

РАЗРАБОТКА PHP-ФРЕЙМВОРКА

В.Я. Шабашов

Алтайский государственный технический университет им. И.И. Ползунова
г. Барнаул

Статья посвящена вопросам создания высокоэффективного легковесного PHP-фреймворка, предназначенного для быстрой разработки информационных систем.

Ключевые слова: PHP, фреймворк, Web-приложение, AJAX, JavaScript.

Разработка приложений с использованием PHP у нас в стране распространена достаточно широко, в том числе и для разработки информационных систем.

Разработка фрагментов информационных систем, которая была выполнена автором с целью изучения возможностей PHP показала, что трудоемкость такой разработки значительно выше чем при использовании технологии Windows Form, ASP.NET, JavaServer Faces.

Трудоемкость разработки определяется тем, что PHP не обеспечивает:

- автоматическую связь между переменными и свойствами элементов управления на форме;
- механизм создания и использования обработчиков событий;
- возможности визуального проектирования.

Применение библиотеки jQuery и технологии AJAX позволяют повысить эффективность разработки приложений PHP. Технология AJAX дает возможность создавать конструкции, аналогичные обработчикам событий, упрощает создание механизма связи между переменными и свойствами элементов управления на формах.

Несмотря на применение jQuery и AJAX производительность при создании информационных систем остается более низкой по сравнению с такими технологиями разработки как Windows Form, ASP.NET, JavaServer Faces.

Технология AJAX повышает эффективность создаваемых приложений, но практически не увеличивает эффективность разработки.

Различными разработчиками сделано много попыток усовершенствовать технологию разработки проектов PHP путем создания фреймворков [2, 3, 4].

Фреймворки создаются для того чтобы приложения имели некоторый единый стиль построения (каркас) и чтобы процесс разработки был более эффективным.

Большинство широко известных фреймворков в качестве каркаса используют технологию MVC (модель-вид-контроллер). Авторы этих фреймворков считают, что построение приложения на основе такого каркаса дает множество преимуществ и упрощает разработку.

Эти выводы ничем не обоснованы. Такой каркас дает определенную дисциплину разработки, но также создает для разработчика массу неудобств и усложняет решение некоторых весьма простых задач. Например, описание того, как разработать приложение типа Hello World с помощью обычного PHP, занимает один абзац текста. Описание разработки приложения Hello World с помощью фреймворков занимает по несколько страниц.

Разделение приложения на отдельные компоненты (модель-вид-контроллер) создает для разработчика неудобства и замедляет процесс разработки. Эта технология не стыкуется с такими общепринятыми в объектно-ориентированном программировании решениями, когда в одном объекте описываются и свойства, и методы (функции).

Эти недостатки частично ослабляются тем, что во фреймворках обычно имеются решения, не связанные с самой сутью каркаса, но которые упрощают процесс разработки. Например, во фреймворке Yii есть механизмы для создания меню, для отображения информации в виде таблиц, для создания и выполнения SQL запросов.

При использовании фреймворков существенно увеличивается размер приложения. Например, при создании простого приложения на базе Yii размер его составляет около 40 Мб. Это примерно в тысячу раз больше простых приложений PHP типа отображения

информации из некоторых таблиц по критериям поиска, выполнения операций CRUD (создание, обновление, удаление) для некоторой таблицы базы данных.

Фреймворки на основе технологии MVC лишают технологию PHP основного преимущества — гибкости. Это ведет к тому, что несмотря на большое количество различных фреймворков они редко используются при разработке информационных систем.

Фреймворк Sh, разработанный автором статьи, является высокоэффективным легковесным PHP-фреймворком, предназначенным для быстрой разработки веб-приложений ориентированных на обработку данных. Основной класс решаемых задач — информационные системы в экономике и бизнесе. Он позволяет существенно ускорить процесс разработки таких приложений. Поддерживается работа со следующими СУБД: MySQL, Microsoft SQL, PostgreSQL, Firebird

Приложение пишется почти полностью на языке **JavaScript**. Технология AJAX здесь реализована в самом концентрированном виде — обращения к Web серверу выполняются только для выполнения запросов к базе данных. При обращении к серверу за счет использования массивов с целыми индексами (не ассоциативными) и за счет использования формата JSON обеспечивается минимизация объема передаваемой по сети информации.

