2021

Web Coding

PHP & SQL

Learn PHP and SQL, the languages behind many of the world's database driven websites.

Checklist

□	1. PHP Programming
□	2. Introduction to the Project
□	3. Setting Up your Website
□	4. Using PHP in a Webpage
□	5. Query Strings
□	6. Online Databases
□	7. Database Connections
□	8. Listing Database Results
□	9. Displaying an Item
□	10. Adding Images
□	11. Editing the Data
□	12. Other Form Fields
□	13. Data Formatting
□	14. Website Security
□	15. Completing your Website

1

PHP Programming

Web Coding
PHP & SQL

In this project, we will be using the PHP programming language to generate pages for the web. We are going to assume that you have a little programming experience and begin by recapping a few ideas quickly. Basic programming is very similar in most general languages; there are data types such as strings and numbers, as well as variables, arrays, ‘if…else’ statements and loops. If you struggle with these initial tasks then it might be wise to get a gentler introduction to coding first. Finally, some experience with setting up a database would be useful, but not essential.

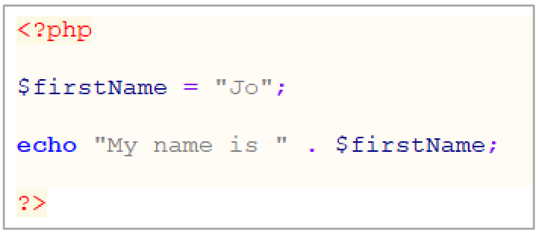
Note: *If you do need a basic introduction to programming then ORB Education offers a number of publications to help, such as coding in Python, VB and JavaScript, as well as resources for HTML/CSS and Access Databases.*

Task 1 – Getting Started with PHP

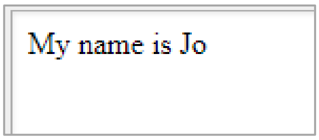
a. Although this project is about creating web pages, we will build a few PHP scripts in an online editor to begin with. These editors offer an easy way to test bits of programming without the need to serve up webpages. Find yourself an online PHP editor to use (at the time of writing phptester.net and www.writephponline.com are both good options).

b. Copy the code on the right and run the script. You should get the output shown. If not, check each character carefully.

c. Copy this script to a text file for your records, saved as **PHP Scripts.txt**.



```
<?php
$firstName = "Jo";
echo "My name is " . $firstName;
?>
```



```
My name is Jo
```

d. Answer the following questions about the script.

1. What code do you think tells the browser to start using PHP? ____________________

2. What symbol is placed at the start of the *firstName* variable? ____________________

3. What symbol is placed at the end of each line or statement? ____________________

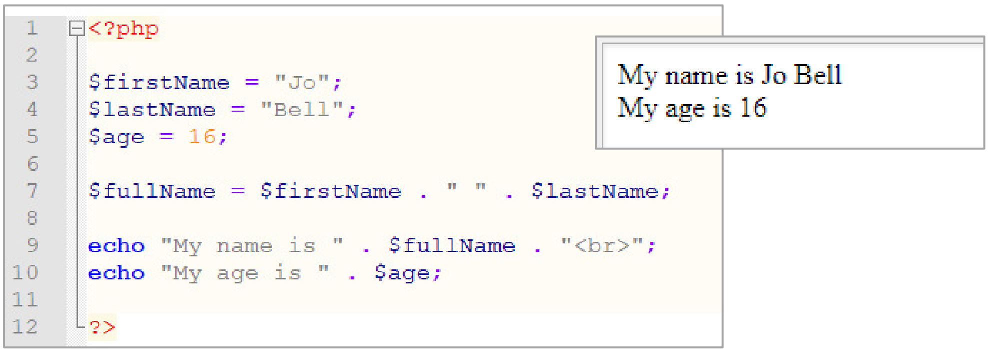
4. What symbol is used to join (concatenate) the two pieces of text? ____________________

5. What code tells the browser to stop using PHP? ____________________

6. What do you think the *echo* statement does? ____________________

Note: *In the online editors, neither the first or last lines actually need to be typed and may be removed if you include them.*

e. Now try and get this script up and running. Add the code to your **PHP Scripts.txt** file, as before.



```
<?php

$firstName = "Jo";
$lastName = "Bell";
$age = 16;

$fullName = $firstName . " " . $lastName;

echo "My name is " . $fullName . "<br>";
echo "My age is " . $age;

?>
```

```
My name is Jo Bell
My age is 16
```

PHP Programming (page 2)

Task 1 – Getting Started with PHP (Cont.)

f. Answer these questions about the new script.

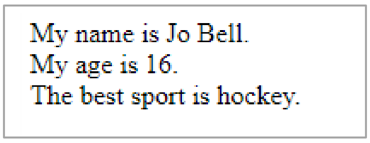
1. Name the 4 variables used (include the $ signs). ______________________________
2. Which HTML tag have we used (HTML tags are enclosed in < and > symbols)? ______________________________
3. What is the purpose of the double quote marks on line 7? ______________________________
4. How is declaring a string (e.g. Jo or Bell) different to declaring a number (e.g. 16)?

   ____
5. What data types are being concatenated on line 10? ______________________________
6. Run a test to find out if PHP is case sensitive. Is the variable *$age* different to *$Age*, for example? ____________

g. Add to the script so that it also displays some other information stored in variables, such as your favourite sport or band.

Edit the code so that a full stop is shown at the end of each line.

Copy this script (and all those that follow in these initial tasks) to your scripts file.



