

ОБРАБОТКА ФОРМ В PHP

LECT. UNIV. NATALIA PLEȘCA

ВСПОМИНАЕМ...

Что такое форма?

Для чего внедряется форма на web-странице?

Примеры элементов формы...

СОДЕРЖАНИЕ

1. Методы GET и POST
используемые для передачи
данных из форм
2. Векторы \$_GET и \$_POST
3. Примеры использования
4. Векторы \$_SERVER и \$_REQUEST

HTML - ФОРМЫ

Форма HTML представляет собой часть документа, созданная с использованием элементов HTML

Назначением формы является сбор информации от пользователей

- После того как пользователь заполнит форму и запустит процесс ее обработки, информация из нее попадает в программу, работающую на сервере

HTML-форма содержит:

- Специальные элементы - **контролы** - элементы управления формой, такие как текстовые поля, кнопки, чекбоксы (флажки), радио кнопки (переключатели) и т.д.
- метки для этих элементов управления (label), обычный текст и т.д.

ФОРМЫ – HTML ЭЛЕМЕНТЫ

Формы это механизм, посредством которого можно передавать данные на веб сервер

Формы содержат элементы управления, которые могут «захватить» данные, введенные пользователем или представить пользователю какие-то данные / информации

Эти данные могут быть переданы PHP-скрипту и потом, могут быть использованы, либо как-то обработаны в PHP программе

Формы это HTML элементы !!!

Общая форма элемента form:

```
<form action="action_page.php" method="POST"  
  target=" blank" accept-charset="UTF-8" autocomplete="off" novalidate  
  enctype="application/x-www-form-urlencoded">
```

элементы формы
</form>

АТТРИБУТЫ ТЕГА „FORM”

Атрибут	Значение	Описание
accept-charset	<i>character_set</i>	Определяет коды символов, которые будут использоваться при заполнении (ISO 8859-1, ISO 8859-2, ISO 8859-15, и т.д.). Он поддерживается всеми браузерами
action	<i>URL</i>	Указывает, куда отправить форму. Он поддерживается всеми браузерами
autocomplete	on off	Определяет, должна ли эта форма иметь свойство автозаполнения или нет. Этот атрибут поддерживается всеми браузерами кроме Оперы. Появился в HTML5
enctype	application/x-www-form-urlencoded (valuearea implicit), multipart/form-data, text/plain	Указывает, как данные формы должны быть зашифрованы при передаче на сервер. Данный атрибут может быть использован только тогда, когда применяется метод = "post". Поддерживается браузерами
method	get post	Определяет метод передачи данных формы. Поддерживается всеми браузерами
name	<i>text</i>	Задаёт имя формы
novalidate	novalidate	Указывает, что форма не должна быть валидирована при передаче. Появился в HTML5. Поддерживается всеми браузерами кроме Safari и IE9-
target	_blank, _self, _parent, _top	Указывает, где появится ответ после заполнения формы. Поддерживается браузерами

МЕТОД „GET” И МЕТОД „POST”

Атрибут **"method"** определяет способ отправки данных формы - данные передаются на страницу/файл, указанный в атрибуте **"action"**)

Данные формы могут быть отправлены в качестве переменной URL (метод **"get"**) или HTTP-транзакции (метод **"post"**)

- При использовании метода GET:
 - Данные формы добавляются в URL в виде пары **"имя&значение"**
 - Длина URL ограничена (около 3000 знаков)
 - Не рекомендуется использование метода GET при отправке конфиденциальных или защищенных данных (будут видны в URL)
 - Рекомендуется использовать данный метод, если пользователь хочет зарегистрировать заполнение формы
- При использовании метода POST:
 - Данные вкладываются в HTTP-запросе (данные не видны в URL)
 - Нет ограничений на размер
 - Заполнение формы не может быть зарегистрирована

СУПЕРГЛОБАЛЫ \$_GET и \$_POST

Атрибут **action** тега **form** ссылается на имя PHP-файла, который будет обрабатывать полученные данные, а атрибут **method** – ссылается на метод, используемый для отправки данных на сервер

Извлечение данных, отправляемых на сервер с помощью метода **GET** (значение по умолчанию) или **POST**, в PHP-скрипте, производится при помощи predefined векторов **\$_GET** или **\$_POST**

Векторы **\$_GET** и **\$_POST** называются в PHP и **суперглобалами**

Суперглобалы были внедрены в PHP 4.1.0, и являются включёнными переменными, которые доступны из любой области видимости

СУПЕРГЛОБАЛЬНЫЕ ПЕРЕМЕННЫЕ В PHP

"Суперглобальны" в PHP - они всегда доступны, в независимости от области применения

