Java学习day06

1. 面向对象static关键字。

静态修饰符Static是一个修饰符，用来修饰成员变量，成员函数。

这种修饰方法可以使调用方式多一种，除了本来有的通过对象的调用方式，还可以直接通过类名来调用。 调用方法：类名.成员变量。

一般，静态变量是用来共享的，如果名字之类的变量不适合用来共享，比如：国籍，血型，某些共性的东西适合用来共享。

2.Static的特点：

1. 随着类的加载而加载；也就是说，静态会随着类的消失而消失，生命周期最长。
2. 优先于对象存在。 明确一点，静态是存在的，对象是后存在的。
3. 被所有对象共享。
4. 可以直接被类名调用

实例变量和类变量的区别

1. 存放位置。

类变量随着类的加载存放与方法区；实例变量随着对象的建立存放在栈内存中。

1. 生命周期：

类变量的生命周期最长，随着类的消失而消失

实例变量(也就是对象的创建)生命周期随着对象的消失而消失

静态使用注意事项：

1. 静态方法只能访问静态成员；非静态方法既可以访问静态也可以访问非静态。
2. 静态方法中不可以定义this，super关键字。因为静态优先于对象存在。所以静态方法中不可以出现this。（因为要用this你必须先通过创建相应的对象才能使用）。
3. 主函数是静态的。

静态有利有弊：

利处：对对象的共享数据进行单独空间的存储，可以直接被类名调用，没有必要每一个对象中都存在一份。

弊处：生命周期过长。

访问出现局限性(静态虽好，只能访问静态成员)

1. 主函数的定义：

Public static void main(String[] args)

Public: 代表函数的访问权限是最大的；

Static：静态变量，表示函数随着类的加载就存在了

Void：主函数的返回类型为空

Main：不是关键字，是一个特殊的关键词，被jvm所识别。

(String[] args):函数的参数，参数的类型是一个数组，该数组中的元素是字符串。字符串类型的数组主函数的固定格式：jvm识别。

1. 关于静态方面的知识：

基本概念的理解：类被实例化意思是，当用类创建对象称为实例化的过程，是一个从抽象到具体的过程。

静态成员与非静态成员的理解：

1. 静态成员先于类的声明存在于内存，调用的方法为：类.静态成员来访问此静态成员。
2. 非静态成员它是通过对类的实例化后才存在于内存，也就是随着对象的创建而创建，有生命周期，它是new之后用实例化的名称调用。