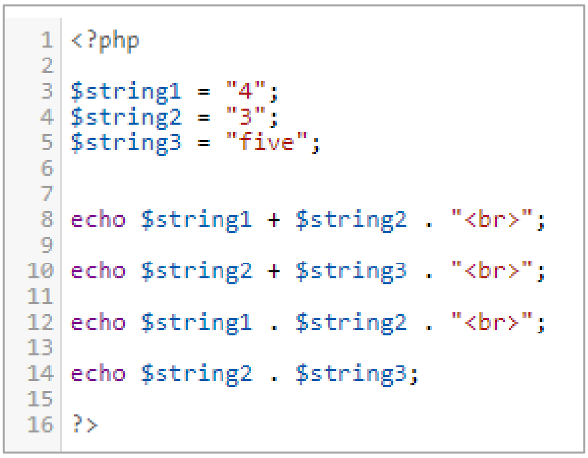
```
My name is Jo Bell.
My age is 16.
The best sport is hockey.
```

Naming variables in PHP

- Start with a $ sign, followed by the actual name.
- The actual name must begin with a letter or underscore, rather than a number or other symbol.
- The name must be made up of only letters, numbers and the underscore.
- Variable names are case-sensitive.
- It's not a rule but camelCase is often used for variables in PHP. compoundWords withA lowerCase firstWord thenA capitalisedSecond.

Task 2 – Data Types

a. Test the script on the right. What does it tell us about data types? Think about strings, numbers and which are being added or concatenated.



```php
<?php

$string1 = "4";
$string2 = "3";
$string3 = "five";


echo $string1 + $string2 . "<br>";

echo $string2 + $string3 . "<br>";

echo $string1 . $string2 . "<br>";

echo $string2 . $string3;

?>
```

b. As we have found above, PHP has the ability to change the data type depending on the context. However, when working with databases it is still sometimes necessary to force data conversions. Match the conversions below to the code.

Conversion			PHP Code
String to integer	●	●	$var2 = (string)$var1;
Number to string	●	●	$var2 = (float)$var1;
String to decimal	●	●	$var2 = (int)$var1;

4

Using PHP in a Webpage

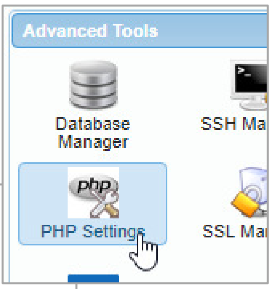
Web Coding

PHP & SQL

With our website in place serving up simple (static) webpages, it's now time to introduce the PHP programming.
PHP is one of the many programming languages that can be used to program a website. To run the PHP code, the file must be located on the webserver; you cannot serve up the file on your own computer unless you install your own webserver.

Task 1 – Testing PHP

a. The first step is to ensure that your webserver is able to run PHP code. In AwardSpace, it is set up by default. Select '**Dashboard / Advanced Tools /PHP Settings**' to check the version being used.

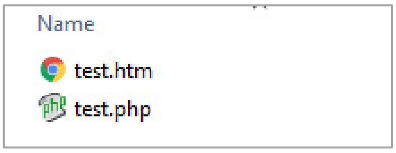


b. Use your text editor to create the code on the right (you may start with a copy of your htm file).

If you are using Notepad++, select PHP from the *Languages* menu to colour the code. Save the file as ***test.php***.

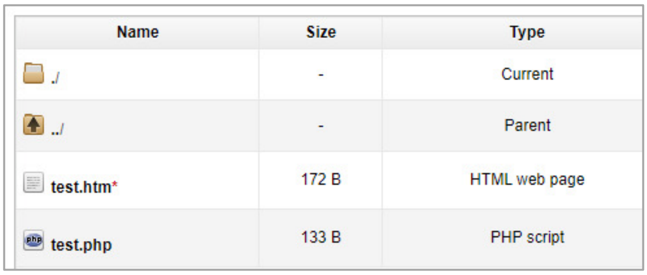
```
<html>
 <head>
  <title>Testing PHP</title>
 </head>
 <body>

 <?php echo "We are testing PHP"; ?>

 </body>
</html>
```

Note: *If your file extensions are not on view then you may have problems changing an htm file to a php one. Use the setting in the 'View' tab (Windows folder) or 'Finder / Preferences / Advanced' (Mac) to display file extensions.*



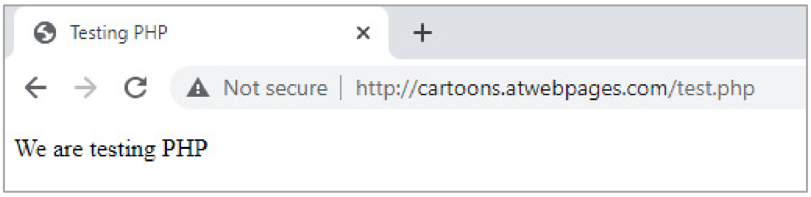
Name
test.htm
test.php

c. Upload the file to your root directory as before ('Dashboard / File Manager / double click to enter root directory' etc.).



Name	Size	Type
./	-	Current
../	-	Parent
test.htm*	172 B	HTML web page
test.php	133 B	PHP script

d. View the page in your browser. If everything is working well then the text 'We are testing PHP' should be displayed. If there is a problem, check that you have typed the code accurately and the URL is correct (including capitalisation). Reupload the file if necessary.



http://cartoons.atwebpages.com/test.php

We are testing PHP

Using PHP in a Webpage (page 2)

Task 2 – HTML in a PHP Page

In the initial activities, we wrote some PHP programs and outputted unformatted text to the screen using the echo function. The next step is to use PHP to control chunks of HTML in a page. For this, you might have different sections of HTML that are used at different times, or you might have sections of HTML that are used repeatedly.

