PHP – Lab Four – Superglobals

Objectives

1. Check PHP settings with `phpinfo()`
2. Understand how data is transferred from forms and then retrieved with `$_POST`
3. Understand how data is transferred from forms and then retrieved with `$_GET`
4. Review for data is transferred with query strings using `$_GET`

Introduction

In this lab we'll check that PHP is up and running on your web space, experiment with PHP variables and get to grips with Superglobals.

Is it working?

To check if PHP is running we'll create a very simple PHP page.

In your *public_html* folder create a subfolder called *superglobals* and create a file called *info.php*.

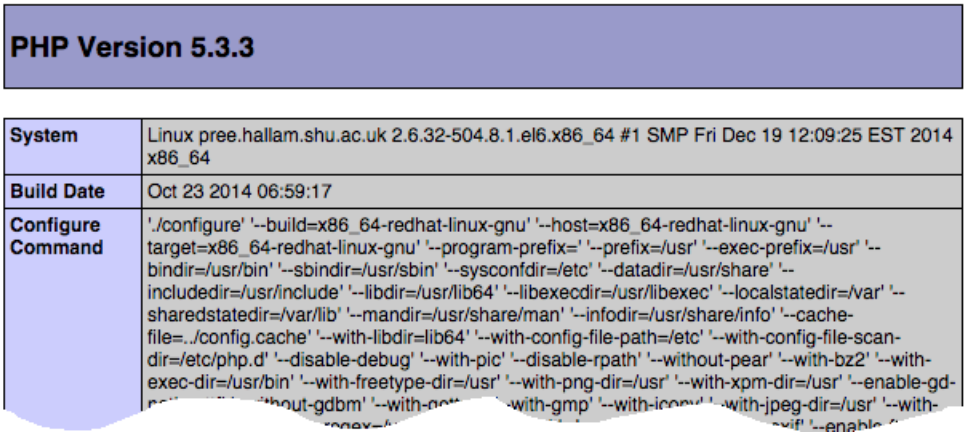
Add the following code to your file.

```
<?php
phpinfo();
?>
```

To view your file you will need to navigate to it through http by using an address such as:

```
http://homepages.shu.ac.uk/~<yourid>/WAD/superglobals/info.php
```

You should see a page such as:



PHP Version 5.3.3

System	Linux pree.hallam.shu.ac.uk 2.6.32-504.8.1.el6.x86_64 #1 SMP Fri Dec 19 12:09:25 EST 2014 x86_64
Build Date	Oct 23 2014 06:59:17
Configure Command	'./configure' '--build=x86_64-redhat-linux-gnu' '--host=x86_64-redhat-linux-gnu' '--target=x86_64-redhat-linux-gnu' '--program-prefix=' '--prefix=/usr' '--exec-prefix=/usr' '--bindir=/usr/bin' '--sbindir=/usr/sbin' '--sysconfdir=/etc' '--datadir=/usr/share' '--includedir=/usr/include' '--libdir=/usr/lib64' '--libexecdir=/usr/libexec' '--localstatedir=/var' '--sharedstatedir=/var/lib' '--mandir=/usr/share/man' '--infodir=/usr/share/info' '--cache-file=../config.cache' '--with-libdir=lib64' '--with-config-file-path=/etc' '--with-config-file-scan-dir=/etc/php.d' '--disable-debug' '--with-pic' '--disable-rpath' '--without-pear' '--with-bz2' '--with-exec-dir=/usr/bin' '--with-freetype-dir=/usr' '--with-png-dir=/usr' '--with-xpm-dir=/usr' '--enable-gd-[illegible]hout-gdbm' '--with-[illegible]-with-gmp' '--with-[illegible]-with-jpeg-dir=/usr' '--with-[illegible]--enable-[illegible]

Working with Variables

Create a second PHP page. This time use a boiler plate HTML document as follows:

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>PHP Variables</title>
</head>
<body>
</body>
</html>
```

Then add the following code in the `<body>` of the document.

```
<?php
$firstname = "Bob";
$surname = "Smith";
echo $firstname;
echo $surname;
?>
```

Experiment with the PHP to output the variables with HTML formatting for example:

```
<?php
$firstname = "Bob"; $surname = "Smith";
echo "<h1>" . $firstname. " ". $surname . "</h1>";
?>
```

Superglobals

One of the key features of web applications is the ability to pass data from one page in your web application to another.

One method to do this is HTML forms. The data from HTML form is sent for processing using either the 'post' or 'get' methods. Where the data is sent and whether it is sent with post or get is controlled by the attributes of the form element ie:

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

Tip: If no `method` value is set the default is for a form to send values using get. If no `action` is set then when the form is submitted it reloads the page - sending the data with it.

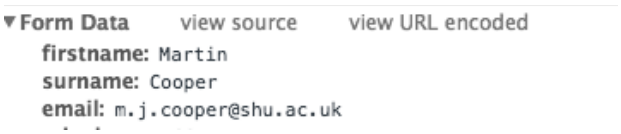
The values that the user entered in the form are then available to the targeted action page for processing. In a HTML form, each form element should have a unique name/id. This is the fieldname of the value to be submitted.

```
<input name="email" type="text" id="email" />
```

That is the name of the name/value pairs that will get submitted when the data is sent. So if you have a form field named 'email' and the users enters the value 'bob@shu.ac.uk' the name/value pair submitted is equal to email=m.j.cooper@shu.ac.uk.

HTTP Protocol

The way data is sent using 'get' and 'post' forms part of the HTTP protocol and therefore is not unique to PHP. You can use modern browser developer tools to see the data been sent in the HTTP requests.



$_POST and $_GET

In PHP the values are made available to the receiving page. How they are retrieved depends on which method was used to send the data in the form ie post or get.

- `$_POST['fieldname']` - retrieves values from a HTML form that has used the method ‘’post’.
- `$_GET['fieldname']` - retrieves values from a HTML form that has used the method ‘get’.

(`$_POST` and `$_GET` are known as superglobal variables.)

The HTML form method ‘post’ sends the name/values pairs to the next page in the header of the http page request. As such the values are hidden from the user.

The ‘get’ methods places the name/values pairs in URL of the page request.

```
ie:
http://www.mysite.com/process.php?email=m.j.cooper@shu.ac.uk
```

Multiply values will be concatenated in the URL with ampersands as follows:

```
http://www.mysite.com/process.php?email=m.j.cooper@shu.ac.uk
&name=Martin
```

This is what is known as a 'query string'.

A 'query string' can be created by appending a HTML link with a ‘?’ and then the name/value pair.

As indicated multiply values can be sent by using the ‘&’ concatenation character.

Task 1 - POST from a Form

1. Open the file *formpost.php*. Notice that the file contains a HTML form with various form elements. Also note that the form has a method of `post`. Add an `action` pointing the form to *formpostprocess.php*.

2. Open the file *formpostprocess.php*.

3. Use the `$_POST` superglobal variable to retrieve and display the values from the form using echo.

Task 2 - GET from a Form

1. Open the file *formget.php*. Notice that the file contains a HTML form with various form elements. Also note that the form has a method of `get`. Add an `action` pointing the form to *formgetprocess.php*.

2. Open the file *formgetprocess.php*.

3. Use the `$_GET` superglobal variable to retrieve and display the values from the form.

Task 3 - GET from a Query String

1. Open the file *querystring.php*.

2. This file contains a link to the page *querystringprocess.php*. Appended the link with a "?" to add a name/value pair for the fieldname "email".

3. Extend the file to include two more variables.

4. Open the file *querystringprocess.php*. Use the appropriate superglobal variable to retrieve and display the values from the hyperlink.