ИСПОЛЬЗУЕМЫЕ ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ ПРИ СОЗДАНИИ ПРОГРАММНОГО МОДУЛЯ ДЛЯ ПЛАТФОРМЫ MAGENTO 2

Грищенко Э.А., Потоцкий Д.С.

Белорусский государственный университет информатики и радиоэлектроники,
Институт информационных технологий,
Минск, Республика Беларусь

Потоцкий Д.С. – ассистент кафедры ИСиТ

Аннотация. В данной статье описывается использование основных паттернов проектирования при создании программного модуля составления отчетности коммерческой деятельности для платформы Magento 2. В работе также рассматривается модульная архитектура, файловая структура модуля, а также паттерн проектирования – внедрение зависимостей. Статья содержит примеры конфигурации для демонстрации функциональных возможностей менеджера объектов Magento 2.

Ключевые слова: Magento, программный модуль, электронная коммерция, паттерны проектирования, объектно-ориентированное программирование.

Введение

Паттерны проектирования являются проверенными стандартами в области разработки программного обеспечения. Они помогают улучшить структуру кода, делая его более читаемым и понятным для разработчиков. В сфере электронной коммерции соблюдение паттернов проектирования особенно важно, так как интернет-магазины часто имеют сложные структуры и требуют высокой производительности и безопасности.

Соблюдение методологии паттернов проектирования позволяет сократить время разработки и снизить затраты на поддержку системы в будущем. Кроме того, использование паттернов уменьшает риск возникновения проблем и повышает качество программного обеспечения.

Паттерн проектирования – это решение для часто встречающейся в разработке программного обеспечения проблемы, которое может быть использовано повторно для улучшения качества и эффективности программ. Паттерны проектирования помогают разработчикам создавать гибкие, структурированные и легко изменяемые системы, а также сокращают время разработки за счет повторного использования решений.

История паттернов проектирования началась в 70-х годах прошлого века с публикации книги Кристофера Александера «A Pattern Language», в которой были представлены понятия паттернов в архитектуре. Затем в 1994 году Эрих Гамма, Ричард Хелм, Ральф Джонсон и Джон Влиссидес издали книгу «Design Patterns: Elements of Reusable Object-Oriented Software», в которой описали 23 паттерна проектирования для объектно-ориентированных систем.

Основная часть. Программный модуль составления отчетности коммерческой деятельности разработан исключительно под платформу электронной коммерции Magento 2, в связи с этим кодовая база программных модулей должна полноценно соблюдать требования и рекомендации платформы. Magento 2 в своей основе кастомизации имеет модульную архитектуру.

Модуль – это логическая группа, то есть каталог, содержащий блоки, контроллеры, помощники и модели, связанные с определенной бизнес-функцией. В соответствии со стремлением Magento к оптимальной модульности модуль инкапсулирует одну функцию и имеет минимальные зависимости от других модулей.

Модули предоставляют бизнес-функции с вспомогательной логикой. Они имеют свой жизненный цикл, который позволяет их устанавливать, удалять и отключать. С точки зрения как пользователей, так и разработчиков расширений, модули являются центральной единицей организации архитектуры Magento 2. Каждый модуль образует логическую группу, состоящую из блоков, хелперов, контроллеров и моделей (PHP и XML файлов). Каждая такая группа является независимой. Таким образом, модульный подход приводит к тому, что

каждый модуль обладает своими уникальными свойствами и особенностями и минимально зависит от других модулей, а его отключение не влияет на работу других модулей.

Платформа Magento 2 предоставляет набор базовой логики: код PHP, библиотеки, ключевые классы, интерфейсы и методы, которые наследуются модулями и другими компонентами.

Для понимания общей архитектуры на рисунке 1 приведена структура файлов модуля и далее опишем назначение основных директорий.

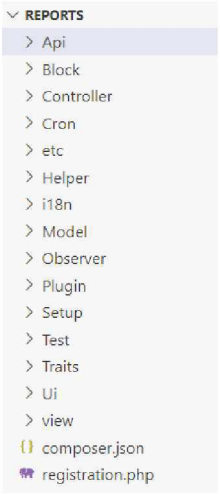

Рисунок 1 – Файловая структура модуля для платформы Magento 2

Файловая структура модуля для платформы Magento 2 обладает высокой степенью организации и стандартизации. Эта структура не только обеспечивает четкое разделение функциональных компонентов, но и создает удобные точки взаимодействия между ними, что существенно упрощает процесс разработки, поддержки и масштабирования приложений.

Файл registration.php играет ключевую роль в процессе регистрации модуля в системе Magento 2, обеспечивая его корректную интеграцию и функционирование.

В папке etc находятся различные файлы конфигурации модуля в формате XML. Эти файлы определяют важные аспекты модуля, такие как внедрение зависимостей, пути для контроллеров и многое другое.

Папка Api содержит интерфейсы моделей и интерфейсы моделей данных, обеспечивая разработчикам безопасное и удобное взаимодействие с методами и свойствами моделей.

