枚举和注解

2018年7月13日

12:21

**第30条：用enum代替int常量**

枚举类型是指由一组固定的常量组成合法值的类型。

枚举类型是实例受控的。他们是单例的泛型化，本质上是单元素的枚举。

如果有多个枚举常量同时共享相同的行为，可以考虑策略枚举。

**第31条：用实例域代替序数**

永远不要根据枚举的序数导出与它关联的值，而是要将它保存在一个实例域中。

除非在编写基于枚举的通用数据结构，最好完全避免使用oridinal方法。

**第32条：用EnumSet代替位域**

用OR位运算将常量合并在一个集合中的表示法成为位域。

正是因为枚举类型要用在集合中，所以没有理由用位域来表示它。

**第32条：用EnumMap代替序数索引**

最好不要用序数来索引数组，而使用EnumMap。

如果所表示的关系是多维的，就使用EnumMap<…, EnumMap<…>>。

**第34条：用接口模拟可伸缩的枚举**

虽然枚举类型不是可扩展的，但是接口类型则是可 扩展的，它是用来表示API中操作的接口类型。

虽然无法编写可拓展的枚举类型，却可以通过编写接口以及实现该接口的基础类型，对它进行模拟。

这样允许客户端编写自己的枚举来实现接口。

如果API是根据接口编写的，那么在可以使用基础枚举类型的任何地方，也可以使用这些枚举。

**第35条：注解优先于命名模式**

所有程序员都应该使用Java平台所提供的任何注解类型。

考虑使用IDE或者静态工具所提供的任何注解。

**第36条：坚持使用Override注解**

要在覆盖超类声明的每个方法声明中使用Override注解，让编辑器防止大量的错误。

在抽象类和接口中，覆盖超类或者超接口方法时，无论是具体的还是抽象的，必须使用Override注解。

**第37条：用标志定义接口定义类型**

标记接口是没有包含方法声明的接口，而只是指明（或者“标明”）一个类实现了具有某种属性的接口。

如果发现编写的目标是ElementType.TYPE的标记注解类型，考虑标记接口是否更加适合。