为什么使用静态？ 这个问题要从两个方面弄下手。

因为静态修饰的内容有成员变量和函数。

什么时候定义静态变量（类变量）？

当对象的出现而来共享数据时，该数据被静态所修饰。

对象中的特有数据要定义成非静态存在于堆内存中。

什么时候定义静态函数？

当功能内部没有访问到静态数据（对象的特有数据），那么该函数就可以被定义为静态。

1. 静态的应用-工具类
2. 对获取最大值最小值做了优化，并且对排序算法做了复习与优化。
3. 面向对象（帮助文档的制作javadoc）

因为类中定义了多少方法，对方并不清楚，所以类中要写说明书。 开始制作程序说明书，java的说明书通过文档注释来完成。用到的符号为/\*\* \*/

生成的方法：javadoc -d myhelp -author -version ArrayTool.java

记住@param arr +中文说明 一定要注意其中的空格，否则会造成识别错误。

只有两种格式的函数才会被javadoc所提取，一种是public，另一种是protected；而private不会被提取。

关于构造函数：一个类中默认会有一个空参数的构造函数，这个默认的构造函数的权限和所属类一致。

如果类没有被public修饰，那么默认的构造函数也没有被public修饰；

反之，如果类被public修饰，那么默认的构造函数也被public修饰；

总之，默认构造函数的权限随着类的变化而变化。

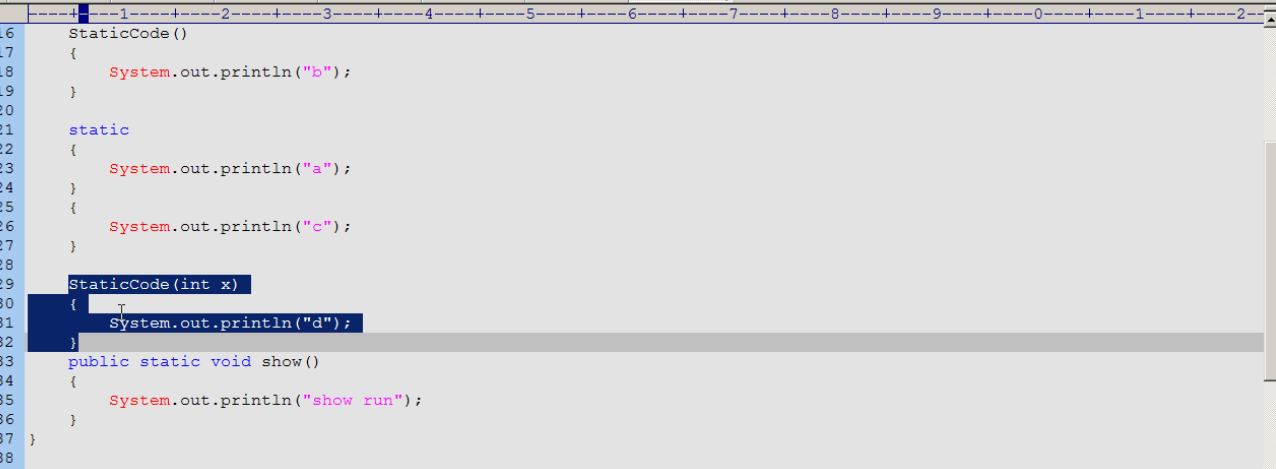
1. 面向对象（静态代码块）

格式为：

Static

{

}



Static

{

}

{

}

StaticCode(int x)

{

}

按照代码的位置顺序分别是：构造代码块、静态代码块、构造代码块与构造函数

执行的顺序分别是静态代码块对类进行初始化；构造代码块给对象初始化的；构造函数给对应对象初始化的。

1. 面向对象，对象初始化过程

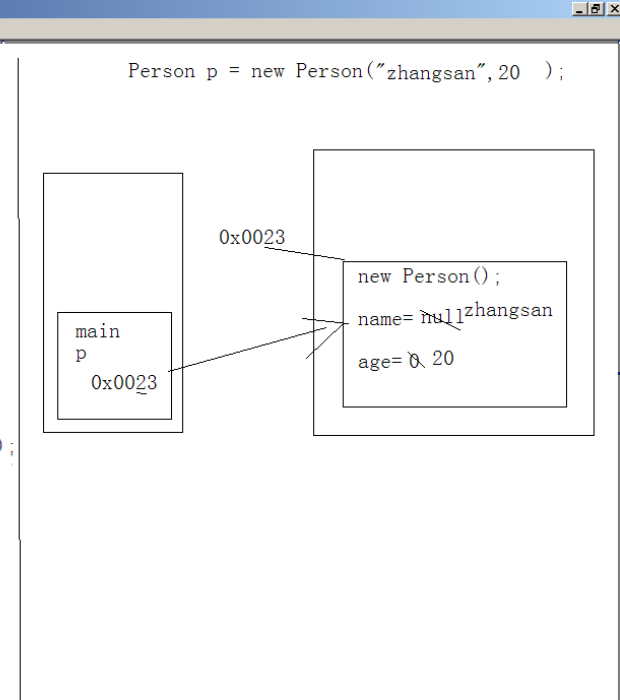
<https://blog.csdn.net/jianghao233/article/details/82777789>

