

Урок №4

Запросы HTTP, параметры URL и формы HTML.

В этом уроке вы:

- узнаете, какие бывают типы запросов HTTP
- узнаете, что такое URL и параметры запроса
- научитесь обрабатывать параметры URL
- научитесь обрабатывать отправку HTML форм

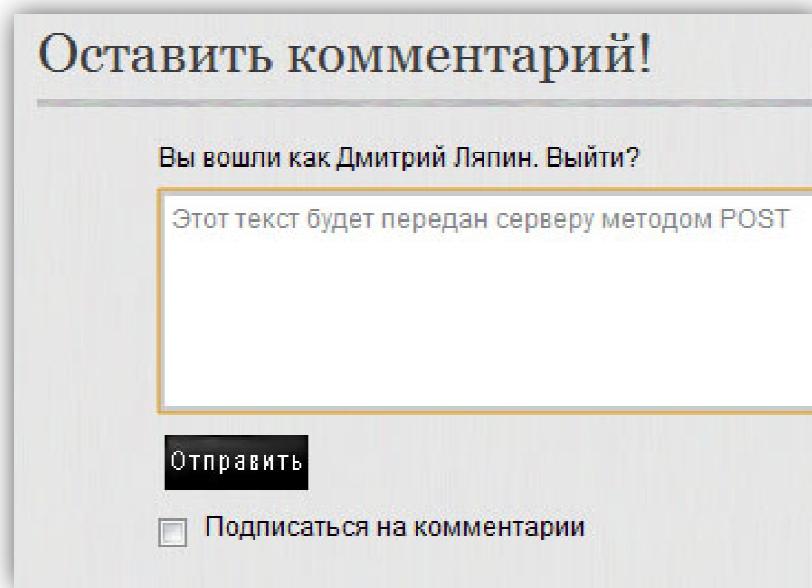
1. Типы запросов HTTP

HTTP (Hyper Text Transport Protocol) – тот самый язык, на котором разговаривают браузеры с веб-серверами, важнейший протокол сети Интернет.

Запросы можно разделить на два вида:

1. GET
2. POST

GET используется, при наборе адреса сайта в строке браузера или перехода по ссылке. POST служит для отправки форм, например, при регистрации на сайте, публикации комментария к статье. Для отправки формы обычно нужно нажать кнопку "Отправить" или наподобие того.



Для простоты понимания различие можно представлять так:

- GET используется для чтения сайтов (читаем Интернет)
- POST используется для публикации информации на сайтах (пишем Интернет)

2. URL и параметры запроса

В обоих случаях требуется URL (Uniform Resource Locator) запрашиваемого документа. URL - это адрес страницы в Интернет.

Как правило, он имеет такой вид:

```
http://<хост>/<путь>
```

Например:

```
http://www.example.ru/about.php
```

Или же такой, если необходимо передать параметры скрипту:

```
http://<хост>/<путь>?<параметры>
```

где <параметры> - это набор пар вида:

<имя>=<значение>

разделенных символом &.

Например:

```
http://www.example.ru/news.php?id=100&show_comments=yes
```

"А для чего скрипту передавать параметры?" - спросит пытливый читатель. Динамическая страница (она же скрипт), в отличие от статической, может выдавать различную информацию. Например, скрипт новостной ленты отображает либо список анонсов последних новостей, либо целиком текст конкретной статьи. Что именно хочет увидеть пользователь, скрипт понимает, исходя из переданных ему параметров.

