Бекенд

лек

Лекція. PHP Форми та додаткові можливості

Зав.каф.КН, PhD Граф М.С.

План:

1. Обробка форм PHP

2. Розширені можливості PHP

Обробка форм PHP

PHP - проста форма HTML

$_GET і $_POST використовуються для збору даних форми

```
<html>
<body>

<form action="welcome.php" method="post">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

Метод POST

Щоб відобразити подані дані, ви можете просто повторити всі змінні. "welcome.php" виглядає так:

```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo
$_POST["email"]; ?>

</body>
</html>
```

Обробка форм PHP

PHP - проста форма HTML

```
<html>
<body>

<form action="welcome_get.php" method="get">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

Метод GET

"welcome_get.php" виглядає так:

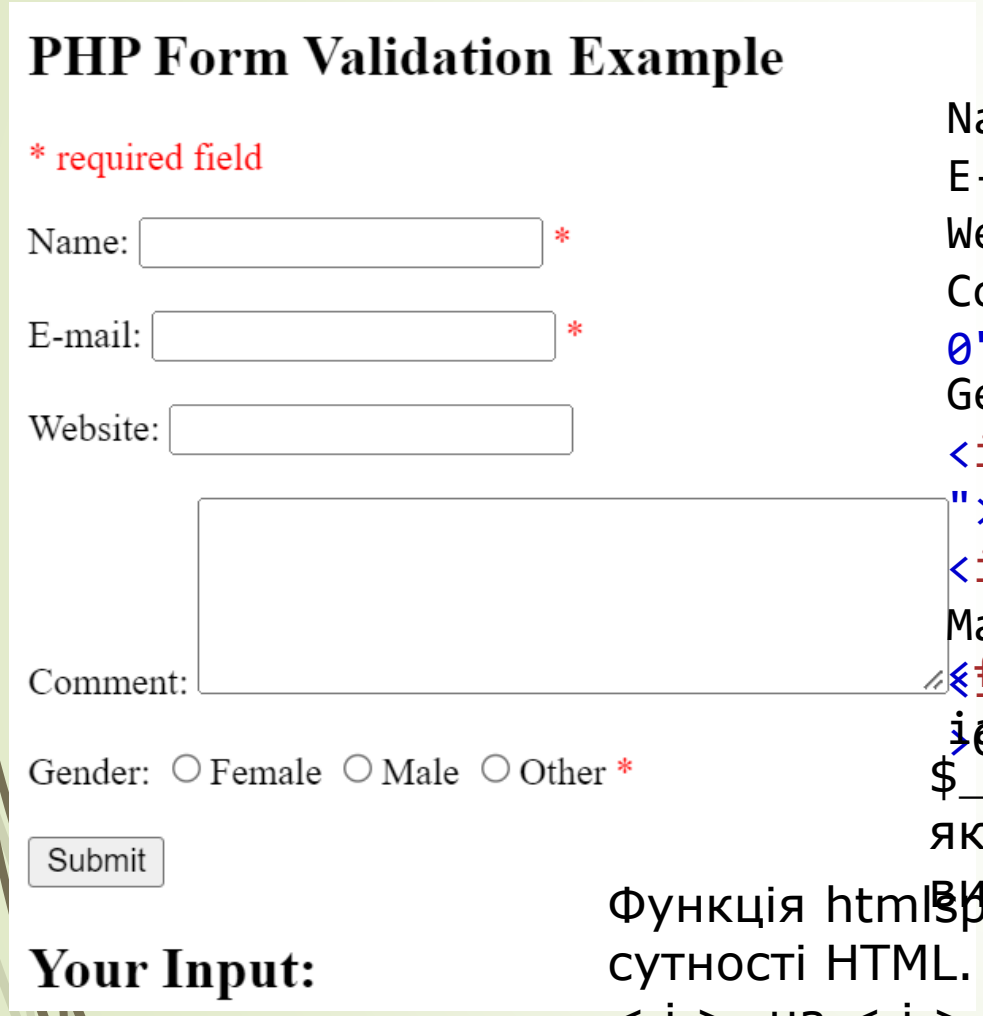
```
<html>
<body>

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

</body>
</html>
```

Обробка форм PHP

Перевірка форми PHP



Текстові поля

```
Name: <input type="text" name="name">
E-mail: <input type="text" name="email">
Website: <input type="text" name="website">
Comment: <textarea name="comment" rows="5" cols="40"></textarea>
```

Радіо кнопки

```
Gender:
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="male">Male
<input type="radio" name="gender" value="other">Other
```

HTML-код форми виглядає так:

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

$_SERVER["PHP_SELF"] — це надглобальна змінна, яка повертає ім’я файлу поточного сценарію, що виконується.

