Взаимодействие с пользователем

Обработка данных, введенных в формы, с помощью PHP

Формы в HTML

- Формы используются для ввода информации пользователем
- **<form>** – тег для задания форм
- Синтаксис:

```
<form>
    …
    Элементы формы
    …
</form>
```

Элемент формы input

- **<input> –** поле для ввода информации пользователем
- **<input> –** один из основных элементов форм
- От атрибута **type** зависит типа поля для ввода

Поле для ввода текстовой информации

- Пример:

```
<form>
    Имя пользователя:<br>
    <input type="text" name="username">
    <br>
    Пароль:<br>
    <input type="password" name="password">
</form>
```

- `type="text"` – поле для ввода текстовой информации в одну строку
- `type="password"` – поле для ввода пароля

Кнопка для передачи информации на сервер

- Пример:

```
<form>
    Имя пользователя:<br>
    <input type="text" name="username">
    <br>
    Пароль:<br>
    <input type="password" name="password">
    <br><br>
    <input type="submit" value="Войти">
</form>
```

- `type="submit"` – создает кнопку, по нажатию на кнопки данные введенные в форму передаются на обработчик формы

Обработчик формы

- Обработчик формы – обычно это веб-страница на сервере со скриптом, который может обработать данные введенные в форму
- Обработчик формы определяется в атрибуте формы **action**

Атрибут action

- **action** – атрибут, который определяет действие, которое происходит при отправке (***submit***) формы
- Обычно данные из формы передаются на определенную страницу на сервере
- Пример. Данные из формы будут переданы на страницу **registration.php**:

```
<form action="registration.php">
```

- Если атрибут **action не заполнен**, данные из формы передаются **на текущую страницу**

Атрибут method

- **method** – атрибут, который определяет **HTTP метод** для передачи данных из формы на сервер
- Обычно используются методы **GET** и **POST**

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

<form action="registration.php" method="post">
```

- Если атрибут **method не заполнен**, данные передаются с помощью метода **GET**

Атрибут name

- Атрибут **name** необходим для корректной обработки формы

```
<html>
<body>

    <form action="welcome.php" method="post">
        Имя: <input type="text" name="name">
        <br>
        Электронная почта: <input type="text" name="email">
        <br>
        <input type="submit">
    </form>

</body>
</html>
```

Обработка данных из формы

- Пример страницы **welcome.php**

```
<html>
<body>

    Добро пожаловать <?php echo $_POST["name"]; ?>
    <br>
    Ваша электронная почта: <?php echo $_POST["email"]; ?>

</body>
</html>
```

- Результат работы:

```
Добро пожаловать, Иван
Ваша электронная почта ivan@gmail.com
```

Передача данных методом GET

- Пример из предыдущего слайда:

```
<html>
<body>

    <form action="welcome_get.php" method="get">
        Имя: <input type="text" name="name">
        <br>
        Электронная почта: <input type="text" name="email">
        <br>
        <input type="submit">
    </form>

</body>
</html>
```

Обработка данных из формы

- Пример страницы **welcome_get.php**

```
<html>
<body>

    Добро пожаловать <?php echo $_GET["name"]; ?>
    <br>
    Ваша электронная почта: <?php echo $_GET["email"]; ?>

</body>
</html>
```

- Результат работы:

```
Добро пожаловать, Иван
Ваша электронная почта ivan@gmail.com
```

GET или POST

- **GET** и **POST** создают ассоциативный массив:

```
array("key" => "value",
      "key2" => "value2",
      "key3" => "value3",
      ...)
```

- **Ключи** – значения атрибута **name** из элементов формы
- **Значения** – данные, которые ввел пользователь соответствующие элементы формы

GET или POST

- Методу **GET** соответствует суперглобальный массив **$_GET**
- Методу **POST** соответствует суперглобальный массив **$_POST**
- В **PHP** суперглобальные массивы доступны из любой функции или файла
- **$_GET** содержит данные, переданные в скрипт как параметры **URL** или через **HTTP GET** запрос
- **$_POST** содержит данные, переданные в скрипт через **HTTP POST** запрос

Когда использовать метод GET

- Метод GET – **метод по умолчанию** в WEB
- Метод GET **не следует использовать для изменения** данных (состояния, state) на сервере – это противоречит протоколу HTTP
- **Не следует передавать чувствительную информацию** с помощью метода GET (пароли, логины и т.п.)
- **Данные,** передаваемые методом GET, **отображаются в адресной** строке браузера
- **Объем данных**, которые можно передать с помощью метода GET**, ограничен**

Когда использовать метод POST

- Метод POST **используется для изменения** данных (состояния, state) на сервере
- **Можно передавать чувствительную информацию** с помощью метода POST (пароли, логины и т.п.)
- **Данные,** передаваемые методом POST, **не отображаются в адресной** строке браузера
- **Объем данных**, которые можно передать с помощью метода POST**, не ограничен**

Переключатель. Radio-button

- **radio-button – выбор только одного значения** для группы элементов с одним значением атрибута **name**
- Пример:

