一、LoD原则核心思想

“只和朋友通信，不和陌生人说话”，即对象间交互应局限于直接关联的 “朋友” 范围，避免通过中间对象间接调用陌生对象的方法，以降低系统耦合度。

二、“朋友圈”认定依据

对象本身：指的是当前对象本身，比如模型（Model）通过firePropertyChange()方法通知自身状态变化，仅与自身属性交互

成员变量：指的是对象持有的成员变量（如字段、属性），比如控制器（Controller）持有registeredModels和registeredViews列表作为成员变量，直接管理模型与视图

方法参数：指的是方法调用时传入的参数，比如视图（View）的构造函数接收控制器（Controller）作为参数，建立直接关联

方法内创建的对象：指的是方法体内通过new关键字创建的对象，比如模型通过new PropertyChangeSupport(this)创建事件支持对象，仅在当前类内部使用

对象的组件：对象包含的复合组件（如容器中的子组件），比如视图（View）中的JSpinner、JSlider等 Swing 组件作为成员变量，通过事件监听器直接交互

三、MVC模式中的LoD实践

（一）model的朋友圈

朋友对象：自身属性、PropertyChangeSupport（事件支持对象）、注册的监听器（如控制器），比如下面的代码：

// 模型通过成员变量PropertyChangeSupport通知监听器（控制器）

protected PropertyChangeSupport propertyChangeSupport;

public void setText(String text) {

// 仅与自身属性和事件支持对象交互

String oldText = this.text;

this.text = text;

firePropertyChange("text", oldText, text); // 通知控制器（朋友）

}

（二）controller的朋友圈

朋友对象：注册的模型（registeredModels）、注册的视图（registeredViews）、PropertyChangeEvent（事件参数），比如下面的代码：

// 控制器通过成员变量管理模型与视图，并处理事件参数

public void propertyChange(PropertyChangeEvent evt) {

for (AbstractViewPanel view : registeredViews) {

view.modelPropertyChange(evt); // 仅与注册的视图（朋友）交互

}

}

（三）view的朋友圈

朋友对象：控制器（构造参数）、Swing 组件（成员变量）、DocumentEvent（事件参数）

// 视图通过构造参数接收控制器，并与Swing组件直接交互

public PropertiesViewPanel(DefaultController controller) {

this.controller = controller; // 控制器为方法参数（朋友）

initComponents(); // 初始化Swing组件（成员变量）

}

private void textDocumentChanged(DocumentEvent evt) {

controller.changeElementText(evt.getDocument().getText(...)); // 仅与控制器（朋友）交互

}