学习 JAVA 第五天 ---- 初始化与清理

作者：DogUnderSunset
原文链接：https://ld246.com/article/1490421569787
来源网站：链滴

关于想法

最近可能因为小宝宝要出生，总是有点心神不宁，看书的时候效率也没那么高了，敲这行字的时候，婆把手伸到衣领子里面.

又定了个决定:

下个月开始（为什么下个月而不是现在，有原因的），晚上有时间不打游戏，去健身房搞事儿.

初始化与清理

讲实话，这章值得记录的真的很多，废话不多说，上干货:

构造器

定义：构造器是个什么玩意呢？构造器是为了做到在对象实例化的时候就被初始化的效果，如下

```
package javaLearningDemo;

class DWName{

    String name;

    {

        name = "whj";

    }

    DWName(){

        System.out.println("在DW中是static，所以只会被调用一次，所以你只能看见我一次,默认构器初始化名字为:"+name);

    }

    DWName(String name){

        this.name = name;

        System.out.println("在DW中是static，所以只会被调用一次，所以你只能看见我一次,默认构器初始化名字为:"+this.name);

    }

}

class DW{

    DWName firstName = new DWName("hahaha");
```

```
    int height;
    DW(DWName name,int height){
        this.name = name;
        System.out.println("定义了name和height"+name.name+height);
    }
    DW(){
        System.out.println("如果没有构造方法，那么默认会有一个无具体操作的无传参默认构造器，是一旦如果有带传参的构造器，如果没有显式默认构造器，那么实例化必须带参数");
    }
    static DWName name = new DWName();
}
public class staticInitDemo {
    public static void main(String[] args) {
        System.out.println(secondDW.name.name);
        System.out.println("你第一个看见的不是我，一定是DW，因为在一个类中，实例初始化优先最高，并且，静态的比非静态的优先级高");
    }
    // DW firstDW = new DW(new DWName(),12);
    static DW secondDW = new DW(new DWName("lalala"),14);
    static DW thirdDW = new DW();
}
```

输出：

```
在DW中是static，所以只会被调用一次，所以你只能看见我一次,默认构造器初始化名字为:whj
在DW中是static，所以只会被调用一次，所以你只能看见我一次,默认构造器初始化名字为:lalala
在DW中是static，所以只会被调用一次，所以你只能看见我一次,默认构造器初始化名字为:hahaha
定义了name和heightlalala14
在DW中是static，所以只会被调用一次，所以你只能看见我一次,默认构造器初始化名字为:hahaha
```

如果没有构造方法，那么默认会有一个无具体操作的无传参默认构造器，但是一旦如果有带传参的构器，如果没有显式默认构造器，那么实例化必须带参数

lalala

你第一个看见的不是我，一定是DW，因为在一个类中，实例初始化优先级最高，并且，静态的比非态的优先级高

从构造器角度来看,有如下注意点:

1. 如果没有构造器，那么会有一个默认构造器(无传参),如果有构造器，那么如果没有显式的默认构器的话，那么初始化的时候必须要带传参，否则错误

2. 构造器可重载

3. 构造器仅在对象实例化的时候会调用，如下

```
class DW{
    DW(){
    print("well");
    }
    static void do(){
    print("Just do it");
    }
}
void main(){
    DW.do();
}
//仅仅打印Just do it
```

重载

重载不多说，大家也都知道的，所以仅做如下注意点记录:

1. 重载的作用是为了实现同一个方法名处理不同的传参类型

2. 重载的区分方法是传参类型差异

3. 切记重载不可以由返回值区分

4. 重载传参可以为可变参数，但是最好不要这样，因为可变参数支持无参，那么若有三个重载传参是单纯的可变参数，那么不传参的调用方式就不知道调用的是哪个方法了.

this指针

首先this指针的意思是调用方法的对象.

需要注意如下几点:

1. 想要在构造其中调用构造器(代码复用),可以用this(传参)

2. 上述方法注意:在构造器中调用构造器的时候，只能存在一次,并且调用必须在这个构造器的顶端，则报错.

3. 除了在类的构造器中使用this调用其它构造器。否在则一个类中的其它地方不可以调用构造器

初始化

成员初始化

首先，什么叫做初始化，初始化是指在创建一个类的实例的时候，不仅要有这个类的空间，还得存在个类的"空白状态"(无构造函数的情况).

那么如果一个对象没有初始化的话，他被声明后其实只是一个null

他的注意点如下:

指定初始化

即在定义成员变量的时候直接赋值

构造器初始化(** 重点 **)

初始化顺序:

1. 按照变量定义的先后执行初始化

2. 任何方法或构造器中，都是初始化方法先执行，详情见上面的示例代码

静态方法的初始化

1. 静态对象A属于某个对象B的时候，仅当B初始化的时候，A才会被初始化，并且切记，只会被初化一次，见上面的示例

2. 在对象中，对象初始化的顺序遵循先静态后非静态

显式的静态初始化

```
public class DW{
  static int i;
  static {
```

```
  i = 47;
  }
}
```