```
<input type="radio" name="sex" value="male" checked> Male
<br>
<input type="radio" name="sex" value="female"> Female
```

- `type="radio"` – создает переключатель
- `value="male"` – значение, которое будет передано на сервер
- `checked` – значение, выбранное по умолчанию

Переключатель. Radio-button

- **checkbox – выбор одного или нескольких значений**
- Пример:

```
<input type="checkbox" name="vehicle1" value="Bike"> Велосипед
<br>
<input type="checkbox" name="vehicle2" value="Car"> Машина
```

- `type="checkbox"` – создает чекбокс
- `value="Car"` – значение, которое будет передано на сервер

Выпадающий список. Drop-Down List

- **<select>** – создает элемент для выпадающего списка
- Пример:

```
<select name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

- **<option>** – элемент выпадающего списка, который доступен для выбора пользователю

Выпадающий список. Drop-Down List

- По умолчанию выбирается первый элемент списка
- Атрибут `selected` – определяет элемент, который выбран по умолчанию
- Пример:

```
<option value="fiat" selected>Fiat</option>
```

Выпадающий список. Drop-Down List

- Атрибут `multiple` – добавляет возможность множественного выбора
- Пример:

```
<select name="cars" multiple>
        …
</select>
```

Многострочный ввод

- **<textarea>** – создает поле для многострочного ввода
- Пример:

```
<textarea name="message" rows="10" cols="30">
Здесь можно написать много текста.
</textarea>
```

- `rows="10"` – высота поля в строках
- `cols="30"` – ширина поля в столбцах (символах)

Атрибуты элементов форм

- Значение по умолчанию:

```
<input type="text" name="firstname" value="Иван">
```

- `value="Иван"` – значение поля по умолчанию

Атрибуты элементов форм

- Доступ только на чтение

```
<input type="text" name="firstname" value="Иван"
readonly>
```

- `readonly` – пользователь не может редактировать поле

Атрибуты элементов форм

- Элемент не используется

```
<input type="text" name="firstname" value="Иван"
disabled>
```

- `disabled` элементы:
  - не редактируются
  - по ним нельзя кликнуть мышкой
  - их значения не передаются при отправке (submit) формы

Валидация данных

- **Валидация** – проверка корректности данных, введенных пользователем
- **Валидация важна** для защиты сайта от хакеров и спаммеров

Спецификация формы

Поле	Правила валидации
Имя	• Обязательно для заполнения • Может содержать только буквы и цифры
Электронная почта	• Обязательно для заполнения • Валидный адрес почты (содержит символы «@», «.» и т.п.)
Сайт	• Необязательно для заполнения • Если заполнен, должен содержать валидный URL-адрес
Комментарий	• Необязательно для заполнения • Должен допускать многострочный ввод текста
Пол	• Обязательно для заполнения • Необходимо выбрать один из двух

Форма

```
<form method="post"
      action="
      <?php
        echo htmlspecialchars($_SERVER["PHP_SELF"]);
      ?>">
Имя:
<input type="text" name="name">
Электронная почта:
<input type="text" name="email">
Сайт:
<input type="text" name="website">
Комментарий:
<textarea name="comment" rows="5" cols="40"></textarea>
Пол:
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="male">Male
</form>
```

Форма

- `htmlspecialchars` – функция, для экранирования специальных символов языка HTML

```
<form method="post"
      action="
      <?php
        echo htmlspecialchars($_SERVER["PHP_SELF"]);
      ?>">
```

- На вход функция принимает строку
- Результат работы функции – строка, в которой заэкранированы все специальные символы: <, >, “, ‘, * и т.п.

Простая валидация

```
<?php
// создание переменные, по умолчанию все переменные пустые
$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;
}
?>
```

Простая валидация

- Функция `trim` – удаляет символы пустого пространства в начале и конце строки
- Функция `stripslashes` – удаляет символ «/» из строки
- Функция `htmlspecialchars` – экранирует специальные символы

Проверка обязательных полей

```
// переменные, в которых будет находиться текст ошибки
$nameErr = $emailErr = $genderErr = $websiteErr = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Имя обязательно для заполнения";
  } else {
    $name = test_input($_POST["name"]);
  }
  if (empty($_POST["email"])) {
    $emailErr = "Электронная почта обязательна для заполнения";
  } else {
    $email = test_input($_POST["email"]);
  }
  …
}
```

Вывод ошибок

```
…
Name: <input type="text" name="name">
<span class="error">* <?php echo $nameErr;?></span>
<br><br>
E-mail:
<input type="text" name="email">
<span class="error">* <?php echo $emailErr;?></span>
<br><br>
…
```

Валидация URL, e-mail и т.п.

- **Регулярные выражения** – мощный механизм валидации

Сохранение введенных данных

```
Name: <input type="text" name="name" value="<?php echo $name;?>">

E-mail: <input type="text" name="email" value="<?php echo $email;?>">

Website: <input type="text" name="website"
         value="<?php echo $website;?>">

Comment: <textarea name="comment" rows="5" cols="40">
         <?php echo $comment;?></textarea>

Gender:
<input type="radio" name="gender"
    <?php if (isset($gender) && $gender=="female") echo "checked";?>
    value="female">Female
<input type="radio" name="gender"
    <?php if (isset($gender) && $gender=="male") echo "checked";?>
    value="male">Male
```