HERIOT WATT UNIVERSITY

F27WD: Web Design & Databases

Databases Lecture 7:

Intro to PHP

Fiona McNeill

4th March 2019

LEADERS IN IDEAS AND SOLUTIONS

HERIOT WATT UNIVERSITY

LEADERS IN IDEAS AND SOLUTIONS

Today's lab

HERIOT WATT UNIVERSITY

Continue with SQL query sheet

- You should keep going with the sheet from last week
- After feedback, I've added some hints and tips which should help if you're finding it hard
- Do ask the lab helpers and/or me if you're stuck

HERIOT WATT UNIVERSITY

Then go on to the next sheet

- If you done all of last week's sheet **including the challenges** you can go on to the next sheet.
- This is looking at relational diagrams and translating them into SQL.
- We will have a PHP lab next week, where you can put into practice what we are learning this week.

HERIOT WATT UNIVERSITY

Assignment

LEADERS IN IDEAS AND SOLUTIONS

HERIOT WATT UNIVERSITY

Assignment

- For the assignment, you have to design, build, populate and query a database, and create a simple web interface to it.
- This is due on Wednesday, 27th March (Week 12) and should be submitted via Vision.
- You will do it in pre-assigned groups.

HERIOT WATT UNIVERSITY

Assignment

- For the assignment, you will be using SQL and PHP.
- You can start on the SQL bit straight away, but don’t attempt the PHP part until after next Monday’s lab, unless you already have PHP experience.

HERIOT WATT UNIVERSITY

Groups

- You will find the group list on Vision. These are the groups you must do the assignment in.
- The groups are organised so that:
  - There are mostly 4 (occasionally 3) in a group
  - Every group has at least 3 students with good attendance records.

HERIOT WATT UNIVERSITY

Groups

- As a group, you will assign effort marks to each member of the group and this will affect your scoring (more details will be given nearer submission time)
- If only three people in a group have an effort score > 0 ==> you will get a 5% bonus
- If only two people in a group have an effort score > 0 ==> you will get a 10% bonus
- If only one person is doing anything ==> get in touch with me asap!

HERIOT WATT UNIVERSITY

Groups

- Get in touch with your group soon - it is every individual's responsibility to contact her/his group.
- If you can't get hold of someone, let me know asap
- Some students with poor attendance records may in fact not be taking the course, so you will be in a group of three. **Let me know about this.**

HERIOT WATT UNIVERSITY

PHP

LEADERS IN IDEAS AND SOLUTIONS

HERIOT WATT UNIVERSITY

What is PHP*?

- PHP is a server scripting language, for creating dynamic and interactive webpages.
- It's handy if you want to get things going quickly.
- It is commonly used for accessing and updating databases via a web interface.
- PHP is *embedded in HTML*. It is used to process and output *data*.

*inspiration for this lecture taken from w3schools

HERIOT WATT UNIVERSITY

Why are doing PHP in a database course?

- PHP is not about databases, and you don't need to know anything about it to understand databases.
- But … PHP is a very common way of accessing databases through web portals

HERIOT WATT UNIVERSITY

Why are doing PHP in a database course?

- Remember that this course is a *practical* course designed to help you build and use databases.
- Doing a bit of PHP will allow you to create databases that people can query through the web, which is an important skill.
- You'll get to play around with this in your assessment.

HERIOT WATT UNIVERSITY

Is PHP for accessing databases?

- Not exclusively - you can use it for lots of things.
- But it is a common way to access databases via a web portal.

HERIOT WATT UNIVERSITY

PHP files

- Can contain text, HTML, CSS and Javascript as well as PHP
- Are executed on the server. The result is returned to the server as HTML.
- Have *.php* extension

HERIOT WATT UNIVERSITY

The PHP language

- PHP has a c-like syntax, similar to other languages you know
- Every line ends in a semi-colon;
- I'm not going to go into the details of the language here - the point is not to learn the language (we don't have time for that) but to understand how it's used to access databases.

HERIOT WATT UNIVERSITY

What can PHP do?

- PHP can generate dynamic page content
- PHP can create, open, read, write, delete, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database
- PHP can be used to control user-access
- PHP can encrypt data

HERIOT WATT UNIVERSITY

An example

```
<?php
echo "Hello World!";
?>
```

Either **echo** or **print** can be used for returning values. Echo is faster and more common.

