Пермский филиал федерального государственного автономного
образовательного учреждения высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»

Факультет экономики, менеджмента и бизнес-информатики

Рязанов Иван Дмитриевич

ОРГАНИЗАЦИЯ ПАТТЕРНОВ ПРОЕКТИРОВАНИЯ. СТРУКТУРНЫЙ ПАТТЕРН «ЗАМЕСТИТЕЛЬ»

Лабораторная работа

студента образовательной программы «Программная инженерия»
по направлению подготовки *09.03.04 Программная инженерия*

Руководитель
к.т.н., доцент кафедры Информационных технологий в бизнесе НИУ ВШЭ-Пермь

А.В. Кычкин

Пермь, 2020 год

Оглавление

Глава 1. Паттерн «Заместитель»

Название и классификация паттерна. Заместитель - паттерн, структурирующий объекты, который предоставляет объект, который контролирует доступ к другому объекту, перехватывая все вызовы.

Назначение. Управление ресурсоемкими объектами, при котором нет необходимости создавать экземпляры таких объектов до момента их реального использования. Proxy является суррогатом другого объекта и управляет доступом к нему.

Заместитель это объект, интерфейс которого идентичен интерфейсу реального объекта. При первом запросе клиента заместитель создает реальный объект, сохраняет его адрес и затем отправляет запрос этому реальному объекту. Все последующие запросы просто переадресуются инкапсулированному реальному объекту.

Применимость. Существуют ситуации, когда можно использовать паттерн Proxy:

1. Виртуальный proxy является заместителем объектов, создание которых обходится дорого. Реальный объект создается только при первом запросе/доступе клиента к объекту.

2. Удаленный proxy предоставляет локального представителя для объекта, который находится в другом адресном пространстве.

3. Защитный proxy контролирует доступ к основному объекту. "Суррогатный"объект предоставляет доступ к реальному объекту, только вызывающий объект имеет соответствующие права.

4. Интеллектуальный proxy выполняет дополнительные действия при доступе к объекту.

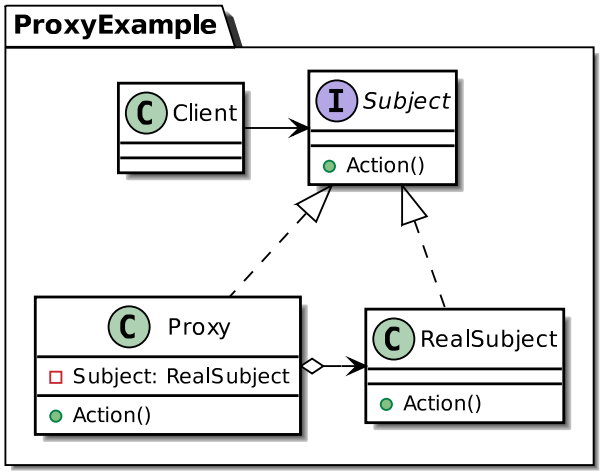

Рис. 1.1. Диаграмма классов паттерна «Заместитель»

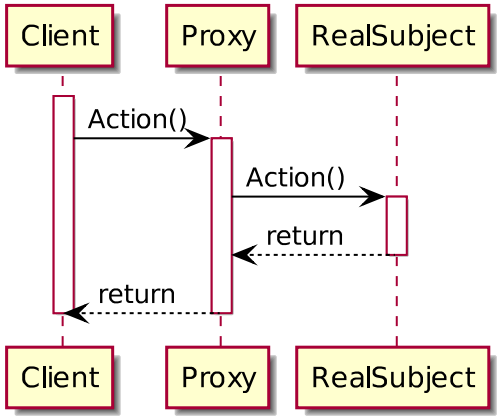

Рис. 1.2. Диаграмма последовательности паттерна «Заместитель»

Участники

1. Proxy - заместитель:
    - хранит ссылку, которая позволяет заместителю обратиться к реальному субъекту. Объект класса Proxy может обращаться к объекту класса Subject, если интерфейсы классов RealSubject и Subject одинаковы;
    - предоставляет интерфейс, идентичный интерфейсу Subject, так что заместитель всегда может быть подставлен вместо реального субъекта;
    - контролирует доступ к реальному субъекту и может отвечать за его создание и удаление;
    - прочие обязанности зависят от вида заместителя:
    - *удаленный заместитель* отвечает за кодирование запроса и его аргументов и отправление закодированного запроса реальному субъекту в другом адресном пространстве;
    - *виртуальный заместитель* может кэшировать дополнительную информацию о реальном субъекте, чтобы отложить его создание.
    - *защищающий заместитель* проверяет, имеет ли вызывающий объект необходимые для выполнения запроса права;
2. Subject - субъект: определяет общий для RealSubject и Proxy интерфейс, так что класс Proxy можно использовать везде, где ожидается RealSubject;
3. RealSubject - реальный субъект: определяет реальный объект, представленный заместителем.

Отношения Proxy при необходимости переадресует запросы объекту RealSubject. Детали зависят от вида заместителя.

Плюсы и минусы Плюсы:

1. Позволяет контролировать сервисный объект незаметно для клиента.
2. Может работать, даже если сервисный объект ещё не создан.
3. Может контролировать жизненный цикл служебного объекта.

Минусы:

1. Усложняет код программы из-за введения дополнительных классов.
2. Увеличивает время отклика от сервиса.

Области применения

1. Системы доставки контента (кеширование важно).
2. ПО, требующее авторизацию пользователей.

Глава 2. Проектирование и реализация

2.1. Проектирование

Для реализации был выбран 2 вариант:

«Описание сценариев Умного дома. Описание различных функций, получаемых путем комбинации датчиков, исполнительных механизмов, панелей оператора, мультимедиа систем, контроллеров управления, сетевых устройств.»

Участники

1. SmartHomeComponent — компонент умного дома: - содержит поле с названием компонета и статус (активен или неактивен). Имеет метод для активации/деактивации.

2. SmartTV, SmartLight, SmartCurtains, SmartFloor — примитивы, компоненты умного дома.

3. SmartGroup — объединенная группа компонентов умного дома.

4. SmartHomeScenario — сценарий умного дома - содержит поля для групп компонентов, которыми сценарий управляет.

5. Client — клиент: - запускает сценарии умного дома через объекты SmartHomeScenario.

2.2. Реализация

Реализация паттерна «Компоновщик» находится в git-репозитории по ссылке: github.com

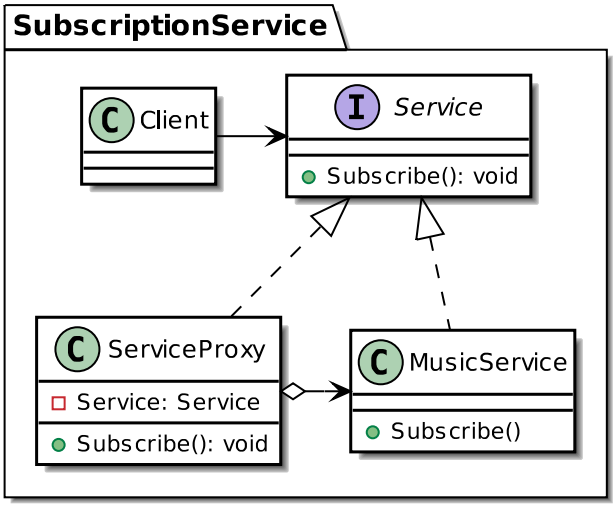

Рис. 2.1. Диаграмма классов