В. А. Пашкевич, Е. М. Березовская
(ГГУ им. Ф. Скорины, Гомель)

РЕАЛИЗАЦИЯ MVC ПАТТЕРНА НА ПРИМЕРЕ СОЗДАНИЯ САЙТА НА PHP

Многие начинают писать проект для работы с единственной задачей, не подразумевая, что это может вырасти в многопользовательскую систему управления, допустим, контентом или производством. И всё вроде здорово и классно, всё работает, пока не начинаешь понимать, что тот код, который написан – состоит целиком и полностью из «костылей» и «хардкода». Код, перемешанный с версткой, запросами и костылями, неподдающийся иногда даже прочтению. Возникает насущная проблема: при добавлении новых фич, приходится с этим кодом очень долго и долго возиться, вспоминая «а что же там такое написано то было?»

Шаблон MVC описывает простой способ построения структуры приложения (рис. 1), целью которого является отделение бизнес-логики от пользовательского интерфейса. В результате приложение легче масштабируется, тестируется, сопровождается, и, конечно же, реализуется.

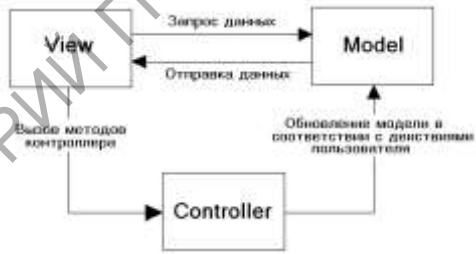

Рисунок 1 – Архитектура MVC

В архитектуре MVC модель предоставляет данные и правила бизнес-логики, представление отвечает за пользовательский интерфейс, а контроллер обеспечивает взаимодействие между моделью и представлением.

Типичную последовательность работы MVC-приложения можно описать следующим образом:

– При заходе пользователя на веб-ресурс, скрипт инициализации создает экземпляр приложения и запускает его на выполнение. При этом отображается вид, скажем, главной страницы сайта.

– Приложение получает запрос от пользователя и определяет запрошенные контроллер и действие. В случае главной страницы, выполняется действие по умолчанию (*index*).

– Приложение создает экземпляр контроллера и запускает метод действия, в котором, к примеру, содержатся вызовы модели, считывающие информацию из базы данных.

– После этого, действие формирует представление с данными, полученными из модели, и выводит результат пользователю.

Модель – содержит бизнес-логику приложения и включает методы выборки (это могут быть методы ORM), обработки (например, правила валидации) и предоставления конкретных данных, что зачастую делает ее очень толстой, что вполне нормально. Модель не должна напрямую взаимодействовать с пользователем. Все переменные, относящиеся к запросу пользователя должны обрабатываться в контроллере. Модель не должна генерировать HTML или другой код отображения, который может изменяться в зависимости от нужд пользователя. Такой код должен обрабатываться в видах.

Вид – используется для задания внешнего отображения данных, полученных из контроллера и модели.

Контроллер – связующее звено, соединяющее модели, виды и другие компоненты в рабочее приложение. Контроллер отвечает за обработку запросов пользователя.

Шаблон MVC используется в качестве архитектурной основы во многих фреймворках и CMS, которые создавались для того, чтобы иметь возможность разрабатывать качественно более сложные решения за более короткий срок. Это стало возможным благодаря повышению уровня абстракции, поскольку есть предел сложности конструкций, которыми может оперировать человеческий мозг. Но, использование веб-фреймворков типа Yii, состоящих из нескольких сотен файлов, при разработке простых веб-приложений (например, сайтов-визиток) не всегда целесообразно.

В результате проведенной работы был реализован шаблон MVC на примере сайта-визитки. Приложение строилось на основе трех базовых классов: model, view, controller и их потомков.