Controlling HTML with IF statements

Page headers

HTML - top section

PHP - If this condition is met…

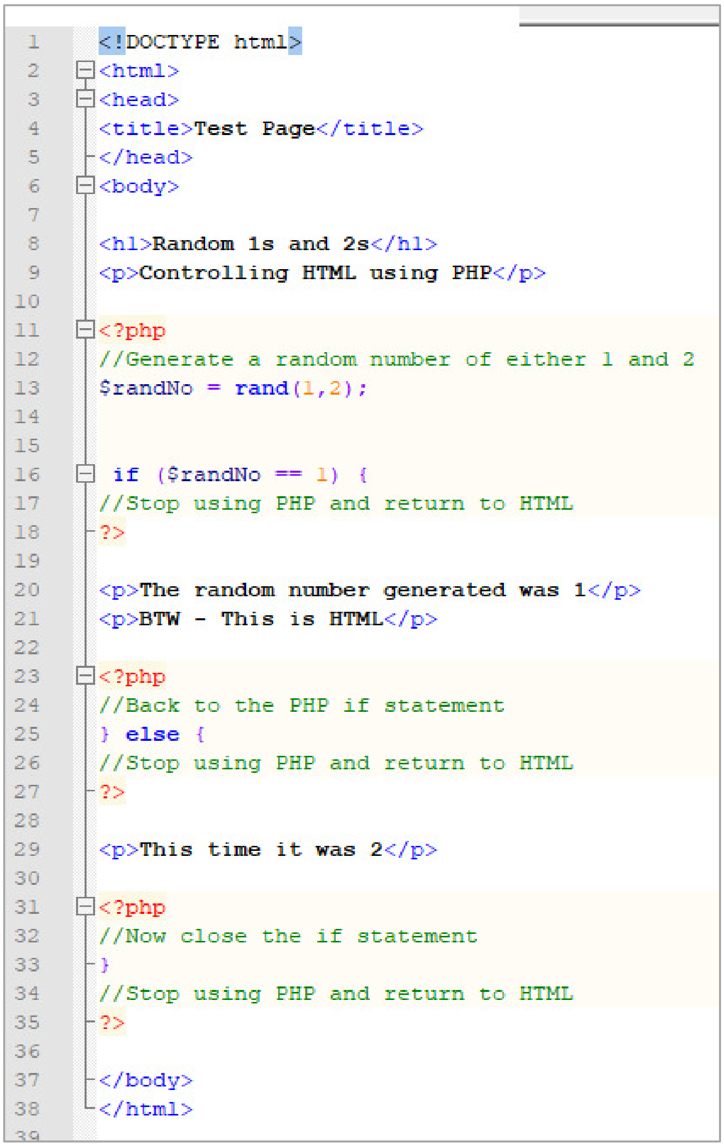
    HTML - Include this content

PHP - Else

    HTML - Include this content

PHP - End If

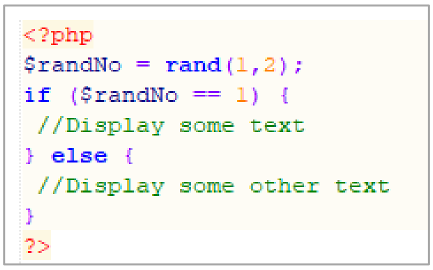
HTML - bottom section



```
<!DOCTYPE html>
<html>
<head>
<title>Test Page</title>
</head>
<body>

<h1>Random 1s and 2s</h1>
<p>Controlling HTML using PHP</p>

<?php
//Generate a random number of either 1 and 2
$randNo = rand(1,2);


 if ($randNo == 1) {
//Stop using PHP and return to HTML
?>

<p>The random number generated was 1</p>
<p>BTW - This is HTML</p>

<?php
//Back to the PHP if statement
} else {
//Stop using PHP and return to HTML
?>

<p>This time it was 2</p>

<?php
//Now close the if statement
}
//Stop using PHP and return to HTML
?>

</body>
</html>
```

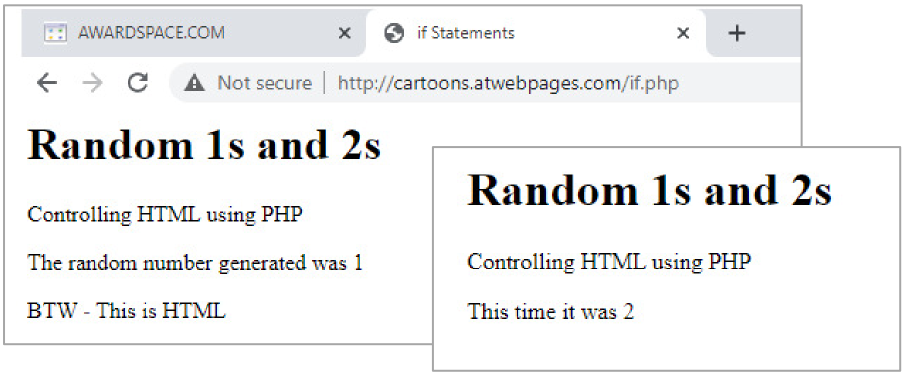
The program on the right uses the *rand* function to generate either the number 1 or 2 randomly. If a 1 is generated then some text is displayed on the screen. If a 2 is generated then some different text is displayed.

