

Урок 2

Архитектура MVC

В этом уроке Вы

- узнаете, что такое архитектура программы;
- познакомитесь с самым популярным шаблоном проектирования – MVC;
- познакомитесь с реализацией этого шаблона в PHP.

1. Архитектура программы

Архитектура - это базовая организация системы, воплощенная в ее компонентах, их отношениях между собой и с окружением, а также принципы, определяющие проектирование и развитие системы. [IEEE 1471]






Одна из основных задач программиста – не просто писать код, а писать хороший код, т.е.:

- ✓ решающий поставленную задачу
- ✓ простой
- ✓ расширяемый
- ✓ аккуратный

Хороший код невозможен без грамотно продуманной архитектуры программы. Её (архитектуру) можно разделить на три основные составляющие:

1. Структура
2. Отношения между компонентами
3. Значимые решения организации.

Структура – это разделение кода на компоненты. По сути, мы решаем, как разбить код на файлы. Рассмотрим возможную структуру приложения на примере создания блога.

 article.php	Просмотр одной статьи
 edit.php	Редактирование статьи
 functions.php	Библиотека функций
 index.php	Просмотр списка всех статей
 new.php	Создание новой статьи

Под каждую точку входа здесь выделено по одному файлу, а все полезные функции вынесены в отдельный файл. Это, действительно, удобно, так как functions.php теперь можно подключать к любой части нашей программы и использовать написанные в нём функции.

В более сложном проекте таких файлов с функциями могло бы быть несколько.

Отношения – это способ взаимодействия созданных компонентов между собой. В описанной выше структуре чётко прослеживается, какие файлы главные, а какой вспомогательный. `index.php`, `edit.php`, `new.php` и `article.php` – точки входа, именно в них строится основная логика работы страниц. А `functions.php` – просто библиотека функций, т.е. компонент, который подчиняется чужим приказам. Его задача - просто принять в точке входа команду на определённое действие и вернуть результат.

При создании структуры сразу продумываются связи между её компонентами.

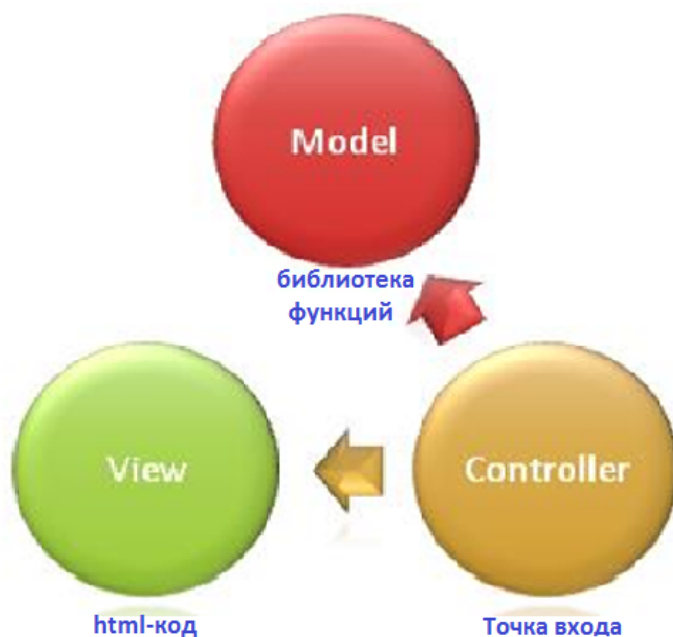
Значимые решения организации – это решения, направленные на организацию программы, а не на проблемы, решаемые с ее помощью. По сути, это просто продумывание первых двух пунктов архитектуры.

Не имея опыта разработки, очень сложно с ходу придумать хорошую архитектуру. Да и ни к чему. Существуют типовые архитектурные решения, которые называются шаблонами проектирования. С одним из них мы и познакомимся в этом уроке.

2. Архитектура MVC

Архитектура MVC является самым распространённым шаблоном проектирования веб-приложений. Её суть заключается в разделении всех компонент системы на три группы: контроллеры, модели и представления (или вьюшки, от англ. *view*). Отсюда и аббревиатура: Model-View-Controller.

Архитектура MVC может принимать различный вид в зависимости от языка программирования и решаемой задачи. Нас будет интересовать взаимодействие её компонент только применительно к веб-программированию. Схематично это можно изобразить следующим образом:



Разберём эти компоненты подробнее.