HERIOT WATT UNIVERSITY

An example - embedding in HTML

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

The PHP syntax is embedded in html

HERIOT WATT UNIVERSITY

An example - embedding in HTML

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

This is the PHP

HERIOT WATT UNIVERSITY

PHP variables

```
<?php
$x = 3;
$y = 7;
$txt = "These are my variables";
?>
```

HERIOT WATT UNIVERSITY

PHP variables

```
<?php
$x = 3;
$y = 7;
$txt = "These are my variables";
?>
```

Variables begin with $, followed the name of the variable

HERIOT WATT UNIVERSITY

PHP variables

```
<?php
$x = 3;
$y = 7;
$txt ="These are my variables";
?>
```

Variables begin with $, followed the name of the variable

You then assign values to variables (put text in quotation marks)

HERIOT WATT UNIVERSITY

PHP variables

- Are weakly typed
- Don’t have to be declared before use.

HERIOT WATT UNIVERSITY

Printing out variables

```
<?php
$txt1 = "I'm learning PHP";
$txt2 = "maths";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "I can do " . $txt2 . "<br>";
echo $x . " + " . $y " = ";
echo $x + $y;
?>
```

HERIOT WATT UNIVERSITY

Printing out variables

```
<?php
$txt1 = "I'm learning PHP";
$txt2 = "maths";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "I can do " . $txt2 . "<br>";
echo $x . " + " . $y " = ";
echo $x + $y;
?>
```

You put html commands and strings in quotation marks

HERIOT WATT UNIVERSITY

Printing out variables

```
<?php
$txt1 = "I'm learning PHP";
$txt2 = "maths";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "I can do " . $txt2 . "<br>";
echo $x . " + " . $y " = ";
echo $x + $y;
?>
```

You put html commands and strings in quotation marks.
Dot is the string concatenation operator.

HERIOT WATT UNIVERSITY

PHP comments

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!”;
#you can comment using hash
// or double forward slash
/* forward slash - star
Allows you to comments over multiple lines
*/
?>

</body>
</html>
```

HERIOT WATT UNIVERSITY

PHP comments

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!”;
#you can comment using hash
// or double forward slash
/* forward slash - star
Allows you to comments over multiple lines
*/
?>

</body>
</html>
```

HERIOT WATT UNIVERSITY

PHP comments

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!”;
#you can comment using hash
// or double forward slash
/* forward slash - star
Allows you to comments over multiple lines
*/
?>

</body>
</html>
```

HERIOT WATT UNIVERSITY

Commands not case sensitive

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!”;
Echo "Hello World!”;
ECHO "Hello World!”;
?>

</body>
</html>
```

All of the above commands do the same thing.

HERIOT WATT UNIVERSITY

But variables are …

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
$name = "Fiona";
echo "My name is " . $name . ”<br>";
echo "Your name is " . $Name . ”<br>";
echo "Everyone's name is " . $naMe . “<br>”;
?>

</body>
</html>
```

Only the first will work

HERIOT WATT UNIVERSITY

For loops

```
<html>
    <body>
        <p>
            <?php
            for ($i = 1; $i <= 10; ++$i)
            {
                echo $i;
            }
            ?>
        </p>
    </body>
</html>
```

For loops have *initial conditions*

HERIOT WATT UNIVERSITY

For loops

```
<html>
    <body>
        <p>
            <?php
            for ($i = 1; $i <= 10; ++$i)
            {
                echo $i;
            }
            ?>
        </p>
    </body>
</html>
```

For loops have initial conditions, *test cases*

HERIOT WATT UNIVERSITY

For loops

```
<html>
    <body>
        <p>
            <?php
            for ($i = 1; $i <= 10; ++$i)
            {
                echo $i;
            }
            ?>
        </p>
    </body>
</html>
```

For loops have initial conditions, test cases and *increments*

HERIOT WATT UNIVERSITY

For loops

```
<html>
    <body>
        <p>
            <?php
            for ($i = 1; $i <= 10; ++$i)
            {
                echo $i;
            }
            ?>
        </p>
    </body>
</html>
```

For loops have initial conditions, test cases and *increments* separated by semi-colons.

HERIOT WATT UNIVERSITY

For loops