Они могут быть доступны из любой функции, любого класса или файла

В PHP можно использовать следующие суперглобалы:

- `$GLOBALS`
- `$_SERVER`
- `$_REQUEST`
- `$_POST`
- `$_GET`
- `$_FILES`
- `$_ENV`
- `$_COOKIE`
- `$_SESSION`

ИСПОЛЬЗОВАНИЕ ВЕКТОРА

\$ _POST

Вектор `$ _POST` (*пишется большими буквами!!!*) используется для сбора данных формы, после того как HTML-форма была отправлена на сервер, при помощи метода „`post`”

Синтаксис:

```
$ _POST["имя_контроля_формы"]
```

В следующем примере будет использован метод „`post`” для отправки формы, файлу „`index_rus.php`”

Обязательно в контроле необходимо использовать атрибут *name*

В файле `index_rus.php` отправленные данные (в данном примере - введенное имя в текстовом поле) принимаются массивом `$ _POST` , через переменную „`name`” (из скрипта)

ПРИМЕР ПЕРЕДАЧИ ДАННЫХ ЧЕРЕЗ ФОРМУ

```
<form action="index_rus.php" method="post">  
  <label>Введите ваше имя:</label>  
  <input type="text" name="name" maxlength="10">  
  <br /><br />  
  <input type="reset" value="Сброс">  
  <input type="submit" value="Отправить">  
</form>
```

Введите ваше имя:

Сброс

Отправить

...после нажатия кнопки „Отправить”

Данные будут «захвачены» следующим скриптом, на сервере:

```
<?php
    echo "<br />";
    $nume = $_POST["nume"]; //в этой переменной, $nume,
    //сохранится значение введенное пользователем
    echo "В форме вы ввели: ",$nume,"<br />";
?>
```

Результат:

В форме вы ввели: Marcela

ИСПОЛЬЗОВАНИЕ СУПЕРГЛОБАЛА `$_POST`

В случае «захвата» данных при помощи вектора `$_POST[...]`, если в каком-то поле отсутствуют данные (какое-то поле не было заполнено пользователем), PHP может сгенерировать предупреждение (`warning`), в зависимости от настроек

Чтобы избежать этого, перед символом `$` (от переменной `$_POST[...]`) ставится символ `@`

Значение использования этого символа состоит в игнорировании предупреждения типа *warning*

....Но лучше проверить и обработать все вводы данных

ИСПОЛЬЗОВАНИЕ ВЕКТОРА \$_GET

Вектор `$_GET` (пишется тоже большими буквами!!!) может быть использован для сбора данных формы, отправленной при помощи метода „get“

Вектор `$_GET` может, также, быть использован и для сбора данных из URL-адреса

Пример (сбора данных из URL адреса):

```
<a href="index_rus.php?subject=PHP&web=W3schools.com">Проверяем метод  
$GET</a>
```

```
<?php  
    echo "<br />";  
    echo "<br />";  
    $subject = $_GET["subject"];  
    $a_web = $_GET["web"];  
    echo "Значения из URL-адреса: " . $subject . " и " . $a_web . "<br />";  
?>
```

Проверяем метод \$GET

Значения из URL-адреса: PHP и W3schools.com

ПРИМЕР ПЕРЕДАЧИ ДАННЫХ ПОСРЕДСТВОМ ФОРМЫ

```
<form action="index_rus.php" method="get">
```

```
  <label>Введите ваше имя:</label>
```

```
  <input type="text" name="nume" maxlength="10">
```

```
  <br /><br />
```

```
  <input type="reset" value="Сброс">
```

```
  <input type="submit" value="Отправить">
```

```
</form>
```



The screenshot shows a web browser window with the address bar displaying `localhost/exemple/_05_Formulare/index_rus.php?nume=Luminita`. Below the address bar, there is a form with the label "Введите ваше имя:" followed by a text input field containing the text "Luminita". Below the input field, there are two buttons: "Сброс" (Reset) and "Отправить" (Submit).

ПРИ ОТПРАВКЕ ФОРМЫ

...данные будут перехвачены вектором `$_GET` и обработаны следующим скриптом:

```
<?php
    echo "<br />";
    $nume=$_GET["nume"];
    echo "Значение из формы: " . $nume . "<br />";
?>
```

Результат:

Значение из формы: Luminita

ФУНКЦИЯ “ISSET” И ОБРАБОТКА ФОРМ

При «захвате» данных из формы, при помощи суперглобалов, рекомендуется проверить если данные были успешно переданы на сервер

Для этого используется функция `isset()`, которая определяет если переменной была присвоена другое значение чем NULL

Синтаксис: `bool isset ($var [, $...])`

Если передаются несколько параметров, и данная передача проверяется при помощи функции `isset()`, тогда функция возвращает ***true***, только если все параметры установлены

