МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Озерский технологический институт — филиал НИЯУ МИФИ

Кафедра прикладной математики

Вл. Пономарев

ПРАКТИКУМ

по программированию web-приложений

Учебно-методическое пособие

Часть 2. Введение в PHP

2018 г.

УДК 681.3.06
П 56

Вл. Пономарев. Практикум по программированию web-приложений. Учебно-методическое пособие. Часть 2. Введение в PHP. Озерск: ОТИ НИЯУ МИФИ, 2018. — 8 с.

В пособии описываются задания практических работ по дисциплине «Современные технологии программирования». Работы третьего семестра изучения дисциплины включают в себя основы программирования web-приложений с использованием языков программирования JavaScript, PHP, XPATH, XSLT и объектной модели документа DOM.

В качестве основного материала при выполнении практических работ пособие предназначено для студентов, обучающихся по направлению подготовки 09.03.01 «Информатика и вычислительная техника» и специальности 09.05.01 «Применение и эксплуатация автоматизированных систем специального назначения».

Рецензенты:
1.
2.

УТВЕРЖДЕНО
Редакционно-издательским
Советом ОТИ НИЯУ МИФИ

Содержание

Общие цели занятий

В ходе практических работ изучается использование языков программирования JavaScript и PHP для формирования интерактивных web-страниц и web-приложений в целом.

Целью работ является создание приложения, моделирующего сетевую игру под названием «Королевский квадрат». Весь процесс создания этого приложения условно разбит на три логические части.

При этом первая часть работ в большей степени посвящена разработке алгоритмов игры на языке JavaScript в виде автономной страницы HTML, которая представляет собой законченное приложение.

Во второй части приложение переносится на сервер, и изучается взаимодействие серверной и клиентской частей кода с использованием языка PHP, а также технологий, связанных с представлением и обработкой информации, представленной в виде документа XML.

В третьей части основной задачей является исследование принципов интерактивной связи сервера с клиентом.

На выполнение каждой работы предположительно отводится от 2-х до 4-х академических часов. Успешное усвоение изучаемого материала возможно при условии, что студент самостоятельно находит необходимую техническую информацию, используя сеть Интернет, конспекты лекций и дополнительную литературу.

1. Работа PHP-01. Формы и альтернативный синтаксис (2 часа)

Цели:
- изучение взаимодействия форм HTML со сценарием PHP.
Задачи:
- передача информации от форм HTML сценарию PHP;
- изучение альтернативного синтаксиса.
Опорные документы:

1.1. Пакет разработчика

Сначала нужно установить пакет разработчика Denwer.

Скачайте с сайта преподавателя и установите этот пакет.

На диске C: появится папка C:\WWW, на рабочем столе появятся ярлыки для запуска и остановки пакета. Запустите пакет.

Откройте браузер, новую вкладку, и введите адрес "localhost". Вероятно, браузер покажет какую-то страницу. Тогда эта страница находится в каталоге C:\WWW\home\localhost\www\.

Создайте в этом каталоге каталог test.

Скачайте с сайта преподавателя шаблон страницы.

Скопируйте из шаблона файлы page.css и html.php в каталог test.

Скопируйте файл html.php в index.php.

Измените строку адреса в браузере на "localhost/test".

Убедитесь, что отображается страница шаблона.

1.2. Формы HTML

Формы передают информацию серверу либо непосредственно в командной строке, либо в теле запроса. В файле index.php сформируем следующую форму:

```
<body>
  <form method='GET' action='page1.php'>
    <input type='text' name='user' value='Петя' />
    <input type='submit' name='send' value='Послать' />
    <input type='hidden' name='time' value='<?=time()?>' />
  </form>
</body>
```

Обновите страницу в браузере.

Убедитесь, что на странице появилось поле и кнопка.

Убедитесь, что в поле отображается «Петя», а не что-то иное.

Если в поле не отображается «Петя», значит, что-то настроено не так.

Прежде всего убедитесь, что файл index.php находится в кодировке utf-8, как в нем указано.

Если это так, то может помочь локальный файл ".htaccess" (у файла нет расширения, в начале имени файла точка) следующего содержания:

```
AddDefaultCharset utf-8
```

Если и это не помогает, требуется настройка файла конфигурации httpd.conf сервера apache.

Предположим, что все хорошо, тогда продолжаем.

В форме указан адрес назначения page1.php.

Скопируйте шаблон html.php в файл page1.php.

Текст тела файла page1.php следующий:

```
<body>
  $_SERVER:<br><? print_r($_SERVER); ?><br>
  $_REQUEST:<br><? print_r($_REQUEST); ?><br>
  $_POST:<br><? print_r($_POST); ?>
</body>
```

Сохраним файл.