```
<html>
    <body>
        <p>
            <?php
            for ($i = 1; $i <= 10; ++$i)
            {
                echo $i;
            }
            ?>
        </p>
    </body>
</html
```

What will this loop produce?

HERIOT WATT UNIVERSITY

For loops

```
<html>
    <body>
        <p>
            <?php
            for ($i = 1; $i <= 10; ++$i)
            {
                echo $i;
            }
            ?>
        </p>
    </body>
</html
```

What will this loop produce?
12345678910

HERIOT WATT UNIVERSITY

For loops

```
<html>
    <body>
        <p>
            <?php
            for ($i = 1; $i <= 10; ++$i)
            {
                echo "<b>Hello</b>! - $i <br/>";
            }
            ?>
        </p>
    </body>
</html>
```

As we have seen, we can embed html commands within php code.

HERIOT WATT UNIVERSITY

For loops

```
<html>
    <body>
        <p>
            <?php
            for ($i = 1; $i <= 10; ++$i)
            {
                echo "<b>Hello</b>! - $i <br/>";
            }
            ?>
        </p>
    </body>
</html>
```

Hello - 1
Hello - 2
Hello - 3
Hello - 4
Hello - 5
Hello - 6
Hello - 7
Hello - 8
Hello - 9
Hello - 10

As we have seen, we can embed html commands within php code.

HERIOT WATT UNIVERSITY

For loops

```
<html>
    <body>
        <p>
            <?php
            for ($i = 1; $i <= 10; ++$i)
            {
            ?>
            <b>Hello</b>! - <?php echo $i; ?>
            <br/>
            <?php
            }
            ?>
        </p>
    </body>
</html>
```

Hello - 1
Hello - 2
Hello - 3
Hello - 4
Hello - 5
Hello - 6
Hello - 7
Hello - 8
Hello - 9
Hello - 10

Another way to do this would be to intersperse php code with html. The bits in bold are the php commands.

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

- A handy thing you might want to do with PHP is to use it to manipulate data input by users.
- This is done through html forms

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

```
<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>
```

This is an HTML form

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

```
<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>
```

The input to the form is sent for processing by a php method - in this case, in the file **welcome.php**.

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

```
<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>
```

There are two methods for doing this: **post** and **get**. This form is using the post method.

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

```
<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>
```

The user will see two prompts: **name** and **e-mail**.

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

```
<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>
```

Next to the prompts will be boxes expecting input. The inputs to both these boxes should be text, and the inputs will be assigned to variables **name** and **email**.

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

```
<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>
```

The type of input is **submit** - this means that a ‘submit’ button will appear and once the user presses it, the input is sent to be processed.

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

So what are we going to do with the data? This information will be contained in the file welcome.php.

```
<html>
<body>

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

</body>
</html>
```

Again, the PHP is wrapped in HTML. This is because the PHP is processing the data and the HMTL is displaying it.

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

So what are we going to do with the data? This information will be contained in the file welcome.php.

```
<html>
<body>

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

</body>
</html>
```

Two lines are printed with the above text on, on different lines.

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

So what are we going to do with the data? This information will be contained in the file welcome.php.

```
<html>
<body>

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

</body>
</html>
```

Next to ‘welcome’, PHP will echo (essentially, return) the variable that is assigned in the post operation to the variable **name.** **Email** will be printed on the line below.

HERIOT WATT UNIVERSITY

Using PHP with HTML forms

So what are we going to do with the data? This information will be contained in the file welcome.php.

```
<html>
<body>

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

</body>
</html>
```

This indicates that the Post method has been used

HERIOT WATT UNIVERSITY

PHP Forms - another example

```
<form action = "lawn.php" method = "post" >
<table>
  <tr>
   <td> Length </td>
   <td> <input type = "text" name = "lawn_length"
               size = "5" /> </td>
  </tr>
  <tr>
    <td> Width </td>
    <td> <input type = "text"  name = "lawn_width"
          size = "5"  /> </td>
  </tr>
</table>
<p> <input type="submit" value="submit" /> </p>
</form>
```

What is this doing?

HERIOT WATT UNIVERSITY

PHP Forms - another example