Функція htmlspecialchars() перетворює спеціальні символи на сутності HTML. Це означає, що він замінить символи HTML, такі як < i >, на < i >. Це запобігає зловмисникам від використання коду, вставляючи код HTML або Javascript (атаки міжсайтового сценарію) у форми.

Обробка форм PHP

Перевірка форми PHP

Змінна $_SERVER["PHP_SELF"] може бути використана хакерами!

Якщо на вашій сторінці використовується PHP_SELF, користувач може ввести косу риску (/), а потім виконати деякі команди міжсайтового сценарію (XSS).

Припустімо, що у нас є така форма на сторінці під назвою "test_form.php":

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

Тепер, якщо користувач вводить звичайну URL-адресу в адресний рядок, як-от «http://www.example.com/test_form.php», наведений вище код буде перекладено на:

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

Якщо користувач вводить таку URL-адресу в адресний рядок:

```
http://www.example.com/test_form.php/%22%3E%3Cscript%3Ealert('hacked')%3C/scrip
t%3E
```

У цьому випадку наведений вище код буде перекладено на:

```
<form method="post" action="test_form.php/"><script>alert('hack
ed')</script>
```

Цей код додає тег сценарію та команду сповіщення. Коли сторінка завантажиться, буде виконано код JavaScript (користувач побачить вікно сповіщення).

Обробка форм PHP

Перевірка форми PHP

Як уникнути експлойтів $_SERVER["PHP_SELF"]?

Експлойтів $_SERVER["PHP_SELF"] можна уникнути за допомогою функції htmlspecialchars().

Код форми

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

Тепер, якщо користувач спробує використати змінну PHP_SELF, це призведе до такого результату:

```
<form method="post" action="test_form.php/&quot
;&gt;&lt;script&gt;alert('hacked')&lt;/script&g
t;">
```

шкоди не завдано

передамо всі змінні через функцію PHP htmlspecialchars()

Обробка форм PHP

```php
<?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Name is required";
  } else {
    $name = test_input($_POST["name"]);
  }

  if (empty($_POST["email"])) {
    $emailErr = "Email is required";
  } else {
    $email = test_input($_POST["email"]);
  }

  if (empty($_POST["website"])) {
    $website = "";
  } else {
    $website = test_input($_POST["website"]);
  }

  if (empty($_POST["comment"])) {
    $comment = "";
  } else {
    $comment = test_input($_POST["comment"]);
  }

  if (empty($_POST["gender"])) {
    $genderErr = "Gender is required";
  } else {
    $gender = test_input($_POST["gender"]);
  }
}
?>
```

Форми PHP - обов'язкові поля

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

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>
Website:
<input type="text" name="website">
<span class="error"><?php echo $websiteErr;?></span>
<br><br>
Comment: <textarea name="comment" rows="5" cols="40"></textarea>
<br><br>
Gender:
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="male">Male
<input type="radio" name="gender" value="other">Other
<span class="error">* <?php echo $genderErr;?></span>
<br><br>
<input type="submit" name="submit" value="Submit">

</form>
```

Обробка форм PHP

Перевірка електронної пошти та URL-адреси

Перевірка імені

```
$name = test_input($_POST["name"]);
if (!preg_match("/^[a-zA-Z-' ]*$/",$name)) {
  $nameErr = "Only letters and white space
allowed";
}
```

перевірка електронної пошти

```
$email = test_input($_POST["email"]);
if (!filter_var($email, FILTER_VALIDATE_EMAIL))
{
  $emailErr = "Invalid email format";
}
```

перевірка URL-адреси

```
$website = test_input($_POST["website"]);
if
(!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[
-a-z0-9+&@#\/%?=~_|!:,.;]*[-a-z0-
9+&@#\/%=~_|]/i",$website)) {
  $websiteErr = "Invalid URL";
}
```

Функція preg_match() шукає в рядку шаблон, повертаючи true, якщо шаблон існує, і false в іншому випадку.

