Android设计模式源码解析之享元模式

本文为 Android 设计模式源码解析 中 享元模式 分析

Android系统版本： 4.0

分析者：lvtea0105，分析状态：未完成，校对者：Mr.Simple，校对状态：未开始

1. 模式介绍

模式的定义

```
通过共享有效支持大量的细粒度对象，节省系统中重复创建相同内容对象的性能消耗，进而提高应用程序的性能。
享元模式可分为单纯享元模式和复合享元模式。
```

模式的使用场景

```
面向对象编程在某些情况下会创建大量的细粒度对象，它们的产生，存储，销毁都会造成资源和性能上的损耗，
可能会在程序运行时形成效率瓶颈，在遇到以下情况时，即可考虑使用享元模式：
（1）一个应用程序使用了大量的对象，耗费大量的内存，降低了系统的效率。
（2）这些对象的状态可以分离出内外两部分。
（3）这些对象按照状态分成很多的组，当把删除对象的外部状态时，可以用相对较少的共享对象取代很多组对象。
（4）应用程序不依赖这些对象的身份，即这些对象是不可分辨的。
在一般的开发中享元模式并不常用，其常常应用于系统底层的开发，以便解决系统的性能问题。
```

1. UML类图

2. 模式的简单实现

简单实现的介绍

（1）享元模式如何实现共享

```
将事物的共性共享，同时又保留它的个性。为了做到这点，享元模式中区分了内蕴状态(Internal State)和
    外蕴状态(External State)。内蕴状态就是共性，外蕴状态就是个性了。
内蕴状态存储在享元内部，不会随环境改变而变化，是可以共享的；
外蕴状态是不可以共享的，它随环境的改变而变化，通常外蕴状态是由客户端来保持的（因为环境的变化是由客户端引起的）。
```

单纯享元模式

————所有的享元对象都是可以共享的

```
抽象享元(Flyweight)角色 ：给出一个抽象接口，以规定出所有具体享元角色需要实现的方法，外蕴状态以参数形式传入此方法。
具体享元(ConcreteFlyweight)角色：实现抽象享元角色定义的接口。如果有内蕴状态的话，则必须为内蕴状态提供存储空间。
享元工厂(FlyweightFactory)角色：负责创建和管理享元角色，保证享元对象可以被系统适当地共享。
    当客户端调用一个享元对象的时候，享元工厂角色会检查系统中是否已经有一个符合要求的享元对象。
    如果已经有了，就提供这个已有的享元对象；如果没有，就创建一个合适的享元对象。
客户端角色：维护所有享元对象的引用，同时还需要存储享元对象所对应的外蕴状态。
```

复合享元模式

————将一些单纯享元使用合成模式加以复合，形成复合享元对象。复合享元对象本
身是不能共享的， 但是它们可以分解成能够进行共享的单纯享元对象。

```
抽象享元(Flyweight)角色 ：给出一个抽象接口，以规定出所有具体享元角色需要实现的方法，外蕴状态以参数形式传入此方法。
具体享元(ConcreteFlyweight)角色：实现抽象享元角色定义的接口。如果有内蕴状态的话，则必须为内蕴状态提供存储空间。
复合享元(ConcreteCompositeFlyweight)角色：复合享元角色所代表的对象是不可以共享的，
    但是一个复合享元对象可以分解成能够进行共享的单纯享元对象。
享元工厂(FlyweightFactory)角色：负责创建和管理享元角色，保证享元对象可以被系统适当地共享。
    当客户端调用一个享元对象的时候，享元工厂角色会检查系统中是否已经有一个符合要求的享元对象。
    如果已经有了，就提供这个已有的享元对象；如果没有，就创建一个合适的享元对象。
客户端角色：维护所有享元对象的引用，同时还需要存储享元对象所对应的外蕴状态。
```

实现源码

Android源码中的模式实现

1. 优点与缺点 优点————大幅度地降低内存中对象的数量，节省系统资源的开销

   缺点————1、为了使对象可以共享，享元模式需要将部分状态外部化，使得
   系统的逻辑变得复杂。 2、读取状态外部化的享元对象，影响了系统速度，使运
   行时间有所加长。