类变量与类方法

一、引出 类变量(静态变量)

```
package com.Static;

public class ChildGame {
    public static void main(String[] args) {
        int count = 0;//统计人数
        Child dsaf = new Child("dsaf");
        Child dghh = new Child("dghh");
        dsaf.join();
        count++;
        dghh.join();
        count++;
        System.out.println("现在有" + count + "个人");
/*
* 1.count是一个独立的对象,Java是一给面向对象的语言
* 2.以后访问count很麻烦,没有使用到OOP
* 3.因此，引出 类变量(静态变量)
* */
    }
}

class Child {
    private String name;

    public Child(String name) {
        this.name = name;
    }

    public void join() {
        System.out.println(name + "加入了我们");
    }
}
```

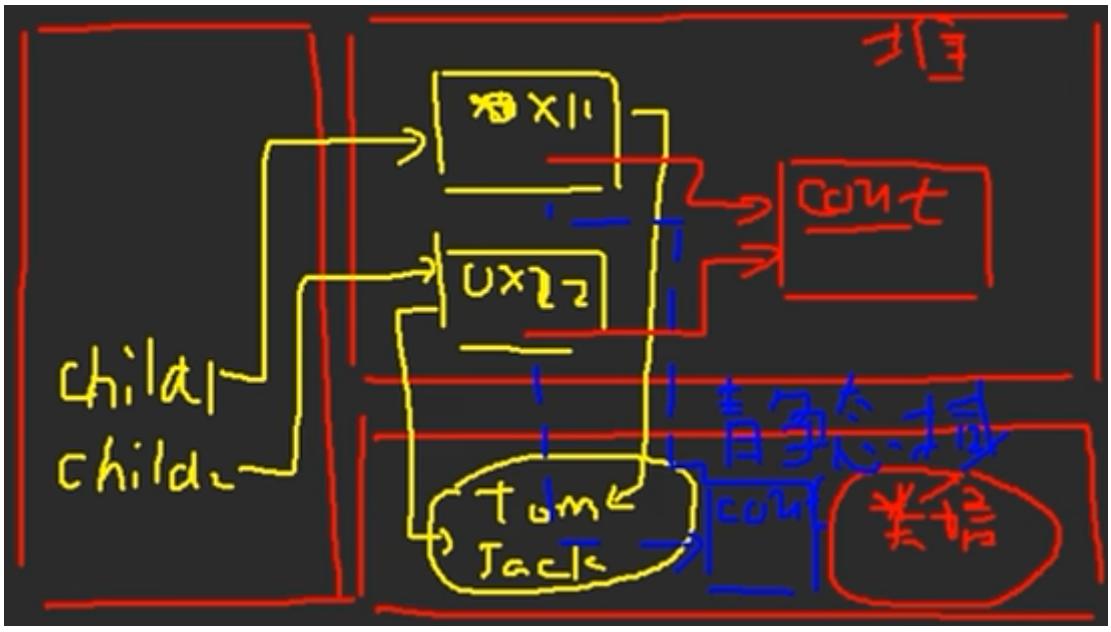
二、什么是类变量?

- **能被多个对象共享的 数据空间 为 类变量.**

```
package com.Static;

public class ChildGame {
    public static void main(String[] args) {
//        int count = 0;//统计人数
        Child dsaf = new Child("dsaf");
        Child dghh = new Child("dghh");
        dsaf.join();
//        count++;
        dsaf.count++;
        dghh.join();
```

```
//          count++;
        dsaf.count++;
//         类变量可以通过类名来访问
        System.out.println("现在有" + Child.count + "个人");
//          以下代码输出是一样的
        System.out.println(dsaf.count);//2
        System.out.println(dghh.count);//2
/*
* 1.count是一个独立的对象,Java是一给面向对象的语言
* 2.以后访问count很麻烦,没有使用到OOP
* 3.因此，引出 类变量(静态变量)
* */
    }
}

class Child {
    private String name;
    //定义一个count变量,此为类变量(或称静态变量) static
    //该变量最大的特点是会被Child类的所有对象实例共享
    public static int count = 0;
    public Child(String name) {
        this.name = name;
    }

    public void join() {
        System.out.println(name + "加入了我们");
    }
}
```

三、静态变量的内存布局

- 

  。

1. **静态变量被对象共享.**
2. **不影响对静态变量的使用.**
3. **static静态变量在类加载的时候就完成了.**
4. **static变量 是与 同一个类 所有对象 共享**
5. **任何一个 该类的对象 去 访问该类同一个static时，都会返回同一个值.**
6. **同样任何一个 该类的对象去 修改同一个static时，也同样是修改同一个值.**

四、静态变量语法

- **定义**

```
//访问修饰符 static 数据类型 变量名;(推荐)
//static 访问修饰符 数据类型 变量名;
```

- 访问类变量

```
//类名.类变量名  (推荐)
//或者    对象名.类变量名
```

1.例子:

```
package com.Static;

public class VisitStatic {
    public static void main(String[] args) {
        //类名.类变量名
        //说明：类变量是随着类的加载而创建的，所以即使没有创建对象实例也可以访问
        System.out.println(A.name);
        A a = new A();
        System.out.println(a.name);
    }
}
class A{
    //类变量 强调:类变量依然遵守访问权限
    public static String name = "赚钱钱不寒掺";
}
```

