在JDK5中新引入的枚举完美地解决了之前通过常量来表示离散量所带来的问题，大大加强了程序的可读性、易用性和可维护性，并且在此基础之上又进行了扩展，使之可以像类一样去使用，更是为Java对离散量的表示上升了一个台阶。因此，如果在Java中需要表示诸如颜色、方式、类别、状态等等数目有限、形式离散、表达又极为明确的量，应当尽量舍弃常量表示的做法，而将枚举作为首要的选择。

语法（定义）

创建枚举类型要使用 enum 关键字，隐含了所创建的类型都是 java.lang.Enum 类的子类（java.lang.Enum 是一个抽象类）。枚举类型符合通用模式 Class Enum<E extends Enum<E>>，而 E 表示枚举类型的名称。枚举类型的每一个值都将映射到 protected Enum(String name, int ordinal) 构造函数中，在这里，每个值的名称都被转换成一个字符串，并且序数设置表示了此设置被创建的顺序。

enum 对象的常用方法介绍

int compareTo(E o)

比较此枚举与指定对象的顺序。

Class<E> getDeclaringClass()

返回与此枚举常量的枚举类型相对应的 Class 对象。

String name()

返回此枚举常量的名称，在其枚举声明中对其进行声明。

int ordinal()

返回枚举常量的序数（它在枚举声明中的位置，其中初始常量序数为零）。

String toString()

返回枚举常量的名称，它包含在声明中。

static <T extends Enum<T>> T valueOf(Class<T> enumType, String name)

返回带指定名称的指定枚举类型的枚举常量。

因为在Java中一个枚举就是一个类，它也可以有属性和方法，并且实现接口。只是所有的枚举都继承自java.lang.Enum类，因此enum不可以再继承其他的类。

下面给出在枚举中声明属性和方法的示例：

1. **package** com.fhp.enumexample;
3. **public** **enum** TypeEnum {
4. VIDEO(1), AUDIO(2), TEXT(3), IMAGE(4);
6. **int** value;
8. TypeEnum(**int** value) {
9. **this**.value = value;
10. }
12. **public** **int** getValue() {
13. **return** value;
14. }
15. }

在这个枚举中，每个枚举的值都有一个对应的int型字段，而且不同的枚举值也会有不同的int数值。同时，它和普通的类一样，可以声明构造器和各种各样的方法。如：

**[java]** [view plain](http://blog.csdn.net/u014527058/article/details/52751488) [copy](http://blog.csdn.net/u014527058/article/details/52751488)

1. TypeEnum type = TypeEnum.TEXT;//type的value属性值为3。
2. System.out.println(type.getValue());//屏幕输出3。

如果要为每个枚举值指定属性，则在枚举中必须声明一个参数为属性对应类型的构造方法（不能是public）。否则编译器将给出The constructor TypeEnum(int, String) is undefined的错误。在此例中，属性为int型，因此构造方法应当为int型。除此之外，还可以为枚举指定多个属性，如：

**[java]** [view plain](http://blog.csdn.net/u014527058/article/details/52751488) [copy](http://blog.csdn.net/u014527058/article/details/52751488)

1. **package** com.fhp.enumexample;
3. **public** **enum** TypeEnum {
4. VIDEO(1, "视频"), AUDIO(2, "音频"), TEXT(3, "文本"), IMAGE(4, "图像");
6. **int** value;
7. String name;
9. TypeEnum(**int** value, String name) {
10. **this**.value = value;
11. **this**.name = name;
12. }
14. **public** **int** getValue() {
15. **return** value;
16. }
18. **public** String getName() {
19. **return** name;
20. }
21. }