Функция print_r выводит содержание массива, она заключена в скобки **<??>**, обозначающие код PHP. Переменная PHP с именем $_SERVER содержит разную информации о запросе, который получен со стороны клиента. Переменная с именем $_REQUEST содержит данные, посылаемые формой методом GET. Переменная с именем $_POST содержит данные, посылаемые формой методом POST.

Отправим форму, нажав кнопку «Послать».

В результате загрузится страница page1.php, интерпретатор PHP выведет на страницу информацию переменных. Изучите эту информацию.

Обратим внимание на строку адреса браузера:

```
http://localhost/test/page1.php?user=Петя&send=Послать&time=000
```

Здесь цифрами 000 обозначено некоторое число, временная метка, возвращаемая функцией time. В строке запроса передаются все элементы формы в формате "имя=значение", где имя — это атрибут name элемента формы, а значение — значение данного элемента. Заметим, что открытым текстом посылаются также элементы, тип которых "hidden" (скрытые).

Информация командной строки попадает на стороне сервера в переменную PHP с именем $_REQUEST, убедитесь в этом.

Теперь изменим метод формы с GET на POST (файл index.php):

```
<form method='POST' action='page1.php'>
```

Снова введем в строке адреса браузера "localhost/test".

Обновим страницу.

Нажмем кнопку «Послать».

Изучим содержание массивов $_REQUEST и $_POST.

Убедимся, что в строке адреса в браузере параметры формы не видны.

Параметры, посылаемые методом GET и POST, можно сочетать.

Для этого еще раз изменим форму в файле index.php:

```
<form method='POST' action='page1.php?abc=xyz&xyz=abc'>
```

Снова введем в строке адреса браузера "localhost/test".

Обновим страницу. Нажмем кнопку «Послать».

Изучим содержание массивов $_REQUEST и $_POST.

Изучим строку адреса браузера.

1.3. Альтернативный синтаксис

При помощи альтернативного синтаксиса и исследования посылаемых формой значений можно совместить два файла в одном, а именно, файлы index.php и page1.php. Альтернативный синтаксис позволит сократить код PHP до минимума, оставляя главным кодом страницы HTML. Исследование значений переменных позволит определить, является ли страница начальной, то есть index.php, или страницей ответа, то есть page1.php.

Разделим страницу на части, соответствующие страницам index.php и page1.php, используя тот факт, что если страница загружена впервые, то не определены переменные, посылаемые формой.

Начальное разбиение выглядит тогда так (файл index.php):

```
<body>
<?if($_REQUEST['user']):?>
<?else:?>
<?endif?>
</body>
```

Мы используем переменную "user", которая при первоначальном открытии страницы не определена. Тогда за второй строкой приведенного кода начинается часть кода HTML, соответствующая странице page1.php, а за строкой, содержащей else, начинается часть кода HTML, соответствующая странице index.php. Остается только вписать код HTML:

```
<body>
<?if($_REQUEST['user']):?>
  <P>Ваше имя <?=$_REQUEST['user']?>.
<?else:?>
  сюда вписываем форму
<?endif?>
</body>
```

Кроме того, нужно изменить получателя формы, например, так:

```
<form method='GET' action=''>
```

Сохраним файл index.php.

Снова введем в строке адреса браузера "localhost/test".

Обновим страницу.

Нажмем кнопку «Послать».

Убедимся, что на странице выводится имя «Петя».

Заметим, что конструкция `<?=значение?>` позволяет выводить в код HTML страницы любое значение, вычисляемое PHP.

1.4. Контрольные вопросы и упражнения

1. Назовите возможные элементы формы (изучите вопрос в Интернет).
2. В чем разница между методами формы GET и POST?
3. В какие массивы PHP передаются значения, посылаемые формами?
4. Как можно узнать IP-адрес отправителя запроса?
5. В чем заключается альтернативный синтаксис операторов PHP?
6. Как в код HTML вставляется значение, вычисляемое PHP?
7. Создайте три страницы, page1.php, page2.php и page3.php. Каждая страница должна выводить свое название, чтобы ее можно было отличить от других страниц. Каждая страница должна иметь гиперссылку на две другие страницы. Теперь пусть каждая страница выводит также счетчик обращений к ней. Для реализации этого задания формы не нужны.

Введите в строку адреса браузера путь к первой странице. Щелкайте на гиперссылки и убедитесь в том, что страницы подсчитывают обращения к ним.

8. Создайте каталог C:\WWW\\home\localhost\www\ksg\.

Поместите в этот каталог файлы index.css и index.html, которые были разработаны в серии работ по JavaScript.

Введите в строку адреса браузера "localhost/ksg", убедитесь, что страница выводится.

Переименуйте страницу index.html в index.php. Убедитесь, что страница выводится.

Используя альтернативный синтаксис, сократите код HTML. Для этого есть предпосылки. В странице много повторяющихся частей, формирующих таблицу игрового поля и таблицу букв алфавита. Выделите в коде HTML повторяющиеся части и оформите их как часть итерации оператора for в альтернативном синтаксисе.