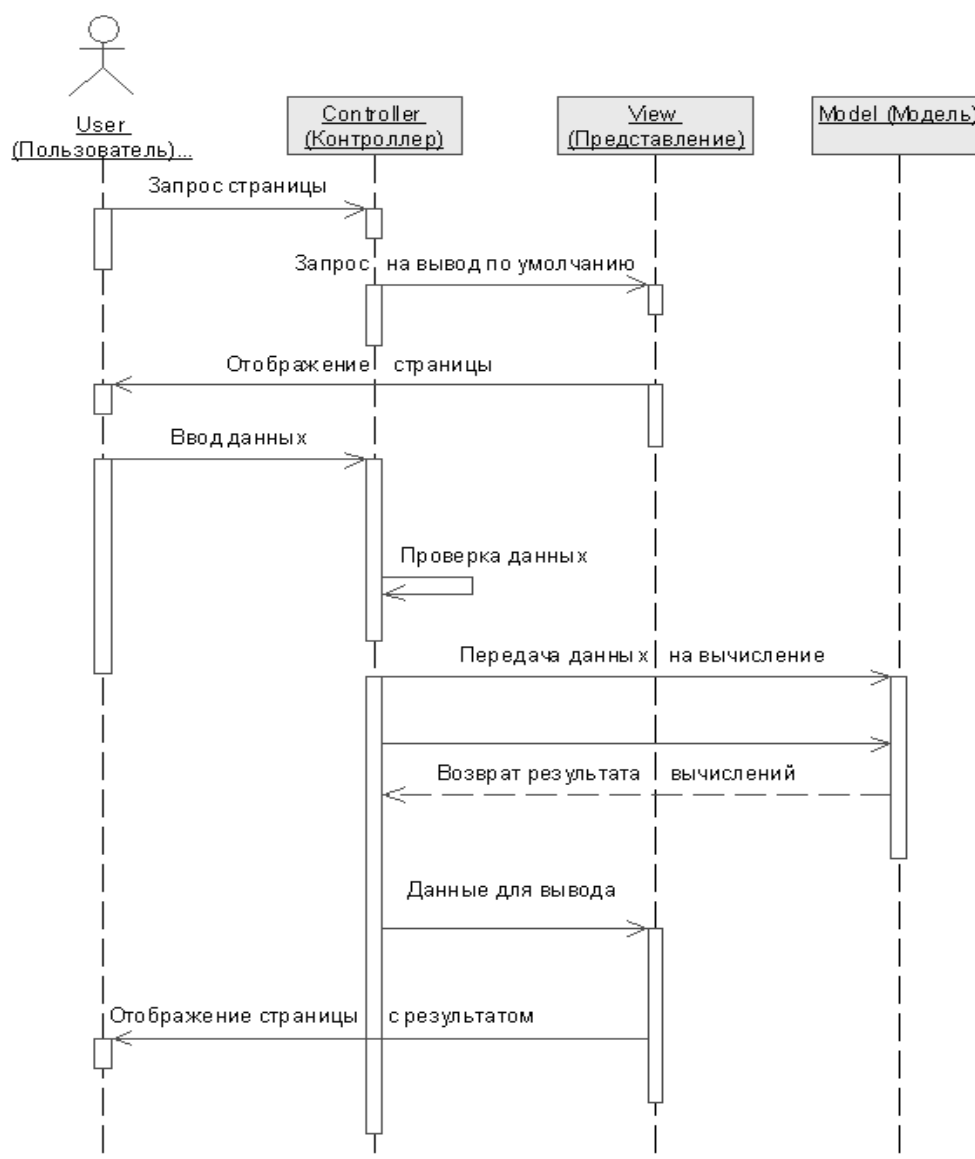
Контроллер принимает запрос, анализирует, что вообще хочет пользователь, и управляет логикой работы всей программы.

Модель – библиотека с функциями. Она изолирована от внешнего мира, т.е., сама не копается в массивах GET и POST и, что очень важно, ничего не выводит на экран. Задача модели – просто принять от контроллера команду и вернуть результат.

Представление – HTML-код с небольшими вставками PHP.

Зачем же введено такое разделение? Дело в том, что в более-менее сложных системах очень важно разделять основную логику работы и решение мелких попутных задач. Модель становится набором решений таких задач, и, благодаря этому, в контроллере остаются только те действия, которые определяют логику работы страницы. HTML-код вынесен в представления также с целью разгрузки контроллера от кода.

Ниже приведена диаграмма последовательности работы всех компонент:



Например, попробуем перенести на данную диаграмму логику работы страницы создания новой статьи. Пользователь может попасть на неё двумя способами: GET и POST. В первом случае он просто видит пустую HTML-форму, во втором – отправляет название и содержание новой статьи на сервер. Поэтому, первое, на что смотрит контроллер, – это метод запроса.

Схема работы при методе GET:

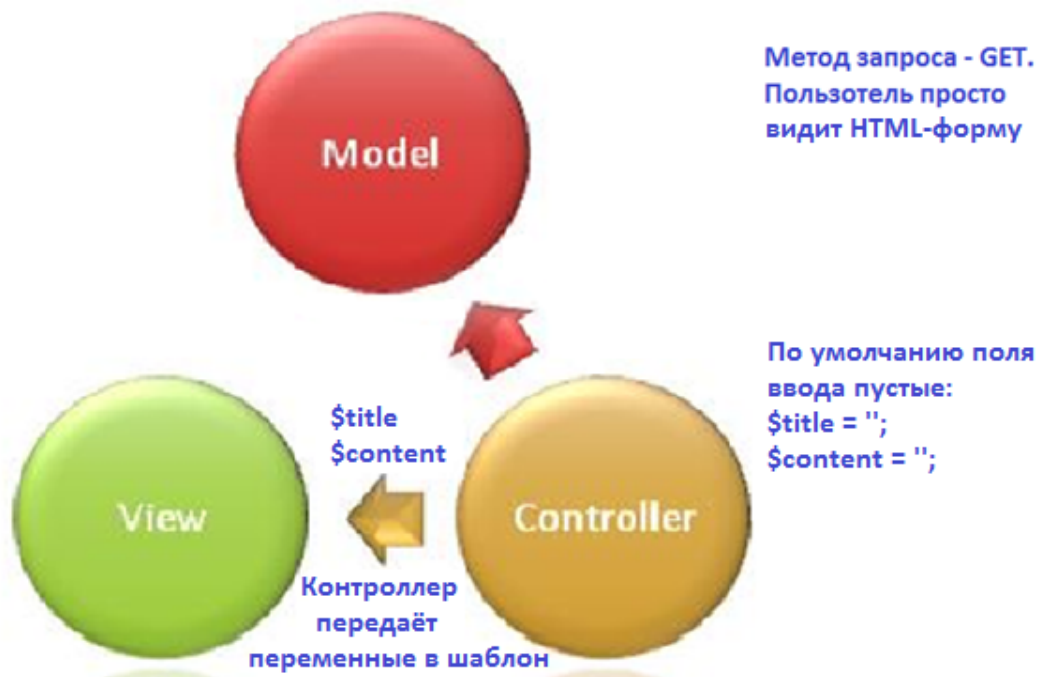
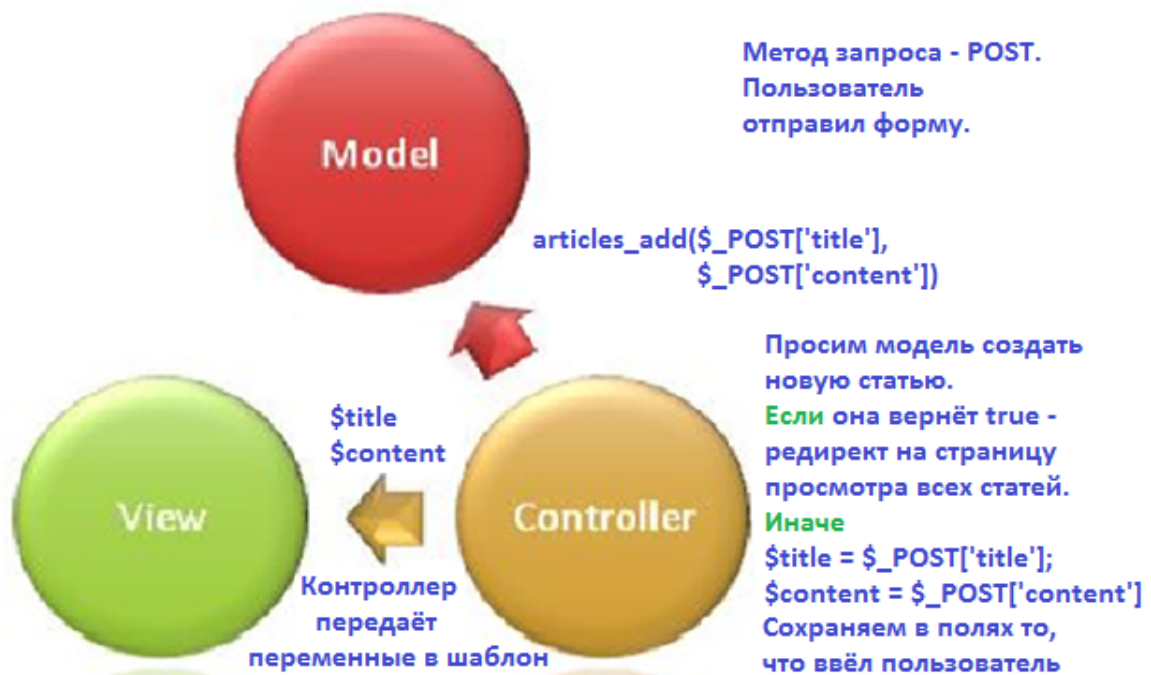


Схема запроса при методе POST:



Ещё раз поясним, зачем же введено данное дробление на компоненты. Помимо уменьшения строк кода в контроллере, данный приём позволяет разделить HTML и PHP-код. Это очень важно при командной разработке, когда программированием и вёрсткой могут заниматься разные люди.

Скачайте и изучите исходники к уроку. В них находится заготовка блога, которую Вам предстоит доделать.

Резюме

Теперь Вы знаете, как грамотно проектировать веб-приложения. Всегда уделяйте этому больше внимание, так как хорошо построенная архитектура всегда спасает от проблем при необходимости изменения или расширения кода.

Самоконтроль

- ✓ Что такое архитектура программы
- ✓ Зачем продумывать архитектуру
- ✓ Из каких основных частей состоит архитектура
- ✓ Что такое связи между компонентами
- ✓ Что такое шаблон проектирования
- ✓ За что отвечает контроллер
- ✓ За что отвечает модель
- ✓ За что отвечает представление
- ✓ Зачем введено такое разделение кода на части

Домашнее задание

Реализовать блог на основе выданных Вам исходников.