Функция `isset()` используется, обычно, вместе с условным оператором `if`

ПРИМЕР ИСПОЛЬЗОВАНИЯ ISSET()

В следующем примере будет использоваться функция *isset()* для того что бы проверить если переменная была установлена

Пример1:

```
<?php
    $nume = "Afanasiu";
    if(isset($nume))
        {echo "Переменная была установлена!";} //true
?>
```

Пример2:

```
<?php
    $nume = "Afanasiu";
    if(isset($prenume))
        {echo "Переменная 'prenume' была установлена!";} //true
    else
        {echo "Переменная 'prenume' не была установлена!";} //false
?>
```

ОБРАБОТКА ФОРМЫ С ИСПОЛЬЗОВАНИЕМ ФУНКЦИИ ISSET()

```
<!DOCTYPE html>
<html>
<head><title>Exemplu setare</title></head>
<body>
<h3>Autentificare</h3>
<form action="setare.php" method="POST">
  Log-in: <input type="text" name="login" /><br /><br />
  Parola: <input type="text" name="password" /><br /><br />
  <input type="submit" value="Verifica">
</form>
<?php
  $login = "Anonim";
  $password = "Anonim";
  if (isset($_POST['login'])) $login = $_POST['login'];
  if (isset($_POST['password'])) $password = $_POST['password'];
  echo "<br /><br />Utilizatorul are log-in-ul: $login si parola: $password";
?>
</body>
</html>
```

РЕЗУЛТАТ

Autentificare

Log-in:

Parola:

Verifica

Utilizatorul are log-in-ul: **Anonim** si parola: **Anonim**

Autentificare

Log-in:

Parola:

Verifica

Utilizatorul are log-in-ul: **Arabela** si parola: **Ada**

Задача

Что произойдет если будут введены такие данные в **input** и ПОТОМ нажмем "submit"?

```
<h3>Autenticare</h3>
<form action="ex_xss.php" method="POST">
  Log-in: <input type="text" name="login" size="40" /><br /><br />
  Parola: <input type="text" name="password" /><br /><br />
  <input type="submit" value="Verifica">
</form>
<?php
  $login = "Anonim";
  $password = "Anonim";
  if (isset($_POST['login'])) $login = $_POST['login'];
  if (isset($_POST['password'])) $password = $_POST['password'];
  echo "<br /><br />Utilizatorul are log-in-ul: $login si parola: $password";
?>
```

Autenticare

Log-in:

Parola:

Utilizatorul are log-in-ul: Anonim si parola: Anonim

ФУНКЦИЯ EMPTY()

Используется в том же контексте что и функция `isset()`

Пример использования функции `empty()` вместе со стилями:

HTML

```
<h1>Date pentru transfer pe server</h1>
<form action="<?php echo $_SERVER['PHP_SELF'];?>" method="post">
  <input type="text" name="name" placeholder="Numele tău"/><br />
  <input type="email" name="email" placeholder="Email-ul tău" /><br />
  <input type="submit" value="Trimite datele" />
</form>
```

PHP

```
<?php
  $name = "Anonymous";
  $email = "Not email";
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
    if (!empty($_POST['name'])) {
      $name = $_POST['name'];
    }
    if (!empty($_POST['email'])) {
      $email = $_POST['email'];
    }
    echo "Va numiți: " . $name . " și adresa dvoastra de email este: " . $email;
  }
?>
```

Результат

Вначале были определены 2 переменные: **name** и **email**, которые будут использованы для сохранения данных из формы

Они были инициализированы. Начальные значения будут использованы в скрипте в случае если не будут введены данные в контролах формы, когда пользователь нажмет кнопку **"submit"**



The image shows a screenshot of a web form with a teal background. The title is "Date pentru transfer pe server" in white text. There are two input fields: the first contains "Iana" and the second contains "iana@gmail.com". Below the fields is a white button with rounded corners and the text "Trimite datele" in orange.

После нажатия на кнопку **"Trimite date"**, данные попадают на сервер и будут использованы в скрипте

Va numiți: Iana și adresa dvoastra de email este: iana@gmail.com

Другое использование функции EMPTY()

При определении HTML элемента:

```
...Log-in: <input type="text" name="login" value="<?php  
if(!empty($_POST['login'])) { echo $login; } ?>">
```

Проверяется если поле было заполнено используя короткий PHP скрипт

При обработки формы:

...

```
if(isset($_POST['submit'])) {
```

```
$login = isset($_POST['login']) ? $_POST['login'] : '';
```

...

