ПАТТЕРН MVVM ДЛЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Лукашеня И. В., Сивоконь А. В.

Лихачевский Д.В. – канд. техн. наук, доцент

В современном мире требования к функциональности и возможностям программного обеспечения постоянно возрастают. В связи с этим увеличивается сложность разработки программного обеспечения и объемам данных, ужесточаются требования и к архитектуре ПО. Паттерн MVVM позволяет улучшить архитектуру и качество программного обеспечения.

Паттерн MVVM позволяет отделить логику приложения от визуальной части (представления). Данный паттерн является архитектурным.

Название шаблона отражает основные его компоненты: модель (Model), представление (View) и модель представления (ViewModel).

Данный паттерн был представлен Джоном Госсманом в 2005 году как модификация шаблона Presentation Model и был первоначально нацелен на разработку приложений в WPF. На данный момент паттерн вышел за пределы WPF и активно применяется в самых различных технологиях, в том числе при разработке под Android, iOS [1].

Модель – это объектное представление имеющихся данных. Модели могут содержать логику, непосредственно связанную этими данными, например, логику валидации свойств модели. В то же время модель не должна содержать никакой логики, связанной с отображением данных и взаимодействием с визуальными элементами управления [2].

Нередко модель реализует интерфейсы INotifyPropertyChanged или INotifyCollectionChanged, которые позволяют уведомлять систему об изменениях свойств модели. Благодаря этому облегчается привязка к представлению, хотя опять же прямое взаимодействие между моделью и представлением отсутствует.

Представление определяет визуальный интерфейс, через который пользователь взаимодействует с приложением. Интерфейс необходимо проектировать так, чтобы он был чрезвычайно легковесным. Интерфейс должен отображать только необходимую информацию и механизмы взаимодействия. Представление MVVM должны разрабатываться с учетом целей, которые оно отображает. Любые интеллектуальные возможности должны встраиваться в другие места приложения.

Модель представления связывает модель и представление через механизм привязки данных. Если в модели изменяются значения свойств, при реализации моделью интерфейса INotifyPropertyChanged автоматически идет изменение отображаемых данных в представлении, хотя напрямую модель и представление не связаны.

Модель представления служит двум целям:

1) модель представления обеспечивает единственное местоположение для всех данных, необходимых представлению. Это вовсе не означает, что модель представления отвечает за получение действительных данных; взамен она обращается к соответствующему код, чтобы получить все данные вместе там, где они легко доступны. В результате между окнами и моделями представлений обычно сохраняется однозначное отношение, но архитектурные отличия существуют, и в каждом конкретном случае они могут варьироваться;
2) вторая задача модели представления касается ее действия в качестве контроллера для представления. Модель представления принимает указания от пользователя и вызывает код для выполнения подходящих действий [2].

Применение паттерна MVVM при разработке программного обеспечения дает следующие преимущества:

1) тестируемость приложений. Приложения, разработанные с использованием MVVM, обладают очень хорошим основанием для проведения модульного тестирования с целью проверки работы отдельных классов и методов;
2) меньшее количество кода. Объем кода, необходимого для управления представлением немного снижается при использовании MVVM, а это означает, что снижается риск допустить ошибки и уменьшается код для написания модульных тестов;
3) улучшенное проектирование приложений. Разработчики и дизайнеры могут самостоятельно работать над разными частями приложения. Есть возможность создать модель представления, которое предоставляет необходимые точки входа для связывания с представлением, которые в конечном представлении можно будет легко привязать. Это позволяет дизайнерам работать над внешним видом приложения, а программистам над бизнес-логикой приложения;

4) легкость понимания логики представления. MVVM предусматривает хорошо организованную и легкую для понимания конструкцию построения графического интерфейса за счет использования механизмов привязок, команд и шаблонов данных.

Список использованных источников:

[1] Определение паттерна MVVM [Электронный ресурс]. – Режим доступа: https://goo.gl/1zVoIB

[2] Троелсен Э. Язык программирования C# 6.0 и платформа .Net 4.6 / Э. Троелсен, Ф. Джепикс – Москва: Apress, 2017 – 1200 с.