Практическая работа №4

По Проектному практикуму

Выполнил студент

Группы К17.1

Шишелов Владимир

1. Сравните использование стратегии в разных языках программирования

Стратегия — поведенческий шаблон проектирования, предназначенный для определения семейства алгоритмов, инкапсуляции каждого из них и обеспечения их взаимозаменяемости. Это позволяет выбирать алгоритм путём определения соответствующего класса.

Стратегию можно реализовать на любом языке программирования, зависит только от синтаксиса.

1. Насколько рациональным является документирование паттерна при сдаче кода заказчику?

Он не является рациональным, так как паттерн очень просто обнаружить опытным разработчиком, а для заказчика информация о паттерне не имеет смысла. Его можно бегло упомянуть, но не большее.

1. Чем можно заменить "Стратегию" в разных ситуациях?

Паттерн «Стретигия» может быть заменен на паттерн «Состояние».

Состояние — поведенческий шаблон проектирования. Используется в тех случаях, когда во время выполнения программы объект должен менять своё поведение в зависимости от своего состояния. Паттерн состоит из 3 блоков: Widget — класс, объекты которого должны менять своё поведение в зависимости от состояния.

1. Опишите процесс использования контрактов в шаблонном методе.

Шаблонный метод определяет контракт между базовым классом и наследниками, поэтому очень важно, чтобы этот контракт был максимально понятным.

1. Опишите пример использования интерфейса в шаблонном методе для обеспечения тестируемости.

За счет использования интерфейса в шаблонном методе, происходит унификация, за которой следует упрощение тестируемости.

1. Дайте объяснение почему переменные шаги алгоритма должны определятся открытыми методами.

Для сокращения размерности проекта.

1. Наведите несколько примеров практического использования паттерна "Итератор".

Iterator — поведенческий шаблон проектирования. Представляет собой объект, позволяющий получить последовательный доступ к элементам объекта-агрегата без использования описаний каждого из агрегированных объектов. - Перебор связного списка.

1. В чем разница между итераторами и генераторами?

В том, что итератор перебирает существующие данные, а генератор перебирает генерируемые данные, т.е по определенному алгоритму генератор выдает новое значение.

1. Дайте характеристики обобщенным и необобщенным итераторам.

Обобщенный итератор является методом объекта, а необобщенные итераторы являются структурами для перебора значений по типу foreach.

1. Что такое делегаты, как они применяются в паттерне "наблюдатель"?

Делегатами являются объекты, указывающие на методы. Делегаты в «наблюдателе» используются для назначения определенного действия в случае наступления того или иного события.

1. Как реализованы события в паттерне "наблюдатель"?

Существует «главный» класс и «ведомые». «Ведомые» подписываются на «главный» и когда главный вызывает метод (как правило называется emit), ведомые реагируют и запускаются.

1. Существует ли проблема утечки памяти в приложениях, построенных с применением паттерна "наблюдатель", не в ASP.NET архитектурах(в других языках разработки)?

Плохой программист может все что угодно реализовать с проблемой утечки памяти.

1. Почему паттерн посетитель является расширяемым?

Потому что он позволяет определить те команды, которые заранее не прописаны в системе классов. Паттерн «Посетитель» позволяет определить новую операцию, не изменяя классов этих объектов.

1. Что даёт применение двойной диспетчеризации?

Выбор выполняемого метода определяется типами посещаемого объекта и посетителя.

1. Приведите пример использования паттерна "Посетитель"

Мы имеем класс “Netcard”, которая предоставляет поток из сетевых пакетов. Нам необходимо обрабатывать эти сырые данные разными способами. Возможность обработать эти данные разными способами даже после запрета на изменение класса становится доступна после добавления посетителя.

1. Приведите практический пример использования паттерна "Команда".

Запросы к базе данных.

1. Какие есть альтернативы паттерну "Цепочка обязянностей"?

Function composition.

1. Опишите процесс создания семейства объектов в абстрактной фабрике.

В конструктор фабрики или в ее метод передается условное название объекта, который должен быть создан.

1. Что такое DI-контейнеры и где они применяются?

DI-контейнер, как и общий паттерн IoC необходим для инверсии контроля, с целью создания слабой связности.

1. Из каких составляющих состоит абстрактная фабрика?

* AbstractFactory – интерфейс для создания семейства продуктов.
* AbstractProduct – семейство продуктов, которое будет использоваться клиентом.
* Product – конкретные типы продуктов.
* Client – клиент фабрики.

1. Зачем необходимо ограждать клиентов от подробностей создания экземпляров класса или иерархии классов?

Так как иерархия и инкапсуляцию является причиной устойчивости проектов, ибо он не позволяет изменять резко систему.

1. Зачем нужен полиморфный фабричный метод?

Полиморфный фабричный метод определяет интерфейс фабрики, а за создание конкретного экземпляра продукта отвечает конкретная фабрика.

1. Какие есть особенности у фабричного метода?

Есть несколько видов фабричного метода:

Классический фабричный метод является частным случаем шаблонного метода. Это значит, что фабричный метод привязан к текущей иерархии типов и не может быть использован повторно в другом контексте.

Полиморфный фабричный метод является стратегией создания экземпляров некоторого семейства типов, что позволяет использовать одну фабрику в разных контекстах. Тип создаваемого объекта определяется типом фабрики и обычно не зависит от аргументов фабричного метода.

Статический фабричный метод является самой простой формой фабричного метода. Статический метод создания позволяет обойти ограничения конструкторов. Например, тип создаваемого объекта может зависеть от аргументов ме- тода, экземпляр может возвращаться из кэша, а не создаваться заново или же фабричный метод может быть асинхронным.

1. Привести примеры применения паттерна строителя

При построении сметы по зданию. Есть класс дом, который реализуется в формате паттерна «строитель». После инициализации указываются необходимые данные, колиество, качество материала, после чего создается общая смета и цена.

1. Что такое Test Fixture?

Test fixture предоставляет из себя окружение для тестирования того или иного элемента/ системы и так далее.

1. Как связаны параметрезированные юнит-тесты с паттерном строитель?

Параметризированные тесты (классы для работы с ними) реализованы в виде паттерна «строитель». В них указываются определенные параметры, что позволяет значительно упростить создание комплексных тестов.

1. В каких ситуациях невозможно использовать паттерн Адаптер?

-

1. Каким образом паттерн Адаптер связан с рефакторингом?

При изменении системы, паттерн позволяет адаптироваться к виду нового интерфейса.

1. Что такое языковые адаптеры, приведите примеры

Языковые адаптеры позволяют реализовать/дополнить определенный синтаксис языка программирования.

1. Какие специфические сценарии могут быть выполнены при помощи фасада?

Фасад позволяет повысить уровень абстракции и инкапсулирует сторонний код.

1. Какие низкоуровневые классы или библиотеки поддерживает фасад?

В зависимости от того, какой фасад спроектировал пользователь.

1. Объясните почему фасад не применим с большими интерфейсами?

Ибо если фасад будет увеличивать уровень абстракции над и так большими интерфейсами, это может привести к огромному количеству лишних надстроек в системе, что приведет к лишней сложности, что ведет к семантической нагрузке.

1. Распишите суть и примеры порождающих патттернов

Порождающие паттерны нацелены на специфику и особенности создания тех или иных объектов, нежели на то, как, из че