A simplified view of what is happening is shown below. It's the same program without the HTML. It's often easier to set up and test a program in this way first, then add the HTML later. It can be less confusing.



```
<?php
$randNo = rand(1,2);
if ($randNo == 1) {
 //Display some text
} else {
 //Display some other text
}
?>
```

Create the program and save as **if.php**. Once the page is open in your browser, reload it repeatedly (try the reload button in your browser or the *F5* key in Windows). You should get a random mixture of the two different pieces of text displayed.



Random 1s and 2s

Controlling HTML using PHP

The random number generated was 1

BTW - This is HTML

Random 1s and 2s

Controlling HTML using PHP

This time it was 2

8 Listing Database Results

With the database connection up and running, it's time to start building our database driven webpages. We will work through a task with the test data first then look at how apply the coding to your own webpages.

Task 1 – Looping through Results

Using your last solution as a starting point, create the page shown on the right. Save as **testpage1.php**.

The structure of the page is as below:

Head section (not shown)

HTML – headings and open table

HTML – first row

PHP - Query the database using SQL

PHP – Set up an *if statement* for results

PHP – Set up a loop

HTML - including PHP database data

PHP – Close loop (i.e. move to next record)

PHP – *else* no results

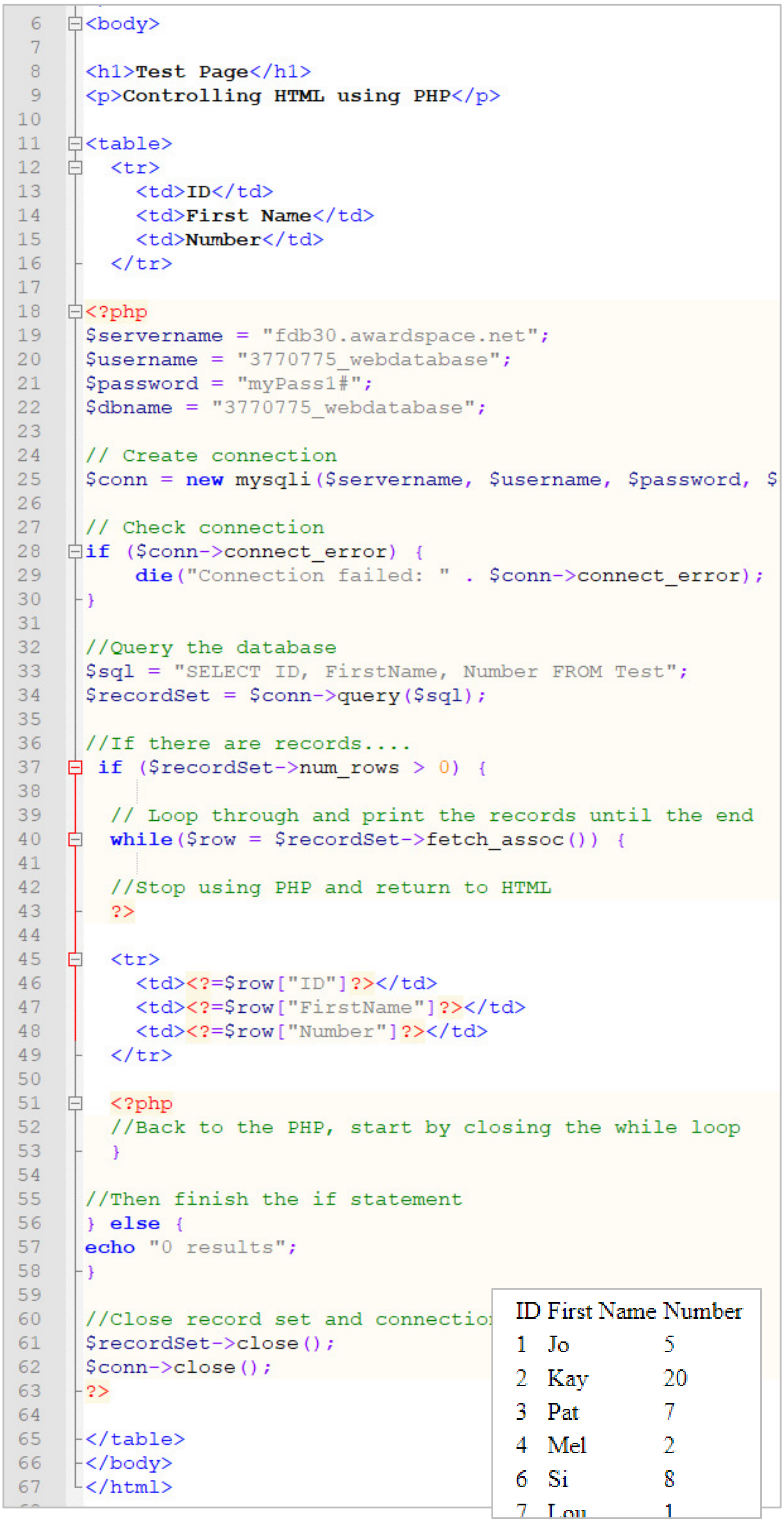
PHP – Close record set and connection

HTML – Close table, body and html

We have used a table here as we are presenting rows of data. Divs could easily be used instead.

Note that the table and first row of headings are set up before the PHP programming. Another row is added for each record returned from the database. The table is closed after the programming has been completed.