Это могло бы работать следующим образом. Получение списка последних новостей:

```
http://www.example.ru/news.php
```

(URL без параметров)

Получение полного текста новостной статьи:

```
http://www.example.ru/news.php?id=1
```

(URL включает в качестве параметра номер новости)

3. Обработка параметров URL

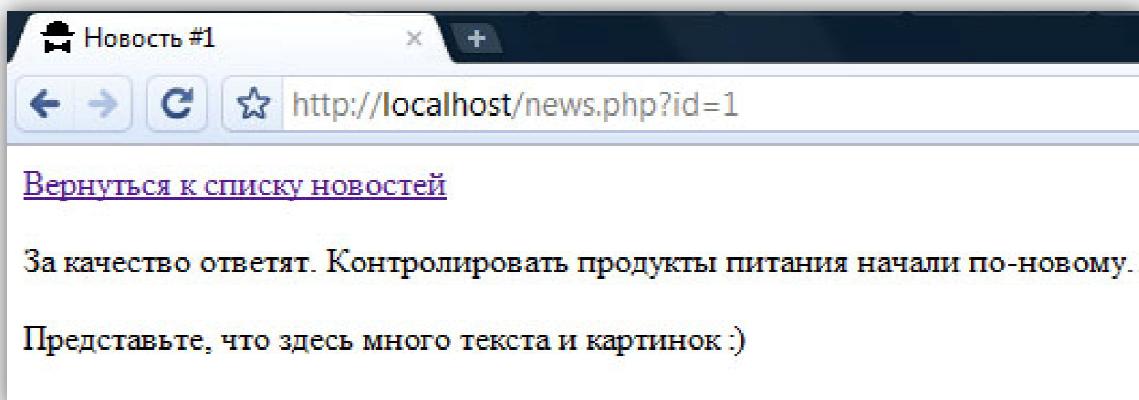
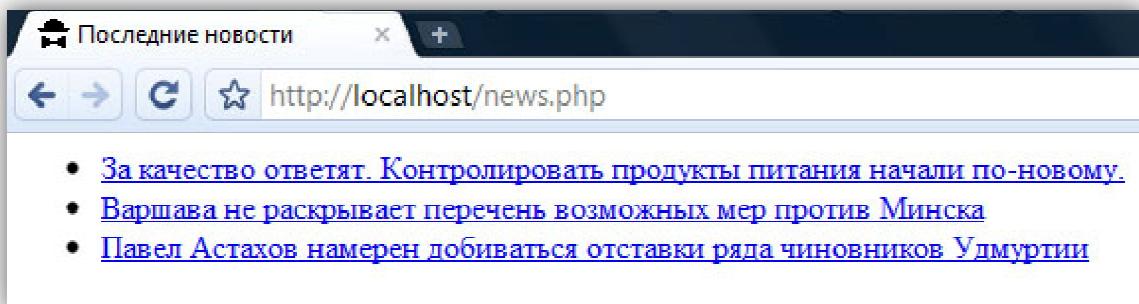
А сейчас мы напишем скрипт этой самой новостной ленты.

У нее будет два режима:

1. показывать список всех новостей (если нет параметров)
2. показывать текст конкретной новости (если ее номер передан в качестве параметра)

Всего новостей у нас будет три штуки:

1. "За качество ответят. Контролировать продукты питания начали по-новому."
2. "Варшава не раскрывает перечень возможных мер против Минска"
3. "Павел Астахов намерен добиваться отставки ряда чиновников Удмуртии"



Внимание! Пример упрощен. Никто никогда не хранит новости в коде скрипта. Хранить подобную информацию следует в базе данных. Но это предмет совсем другого урока! Сейчас же нам важно научиться обрабатывать параметры, переданные через URL.

Создайте файл news.php:

```
<?php
// Функция вывода всего списка новостей.
function show_list($news)
{
    echo '<html>';
    echo '<head>';
    echo '<title>Последние новости</title>';
    echo '</head>';
    echo '<body>';
    echo '<ul>';

    for ($i = 0; $i < count($news); $i++)
    {
        echo '<li>';
        echo '<a href="news.php?id=' . ($i + 1) . '">';
        echo $news[$i];
        echo '</a>';
        echo '</li>';
    }

    echo '</ul>';
    echo '</body>';
    echo '</html>';
}

// Функция вывода конкретной новости.
function show_item($news, $id)
{
    echo '<html>';
    echo '<head>';
    echo "<title>Новость #$id</title>";
```

```

echo '</head>';
echo '<body>';
echo '<a href="news.php">Вернуться к списку новостей</a>';
echo '<p>';
echo $news[$id - 1];
echo '</p>';
echo '<p>';
echo 'Представьте, что здесь много текста и картинок :)';
echo '</p>';
echo '</body>';
echo '</html>';

}

// Точка входа.

// Создаем массив новостей.
$news = array();
$news[0] = 'За качество ответят. Контролировать продукты питания начали по-новому.';
$news[1] = 'Варшава не раскрывает перечень возможных мер против Минска';
$news[2] = 'Павел Астахов намерен добиваться отставки ряда чиновников Удмуртии';

// Был ли передан id новости в качестве параметра?
if (isset($_GET['id']))
{
    show_item($news, $_GET['id']);
}
else
{
    show_list($news);
}
?>