首先先来一个知识回顾：java在运行程序时先要开辟一个内存空间，其中java虚拟机也是。

Java内存可以划分为5片：

1. 方法区； 2.本地方法区； 3.寄存器； 4.堆内存区； 5.栈内存区

简单的说一下堆和栈：一般栈内存区是用来存放局部变量，而堆内存区是存放对象与数组的，凡是new出来的东西都要放在堆内存区。所以栈内存区具有临时性。



开始进行分析：

/\* Person p = new Person("zhangsan", 20); 这句话都执行了什么？

1. 因为有new这个关键字，所以通过jvm这个虚拟机找到Person.class 这一个文件

2. 静态代码块对类的初始化，如果有的话是对Person.class这个类进行初始化

3. 在堆内存中开辟空间，分配内存地址。（我的理解是创建了对象）

4. 在堆内存中建立对象的特有属性，默认的初始化代码的执行

5. 对属性的显示的代码块的初始化。

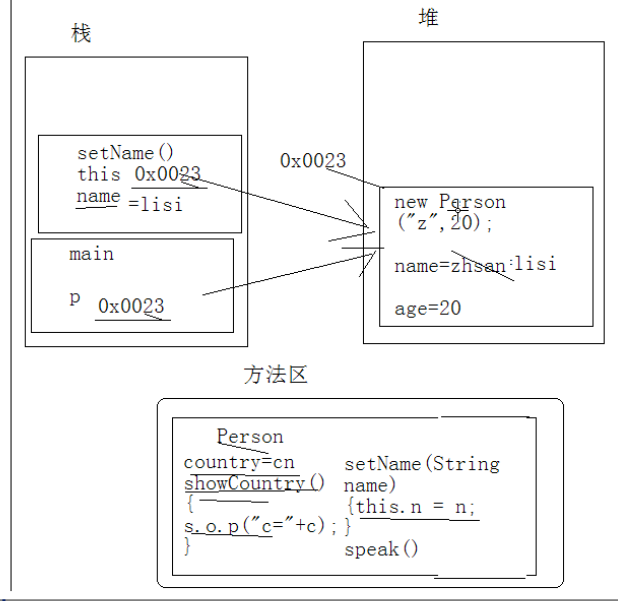
6. 对对象进行构造代码块的初始化

7. 对对象的构造函数的初始化。

8. 将内存地址赋给堆内存中的p变量。

\*/

9.对象调用成员的过程：



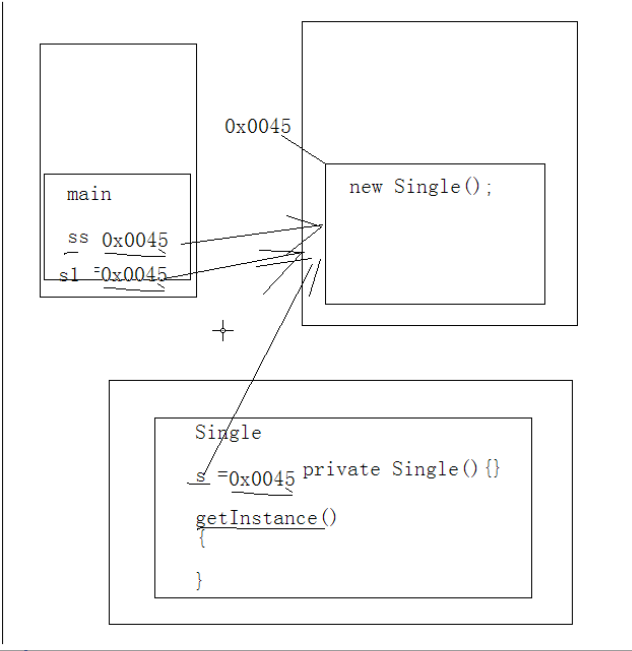
首先，略去初始化步骤。 运行程序的时候main方法就进入栈中，对象进入堆内存中，其他的方法进入方法区；

接着，在调用类名setName之后，这个方法与对应的局部变量name进入栈内存。

然后this和对象p同时指向堆内存空间，所以可以说，this可以理解为p。

10.面向对象，单例设计模式：在82讲，解决一个类在内存中只存在一个对象。

对应的内存图：



完成了单例设计模式的功能：只有在本类中有一个对象，其他对象不能建立，并且该对象能被提供的方法访问。

11.面向对象（单例设计模式方法2）： 设计模式可以分为饿汉式与懒汉式；

饿汉式指的是先初始化对象，懒汉式初始化对象是放在方法中的。一般懒汉式初始化不常用，但是面试题多考

在懒汉式中可以加入synchronized关键字 ， 这是一个同步的关键字，只有在一个线程执行完毕后，另一个线程通过敲门的方式，询问当前是否为空，如果是空的话执行。但是存在效率较低的问题

这是一个双重判断的synchronized语句 ，能够提高效率 现在还不是特别理解 涉及多线程的问题