```
<form action = "lawn.php" method = "post" >
<table>
  <tr>
   <td> Length </td>
   <td> <input type = "text" name = "lawn_length"
               size = "5" /> </td>
  </tr>
  <tr>
    <td> Width </td>
    <td> <input type = "text"  name = "lawn_width"
          size = "5"  /> </td>
  </tr>
</table>
<p> <input type="submit" value="submit" /> </p>
</form>
```

It is sending data via the **post** method to be processed by the file **lawn.php.**

HERIOT WATT UNIVERSITY

PHP Forms - another example

```
<form action = "lawn.php" method = "post" >
<table>
  <tr>
   <td> Length </td>
   <td> <input type = "text" name = "lawn_length"
               size = "5" /> </td>
  </tr>
  <tr>
    <td> Width </td>
    <td> <input type = "text"  name = "lawn_width"
          size = "5"  /> </td>
  </tr>
</table>
<p> <input type="submit" value="submit" /> </p>
</form>
```

The user will see two prompts, asking for a **width** and a **length.**

HERIOT WATT UNIVERSITY

PHP Forms - another example

```
<form action = "lawn.php" method = "post" >
<table>
  <tr>
   <td> Length </td>
   <td> <input type = "text" name = "lawn_length"
              size = "5" /> </td>
  </tr>
  <tr>
    <td> Width </td>
    <td> <input type = "text"  name = "lawn_width"
          size = "5"  /> </td>
  </tr>
</table>
<p> <input type="submit" value="submit" /> </p>
</form>
```

These will be next to two input boxes of **size 5**, which will save **text** to two variables, **lawn_length** and **lawn_width.**

HERIOT WATT UNIVERSITY

PHP Forms - another example

```
<form action = "lawn.php" method = "post" >
<table>
  <tr>
   <td> Length </td>
   <td> <input type = "text" name = "lawn_length"
               size = "5" /> </td>
  </tr>
  <tr>
    <td> Width </td>
    <td> <input type = "text"  name = "lawn_width"
          size = "5"  /> </td>
  </tr>
</table>
<p> <input type="submit" value="submit" /> </p>
</form>
```

There will be a submit button which will be the trigger to send the data to be processed.

HERIOT WATT UNIVERSITY

PHP Forms - another example

```
<form action = "lawn.php" method = "post" >
<table>
  <tr>
   <td> Length </td>
   <td> <input type = "text" name = "lawn_length"
              size = "5" /> </td>
  </tr>
  <tr>
    <td> Width </td>
    <td> <input type = "text"  name = "lawn_width"
          size = "5"  /> </td>
  </tr>
</table>
<p> <input type="submit" value="submit" /> </p>
</form>
```

The rest of the HTML is creating a table.

HERIOT WATT UNIVERSITY

PHP Forms - another example

Lawn.php

```
<html>
<head>
<title>Displaying the lawn area</title>
</head>
<body>
<p> The area is
<?php
$width = $_POST["lawn_width"];
$length = $_POST["lawn_length"];
print $width*$length;
?>
Sq m. </p>
</body>
</html>
```

HERIOT WATT UNIVERSITY

PHP Forms - the GET method

```
<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>
```

This is exactly the same HTML code as for the POST method - the difference lies in what happens in the PHP

HERIOT WATT UNIVERSITY

PHP Forms - the GET method

What goes on inside the GET method?

```
<html>
<body>

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

</body>
</html>
```

Again, this is the same code except we use the GET method instead of the POST method.

HERIOT WATT UNIVERSITY

PHP Forms - POST v GET

So what is the difference between these two approaches?

- **Get** attaches the input data to the URL as name/value pairs
  The URL will be …lawn.php?lawn_length=2.1&lawn_width=10
- **Post** sends the input data within a message as name/value pairs
  The URL will be …lawn.php
- This means that the parameters of **get** are visible to everyone; the parameters of **post** are not. However, this does not mean that post is secure!
- **Get** is good for things like returning google searches but should not be used for, e.g., information about users names and passwords.

HERIOT WATT UNIVERSITY

Security

- If you are getting people to send information in forms, you need to think about security.
- You need to wrap the things you send in a security function when writing PHP, otherwise you are at risk from hackers and spammers.
- PHP looks simple but you can get into trouble quickly if you don’t know what you’re doing.

HERIOT WATT UNIVERSITY

Summary

- Today, we looked at the basic ideas behind PHP and how to use PHP with HTML forms.
- On Wednesday we will look at accessing databases using PHP.