Code such as <?php echo $row["ID"]?> has been used to display data from the database in the table's data cells.



```
<body>

<h1>Test Page</h1>
<p>Controlling HTML using PHP</p>

<table>
  <tr>
    <td>ID</td>
    <td>First Name</td>
    <td>Number</td>
  </tr>

<?php
$servername = "fdb30.awardspace.net";
$username = "3770775_webdatabase";
$password = "myPass1#";
$dbname = "3770775_webdatabase";

// Create connection
$conn = new mysqli($servername, $username, $password, $

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

//Query the database
$sql = "SELECT ID, FirstName, Number FROM Test";
$recordSet = $conn->query($sql);

//If there are records....
 if ($recordSet->num_rows > 0) {

  // Loop through and print the records until the end
  while($row = $recordSet->fetch_assoc()) {

  //Stop using PHP and return to HTML
  ?>

  <tr>
    <td><?=$row["ID"]?></td>
    <td><?=$row["FirstName"]?></td>
    <td><?=$row["Number"]?></td>
  </tr>

  <?php
  //Back to the PHP, start by closing the while loop
  }

//Then finish the if statement
} else {
echo "0 results";
}

//Close record set and connectio
$recordSet->close();
$conn->close();
?>

</table>
</body>
</html>
```

ID	First Name	Number
1	Jo	5
2	Kay	20
3	Pat	7
4	Mel	2
6	Si	8
7	Lou	1

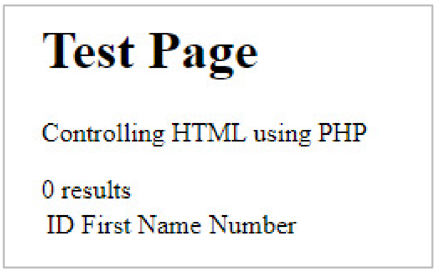
Listing Database Results (page 2)

Task 2 – Improving the Presentation

Once your page is up and running, save a copy of the file as **testpage2.php**. Try out some of the things below.

1. Change the SQL query so that no results are retrieved (‘e.g. …WHERE Number > 100’). You will find that the table and the first row of headings will be created whether there are any results or not (as shown on the right). Additionally, the PHP echo statement ‘0 results’ ends up being displayed above the table.

   Adapt the code so that the table is only created if there is data to present. This will require shifting around some sections of code. Also, if there aren’t any results then display the *0 results* message in a paragraph.



Test Page

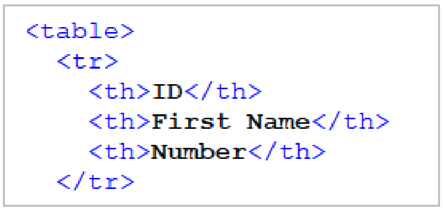
Controlling HTML using PHP

0 results

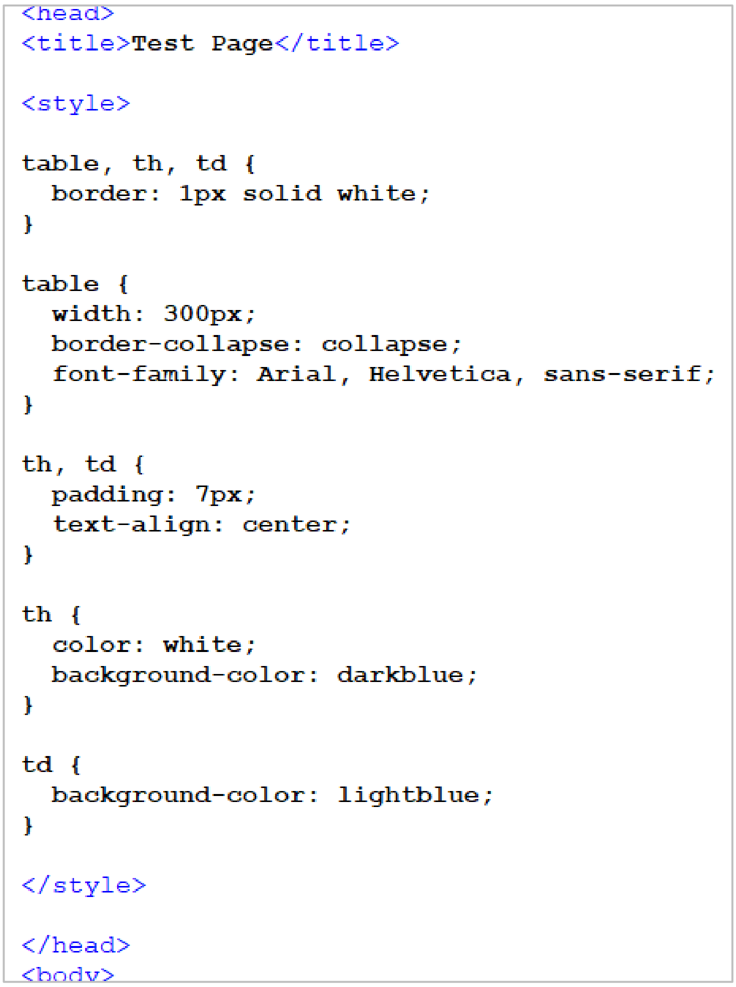
ID First Name Number

2. Add a CSS stylesheet to style your table. This can be placed at the top of the page in the head section at this point (see below-right). Note the following:

