# **更改阻止程序**

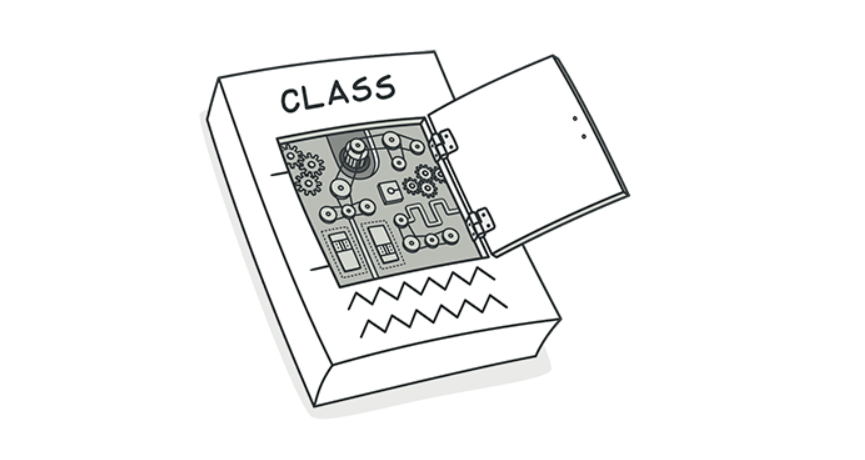
这些气味意味着，如果您需要在代码中的一个位置更改某些内容，那么同时需要在代码的其他位置进行更多的修改。因此，程序开发变得更加复杂和昂贵。

# **发散变化**

发散性变化类似于**[霰弹枪手术](https://refactoring.guru/smells/shotgun-surgery)**，但实际上气味相反。发散更改是指对单个类进行许多更改。霰弹枪手术是指同时对多个类进行单个更改。

### **体征和症状**

当您对类进行更改时，您会发现自己必须更改许多不相关的方法。例如，添加新的产品类型时，您必须更改查找、显示和订购产品的方法。



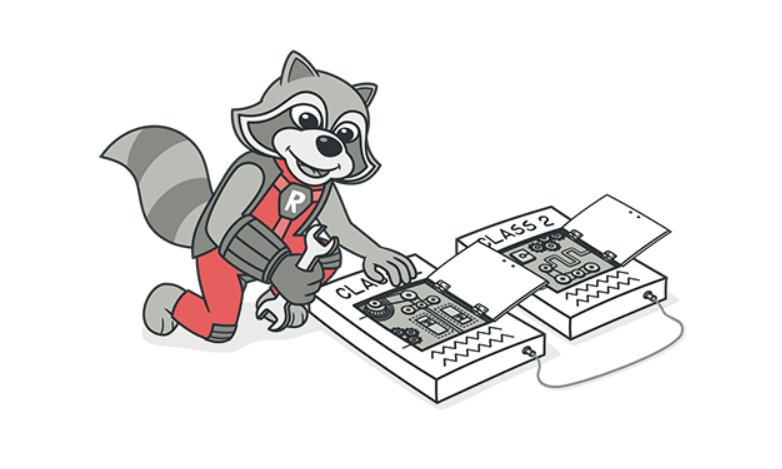
### **问题的原因**

通常这些地方的修改是由于程序结构不佳或"复制粘贴编程"造成的。

### **治疗**

### 通过**["提取](https://refactoring.guru/extract-class)类**"拆分类的行为。

如果不同的类具有相同的行为，则可能需要通过继承**[（"提取父类"和"提取子类](https://refactoring.guru/extract-superclass)**"）来组合**[这些类](https://refactoring.guru/extract-subclass)**。



### 收益

改进了代码组织。

减少代码重复。

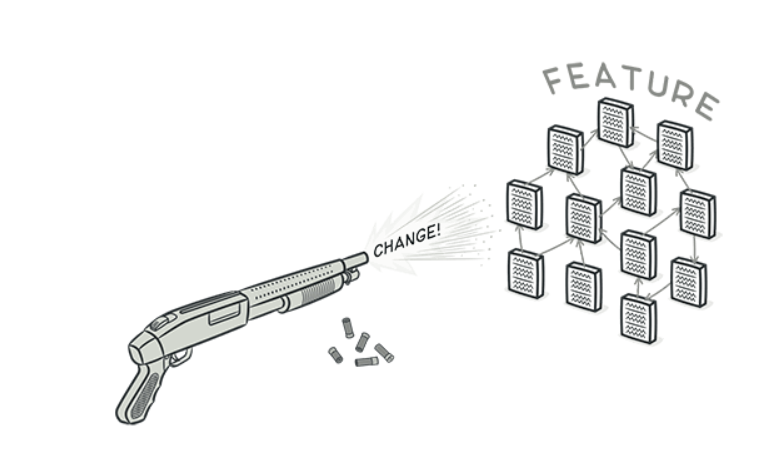
简化支持。

# **霰弹枪手术**

霰弹枪手术类似于**[发散性变化，](https://refactoring.guru/smells/divergent-change)**但实际上是相反的气味。发散更改是指对单个类进行许多更改。霰弹枪手术是指同时对多个类进行单个更改。

