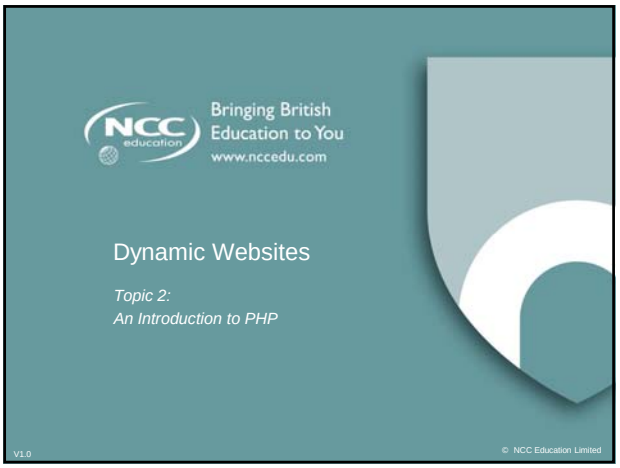
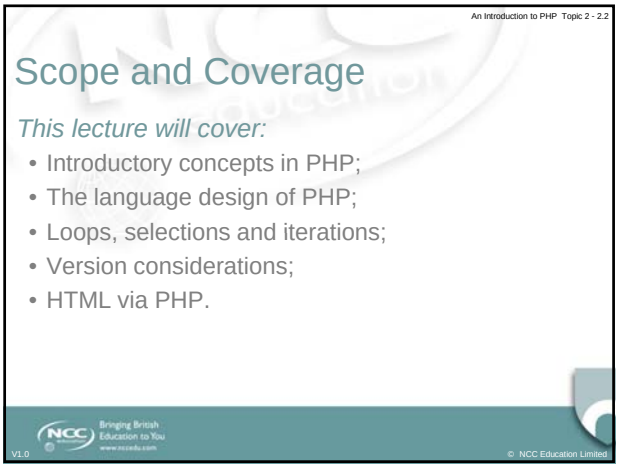
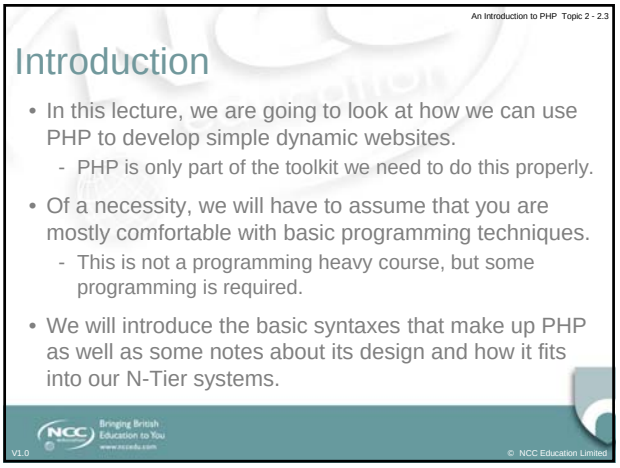

Bringing British
Education to You
www.nccedu.com
Dynamic Websites
Topic 2:
An Introduction to PHP
V1.0
© NCC Education Limited


An Introduction to PHP Topic 2 - 2.2
Scope and Coverage
This lecture will cover:
• Introductory concepts in PHP;
• The language design of PHP;
• Loops, selections and iterations;
• Version considerations;
• HTML via PHP.
V1.0
© NCC Education Limited


An Introduction to PHP Topic 2 - 2.3
Introduction
• In this lecture, we are going to look at how we can use PHP to develop simple dynamic websites.
- PHP is only part of the toolkit we need to do this properly.
• Of a necessity, we will have to assume that you are mostly comfortable with basic programming techniques.
- This is not a programming heavy course, but some programming is required.
• We will introduce the basic syntaxes that make up PHP as well as some notes about its design and how it fits into our N-Tier systems.
V1.0
© NCC Education Limited

PHP - 1

- PHP is a ***server side scripting language***.
  - You request a page on the internet.
  - The server interprets the PHP it has been given.
  - It returns the results of that interpretation to you as an HTML page.
- PHP makes use of the general structure of HTTP on the internet.
  - As such, it suffers from the same limitations as HTML, primarily ***statelessness***.

PHP - 2

- PHP programs are written in a different way to desktop applications. You need several tools:
  - A web-server with PHP installed.
    - That will be taken care of for you.
  - Some kind of programming environment.
    - Normally we write PHP code using a simple text editor (not a word processor)
    - Some good choices for this are Notepad++ or Jedit.
  - An internet browser to interact with the application.
    - Any of these will be fine for now.