```

Теперь подробно разберем, что же мы написали.

Вначале объявляем две функции, которые будут генерировать HTML. Первая отображает список новостей, вторая - текст конкретной новости. Управление будет передано в эти функции только тогда, когда мы их вызовем. Вернемся к ним позже.

Выполнение скрипта начинается с того места, где комментарием помечена точка входа. Мы создаем массив, состоящий из трех новостей.

Напоминаю! Нумерация элементов в массиве начинается с нуля.

Далее проверяем, был ли передан id новости в качестве параметра. Параметры, переданные через URL, хранятся в системной переменной `$_GET`. Она представляет собой ассоциативный массив (или, по-другому, словарь).

Напоминаю! Ассоциативный массив (или словарь) - это такая структура данных, которая содержит пары ключ-значение.

Ключи словаря `$_GET` - это имена параметров. Функция `isset()` возвращает `true`, если переменная определена. Таким образом,

```
if (isset($_GET['id']))
```

следует читать: "если URL запроса содержит параметр `id`".

Теперь возвращаемся к функциям. Здесь все просто, но я хотел бы обратить внимание на два момента. Во-первых, может быть не совсем понятно, для чего мы в одном месте прибавляем к `$i` единицу, а в другом вычитаем. Сделано это для того, чтобы пользователь

видел URL первой новости так: "news.php?id=1", а не "news.php?id=0". Это хороший тон и не более того.

Во-вторых, обратите внимание на строчку:

```
echo "<title>Новость #$id</title>";
```

Двойные кавычки отличаются от одинарных тем, что если внутри них встречаются имена переменных (со знаком \$), то они заменяются на значения этих самых переменных. Стока в одинарных кавычках остается как есть.

4. Обработка отправки HTML формы

В предыдущем примере мы отправляли запросы методом GET. Теперь познакомимся ближе с методом POST и формами HTML. Для этого напишем крайне полезный скрипт - сумматор. Он будет складывать числа.

Создайте файл sum.php:

```
<?php
// Функция вывода формы ввода.
function show_form()
{
    echo '<html>';
    echo '<head>';
    echo '<title>Сумматор</title>';
    echo '</head>';
    echo '<body>';
    echo '<form action="sum.php" method="post">';
    echo '<input type="text" name="a" />';
    echo '+';
    echo '<input type="text" name="b" />';
    echo '<input type="submit" value="=" />';
    echo '</form>';
    echo '</body>';
    echo '</html>';
}

// Функция вывода результата.
function show_result($a, $b)
{
    $result = $a + $b;

    echo '<html>';
    echo '<head>';
    echo '<title>Сумматор</title>';
    echo '</head>';
    echo '<body>';
    echo '<p>';
    echo "$a + $b = <b>$result</b>";
    echo '</p>';
    echo '<p>';
    echo '<a href="sum.php">Хочу суммировать еще</a>';
    echo '</p>';
    echo '</body>';
    echo '</html>';
}

// Точка входа.

// Показываем результат операции или форму ввода.
if (isset($_POST['a']) && isset($_POST['b']))
```

```

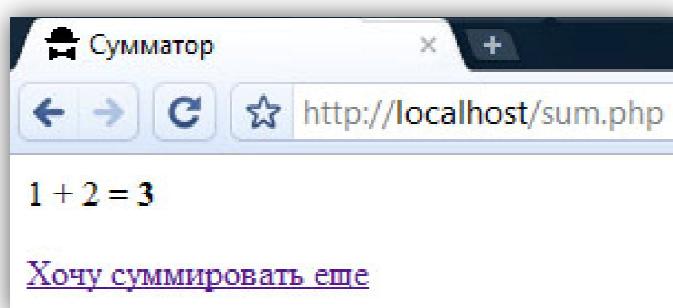
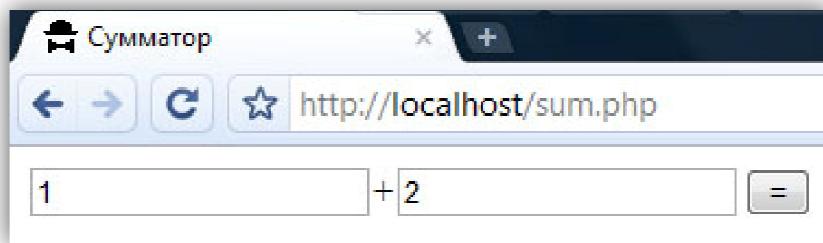
{
    show_result($_POST['a'], $_POST['b']);
}
else
{
    show_form();
}
?>

