



情境2：类与对象的使用

Java面向对象程序设计

1.对象的使用

- (1) 对象
- (2) 对象的使用
- (3) 实例

对象

对象的定义

<类名><对象名> = **new**<类名> (参数1, 参数2, ...);

或者:

<类名> <对象名>;

<对象名> = **new** <类名> (参数1, 参数2, ...);

创建对象实际执行了三个操作：说明对象、实例化对象和初始化对象。

■ **说明对象**：就是给对象命名，也称定义一个实例变量。

一般形式为：<类名> <对象名>;

■ **实例化对象**：给对象分配存储空间，用来保存对象的数据和代码。**new**运算符用来实现对象的实例化。

<对象名> = **new** <类名> (参数1, 参数2, ...);



对象的使用

- 初始化对象通过调用该对象所在类的构造方法来实现的。
- 引用对象的方法一般形式：
<对象名>. <方法名> ([<参数1>,<参数1>,...]);
例：程序**TwoArea.java**，是引用对象的例子。

执行结果：



```
D:\java>java TwoArea
两个矩形面积差=60
r1的原点为(4,6)
r2的原点为(4,12)
```



/*定义一个类Point*/

class Point

{

public int x; //定义成员变量

public int y; //定义成员变量

public Point() //Point构造方法，不带参数

{

this.x=0; //引用当前x

this.y=0; //引用当前y

}

**public Point(int x,int y) //Point构造方法，
带两个参数**

{

this.x=x; //引用当前x

this.y=y; //引用当前y

}

}





/*定义一个类Graph*/

class Graph

{

public int width=0; //定义成员变量

public int height=0; //定义成员变量

public Point origin; //定义成员变量

public Graph() //Graph构造方法1, 不带参数

{

origin=new Point(0,0);

}

public Graph(Point p) //Graph构造方法2, 带一个参数

{

origin=p;

}

public Graph(Point p,int w,int h) //Graph构造方法3, 带三个参数

{

origin=p;

width=w;

height=h;

}





```
public void move1(int x,int y) //成员方法move1
{
    origin.x=x;
    origin.y=y;
}
public void move2(int a,int b) //成员方法move2
{
    origin.x+=a;
    origin.y+=b;
}
public int area() //成员方法area
{
    return width * height;
}
}
```



```
public class TwoArea /*定义主程序类*/
{
    public static void main(String args[])
    {
        int x=3,y=4;
        int w=4,h=5;
        Point p1=new Point(x,y); //将原点p1定在(3, 4)坐标点
        Point p2=new Point(6,8); //将原点p2定在(6, 8) 坐标点
        Graph r1=new Graph(p1,w,h);
        Graph r2=new Graph(p2,8,10);
        r1.move1(4,6); //把矩形r1的原点移到 (4, 6)
        r2.move2(-2,4); //把矩形r2的原点x坐标减2, y坐标加4
        System.out.println("两个矩形面积差="+r2.area() - r1.area());
        /*调用成员方法计算矩形r2与r1面积差*/
        System.out.print("r1的原点为("+r1.origin.x+",");
        System.out.println(r1.origin.y+"");
        System.out.print("r2的原点为("+r2.origin.x+",");
        System.out.println(r2.origin.y+"");
    }
}
```





Thank you!