### **体征和症状**

进行任何修改都需要对许多不同的类进行许多小的更改。



### **问题的原因**

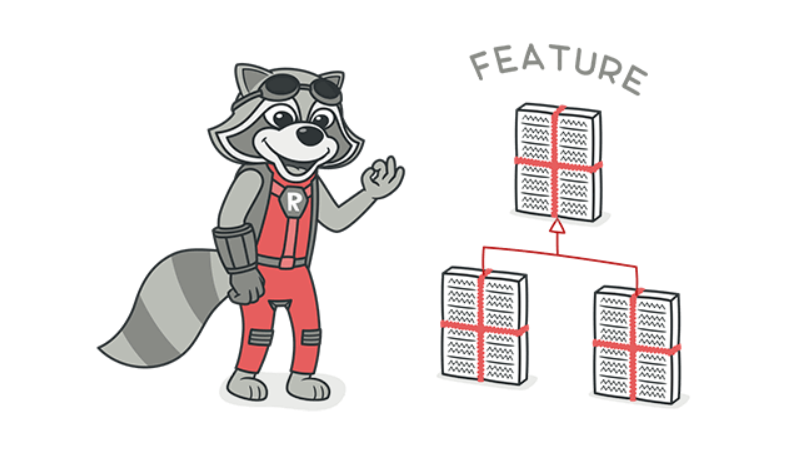
一个单一的行为被划分为许多类。这可能发生在过度使用‘**[发散变化](https://refactoring.guru/smells/divergent-change)**’之后。



### **治疗**

使用**["移动方法](https://refactoring.guru/move-method)["和"移动字段"](https://refactoring.guru/move-field)**将现有类行为移动到单个类中。如果没有适合此目的的类，请创建一个新类。

如果将代码移动到同一类使原始类几乎为空，请尝试通过**[内联类](https://refactoring.guru/inline-class)**来摆脱这些现在冗余的类。



**收益**

更好的组织结构。

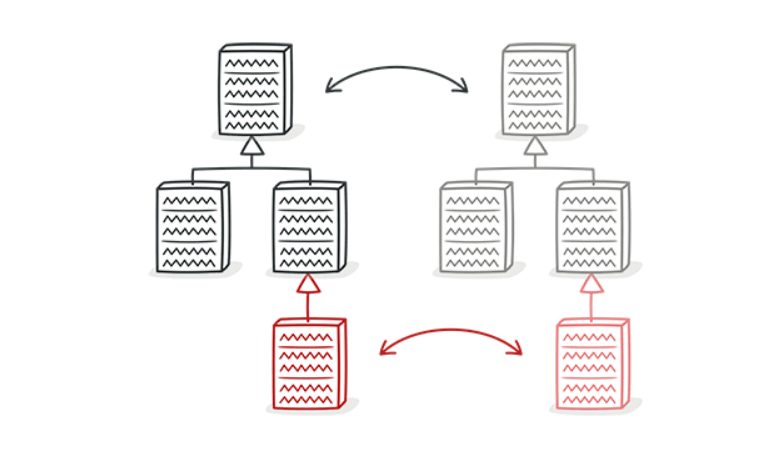
减少代码重复。

更易于维护。

# **并行继承层次结构**

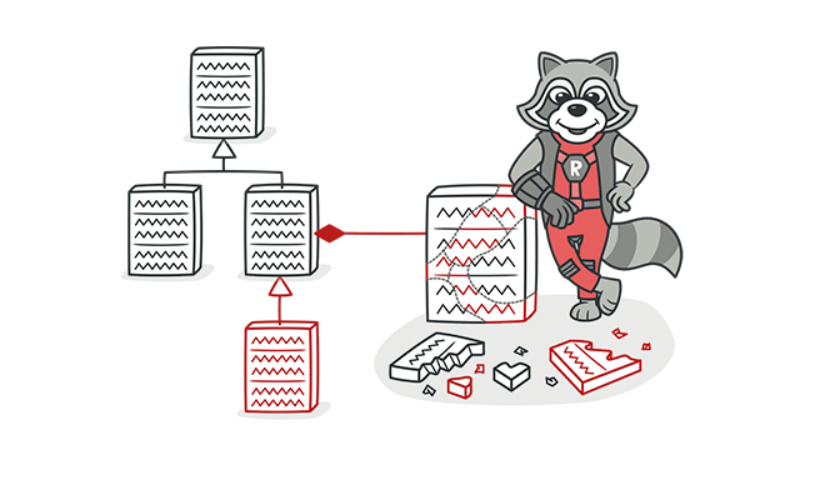
### **体征和症状**

每当为类创建子类时，您都会发现自己需要为另一个类创建子类。



### **问题的原因**

只要等级制度层数比较低，那么则不会出现问题。但是随着新类的加入，进行更改变得越来越困难。



### **治疗**

您可以通过两个步骤消除重复的并行类层次结构。首先，使一个层次结构的实例引用另一个层次结构的实例。然后，通过使用**["移动方法"和"移动](https://refactoring.guru/move-method)[字段"](https://refactoring.guru/move-field)**删除所引用的类中的层次结构。

### **收益**

减少代码重复。

可以改进代码的组织结构。