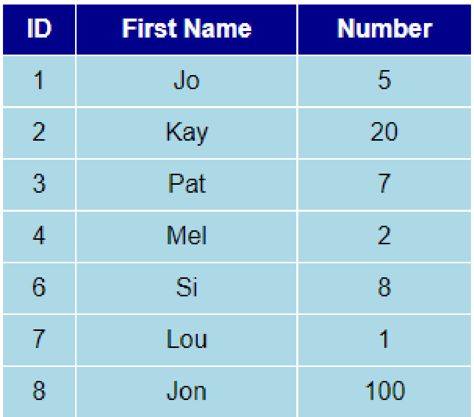
   - The HTML table header tag <th> can be used as an alternative to <td> as on the right. This allows you to style the top row of cells differently.
   - The selectors for the table, th and td tags can be set up as shown on the right. Some definitions target all three elements (table, th and td); others only target one or two.
   - Play with the design and see if you can come up with one you like. You can use elements of this in your own pages.



```
<table>
  <tr>
    <th>ID</th>
    <th>First Name</th>
    <th>Number</th>
  </tr>
```



```
<head>
<title>Test Page</title>

<style>

table, th, td {
  border: 1px solid white;
}

table {
  width: 300px;
  border-collapse: collapse;
  font-family: Arial, Helvetica, sans-serif;
}

th, td {
  padding: 7px;
  text-align: center;
}

th {
  color: white;
  background-color: darkblue;
}

td {
  background-color: lightblue;
}

</style>

</head>
<body>
```

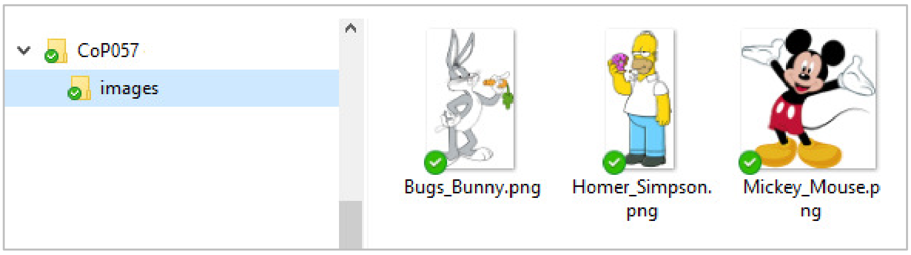


ID	First Name	Number
1	Jo	5
2	Kay	20
3	Pat	7
4	Mel	2
6	Si	8
7	Lou	1
8	Jon	100

10 Adding Images

An image of your item should be displayed on the page. Although it is possible to store the actual data that makes up the image in a database, it's better practice to keep the image files separate and store only the filenames in the database. You will need to find an image for each member, team or band etc. and save them to an *images* folder.

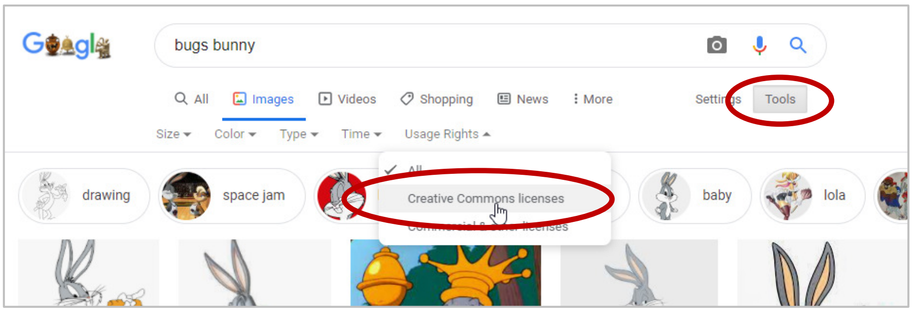
Task 1 – Displaying an Image



a. Open the directory (folder) on your computer where you are saving the webpage files and create a new folder within called *images*.

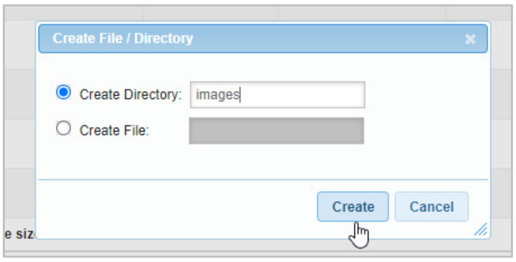
b. Add an image to this folder for each item. Don't use spaces in your image filenames as they tend to cause trouble. Underscores are fine.

If you need to find pictures online then use the Google tools to locate legal or public domain images. Anything marked with a *Creative Commons* licence should be fine.



Note: *You are publishing your website on the internet so it is important that your images are legal.*

c. Go back to the *File Manager* in Award Space and create a directory within your root folder, also named *images*. Upload your image files to this directory.



● Create Directory: images

○ Create File:

d. Start by naming a selected image in your HTML and include some temporary *alt text* and an *image title*. The image will be placed in its own div within the content div, something like below. The source has the form 'images/your_image_filename'.

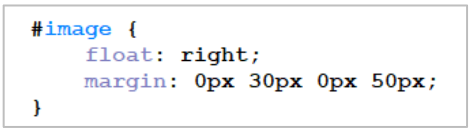
```
<div id="content">
 <div id="image">
   <img src="images/Bugs_Bunny.png" width="200" alt="TempAlt" title="TempTitle" />
 </div>
 <?=$charDescLong?>
</div>
```

Note: *If you fix a width for the image but not a height, then the image will scale to the width you have set. You may decide to fix the height instead. You can't fix both the width and height unless all your images are the same size or they will distort to fit the space assigned.*

Adding Images (page 2)

Task 1 – Displaying an Image (Cont.)