Blocks – это хранилище вью-моделей, которые служат прослойкой между данными и представлением в архитектурном паттерне MVVM. Они играют важную роль в разделении логики и представления.

Controller отвечает за обработку веб-запросов. Структура URL-запроса определяется стандартно и состоит из нескольких сегментов, включая frontName из файла etc/routes.xml, путь к файлу контроллера и его класс.

Cron содержит скрипты для планировщика задач, такие как проверка заброшенных корзин и рассылка уведомлений.

Helper служит для хранения классов-помощников, хотя их использование рекомендуется минимизировать в силу их анти-паттерна.

Директория i18n содержит файлы перевода в формате .CSV для различных локалей.

Model – это пространство хранения моделей, отвечающих за работу с данными и бизнес-логику.

Observer предназначен для хранения событий, позволяющих изменять поведение кода до, во время и после его выполнения.

Наконец, Plugin используется для изменения поведения публичных методов, обеспечивая безопасное взаимодействие с архитектурой Magento 2.

В папке view хранятся все статичные файлы модуля, включая JavaScript, CSS, HTML и изображения. Эти файлы определяют внешний вид и поведение модуля в пользовательском интерфейсе.

Также в рамках статьи хотелось бы подробно рассмотреть основной паттерн проектирования в процессе создания программных модулей для платформы Magento 2, а именно - Dependency Injection (Внедрение зависимостей).

Создание объектов является одной из самых сложных операций в рамках объектно-ориентированного программирования [1], классы могут иметь множество зависимостей, зависимости могут иметь свои зависимости и так далее.

Внедрение зависимостей (англ. Dependency injection, DI) – процесс предоставления внешней зависимости программному компоненту. Является специфичной формой «инверсии управления» (англ. Inversion of control, IoC), когда она применяется к управлению зависимостями. В полном соответствии с принципом единственной ответственности объект отдаёт заботу о построении требуемых ему зависимостей внешнему, специально предназначенному для этого общему механизму.

Таким механизмом в платформе Magento 2 является специальный класс ObjectManager. Благодаря менеджеру объектов, в процессе программирования разработчику вообще не стоит задумываться над созданием объектов, а лишь при необходимости настраивать некоторые конфигурации создаваемых объектов.

XML-тег «preference» – используется для сопоставления реализации(класса) и абстракции(интерфейс), когда сигнатура конструктора класса запрашивает объект через его интерфейс. Диспетчер объектов использует эти сопоставления, чтобы определить, какова реализация по умолчанию для этого класса в определенной области [2]. Это используется для соблюдения так называемых Service Contracts в Magento 2. Контракт службы включает в себя интерфейсы данных, которые сохраняют целостность данных, и интерфейсы служб, которые скрывают детали бизнес-логики от запрашивающих службы, таких как контроллеры, веб-службы и другие модули. Таким образом мы определяем непрямые и легко заменяемые зависимости.

Благодаря тегу virtualType можно создавать “виртуальные” типы, не затрагивая PHP код. Виртуальный тип позволяет изменять аргументы конкретной внедряемой зависимости и изменять поведение определенного класса. Это позволяет вам использовать настроенный класс, не затрагивая другие классы, зависящие от оригинала.

Тег type – описывает образ жизни объекта и способы его создания, в некотором виде представляет собой дополнительный помощник для менеджера объектов.

Внедрение зависимостей – это шаблон проектирования, который позволяет объекту A объявлять свои зависимости от внешнего объекта B, который предоставляет эти зависимости. Зависимости, объявленные A, обычно представляют собой интерфейсы классов, а зависимости, предоставляемые B, представляют собой конкретные реализации этих интерфейсов.

Это позволяет ослабить связанность кода, поскольку объекту A больше не нужно заниматься инициализацией собственных зависимостей. Объект B решает, какие реализации предоставить объекту A, на основе конфигурации или желаемого поведения.

Заключение

Программный код Magento 2 представляет собой сложную систему, в основе которой лежит более десятка паттернов проектирования. Эти паттерны являются обязательными для соблюдения разработчиками, и это не просто формальное требование, а ключевой элемент для создания высококачественного и эффективного кода. Соблюдение паттернов обеспечивает лаконичность, грамотность и надежность программного продукта. В то время как структура каталогов, хоть и может показаться простой на первый взгляд, на самом деле играет ключевую роль в организации кода. Она предоставляет разработчикам удобные места для размещения различных компонентов модуля, начиная от интерфейсов и моделей, заканчивая контроллерами и статическими файлами. Благодаря этой структуре поддержка и развитие модуля становятся более простыми и прозрачными.

Список использованных источников:

1. Зандстра, М. PHP 8 объекты, шаблоны и методики программирования, 6-е издание / М. Зандстра.: Apress, 2021. – 860 с.
2. Документация Magento 2 [Электронный ресурс] – Режим доступа: https://developer.adobe.com/commerce/docs/. – Дата доступа: 10.02.2024.