853769620

flyweight模式：存储固定数量的对象（Map），调用方使用同一个对象【频繁读，很少写】

场景

（1）我们现在要缓存一些数据，比如说权限，根据权限的ID作为key，权限数据自己本身是value

（2）但是我们不需要使用太复杂的ehcache框架之类的去使用，特别特别的简单的

（3）我们就想基于自己的Map去实现一套东西

public **interface Flyweight** {

   void operate();

}

public class **ConcreteFlyweight implements Flyweight** {

   public void operate(String state) {

   }

}

public class **FlyweightFactory** {

   private Map<String, Flyweight> map =

       new HashMap<String, Flyweight>();

   public Flyweight getFlyweight(String key) {

       Flyweight flyweight = map.get(key);

       if(flyweight == null) {

           flyweight = new ConcreteFlyweight();

           map.put(key, flyweight);

      }

       return flyweight;

  }

}

享元模式，从最基本的角度来说**，其实就是维护固定数量的对象实例，不同的对象实例，只有一个，然后保存在内部的一个map中。可以算是跟单例模式的一个扩展和结合吧。**

这边我可以给大家说一个享元模式的常见使用场景，其实就是jvm内缓存实现，如果我们不想引入ehcache之类缓存框架，就想简单一点，通过自己实现的一个map来缓存对应的一些数据，比如说每个权限的数据，或者每个员工的数据，在内存里自己维护一个缓存工厂，就是基于享元模式的思想来实现的。

这块我们在电商系统里，可以在权限那块，做一个简单的冷数据jvm缓存方案呢，来给大家实践。