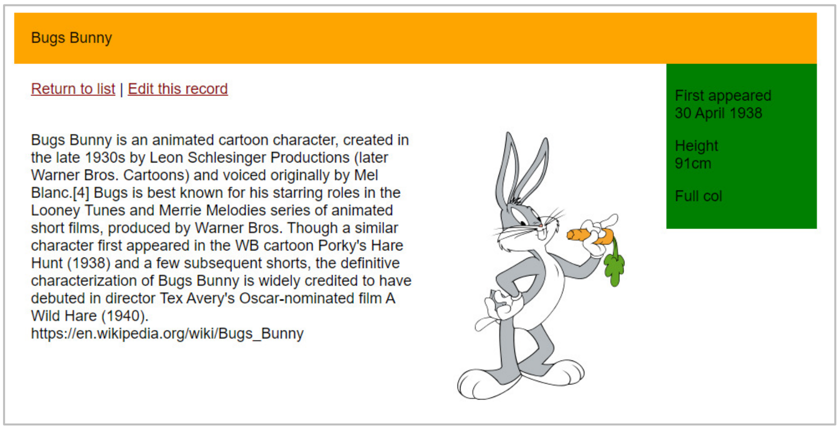
e. Add a selector to your CSS file and float the image to the right. Remember that float will cause the other content to flow around the object.



```
#image {
    float: right;
    margin: 0px 30px 0px 50px;
}
```

f. You may set a margin for the image. A single margin can be fixed using code such as *margin-left: 50px;*. The CSS above-right fixes all four margins around the image in one declaration. Values are placed in the order Top, Right, Bottom, Left.

g. It may be time to remove or replace some of the temporary background colours. Our current page is shown on the right. There is still work to do on the design. We will return to that later.



Bugs Bunny

Return to list | Edit this record

Bugs Bunny is an animated cartoon character, created in the late 1930s by Leon Schlesinger Productions (later Warner Bros. Cartoons) and voiced originally by Mel Blanc.[4] Bugs is best known for his starring roles in the Looney Tunes and Merrie Melodies series of animated short films, produced by Warner Bros. Though a similar character first appeared in the WB cartoon Porky's Hare Hunt (1938) and a few subsequent shorts, the definitive characterization of Bugs Bunny is widely credited to have debuted in director Tex Avery's Oscar-nominated film A Wild Hare (1940).
https://en.wikipedia.org/wiki/Bugs_Bunny

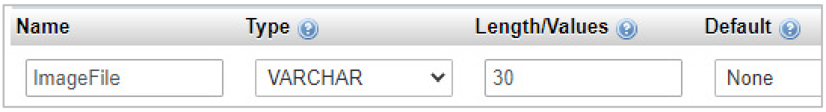
First appeared
30 April 1938

Height
91cm

Full col

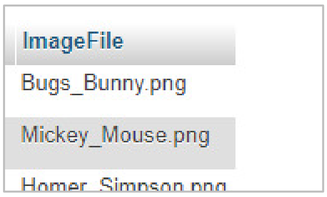
Note: *You may need to force a reload of the page after editing your CSS file. This depends on the caching systems in place between the server and your browser. Hold down the 'Ctrl' key and press 'F5' in Windows or hold down 'Cmd + Shift' and press 'R' on a Mac.*

h. Add a field to your database table for the image filename. We will use the cartoon name as the image title and alt text although you can add separate fields for these if you choose.



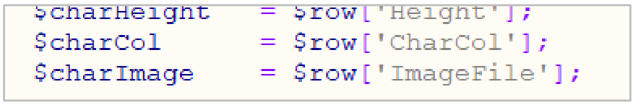
Name	Type	Length/Values	Default
ImageFile	VARCHAR	30	None

i. Update each record so that the relevant image name is included.



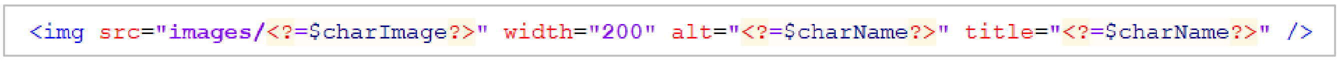
ImageFile
Bugs_Bunny.png
Mickey_Mouse.png
Homer_Simpson.png

j. Collect the image filename from the record set in the PHP section of your webpage code. Remember that the SQL SELECT statement uses the * wildcard to collect all records so the data should be available.



```
$charHeight     = $row['Height'];
$charCol        = $row['CharCol'];
$charImage      = $row['ImageFile'];
```

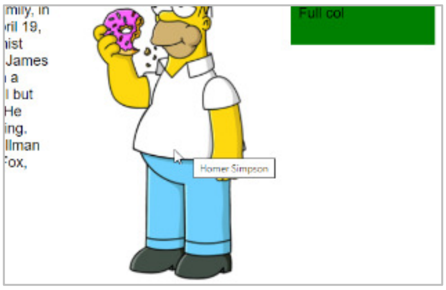
k. Replace the filename, alt text and title in your HTML with the PHP variables.



```
<img src="images/<?=$charImage?>" width="200" alt="<?=$charName?>" title="<?=$charName?>" />
```

l. Check that each image appears when you run through your list of items. If there are problems, double check your coding as well as the filenames. Remember that filenames are case sensitive on this server.

