**PHP. Основы**

## Обработка формы

PHP Суперглобальные переменные $\_GET и $\_POST ( и $\_REQUEST) используются для сбора данных формы.

<form action="welcome\_get.html" method="get">

Name: <input type="text" name="name"><br>

E-mail: <input type="text" name="email"><br>

<input type="submit">

</form>

В welcome\_get.html:

Welcome <?php echo $\_GET["name"]; ?><br>

Your email address is: <?php echo $\_GET["email"]; ?>

И GET, и POST создают массив (например, array( key1 => value1, key2 => value2, key3 => value3, ...)). Этот массив содержит пары ключ / значение, где ключи (key) - это имена элементов управления формы, а значения (value) - входные данные пользователя. $\_GET - это массив переменных, переданных в текущий скрипт через параметры URL. $\_POST - это массив переменных, передаваемых текущему скрипту с помощью HTTP-метода POST.

GET также имеет ограничения на количество информации для отправки. Ограничение составляет около 2000 символов. Однако, поскольку переменные отображаются в URL-адресе, можно добавить страницу в закладки. Это может быть полезно в некоторых случаях. Примечание: GET никогда не должен использоваться для отправки паролей или другой конфиденциальной информации!

Информация, отправляемая из формы методом POST, невидима для других (все имена / значения встроены в тело HTTP-запроса) и не имеет никаких ограничений на количество отправляемой информации. Кроме того, POST поддерживает расширенные функции, такие как поддержка двоичного ввода из нескольких частей при загрузке файлов на сервер. Однако, поскольку переменные не отображаются в URL-адресе, невозможно добавить страницу в закладки.

## Проверка формы

Правила валидации (проверки) для формы выше:

|  |  |
| --- | --- |
| **Поле** | **Правила валидации** |
| Имя | Обязательно. + Должно содержать только буквы и пробелы |
| E-mail | Обязательно. + Должно содержать действительный адрес электронной почты (с @ and .) |
| Веб-сайт | Необязательно. Если присутствует, то должно содержать действительный URL |
| Комментарий | Необязательно. Многострочное поле ввода (текстовое поле) |
| Пол | Обязательно. Необходимо выбрать только один вариант |

HTML-код формы выглядит следующим образом:

<form method="post" action="<?php echo htmlspecialchars($\_SERVER["PHP\_SELF"]);?>">

где $\_SERVER["PHP\_SELF"] является суперглобальной переменной, которая возвращает имя исполняемого в данный момент скрипта. Таким образом, $\_SERVER["PHP\_SELF"] отправляет отправленные данные формы на саму страницу, вместо того, чтобы переходить на другую страницу. Таким образом, пользователь будет получать сообщения об ошибках на той же странице, где и форма.

Функция htmlspecialchars() преобразует специальные символы в объекты HTML. Это означает, что он заменит HTML символы, такие как < и > с помощью &lt; и &gt;. Это предотвращает использование кода злоумышленниками путем внедрения кода HTML или Javascript (межсайтовый скриптинг) в формы.

**Проверка:**

1. Первое, что мы сделаем, это передадим все переменные через PHP функцию htmlspecialchars().
2. Убрать ненужные символы (лишний пробел, табуляцию, символ новой строки) из введенных пользователем данных (с помощью PHP функции trim())
3. Удалить обратную косую черту (\) из введенных пользователем данных (с помощью PHP функции stripslashes())

// определить переменные и установить пустые значения

$name = $email = $gender = $comment = $website = "";

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$name = test\_input($\_POST["name"]);

$email = test\_input($\_POST["email"]);

$website = test\_input($\_POST["website"]);

comment = test\_input($\_POST["comment"]);

$gender = test\_input($\_POST["gender"]);

}

function test\_input($data) {

$data = trim($data);

$data = stripslashes($data);

$data = htmlspecialchars($data);

return $data;

}

1. Проверить обязательные поля
2. Показать сообщения об ошибках

E-mail: <input type="text" name="name">

<span class="error">\* <?php echo $nameErr;?></span>

1. Проверка имени

if (empty($\_POST["name"])) {

$nameErr = "Name is required";

} else {

$name = test\_input($\_POST["name"]);

// проверьте, содержит ли имя только буквы и пробелы

if (!preg\_match("/^[a-zA-Z ]\*$/",$name)) {

$nameErr = "Разрешены только буквы и пробелы";

}

}

1. Проверка email

if (empty($\_POST["email"])) {

$emailErr = "Email is required";

} else {

$email = test\_input($\_POST["email"]);

// проверьте, правильно ли сформирован адрес электронной почты

if (!filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

$emailErr = "Invalid email format";

}

}

1. Проверка URL

if (empty($\_POST["website"])) {

$website = "";

} else {

$website = test\_input($\_POST["website"]);

// проверить правильность синтаксиса URL-адреса (это регулярное выражение также допускает тире в URL-адресе)

if (!preg\_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~\_|!:,.;]\*[-a-z0-9+&@#\/%=~\_|]/i",$website)) {

$websiteErr = "Invalid URL";

}

}