五大原则之----里氏替换原则（LSP）

阐述：子类型（subtype）必须能够替换掉它们的基类型（basetype）

先提出一个问题：正方形是不是一种特殊的长方形（IS - A关系）？

先不要回答这个问题，看下面的分析。

理解：LSP原则的一个例子，假如有个people的基类，两个字类man类和woman类，都继承于people类。那么针对people类的任何操作，比如fun吃饭、fun睡觉、fun走路，对于man类和woman类都成立。这个很好理解，不管是man还是woman，归根结底，还都是一个people。

（一）正常思维

如下例子：

```
class CShape
{
public:
 CShape(void);
 ~CShape(void);
public:
 virtual void Draw();
};

class CCircle:public CShape
{
public:
 CCircle(void);
 ~CCircle(void);
public:
 virtual void Draw();
};

class CSquare:public CShape
{
public:
 CSquare(void);
 ~CSquare(void);
public:
 virtual void Draw();
};
```

在使用CShape对象的任何地方，都可以使用CCircle对象或者CSquare对象。

（二）、特殊情况呢?

回到最初的问题，正方形是不是矩形的问题。

```
class CRectangle
{
public:
 CRectangle(void);
 ~CRectangle(void);
protected:
 int  width;
 int  height;
};

class CSquare:public CRectangel
{
public:
 CSquare(void);
 ~CSquare(void);
};
```

假如有个函数

```
void g(CRectangle * r)

   r.width = 4;

   r.height = 5;

   if( r.Area() != 20)

     break;
```

请问，对于函数g来说，能用一个CSquare对象，代替CRectangle对象吗？很明显，不能！

很明显，违反了LSP原则。

那么，正方形到底是不似乎矩形呢？也就是说CSquare和CRectangle之间，是否存在(IS - A)关系呢？

1、从属性方面讲，正方形是矩形，是一种特殊矩形，即width = height;

2、从行为方式将，正方形可能不是矩形。

比如，对于函数g来说，描述了矩形的一种行为方式，很明显，正方形不符合这种行为方式。

OOD中的IS-A关系，是就行为方式而言的，行为方式是可以进行合理假设的。而行为方式，才是我们进行面向对象软件设计真正所关注的问题。

因此，可以讲，正方形不是一个矩形。

（三）、怎么处理此类问题呢？

1、基于契约进行设计。

每个类设计时，都会有一些假设，每个方法，都有前置条件，后置条件，这些条件都是契约。对这些方法，要注明契约。

要想从基类派生子类，就必须满足这些契约。如果不满足这些契约，就不能继承出子类。（即使他们看起来很像，比

如正方形与矩形）

2、但是我们又需要LSP原则，怎么办呢?

从CRectangle类和CSquare类，提取出公共部分，做为一个基类。比如CShape类。

CRectangle和CSquare都继承自CShape类。

具体一些例子，参考《敏捷软件开发》相关章节

Links