概述
举例说明
使用场景
代码
圆形类
获取圆形的工厂
测试类
代码简述

概述

享元模式（Flyweight Pattern）主要用于减少创建对象的数量，以减少内存占用和提高性能

举例说明

颜色有很多种，如果需要具体颜色的圆形，就去创建一个这样颜色的圆形，就会消耗内存比较多。

但是用一个池子，装起来具体颜色的圆形，如果这个颜色的存在，就直接取。不存在才去创建并且放入池子。就会节约内存

使用场景

1、系统有大量相似对象
2、需要缓冲池的场景

代码

圆形类

```java
/**
 * 圆形类
 */
public class Circle{
   private String color;

   public Circle(String color) {
      this.color = color;
   }

    public void draw() {
       System.out.println("Circle: Draw() [Color : " + color);
    }
}
```

获取圆形的工厂

```
/**
 * 图形工厂
 */
public class CircleFactory {

   private static final HashMap<String, Circle> circleMap = new HashMap<String, Circle>();

   public static Circle getCircle(String color) {
      Circle circle = circleMap.get(color);
       if (circle == null) {
          circle = new Circle(color);
          circleMap.put(color, circle);
          System.out.println("Creating circle of color : " + color);
       }
       return circle;
    }
}
```

测试类

```
public class Main {

   private static final String colors[] = { "Red", "Green", "Blue", "White","Black" };

   public static void main(String[] args) {

      for (int i = 0; i < 10; ++i) {
         Circle circle = CircleFactory.getCircle(getRandomColor());
         circle.draw();
       }
    }

    private static String getRandomColor() {
       return colors[(int) (Math.random() * colors.length)];
    }
}
```

代码简述

说明： 当需要白色的圆形，会去看是否存在一个白色的圆形，如果不存在就创建一个，否则直接返回一个