Другой пример

```
<?php
    $login = $password = "";
    $err_log=$err_pass="";
    if (!empty($_POST['login'])) $login = $_POST['login'];
    else $err_log = "Заполни поле!";
    if (!empty($_POST['password'])) $password = $_POST['password'];
    else $err_pass = "Заполни поле!";
    if (($err_log=="")and($err_pass==""))
        echo "<br /><br />Пользователь имеет логин: <b>$login</b> и пароль: <b>$password</b>";
?>
<!DOCTYPE html>
<html>
<head><title>Primer</title><meta charset="utf-12" /><style> .err {color:red; font-size: 10px;}</style></head>
<body>
<h3>Аутентификация</h3>
<form action="isset_empty.php" method="POST">
    <label>Логин: <input type="text" name="login" value="<? echo $login; ?>" /></label>
        <span class="err">*<? echo $err_log; ?></span><br /><br />
    <label>Пароль: <input type="text" name="password" value="<? echo $password; ?>" /></label>
        <span class="err">*<? echo $err_pass; ?></span><br /><br />
    <input type="submit" value="Проверь">
</form></body></html>
```

РЕЗУЛЬТАТЫ

Аутентификация

Логин: *Заполни поле!

Пароль: *Заполни поле!

Проверь

Аутентификация

Логин: *

Пароль: *Заполни поле!

Проверь

Пользователь имеет логин: Алла и пароль: Бала

Аутентификация

Логин: *

Пароль: *

Проверь

АНАЛИЗ ПЕРЕДАЧИ ДАННЫХ ЧЕРЕЗ ДРУГИЕ КОНТРОЛЫ ФОРМЫ

Для текстового `input`-а, уже была проанализирована передача данных

Также «перехватываются» данные из `input`-а типа `password`, `number`, `email` или тега `textarea`

PS: Не забывайте, что элемент `textarea` имеет контент, и не имеет атрибут `value`

ОБРАБОТКА КОНТРОЛА ТИПА “SUBMIT”

В случае `input`-а типа `submit`, значение (которое совпадает с текстом на кнопке) может быть «перехвачено» только в том случае, если в теге `<input type="submit"...>` используется и атрибут `name`

В случае если форма имеет только один контрол типа `submit`, нет необходимости проверить значение (из атрибута *value*)

Но в случае если форма имеет две или более кнопки типа `submit`, которым мы хотим поставить в соответствие разные функции, очень важно знать какая кнопка была нажата пользователем

ПРИМЕР

```
<form action="index_rus.php" method="post">
  <table cellspacing="0" cellpadding="4">
    <tr> <td>Введите ваше имя:</td>
      <td><input type="text" name="name"
maxlength="15"></td>
    </tr>
    <tr> <td colspan="2" align = "center">
      <input type="submit" value="Приветствие"
name="operatia" />
      <input type="submit" value="Прощание"
name="operatia" />
    </td>
  </tr>
</table>
</form>
```

Введите ваше имя:

Приветствие

Прощание

ОБРАБОТКА ФОРМЫ

```
<?php
    $nume = $_POST["nume"];
    $op=$_POST['operatia']; //контрол с именем 'operatia' это кнопка типа
    submit
    //захватываем его значение в перем. $op, чтобы понять какая из
    кнопок была нажата
    //выводим их
    if($op=="Приветствие"){
        echo "<br />Добро пожаловать к нам " . $nume . "!!!";
    } else {echo "<br />До новой встречи " . $nume . "!!!";}
?>
```

Добро пожаловать к нам Iana!!!

ил
и

До новой встречи Iana!!!

ОБРАБОТКА ДАННЫХ ПЕРЕДАННЫХ ЧЕРЕЗ „RADIO”

В случае использования контрола (ввода) типа **radio**, все теги типа `<input type="radio"...>` должны иметь, для атрибута **name**, то же имя, а в качестве значения, для атрибута **value** – разные значения, при помощи которых и будет производится идентификация опции, выбранной пользователем

Значение, выбранное пользователем , будет послано скрипту, на сервере, для обработки

ПРИМЕР ОБРАБОТКИ ФОРМЫ С „RADIO” КНОПКАМИ

```
<form action="index_rus.php" method="post">
  <table cellspacing="0" cellpadding="4">
    <tr><td>Введите ваше имя:</td>
    <td><input type="text" name="name" maxlength="15"></td>
  </tr></table>
  <p><b>Выберете правильный вариант ответа: </b></p>
  <input type="radio" value="1" name="op">Колорадский жук является
хищником<br />
  <input type="radio" value="2" name="op">Кузнечик является
хищником<br />
  <input type="radio" value="3" name="op">Божья коровка является
хищником<br /><br />
  <input type="reset" value="Сброс">
  <input type="submit" value="Отправить">
</form>
```


РЕЗУЛЬТАТ

Введите ваше имя:

Выберете правильный вариант ответа:

- Колорадский жук является хищником
- Кузнечик является хищником
- Божья коровка является хищником

Сброс

Отправить

СКРИПТ ДЛЯ ОБРАБОТКИ ДАННЫХ

```
<?php
$nume = $_POST["nume"];
$var=$_POST['op'];
switch($var) {
  case 1:
    echo "Колорадский жук не является хищником. Пересмотри определение понятия - хищник- " . $nume . "!!!";
    break;
  case 2:
    echo "Кузнечик не является хищником. Пересмотри определение понятия -хищник- " . $nume . "!!!";
    break;
  case 3:
    echo "Божья коровка является хищником, потому что питается насекомыми- вредителями, которые съедают капусту на огородах. Молодец " . $nume . "!!!";
    break;
}
```

РЕЗУЛЬТАТ ОБРАБОТКИ

Колорадский жук не является хищником. Пересмотри определение понятия -хищник- Anna!!!

Кузнечик не является хищником. Пересмотри определение понятия -хищник- Anna!!!

Божья коровка является хищником, потому что питается насекомыми-вредителями, которые съедают капусту на огородах. Молодец Anna!!!

ОБРАБОТКА ФОРМЫ СОДЕРЖАЩАЯ КОНТРОЛ ТИПА «CHECKBOX»

В случае использования контрола (ввода) типа **checkbox**, каждый такой контрол имеет свое собственное имя

Если какой-то контрол типа **checkbox** был выбран пользователем, форма отправит значение из атрибута **name** тега `<input type="checkbox"…>` для обработки скриптом (как `string` - последовательность символов)

Но если данный атрибут отсутствует, для обработки, скрипту, будет послано последовательность символов „**on**”

А если , пользователь ничего не выбрал из списка, серверу ничего не передастся, и поэтому запросом `$_POST[...]` может быть сгенерировано предупреждение (зависит и от опций PHP)

- Чтобы избежать этого, в независимости от настроек, используется символ **@** перед запросом `$_POST[...]`

Пример использования кнопки "checkbox"

HTML

```
...<label>Doriți livrarea?<input type="checkbox" checked class="linie"
id="LivrareDaNu" onclick="LivrareDaNuClick();" title='Daca nu doriti livrarea -
scoateti bifa!' name="da_nu" /></label>...
```

JavaScript

```
function LivrareDaNuClick(){
    document.getElementById("LivrareDaNuContent").style.display =
(document.getElementById("LivrareDaNu").checked) ? 'block' : 'none';
}
```

PHP

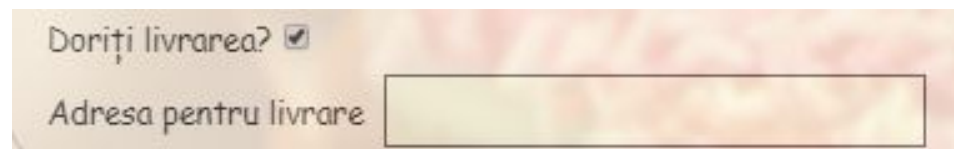
```
...$da_nu = "".$_POST['da_nu'];
if($da_nu=='on') $da_nu = 1; else $da_nu = 0;
$adresa = "".$_POST['adresa'];
if($da_nu==0) $adresa = "";
```



Numarul dvoastra de telefon

Doriți livrarea?

Selectati produsul **Gogosi** ▼



Doriți livrarea?

Adresa pentru livrare

ПРИМЕР

```
<form action="index_rus.php" method="post">
  <table cellspacing="0" cellpadding="4">
    <tr><td>Введите ваше имя:</td>
    <td><input type="text" name="nume" maxlength="15"></td>
  </tr>
</table>
<p><b>Выберите иностранные языки которыми владеете: </b></p>
<input type="checkbox" name="en">Английский<br />
<input type="checkbox" name="fr">Французкий<br />
<input type="checkbox" name="ru">Румынский<br />
<input type="checkbox" name="ge">Немецкий<br />
<input type="checkbox" name="it">Итальянский<br />
<input type="checkbox" name="es">Испанский<br />
<br />
<input type="reset" value="Сброс">
<input type="submit" value="Отправить">
</form>
```

РЕЗУЛЬТАТ СОЗДАНИЯ ФОРМЫ

Введите ваше имя:

Выберите иностранные языки которыми владеете:

- Английский
- Французский
- Румынский
- Немецкий
- Итальянский
- Испанский

СКРИПТ КОТОРЫЙ БУДЕТ ОБРАБАТЫВАТЬ ДААННЫЕ

?php

```
$nume = $_POST["nume"];
```

```
$en=@$_POST['en'];
```

```
$fr=@$_POST['fr'];
```

```
$ru=@$_POST['ru'];
```

```
$ge=@$_POST['ge'];
```

```
$it=@$_POST['it'];
```

```
$es=@$_POST['es'];
```

```
$l=0;//$l это флаг, который =1 когда какой-то язык был выбран
```

```
echo "Вот языки которыми вы владеете, " . $nume . " :<br />";
```

```
if($en=="on") {echo " - английский<br />";$l=1;}
```

```
if($fr=="on") {echo " - французский<br />";$l=1;}
```

```
if($ro=="on") {echo " - румынский<br />";$l=1;}
```

```
if($ge=="on") {echo " - немецкий<br />";$l=1;}
```

```
if($it=="on") {echo " - итальянский<br />";$l=1;}
```

```
if($es=="on") {echo " - испанский<br />";$l=1;}
```

```
if($l==0)//если флаг остается на значение 0, выводим пользователю
```

```
//сообщение что бы он знал что не выбрал ничего
```

```
echo "Не знаешь ни одного иностранного языка " . $nume . " ?";
```

?>

РЕЗУЛЬТАТ ОБРАБОТКИ

Вот языки которыми вы владеете, Inna:

- английский
- французкий
- румынский

Или, если ничего не было выбрано

Введите ваше имя:

Выберите иностранные языки которыми владеете:

- Английский
- Французкий
- Румынский
- Немецкий
- Итальянский
- Испанский

Сброс

Отправить

Вот языки которыми вы владеете, Inna:

Не знаешь не одного иностранного языка Inna?

ОБРАБОТКА ТЕГА „SELECT”

В случае использования контрола `select`, PHP сможет извлечь передаваемые данные через имя, зафиксированное в атрибуте `name`, тега `<select ...>`

Значения, отправленные на сервер будут те которые зафиксированы в атрибуте `value` тегов `option`, включенные внутри тега `select`

ПРИМЕР „SELECT”

```
<form action="index_rus.php" method="post">
  <table cellspacing="0" cellpadding="4">
    <tr><td>Введите ваше имя:</td>
    <td><input type="text" name="nume" maxlength="15"></td>
  </tr>
</table>
<p><b>Выберите язык который вы знаете лучше: </b></p>
<select name="op">
  <option value="1">Английский</option>
  <option value="2">Французский</option>
  <option value="3">Румынский</option>
  <option value="4">Немецкий</option>
</select>
<br /><br />
<input type="reset" value="Сброс">
<input type="submit" value="Отправить">
</form>
```

ПОЛУЧЕННАЯ ФОРМА

Введите ваше имя:

Выберите язык который вы знаете лучше:

Французкий ▾
Английский
Французкий
Румынский
Немецкий

ВВЕСТИ

вы знаете лучше, :

СКРИПТ КОТОРЫЙ ОБРАБОТАЕТ ФОРМУ

```
<?php
    $nume = $_POST["nume"];
    $op=$_POST['op'];
    echo "Язык который вы знаете лучше, " . $nume . ":";
    switch($op) {
    case 1:
        echo " английский!";
        break;
    case 2:
        echo " французский!";
        break;
    case 3:
        echo " русский!";
        break;
    case 4:
        echo " немецкий!";
        break;
    }
}
```

?>

Язык который вы знаете лучше, Lilia: французский!

ПЕРЕДАЧА НЕСКОЛЬКИХ ЗНАЧЕНИЙ, ИЗ НЕСКОЛЬКИХ КОНТРОЛОВ

Один из плюсов использования форм при передаче данных состоит в том что можно передать несколько значений, из нескольких контролов

В данном случае данные передаются как элементы вектора или иногда как элементы матрицы

Для этого, атрибут **name** контролов формы, должен быть сопоставлен соответствующему элементу вектору (или матрицы), с целью принятия его значения

Будет использован следующий синтаксис: **name=«имя_вектора[индекс]»**

Или, соответственно: **name=«имя_матрицы [индекс_строки][индекс_столбца]»**

ПРИМЕР

```
<form method="post" action="index_rus.php">
  <table cellspacing="0" cellpadding="4" width = "50%">
    <h2>Ежедневный отчет за <?php echo " " . date("d/m/Y") ?> </h2>
    <br />
    <tr><th align = "right">Введите ваше имя:</th>
    <th><input type="text" name="name" maxlength="15" value='<?php echo $_POST["name"] ?>' /></th>
    </tr>
    <tr><td align = "right">Введите количество проданных сегодня батонов "Botanica": </td>
    <td><input type="number" min="1" max="50" name="cantitate[0]" /></td></tr>
    <tr><td align = "right">Введите количество проданных сегодня батонов "Orasaneasca": </td>
    <td><input type="number" min="1" max="50" name="cantitate[1]" /></td></tr>
    <tr><td align = "right">Введите количество проданного сегодня хлеба "De capitala": </td>
    <td><input type="number" min="1" max="50" name="cantitate[2]" /></td></tr>
    <tr><td align = "center">
      <input type="reset" value="Сброс данных">
      <input type="submit" value="Отправка данных">
    </td></tr>
  </table>
</form>
```