A PHP Script

```
<html>
  <head>
    <title>My First PHP Script</title>
  </head>
  <body>

  <?
    echo "<p>Hello World!</p>"
  ?>

  </body>
</html>
```

My First PHP Script

- PHP works like standard HTML, except you can set sections of the page to be interpreted by the server.
- PHP sections are marked by blocks.
  - <? starts a block of PHP
  - ?> ends a block of PHP
  - All of your PHP code goes in this block.
- The echo function is used to output some text to the browser.
  - This script will display the text “Hello World” in a browser.

The Produced HTML

- We never see the PHP in our browser, because the processing is done on the server.
  - What we get back is the processed HTML:

```
<html>
  <head>
    <title>My First PHP Script</title>
  </head>
  <body>

  <p>Hello World!</p>
  </body>
</html>
```


User Input

- As with all programming, it is important that we are able to get and manipulate user information.
- This is handled in PHP through the use of ***form elements***.
- We create an HTML page that links to our PHP script, and when the form element is triggered, its information will be passed to the script.
- Note that this page uses no PHP itself.
  - It is the ***front-end*** to our PHP script.

HTML Forms

This is the PHP page that will be loaded when the user hits the 'ask' button.

```
<html>
 <head>
  <title>Test Form</title>
 </head>
 <body>
 <form action = "test_variables.php" method = "post">

   <p>What is your name?</p>
   <input type = "text" name = "name">
   <p>What is your question?</p>
   <input type = "text" name = "question">

   <input type = "submit" value = "ask">
   <input type = "reset" value = "Clear values">

 </body>
</html>
```


Variables - 1

- When the user presses 'ask', the browser will send the info they have entered into the textboxes to the page test_variables.php.
- We will not do much with them yet.
  - We will just print them out to the screen.
- Before we do that, we need to talk a little bit about ***variables*** in PHP.
  - These work differently depending on what version of PHP you are using.

Variables - 2

- The concept of variables in PHP is identical to that in other languages – they let us deal with the unknown.
  - For example, we do not know what a user will type for their name or for their question.
- In PHP, variable names are always preceded by a $.
  - Such as $myVariable

Variables - 3

- We do not provide the type of a variable.
  - Just a name.
- When the browser sends the contents of our text boxes to the PHP script, it provides them as part of a hash table it maintains called $_POST.
  - The elements have the same name as we give them in the form elements.

Test_variables.php

```
<<html>
 <head>
  <title>Testing Variables</title>
 </head>

 <body>

 <?
  $name = $_POST["name"];
  $question = $_POST["question"];

  echo "<p>You entered $name for the name.</p>";
  echo "<p>You entered $question for the question.</p>";

 ?>

 </body>
</html>
```


Why Use PHP? - 1

- Because it is quick to setup an interface.
  - As you can see, input and output are simple to accomplish.
- HTML is a very rich output language.
  - You can lay things out in PHP much better than you can in any other programming language.
  - This is because rendering the output is handled on the client, and not in our PHP.
    - We simply provide our output as HTML.

Why Use PHP? - 2

- Database connectivity is built into the core of the language.
  - It is very easy to hook up to a database.
- It is quite easy to learn.
  - Lots of the complicated things that are present in other languages are simplified.

Why Not Use PHP?

- It is designed for running over the internet, with all the complications that brings.
- Architecturally, it has numerous disadvantages compared to more strict programming languages.
- It is hard to find good 'example' programs.
- Persistent data representation requires the use of other applications.
  - Like mySQL

Some More PHP

- Let us look at doing something a little more complicated in PHP.
  - A program that answers our questions.
- We need to use arrays to handle this.
- PHP does not distinguish between variables of one type, and variables of another in code.
  - They are all just 'variables'.
- In technical terms, it is ***loosely typed***.
  - This means you have to be careful.