Доступ к данным выполняется с использованием унифицированных модулей, написанных на языке PHP. Основным входным параметром такого модуля является запрос, написанный на языке SQL. Для каждой СУБД создаются свои собственные модули. Эти модули являются едиными для всех приложений. В результате работы модуля формируется массив с данными. Структура этого массива является единой для всех СУБД. Обращение к модулям выполняется с помощью AJAX.

Имеется набор функций Sh, написанных на языке JavaScript, с помощью которых выполняются различные манипуляции с данными. Эти функции являются едиными для всех СУБД. Функции обращаются к модулям доступа, отображают данные в виде удобном для пользователя и выполняют операции добавления, корректировки и удаления данных. При манипуляции с данными используются скрипты, формируемые динамически функциями Sh.

Функции Sh предусматривают создание обработчиков событий, которые подключают-

ся на разных этапах работы с данными и могут дополнять и корректировать специфику выполнения операций по манипулированию данными.

Характерной особенностью является то, что пользователь пишет SQL запросы на получение информации из базы данных и задает их в качестве параметра при обращении к функциям Sh, а запросы на обновление базы данных формируются и выполняются автоматически.

Путем обращения к функциям Sh можно с помощью одного запроса получать данные из нескольких связанных таблиц базы данных, но операции по обновлению данных можно выполнять только с одной из них. Информацию, отображаемую в виде таблиц можно разбивать на страницы. При этом запрос к базе данных получает при каждом обращении только одну страницу. Это обеспечивает возможность работы с таблицами, которые содержат большое количество записей (протестирована работа с таблицей, содержащей 2,5 миллиона записей).

Порядок работы с данными определяет модель, которая формируется с помощью инструментария Sh. Модель — это объект языка JavaScript, который содержит свойства, описывающие структуру данных; свойства, определяющие вид отображаемых данных, а также действия, которые пользователь может с ними выполнять. Создается такая модель с помощью инструментария ShModelTools, который является Web приложением, входящим в состав фреймворка.

Для запуска приложения нужен модуль PHP, который выполняет обращение к модулю аутентификации и определяет контейнеры, в которых будут располагаться отображаемые данные. Такой модуль, как правило является очень легким (менее трех десятков строк) и формируется на основе шаблона, входящего в состав фреймворка. При разработке такого модуля обычно достаточно изменить в шаблоне всего 2–3 строки.

Этот модуль PHP обычно является корневым модулем приложения (index.php), но приложение Sh можно также использовать как компонент некоторого приложения PHP более высокого уровня, при этом другие его компоненты могут не являться приложениям Sh.

Фреймворк предполагает отображение данных с помощью Интернет-обозревателя, **поддерживающего HTML5**. Протестирован на обозревателях Google Chrome и Microsoft Edge.

Дистрибутив фреймворка Sh включает следующие компоненты:

- модули `php`, обеспечивающие аутентификацию и выполнение запросов по доступу к данным;
- модуль, содержащий функции JavaScript, с помощью которых выполняются различные действия по манипулированию данными (отображение, изменение, добавление, удаление);
- инструментарий для проектирования модели данных;
- шаблон для создания `php` приложения с использованием фреймворка Sh.

В состав первого компонента входят модули `JsonQuery`, `JsonNonQuery` и `Authentication`, которые выполняют операции чтения данных, выполнения запросов по изменению данных и аутентификации пользователей. Эти модули являются едиными для всех задач. Для каждой СУБД существует своя группа модулей, так как каждая СУБД имеет свою специфику работы с данными. В настоящее время такие модули созданы для следующих СУБД: MySQL, MS SQL, PostgreSQL, Firebird.