ЗАПОЛНЕННАЯ ФОРМА

Ежедневный отчет за 21/02/2015

Введите ваше имя:

Введите количество проданных сегодня батонov
"Botanica":

Введите количество проданных сегодня батонov
"Orasaneasca":

Введите количество проданного сегодня хлеба
"De capitala":

СКРИПТ ОБРАБАТЫВАЮЩИЙ ДААННЫЕ

```
<?php
    $nume = $_POST["nume"];
    $a=$_POST['cantitate'];
    $preturi = array(4, 2.9, 5.5);
    echo $nume . ' сегодня вы продали продукцию на ';
    $p=0;
    for($i=0;$i<count($a);$i++)
        $p+=$a[$i]*$preturi[$i];
    echo 'сумму: ' . $p . " леи";

?>
```

Результат обработки:

Anna сегодня вы продали продукцию на сумму: 278.6 леи

НЕОБХОДИМОСТЬ ОБРАБОТКИ ФОРМ

Обработка форм имеет много общего с валидацией данных. И поэтому, обычно создаются **правила валидации**

Правило валидации ограничивает или контролирует то, что пользователь может ввести в элемент управления

Правила валидации могут быть созданы имея ввиду:

- **Тип данных** необходимых для ввода – если требуется ввести числовое значение, то оно должно состоять только из чисел; если требуется имя – то данное поле может состоять только из букв; если требуется ввести дату – оно должна соблюдать необходимый формат и т.д.
- **Размер поля** – к примеру для имени можно ограничить поле до 10-15 символов, то что не позволит ввести разный спам
- **Свойства контроля** – обязательные поля и необязательные, можно задать мин. либо макс. значение, либо если просится дата то не позднее чем ...
- **Маски ввода** – используются для того что бы обязать пользователей ввести данные в определенном формате. Например дату в американском формате - мм/дд/гг

Для каждого контроля можно использовать одно или несколько правил валидации

СУПЕРГЛОБАЛ \$ _SERVER

Переменная \$ _SERVER - это массив, содержащий информацию о заголовках, путях и местоположении скриптов

Записи в этом массиве создаются веб-сервером. Нет гарантии, что каждый веб-сервер предоставит любую из них; сервер может упустить некоторые из них или предоставить другие, не указанные здесь.

Наиболее распространенные элементы, которые могут быть использованы в переменной \$ _SERVER:

Элемент	Описание
<code>\$ _SERVER['PHP_SELF']</code>	Возвращает имя файла, в котором выполняется текущий скрипт
<code>\$ _SERVER['GATEWAY_INTERFACE']</code>	Возвращает версию CGI, используемую web-сервером. CGI (Common Gateway Interface) - это стандартный метод используемый для генерирования динамического контента web страниц или приложений. CGI - обеспечивает интерфейс между сервером и программами, которые генерируют веб-контент
<code>\$ _SERVER['SERVER_ADDR']</code>	Возвращает IP адрес хост-сервера
<code>\$ _SERVER['SERVER_NAME']</code>	Возвращает имя хост-сервера

СУПЕРГЛОБАЛ \$_SERVER

<code>\$_SERVER['SCRIPT_NAME']</code>	Возвращает путь и имя скрипта, который в данный момент выполняется
<code>\$_SERVER['SERVER_PROTOCOL']</code>	Возвращает название и обновление протокола информации
<code>\$_SERVER['REQUEST_METHOD']</code>	Возвращает используемый метод для передачи данных (к примеру POST)
<code>\$_SERVER['HTTPS']</code>	Скрипт запрашивается через защищенный HTTP протокол
<code>\$_SERVER['REMOTE_ADDR']</code>	Возвращает IP-адрес с которого пользователь просматривает страницу
<code>\$_SERVER['HTTP_USER_AGENT']</code>	Возвращает информацию о сервере и среде выполнения
<code>\$_SERVER['PHP_AUTH_USER']</code>	Когда происходит HTTP аутентификация пользователя, данному параметру ставится в соответствие значение имени (log-in) введенного пользователем
<code>\$_SERVER['PHP_AUTH_PW']</code>	При аутентификации, данному параметру ставится в соответствие пароль пользователя

Другие элементы и детали: <http://php.net/manual/en/reserved.variables.server.php>

