## 设计模式详述

## Visitor

****访问者**模式**是一种行为设计模式， 它能将算法与其所作用的对象隔离开来。访问者模式通过在访问者对象中为多个目标类提供相同操作的变体， 让你能在属于不同类的一组对象上执行同一操作。该模式会将所有非主要的行为抽取到一组访问者类中， 使得程序的主要类能更专注于主要的工作。

优点：1、开闭原则。 你可以引入在不同类对象上执行的新行为， 且无需对这些类做出修改。

1. 单一职责原则。可将同一行为的不同版本移到同一个类中。
2. 访问者对象可以在各种对象交互时收集一些有用的信息。当你想要遍历一些复杂的对象结构（例如对象树），并在结构中的每个对象上应用访问者时，这些信息可能会有所帮助。

缺点：1、每次在元素层次结构中添加或移除一个类时， 你都要更新所有的访问者。

2、在访问者同某个元素进行交互时， 它们可能没有访问元素私有成员变量和方法的必要权限。

API描述

Visitor场景为：迪士尼乐园会接受各种各样的游客，其中因为游客团体的不同，迪士尼的票务系统也提供不同的种类的票供游客选择，也提供不同的票务窗口。因为游客的数量不同，门票分为团体窗口和个人窗口，因为游客身份的不同，门票分为全价票和半价票

EventListener是用于处理Editor产生的状态改变，进而向特定的EventListener发送值得关注的事件，且允许新预约者加入和当前预约者离开列表的订阅框架。EventListener接口声明了通知接口，仅包含一个update()方法。RollerCosterListener和CarrouselListener分别对应了预约摩天轮和预约旋转木马的预约者。

Class diagram

## Filter

Filter Pattern（过滤器模式）又被称为Criteria Pattern（标准模式 ）允许开发人员使用不同的标准来过滤一组对象，通过逻辑运算以解耦的方式把它们连接起来。这种类型的设计模式属于结构型模式，它结合多个标准来获得单一标准。

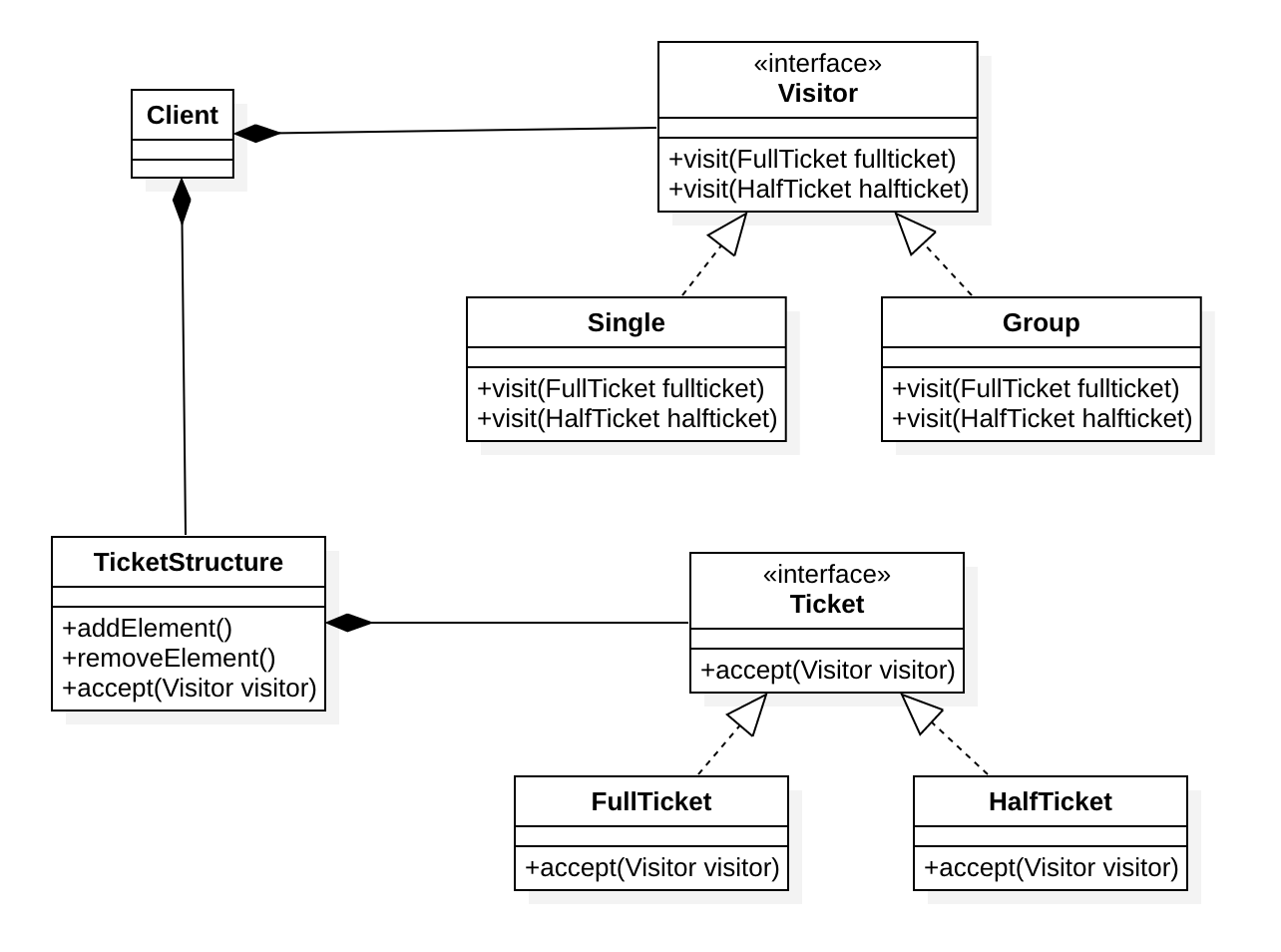
优点：简单，解耦，使用方便。

缺点：在构建过滤规则的时候会有些繁琐。

API描述

Filter场景为：迪士尼乐园有游乐设施如摩天轮和旋转木马，每个游乐设施都对游客的年龄或身高有一定限制，在已预约该项目的游客中，需要以不同的条件筛选出符合游玩条件的游客并显示。

Class diagram



## Facade

Facade（外观模式）是一种结构型设计模式， 能为程序库、 框架或其他复杂类提供一个简单的接口。提供了一种访问特定子系统功能的便捷方式， 其了解如何重定向客户端请求， 知晓如何操作一切活动部件。

优点：1、你可以让自己的代码独立于复杂子系统。

缺点：1、外观可能成为与程序中所有类都耦合的**[上帝对象](https://refactoringguru.cn/antipatterns/god-object)**。

API描述

迪士尼内部的餐厅有着明确的分工，分为前台、厨房、服务员三个部门，分别负责点单、烹饪、上菜。因此有三个子系统FrontDesk、Kitchen、Waiter,有各自的成员函数。

Class diagram