```

Принцип работы скрипта похож на новостную ленту.

Тут также два режима:

1. режим ввода параметров операции сложения
2. режим показа результата



Форму ввода генерирует функция `show_form()`. Вот ее HTML код:

```

<form action="sum.php" method="post">
    <input type="text" name="a" />
    +
    <input type="text" name="b" />
    <input type="submit" value="=" />
</form>

```

Тег `input` бывает различных типов (что определяется атрибутом `type`). Нам нужно два текстовых поля (`type="text"`) и одна кнопка (`type="submit"`). Текстовым полям мы дали имена ("a" и "b"), чтобы из скрипта получить введенные туда пользователем значения. `input` должен обязательно находиться внутри `form`. У тега `form` мы указали адрес скрипта обработки отправки формы и метод передачи данных. Вместо "post" тут можно было бы указать "get", тогда параметры и значения передались бы через URL (как правило, используют все же "post").

Параметры, переданные методом POST, содержатся в системной переменной `$_POST`. Это так же, как и `$_GET`, ассоциативный массив (словарь). Наш скрипт проверяет, были ли переданы методом POST аргументы операции сложения. Если да, пользователю возвращается результат, иначе - форма ввода.

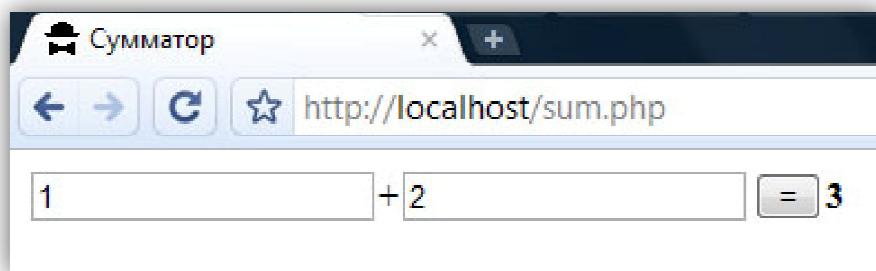
Резюме

В этом уроке мы, наконец, перешли от скучной теории к долгожданной практике! Теперь вы можете писать боевые скрипты, обрабатывающие настоящие HTTP запросы.

Теперь вы понимаете разницу между GET и POST, знаете, что такое URL и как передать через него параметры запросу. И самое главное, что вы знаете, как обработать эти параметры и как обработать отправку формы.

Домашнее задание

1. Обязательно сделайте скрипты, приведенные в качестве примеров в этом уроке.
2. Измените сумматор таким образом, чтобы два режима его отображения объединить в один. Вот как это должно выглядеть:



3. Превратите получившийся сумматор в калькулятор с четырьмя операциями: сложение, вычитание, умножение, деление. Не забудьте обработать деление на ноль!