The Magic Eight Ball - 1

- Our program is going to take questions from users, and then give random answers.
  - Much like with a 'magic eight ball'.
- We declare an array of possible answers using the ***array*** keyword:

```
$responses = array (
  "I have no idea.",
  "I don't know why you're asking me, I don't know.",
  "Please stop asking questions, I don't know.",
  "That's an interesting question.  I don't know the
answer.",
);
```


The Magic Eight Ball - 2

- When we get a question, we do not really care what the question is.
  - We just care that a random answer is given.
- The items inside a list (or an array) are identified by a numeric index.
  - The first element in an array is identified by the index 0, the second by 1, and so on.
- Programmers start counting from zero, which is useful to remember.
  - Thus, if we wanted to always give the first answer:
    - $answer = $responses[0];

Picking A Random Number

- If we want to get a random index from an array, we do it like so:
  - $random_response = array_rand ($responses);
- Array_rand is a function that is built in to PHP, we do not need to write it ourselves.
- With this line of code, the variable $random_response contains a valid random index number.

The Magic Eight Ball - 3

```
<html>
  <head>
    <title>Magic Eight Ball</title>
  </head>

  <body>
  <?
    $responses = array (
      "I have no idea.",
      "I don't know why you're asking me, I don't know.",
      "Please stop asking questions, I don't know.",
      "That's an interesting question.  I don't know the answer.",
    );
    $random_response = array_rand ($responses);
    $answer = $responses[$random_response];
    $name = $_POST["name"];
    echo "<p>I have an answer for you, $name - $answer</p>"
  ?>
  </body>
</html>
```


Loops in PHP -1

- PHP offers the full complement of loops for you to use.
  - Syntactically these are very similar to C/Java, except that variables are referenced with the $ notation.

```
<?
   $i = 0;
   while ($i < 10) {
     echo "<p>The number is " . $i . "</p>";
     $i += 1;
   }
 ?>
```

While Loop

Loops in PHP - 2

- Note here too that we are using a slightly different way of outputting the values.
  - This is not specific to for loops, it is just to show you different ways of accomplishing the same thing.
  - The dot is the ***concatenation*** operator

```
<?
   for ($i=0; $i < 10; $i++) {
     echo "<p>The number is " . $i . "<p/>";
   }
 ?>
```

For Loop

Loops in PHP - 3

- There are two other kinds of loop in PHP that can be useful.
  - The do-while loop, which is common to most programming languages.
  - The foreach loop, which is slightly more unusual.
- You are invited to research these loops for yourself.
  - We do not have enough time in the module to go over all the syntax you might find useful or interesting.

Selection in PHP

- As with for and while loops, the syntax for selection in PHP is syntactically similar to C/Java.

```
<?
   for ($i=0; $i < 10; $i++) {
     if ($i % 2 == 0) {
       echo "<p>The number is " . $i . " and it's even.</p>";
     }
     else {
       echo "<p>The number is " . $i . " and it's odd.</p>";
     }
   }
 ?>
```

Selection

String Comparison in PHP

- You might be tempted to use the == operator to compare strings in PHP.
  - You will not get the behaviour you want doing that.
  - The == in PHP is called a ***loose comparison operator.***
    - It tries to do some ***type juggling*** to make sure a comparison between two types of data is sensible.
  - PHP offers ***strict comparison operators*** too
    - ===
    - !==
  - These should be used for string comparisons.

Type Casting

- While PHP is loosely typed, it is often still valuable to be able to change the contents of a variable from one type to another.
- This is done through type casting:
  - $num = 10
  - $strnum = (string)$num;
- You need to keep track of what is contained within variables.
  - It is a good idea to be consistent with your typing.

PHP and Version Differences - 1

- In an early slide, the point was made that it is difficult to find good example programs.
  - Part of that problem is due to version and configuration differences.
- PHP is a very flexible language, but it changes much depending on its context and version.
- During this course, we will assume you are using version 5 of PHP.
  - Make sure that any example code you research is also using PHP version 5.

PHP and Version Differences - 2

- PHP allows for much to be configured on a server.
  - For example, it is possible to configure PHP so that the work of getting form elements from $_POST is handled for you.
- If example code you research does not work on your system, it may be because of configuration differences.
  - The example code that we discuss in this module should work in all contexts.

Conclusion

- PHP is a C-Type language.
  - The syntax is syntactically very similar to C, C++ and Java.
- It is a server-side scripting language.
  - All the processing of the code is done on the server side.
- We can make use of the fact our output goes to a browser by using HTML markup.
  - This greatly increases how effective our input and output can be.
- There are often substantial version differences between installations of PHP on a server.
  - You need to be careful of this.

Terminology

- ***Loosely typed***
  - A programming language that does not require the type of variables to be declared.
- ***Type juggling***
  - The automatic type conversions that PHP performs.
- ***Type casting***
  - Changing the type of a variable from one kind of data to another.

Topic 2 – Introduction to PHP

Any Questions?