2. 例子:

```
package com.Static;

public class StaticMethod {
    public static void main(String[] args) {
        Stu.payFee(100);
        Stu.showFee();//100
        Stu.payFee(200);
```

```
        Stu.showFee();//300因为在payFee()已经累计了
        //对象调用静态方法
        Stu s1 = new Stu("兔八哥");
        Stu s2 = new Stu("三娃");
        s1.payFee(100);
        s1.showFee();
        s2.payFee(100);
        s2.showFee();
    }
}
class Stu{
    //普通成员
    private String name;
    //定义静态变量 来累计学费
    private static double fee = 0;

    public Stu(String name) {
        this.name = name;
    }
    public static void payFee(double fee){
        Stu.fee += fee;//fee为静态变量，可以直接写成 （类变量.静态变量名）
        //不能用this.fee因为fee现在已经是该类对象共有，而非某个该类的对象了
    }

    public static void showFee(){
        System.out.println("收到的学费累计："+Stu.fee);
    }
}
```

五、静态方法使用场景

1. **如果我们希望不创建实例，也可以调用某个方法(即当作工具来使用)，这时把方法做成静态方法时非常合适**

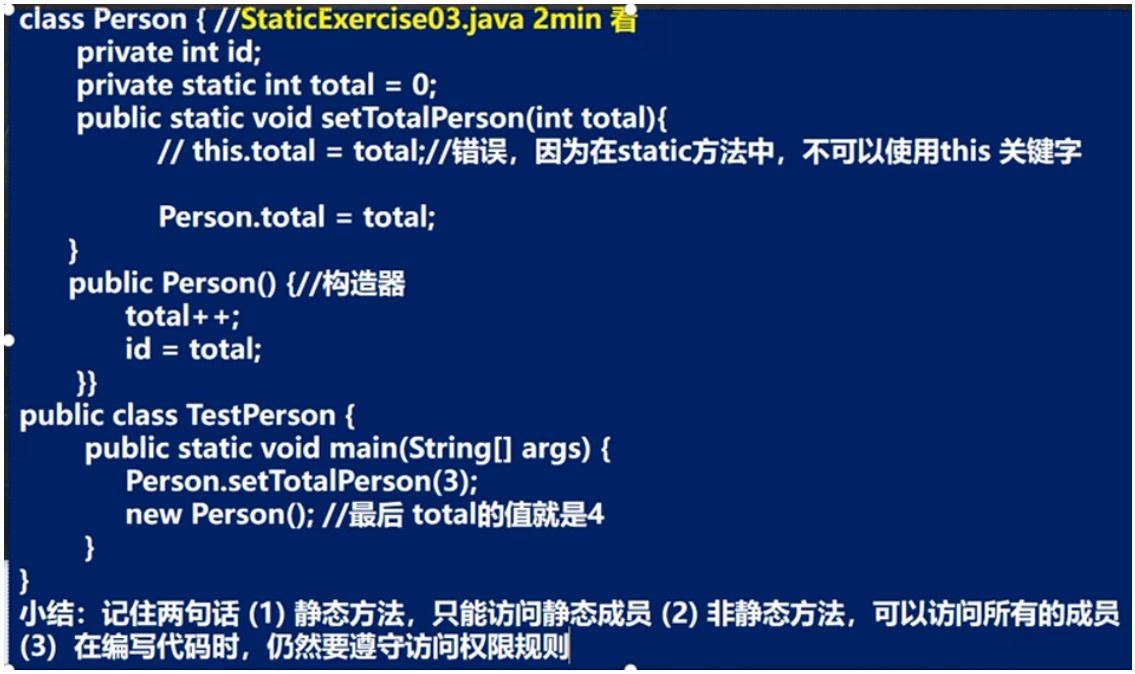
六、类变量 和 类方法 的使用方法

1. **类方法 和 类变量 都是 随着类加载而加载，将结构信息存储在方法区，类方法中无this参数 和 super().**
2. **普通方法中隐含着this参数**
3. **类方法可以通过 类名 调用 ， 也可以通过 对象名 调用**
4. **普通方法 和 对象 有关，需要通过 对象名 调用**
5. **口诀：非静态方法 可以访问 静态成员 和 非静态成员.**
6. **静态方法中，只能访问 静态变量 或 静态方法.**
7. **口诀:静态方法只能访问静态成员.**

七、练习

1.输出什么?

```
package com.Static.Excer;

public class Test {
    public static void main(String[] args) {
        new Test().count();//9
        new Test().count();//10
        System.out.println(Test.count);//有调用了一次11
    }
    static int count = 9;
    public void count(){//非静态变量可以访问所有成员
        System.out.println("count = " + (count++));
    }
}
```

2. 

。