolean Char

引用类型：对象和数组都是引用类型（同c#），引用类型的默认值都是null

一个引用变量可以用来引用任何与之兼容的类型（如父类变量引用子类对象）

Java常量修饰符：final，与c#中不同（c#中为const），一般常量大写

与c#一样，字符使用单引号’’，字符串使用双引号””

Char c=’a’

String str=”a string”;

数据转换的规则：

1. 不能将boolean类型进行类型转换
2. 不能把对象类型转换为不相关的对象
3. 在把容量大的类型转换为容量小的类型的时必须使用强制类型转换；即超类转子类需要前置转换
4. 转换过程中可能导致溢出或损失经度（大精度转小精度）

自动类型转换：

Char可以和int类型自动转换

Java变量类型：

成员变量、局部变量、类变量

**局部变量：方法体中、方法的参数和语句块**

局部变量在执行完后自动销毁

访问修饰符不能修饰局部变量

局部变量只在声明它的方法构造方法和语句块中可见

局部变量实在栈上分配的

局部变量没有默认值（类没有默认值），所以局部变量被申明后，必须进行初始化才可以使用

**实例变量（成员变量）:**

实例变量声明在一个类中，在方法构造方法和语句块之外

当一个对象实例化后，每一个实例变量就跟着被确定

实例变量在对象创建的时候创建，在对象被销毁的时候销毁

实例变量可以声明在使用前或者使用后。

实例变量具有默认值，数值型变量的默认值为0，布尔型变量的默认值为false,引用类型的变量的默认值为null------与c#相同

类变量—静态变量—关键字static

无论一个类创建了多少个对象，类只拥有变量的一份拷贝

静态变量除了声明为常量外很少使用—public static final TEST;

静态变量存储在静态存储区内，在程序开始时创建，在程序结束时销毁

静态变量的默认值和实例变量的默认值相同，值类型为0,布尔类型默认值为false，引用类型的默认值为null.

类变量被声明为public static final类型时一般要大写

访问控制修饰符：

默认的为default，不使用这个关键字



接口里的变量都隐式的声明为public static final,而接口里的方法都默认声明为public

接口的成员变量不能申明为protected

访问控制和继承：

1. 父类中声明为public的方法在子类中也必须为public
2. 父类中声明为protected的方法在子类中要么声明为protected要么声明为public，不能声明为private----防止它的子类在继承的时候不能访问基类的这个方法

3、继承后的访问控制符只能更加的宽松

Synchronized和volatile修饰符，主要用于线程的编程。

Final修饰符：

Final修饰的变量能被显示的初始化并且只能被初始化一次，被声明为final的对象的引用不能指向不同的对象，但是final对象里的数据可被改变，也就是说final对象的值可以改变。

Final修饰符经常和static搭配起来声明常量

类中的final修饰的方法可以被子类继承但是不能被子类修改

Final类不能被继承

**Synchronized修饰符**

Synchronized关键字声明的方法同一时间只能被一个线程访问。Synchronized修饰符可以应用于四个访问修饰符。

**Transient修饰符**

序列化的对象包含被transient修饰的实例变量时，java虚拟机(JVM)跳过该特定的变量。

该修饰符包含在定义变量的语句中，用来预处理类和变量的数据类型。

**Volatile修饰符**

Volatile 修饰的成员变量在每次被线程访问时，都强制从共享内存中重新读取该成员变量的值。而且，当成员变量发生变化时，会强制线程将变化值回写到共享内存。这样在任何时刻，两个不同的线程总是看到某个成员变量的同一个值。

一个volatile对象引用可能是null。

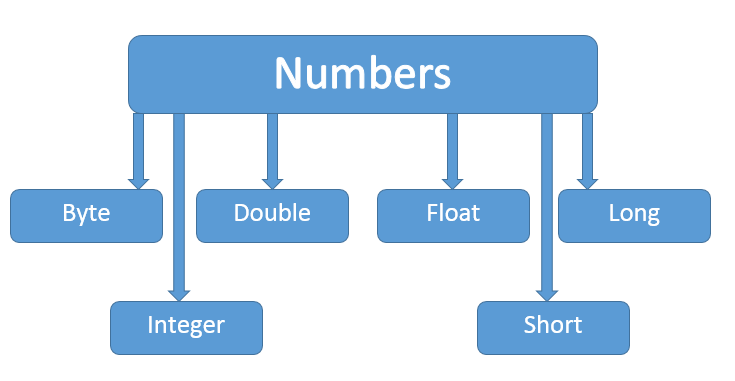
Java的运算符：

Instanceof运算符：该运算用来操作对象实例，检查一个对象是否是一个特定的类型(类型或者接口类型)

**Testt**=**new**Test();

**System**.***out***.println(t**instanceofTest**);

Java的Number类：



|  |  |
| --- | --- |
| 类型 | 字节数 |
| Byte | 1 |
| Short | 2 |
| Char | 2 |
| Int | 4 |
| Long | 8 |
| float | 4 |
| Double | 8 |

自动装箱和拆箱

这种由编译器特别支持的包装称为装箱，所以当内置数据类型被当作对象使用的时候，编译器会把内置类型装箱为包装类。相似的，编译器也可以把一个对象拆箱为内置基本数据类型。

Public static void main(Stringargs[]){

Integer x =5;

Integer y=x; //这里做了拆箱，以便能执行赋值操作

Y=10; //做了拆箱，并不会导致x的值改变

System.out.println(x);//输出5

}

**Java Character类**

在某些情况下，Java编译器会自动创建一个Character对象。

例如，将一个char类型的参数传递给需要一个Character类型参数的方法时，那么编译器会自动地将char类型参数转换为Character对象。这种特征称为装箱，反过来称为拆箱。

**Java String类**

String类是不可改变的，底层实现是char数组，所以你一旦创建了String对象，那它的值就无法改变了。如果需要对字符串做很多修改，那么应该选择使用[StringBuffer（线程安全但性能低） & StringBuilder（非线程安全但性能高） 类](http://www.runoob.com/java/java-stringbuffer.html)。

格式化字符串：%d整形变量，%f浮点型变量，%s字符串变量类似c#中格式化字符串中的占位符 {0}

**Java StringBuffer和StringBuilder类**

String是不能改变的，而StringBuilder和StringBuffer类的对象能够被多次访问，并且不产生新的为使用对象。StringBuilder和StringBuffer之间最大的不同是StringBuilder的方法不是线程安全的（不能同步访问），StringBuilder相较于StringBuffer具有速度优势，因此多数情况下建议使用StringBuilder，然而在应用程序要求线程安全的情况下，则必须使用StringBuffer类。

数组的长度是固定的，可以使用for循环，也可以使用foreach循环（即增强型for循环）

多维数组：int a[][]

Array类的静态方法：



Java日期时间：

两个构造函数：Date()当前时间来初始化和 Date(long millisec) –代表从1970年1月1日起的毫秒数



Java日期的比较：

1. getTime()方法比较两个值
2. 使用方法before