ИСПОЛЬЗОВАНИЯ СУПЕРГЛОБАЛА \$ _SERVER

```
<?php
    echo $_SERVER['PHP_SELF'] . " - PHP_SELF";
    echo "<br />";
    echo $_SERVER['SERVER_NAME'] . " - SERVER_NAME";
    echo "<br />";
    echo $_SERVER['HTTP_HOST'] . " - HTTP_HOST";
    echo "<br />";
    echo $_SERVER['GATEWAY_INTERFACE'] . " - GATEWAY_INTERFACE";
    echo "<br />";
    echo $_SERVER['HTTP_USER_AGENT'] . " - HTTP_USER_AGENT";
    echo "<br />";
    echo $_SERVER['SCRIPT_NAME'] . " - SCRIPT_NAME";

?>
```

РЕЗУЛЬТАТ ПРИМЕРА

`/exemple/_05_Formulare/index_test_supergl.php - PHP_SELF`

`localhost - SERVER_NAME`

`localhost - HTTP_HOST`

`CGI/1.1 - HTTP_REFERER`

`Mozilla/5.0 (Windows NT 5.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/40.0.2214.111 Safari/537.36 - HTTP_USER_AGENT`

`/exemple/_05_Formulare/index_test_supergl.php - SCRIPT_NAME`

ДРУГОЙ ПРИМЕР

`$_SERVER['SCRIPT_NAME']` – полезно использовать для страниц, которые должны указывать на самих себя.

```
<form action="<?php $_SERVER['SCRIPT_NAME']?>" method="post">
```

СУПЕРГЛОБАЛ \$_REQUEST

Используется когда происходит сбор данных от пользователя, при помощи формы

Создается ассоциативный массив, который по умолчанию содержит данные переменных *\$_GET*, *\$_POST* и *\$_COOKIE*

ПРИМЕР ИСПОЛЬЗОВАНИЯ СУПЕРГЛОБАЛА \$_REQUEST

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_REQUEST['name'];
    if (empty($name)) {
        echo "Введите, пожалуйста, ваше имя!!!";
    } else {
        echo $name . "<br />";
    }
    $cantitate = $_REQUEST['cantitate'];
    if (empty($cantitate[0]) || empty($cantitate[1]) ||
        empty($cantitate[2])) {
        echo "Введите, пожалуйста, количество во всех полях!!!";
    } else {
        foreach ($cantitate as $value) {
            echo $value . " ";
        }
    }
}
?>
```

РЕЗУЛЬТАТ ПРИМЕРА

Ежедневный отчет за 21/02/2015

Введите ваше имя:

Введите количество проданных сегодня батонov
"Botanica":

Введите количество проданных сегодня батонov
"Orasaneasca":

Введите количество проданного сегодня хлеба
"De capitala":

Anna

Введите, пожалуйста, количество во всех полях!!!

ПРИМЕР II, ПРИМЕНЕНИЯ ГЛОБАЛОВ

```
<head>
<title>Пример обработки</title><meta charset="UTF-8">
<style>
    .error {    color:red; text-weight:bold; font-size:10 px;    }
</style>
</head>
<body>
<?php
    if ($_SERVER['REQUEST_METHOD'] == "POST")
        {if (empty($_POST["prename"]))
            { $prenameErr = "Необходимо заполнить поле!!!"; }
        }
?>
<form action="<?php echo htmlspecialchars($_SERVER['PHP_SELF'])?>" method="post">
Введите свое имя: <input type="text" name="prename" value="<?php echo $prename; ?>">
<span class="error">*<?php echo $prenameErr; ?></span><br /><br />
<input type="submit" value="Передать" />
</form>
</body></html>
```

РЕЗУЛЬТАТ ПРИМЕРА

Введите свое имя: *

Передать

Введите свое имя: *Необходимо заполнить поле!!!

Передать

ГДЕ ИСПОЛЬЗУЮТСЯ ДАННЫЕ ИЗ ФОРМЫ

Данные из формы могут быть

1. Обработаны прямо в скрипте, после принятия используя для этого переменные или массивы для их временного хранения
2. Сохранены в файлах данных, а потом, по необходимости выведены и обработаны
3. Отправлены на какой-то e-майл адрес для обработки каким-то пользователем
4. Сохранены в БД, а потом выведены и по необходимости обработаны используя арифметические операции, логические, сортировки, фильтрации с целью предоставления специальным пользователям для анализа



Знания

Что мы учили/прошли сегодня?

Способности/Навыки

Что мы можем сделать с
полученными знаниями?

Отношения / Поведение

Что, как, где и для чего мы можем
применить, знания и навыки
полученные сегодня?

A solid blue horizontal bar spans the bottom of the slide.