```
<?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Name is required";
  } else {
    $name = test_input($_POST["name"]);
    // check if name only contains letters and whitespace
    if (!preg_match("/^[a-zA-Z-' ]*$/",$name)) {
      $nameErr = "Only letters and white space allowed";
    }
  }

  if (empty($_POST["email"])) {
    $emailErr = "Email is required";
  } else {
    $email = test_input($_POST["email"]);
    // check if e-mail address is well-formed
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
      $emailErr = "Invalid email format";
    }
  }

  if (empty($_POST["website"])) {
    $website = "";
  } else {
    $website = test_input($_POST["website"]);
    // check if URL address syntax is valid (this regular expression also allows dashes in the URL)
    if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~_|!:,.;]*[-a-z0-9+&@#\/%=~_|]/i",$website)) {
      $websiteErr = "Invalid URL";
    }
  }

  if (empty($_POST["comment"])) {
    $comment = "";
  } else {
    $comment = test_input($_POST["comment"]);
  }

  if (empty($_POST["gender"])) {
    $genderErr = "Gender is required";
  } else {
    $gender = test_input($_POST["gender"]);
  }
}
?>
```

Функція PHP Date() `date(format,timestamp)`

Ось деякі символи, які зазвичай використовуються для дат:

- d - Позначає день місяця (від 01 до 31)
- m - означає місяць (від 01 до 12)
- Y – означає рік (чотирма цифрами)
- l (малий регістр 'L') - представляє день тижня

Інші символи, як-от "/", "." або "-", також можна вставляти між символами для додаткового форматування.

```php
<?php
echo "Today is " . date("Y/m/d") . "<br>";
echo "Today is " . date("Y.m.d") . "<br>";
echo "Today is " . date("Y-m-d") . "<br>";
echo "Today is " . date("l");
?>
```

Отримати час

Ось кілька символів, які зазвичай використовуються для позначення часу:
•H - 24-годинний формат години (з 00 до 23)
•h - 12-годинний формат години з нулями на початку (від 01 до 12)
•i - хвилини з нулями на початку (від 00 до 59)
•s - секунди з нулями на початку (від 00 до 59)
•a – нижній регістр Ante meridiem і Post meridiem (до полудня або після полудня)

```php
<?php
echo "The time is " . date("h:i:sa");
?>
```

Отримайте свій часовий пояс

```php
<?php
date_default_timezone_set("America/New_York");
echo "The time is " . date("h:i:sa");
?>
```

Створення дати за допомогою mktime()

Додатковий параметр *timestamp* у функції date() визначає позначку часу. Якщо опущено, використовуватимуться поточні дата й час (як у наведених вище прикладах).

```
mktime(hour, minute, second, month, day, year)
```

```php
<?php
$d=mktime(11, 14, 54, 8, 12, 2014);
echo "Created date is " . date("Y-m-d h:i:sa",
$d);
?>
```

Створення дати з рядка за допомогою strtotime()

```
strtotime(time, now)
```

```php
<?php
$d=strtotime("10:30pm April 15 2014");
echo "Created date is " . date("Y-m-d h:i:sa",
$d);
?>
```

PHP досить розумно перетворює рядок на дату, тому ви можете вводити різні значення:

```php
<?php
$d=strtotime("tomorrow");
echo date("Y-m-d h:i:sa", $d) . "<br>";

$d=strtotime("next Saturday");
echo date("Y-m-d h:i:sa", $d) . "<br>";

$d=strtotime("+3 Months");
echo date("Y-m-d h:i:sa", $d) . "<br>";
?>
```

Розширені можливості PHP

Файли cookie PHP

```
setcookie(name, value, expire, path, domain, secure, httponly);
```

```
<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name,
$cookie_value, time() +
(86400 * 30), "/"); // 86400 = 1 day
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
  echo "Cookie named '" .
$cookie_name . "' is not set!";
} else {
  echo "Cookie '" . $cookie_name . "'
is set!<br>";
  echo "Value is:
" . $_COOKIE[$cookie_name];
}
```

Щоб змінити файл cookie, просто встановіть (знову) файл cookie за допомогою `setcookie()`функції:

```
<?php
$cookie_name = "user";
$cookie_value = "Alex Porter";
setcookie($cookie_name, $cookie_value, time() +
(86400 * 30), "/");
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
  echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
  echo "Cookie '" . $cookie_name . "' is set!<br>";
  echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

</body>
</html>
```

Видалити файл cookie

```
setcookie("user", "", time() - 3600);
```

Сеанс — це спосіб зберігання інформації (у змінних), яка буде використовуватися на кількох сторінках.

Сеанс починається з `session_start()`функцією.

Змінні сеансу встановлюються за допомогою глобальної змінної PHP: $_SESSION.

Файл demo_session1.php.
На цій сторінці ми починаємо новий сеанс PHP і встановлюємо деякі змінні сеансу:

```
<?php
// Start the session
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are
set.";
?>

</body>
```

Розширені можливості PHP

Сеанси PHP

Отримання значення змінних сеансу PHP

"demo_session2.php". З цієї сторінки ми отримаємо доступ до інформації про сеанс, яку ми встановили на першій сторінці ("demo_session1.php").

Також зауважте, що всі значення змінних сеансу зберігаються в глобальній змінній $_SESSION:

```php
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Echo session variables that were set on previous page
echo "Favorite color is " . $_SESSION["favcolor"] . ".<br>";
echo "Favorite animal is " . $_SESSION["favanimal"] . ".";
?>

</body>
</html>
```

```php
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
print_r($_SESSION);
?>

</body>
</html>
```

Щоб змінити змінну сеансу, просто перезапишіть її:

```php
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// to change a session variable, just overwrite it
$_SESSION["favcolor"] = "yellow";
print_r($_SESSION);
?>

</body>
</html>
```

Щоб видалити всі глобальні змінні сеансу та знищити сеанс, використовуйте `session_unset()` та `session_destroy()`:

```php
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// remove all session variables
session_unset();
// destroy the session
session_destroy();
?>

</body>
</html>
```

Перевірка даних = Визначте, чи дані належної форми.

Очищення даних = видалення будь-яких недозволених символів із

`filter_list()`функцію можна використовувати для переліку того, що пропонує розширення фільтра PHP:

```
<table>
  <tr>
    <td>Filter Name</td>
    <td>Filter ID</td>
  </tr>
  <?php
  foreach (filter_list() as $id =>$filter) {
    echo '<tr><td>' . $filter . '</td><td>' .
filter_id($filter) . '</td></tr>';
  }
  ?>
</table>
```

Ви завжди повинні перевіряти зовнішні дані!
Недійсні подані дані можуть призвести до проблем із безпекою та зламати вашу веб-сторінку!
Використовуючи PHP-фільтри, ви можете бути впевнені, що ваша програма отримує правильні вхідні дані!

Розширені можливості PHP

Фільтри PHP

Функція `filter_var()`фільтрує одну змінну за допомогою вказаного фільтра. Для цього потрібні дві частини даних:
- Змінна, яку ви хочете перевірити
- Тип перевірки для використання

видалення всіх тегів HTML із рядка:

```
<?php
$str = "<h1>Hello World!</h1>";
$newstr = filter_var($str,
FILTER_SANITIZE_STRING);
echo $newstr;
?>
```

`ilter_var()`функція для перевірки, чи є змінна $int цілим числом. Якщо $int є цілим числом, вихід коду нижче буде: «Ціле число дійсне». Якщо $int не є цілим числом, результатом буде: «Ціле число недійсне»:

```
<?php
$int = 100;

if (!filter_var($int,
FILTER_VALIDATE_INT) === false) {
  echo("Integer is valid");
} else {
  echo("Integer is not valid");
}
?>
```

чи є змінна $ip дійсною IP-адресою:

```
<?php
$ip = "127.0.0.1";

if (!filter_var($ip, FILTER_VALIDATE_IP) === false) {
  echo("$ip is a valid IP address");
} else {
  echo("$ip is not a valid IP address");
}
?>
```

Дякую за увагу!