Note: *Depending on your browser, either the title or the alt text should be displayed when you hold your mouse over the image*



Data Formatting

Data from the database is used in many ways. For example, you might want to automate the process of writing emails to your website members and begin the letter with a salutation such as 'Dear Jo' or 'Dear Pat'. The examples below show the salutations that would be created for different pieces of data from a first name field. We have imagined that the user filled out the member form themselves. This type of mismatched, somewhat confusing data is common.

Text in first name field		***Salutation created***
Jo	*i.e. the user entered a name, as expected*	*Dear Jo*
jo	*i.e. the user didn't capitalise their name*	*Dear jo*
J	*i.e. the user entered only an initial*	*Dear J*
Jo Kennedy	*i.e. the user entered their whole name*	*Dear Jo Kennedy*
jkhsd	*i.e. the user entered nonsense*	*Dear jkhsd*
	i.e. the user left the field blank	*Dear*
Dr Jo	*i.e. the user included a title*	*Dear Dr Jo*

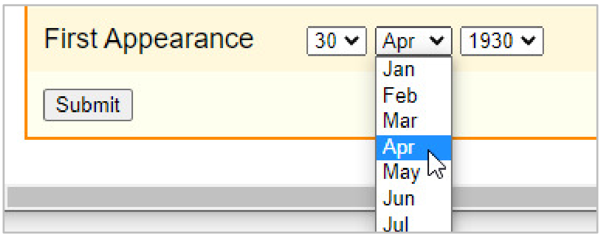
It's obviously in our interest to try and ensure that the data entered into our database is as well formatted as possible. Although there is not always a perfect solution, we can certainly put things in place to improve the quality of the data. Data formatting is the controlling of data so that it meets our pre-set conditions.

Task 1 – Form Fields

We learned earlier that the different types of form fields (e.g. radio buttons, checkboxes etc.) can be used to control the data entered into a database. We can also use these mechanisms to create formatted data.

Dates can be a huge problem on websites. If you ask a user to enter their birthdate into a text box you will get 100 different formats. These may include shorthand (20/1/05), different separators (20-1-05), full months (20 January 05), full years (20 Jan 2005), misspellings (20 Septmber 05), US formatting with the month first (1/20/05) etc. Not only is this data difficult to work with, many of the examples wouldn't be recognised by the database as a date.

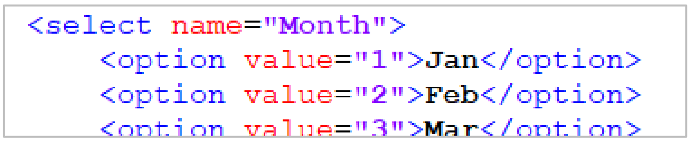
A common solution to this problem is to create separate drop-down lists for the day, month and year. These separate pieces of information can then be compiled into a standard date format that the database will understand.



First Appearance 30 Apr 1930

Have a go at building a solution like this in your own pages. If you have a date field already in place you may use this. If not, add one to your database to use in your pages.

It's generally safer to use number values for the day, month and year in your drop-down lists. Remember that the displayed value can be different to the one forwarded to the action page.



```
<select name="Month">
    <option value="1">Jan</option>
    <option value="2">Feb</option>
    <option value="3">Mar</option>
```

Data Formatting (page 2)

Task 1 – Form Fields (Cont.)

In your action page, code such as that on the right will concatenate the separate pieces of data and create something that our database will recognise as a date.

```
$faDate = ($faYear . "-" . $faMonth . "-" . $faDay);
```

Note: *There are date and time functions in PHP that you could use here but they are not really needed at this stage.*

When returning to your editing page, you will want the correct day, month and year pre-selected in the lists when the page loads. This requires separating the date from the database into these constituent parts and working with each individually.

The code on the right creates 3 variables for the day, month and year. Our example uses the date function with the following arguments:

```
$charBirth    = $row['Birth'];

$faDate = strtotime($charBirth);

$faDay        = date("j", $faDate);
$faMonth      = date("n", $faDate);
$faYear       = date("Y", $faDate);
```

j *the day without leading zeros e.g. 1, 5, 11, 31.*

n *the month without leading zeros e.g. 2, 8, 12.*

Y *the year in 4 digits e.g. 1970, 2021.*

There are lots of other formats you could utilise for these elements, listed here: https://www.php.net/manual/en/function.date.php

Task 2 – JavaScript

JavaScript makes things happen on a webpage. The JavaScript programming language has a wide range of uses, including:

- *displaying different text depending on user actions;*
- *performing calculations and displaying the results;*
- *validating the data entered into a form.*

These abilities can help ensure that data is of the correct format before it is passed to our action page.

Note: *It's not our aim here to teach JavaScript coding as this is covered in our publication 'Web Coding: JavaScript'. The information below indicates how the JavaScript might be used in your editing page.*

Some points about JavaScript

1. Today's browsers will generally ignore JavaScript if there are problems with it.

2. JavaScript can be executed at different times e.g. when the page is first displayed, when a button is clicked, when the mouse is placed over an object or when a form is submitted. We will be looking at form submission.

3. The use of JavaScript to check forms is an example of *client-side validation*. This means that the data is checked in the webpage before anything is sent onwards. Websites may also use *server-side validation*, where the data is checked in the action page and then the user returned to the form if there are problems.

   Note: *There is also a set of techniques called AJAX which use JavaScript in the webpage to send data directly to the database without leaving the page. AJAX is not covered here.*