享元模式

Flyweight

郭嘉

模式定义：

运用共享技术有效地支持大量细粒度的对象

```java
package com.tuling.designpattern.flyweight;

import java.util.Map;
import java.util.concurrent.ConcurrentHashMap;

/**
 * @author 腾讯课堂-图灵学院 郭嘉
 * @Slogan 致敬大师，致敬未来的你
 */
 public class FlyWeightTest {
  public static void main(String[] args) {
  TreeNode treeNode1=new TreeNode( 3,4, TreeFactory.getTree("xxx","xxxxxxxxx" ));
  TreeNode treeNode2=new TreeNode( 5,4, TreeFactory.getTree("xxx","xxxxxxxxx" ));
  TreeNode treeNode3=new TreeNode( 13,24, TreeFactory.getTree("yyy","xxxxxxxxx" ));
  TreeNode treeNode4=new TreeNode( 15,24, TreeFactory.getTree("yyy","xxxxxxxxx" ));
 }
}

class TreeNode{

 private int x;
 private int y;
 private Tree tree;

 public TreeNode(int x, int y, Tree tree) {
 this.x=x;
 this.y=y;
 this.tree=tree;
 }

 public int getX() {
 return x;
 }

 public void setX(int x) {
 this.x=x;
 }

 public int getY() {
 return y;
 }

 public void setY(int y) {
 this.y=y;
 }

 public Tree getTree() {
 return tree;
 }

 public void setTree(Tree tree) {
 this.tree=tree;
 }
}

class TreeFactory{
 private static Map<String,Tree> map=new ConcurrentHashMap<>( );

 public static Tree getTree(String name,String data){

 if (map.containsKey( name )){
 return map.get( name );
 }
 Tree tree=new Tree( name,data );
 map.put( name,tree );
 return tree;
 }
}

//
class Tree{
 private final String name;
 private final String data;

 public Tree(String name, String data) {
 System.out.println(" name: "+name +" tree created. ");
 this.name=name;
 this.data=data;
 }
 public String getName() {
 return name;
 }
 public String getData() {
 return data;
 }

}

```

优点：

如果系统有大量类似的对象，可以节省大量的内存及CPU资源

JDK源码中的应用

```
String,Integer,Long...
com.sun.org.apache.bcel.internal.generic.InstructionConstants
```