При обращении к модулям `JsonQuery` и `JsonNonQuery` задаются определенные параметры. Основным параметром является строка, содержащая SQL запрос на выполнение операции с базой данных. Результаты запроса – это массив с цифровыми индексами, представленный в виде JSON документа. Массив имеет трехуровневую структуру – элементами массива являются другие массивы и также с цифровыми индексами. Модуль аутентификации обеспечивает получение от пользователя реквизитов доступа (имя и пароль), которые используются при обращении к базе данных. Этот модуль может заменяться разработчиком приложения на собственный модуль, который определяет реквизиты доступа (имя и пароль).

Вторым компонентом системы является модуль `ShTables.js`, который содержит функции JavaScript, с помощью которых выполняются различные действия по манипулированию данными (отображение, изменение, добавление, удаление).

Для того, чтобы воспользоваться этими функциями необходимо создать модель, которая содержит описание данных и порядок работы с этими данными. Модель, представляет собой объект, который содержит самую разнообразную информацию (названия таблиц базы данных, имена полей, типы данных, определение первичного ключа, название

столбцов таблицы, порядок работы с внешними ключами и др.).

Для создания такого объекта используется третий компонент, который на основании введенного SQL запроса извлекает из базы данных различную метainформацию (имена полей, типы данных, размерность полей, сведения о первичном ключе и т.д.) и позволяют пользователю в удобном виде описать порядок работы с данными). Если речь идет просто об отображении данных, то действия разработчика приложения сводятся только ко вводу SQL запроса. Все элементы модели формируются автоматически.

Каким образом создается приложение.

Разработчик создает `php` модуль, который содержит контейнеры, в которых будет отображаться различная информация (в шаблоне имеется заготовка для написания такого модуля). Здесь должно выполняться обращение к модулю аутентификации, который определяет имя пользователя и пароль, используемые для доступа к базе данных. В качестве контейнера может выступать любой `html` элемент, в который можно занести таблицу `table`, например, `div`, `td`, `p` и т.д.

Модуль аутентификации поставляется в составе дистрибутива. Если используемый в нем порядок аутентификации не устраивает, то пользователь может заменить его на собственный модуль аутентификации.

Функции, которые используются для создания приложения делятся на три группы:

- функции для отображения данных в табличном виде;
- функции для работы с критериями поиска;
- прочие функции.

Рассмотрим кратко эти функции.

Для отображения данных в табличном виде используются функции: `DbToTable`, `DbToRoll` и `DbToList`, `PhpToTable`, `JsFuncToTable`, `JsArrayToTable`.

Функция **`DbToTable`** дает возможность отобразить в виде таблицы информацию из базы данных, и реализует операции CRUD для этой информации (создание, обновление и удаление). Основными параметрами при обращении к этой таблице являются: SQL запрос, модель, порядок разбиения на страницы, контейнеры в которых будут выполняться операции CRUD.

Функция **`DbToRoll`** похожа на предыдущую, но в отличие от нее отображает информацию в виде рулона. Каждое поле таблицы базы данных отображается в виде строки таблицы. В этой строке содержится только

две колонки: название поля и его содержимое. Это дает возможность отобразить таблицы с большим количеством полей на небольшом экране.

Функция **DbToList** предназначена для отображения информации из базы данных в виде таблицы и для выполнения операций обновления данных. В отличие от **DbToTable** информация выдается в виде листов, напоминающих лист Excel. Другим важным отличием от **DbToTable** является то, что информация заносится в базу данных после выполнения действий по всей странице при нажатии на кнопку "Сохранить". Это ведет к более эффективной работе – за один запрос к Web серверу обновляются все строки, в которые были внесены изменения. Эта функция не позволяет добавлять в таблицу новые записи.

Функция **PhpToTable** предназначена для отображения информации, сформированной **php** модулем, не входящем в состав фреймворка **Sh**. Этот модуль должен формировать данные в соответствии требованиями, определяемыми системой **Sh**. Если пользователь предполагает поддержку постраничного режима выдачи данных, то модуль должен выдавать сведения порциями. Операции **CRUD** эта функция не поддерживает (только отображение данных).

Функция **JsFuncToTable** предназначена для отображения информации сформированной функцией пользователя, написанной на JavaScript. Такая функция должна формировать данные в соответствии требованиями, определяемыми системой **Sh**. Если пользователю необходима поддержка постраничного режима выдачи данных, то эта функция должна выдавать сведения порциями. Операции **CRUD** эта функция не поддерживает.

Функция **JsArrayToTable** предназначена для отображения информации сформированной в виде массива JavaScript. Массив должен быть создан в соответствии требованиями, определяемыми системой **Sh**. Поддерживается постраничный режим отображения данных, разбиение массива на страницы выполняет сама функция – выдача данных порциями здесь не требуется. Операции **CRUD** эта функция не поддерживает.

Очень часто при получении информации из базы данных необходимо считывать не все записи, а только те, которые удовлетворяют **определенным критериям поиска**. Если критерий не является обычным текстовым полем, а является показателем, выбираемым из справочника, то для работы с

таким справочником можно использовать функции для работы с критериями поиска: **SearchCritBut**, **SearchCritSelect** и **SearchCritFunc**.

Функция **SearchCritBut** предназначена для отображения критерия поиска в виде элемента **input** рядом с которым расположена кнопка для открытия справочника. При нажатии на такую кнопку считывается справочник, отображается в виде таблицы и формируются обработчики событий, обеспечивающие обработку действий пользователя при выборе строки справочника.

Функция **SearchCritSelect** предназначена для отображения критерия поиска в виде элемента **select**. Этот элемент позволяет отображать таблицу-справочник, из которого можно выбрать необходимую строку. При выборе строки в поле **input**, определенное пользователем заносится ключ выбранной строки.

Функция **SearchCritFunc** обычно используется в том случае, если справочник не является простой таблицей, состоящей из двух полей (ключа и названия), а также если требуется создавать фильтры для поиска данных в справочниках большого объема. Она предназначена для отображения критерия поиска в виде элемента **input** рядом с которым расположена кнопка для открытия справочника. В отличие от **SearchCritBut** выбор из справочника выполняется с помощью специально написанной функции. Эта функция может использовать для отображения справочника любые функции **Sh** кроме **DbToRoll**. При нажатии на кнопку, расположенную рядом с элементом **input**, выполняется обращение к функции. Автоматически формируются обработчики событий, обеспечивающие обработку действий пользователя при выборе строки справочника или при отмене.

Кроме этого имеется ряд вспомогательных функций (**SelectScalar**, **GetExternValue**, **DbPrefix**), которые обычно используются при написании обработчиков событий для функций **Sh**.

Предлагаемый фреймворк позволяет существенно повысить скорость разработки программного обеспечения информационных систем и повысить его надежность за счет использования готовых блоков и функций.

СПИСОК ЛИТЕРАТУРЫ

1. Шабашов, В. Я. Учебное пособие по дисциплине "Web- программирование" Создание информационных систем с помощью фреймворка **Sh** / В. Я. Шабашов ; Алт. гос. техн. ун-т им. И. И. Ползунова, Кафедра ИСЭ. – Барнаул : Изд-во АлтГТУ,

2016. – 74с. Режим доступа:
http://new.elib.altstu.ru/eum/download/ise/Shabashov_sh.pdf.

2. The Definitive Guide to Yii 2 [Электронный ресурс] / Режим доступа:
<http://www.yiiframework.com/doc-2.0/guide-index.html>.

3. Symfony Documentation [Электронный ресурс] / Режим доступа:
<https://symfony.com/doc/current/index.htm>.

4. Zend Framework Tutorials [Электронный ресурс] / Режим доступа:
<https://docs.zendframework.com/tutorials/>

5. One Framework. Mobile & Desktop [Электронный ресурс] / Режим доступа: <https://angular.io>.

Шабашов Владимир Яковлевич – к.т.н. кафедра ИСЭ, тел.: (3852) 29-08-70, e-mail: vl.shabashov@mail.ru.