# PHP

control structures - loops cont.
associative arrays
functions

## Control structures / flow control

- if ... else
- Switch
- for loops
- while ... do ..
- do ... while ...

} Did these last week

Much of this material is explained in PHP programming 2nd Ed. Chap 2

## Loops / Iteration / doing things over and over and over and over ....

Three standard loop types

for
while ...
do ... while

Choose a for-loop if the number of times the loop will run is known 'in advance'

...or you are processing an array

## Loops / Iteration / doing things over and over and over and over ....

### Three standard loop types

for
while ...
do ... while

Choose a while-loop if the loop will run 0 or more times till some condition becomes false

Choose a do-while-loop if the loop will run 1 or more times till some condition becomes false

## Loops / Iteration / doing things over and over and over and over ....

### Three standard loop types

for
while ...
do ... while

With these we typically don't know in advance how many times the loop will execute...

..but we must know the minimum number of times the loop will work to choose between while or do ... while

i.e. 0 or 1

## while loop

### The structure of a while statement is:

```
while (condition)
   statement
```

Loop continues whilst condition is true

### or with many statements -

```
while (condition){
   statement;
   statement;
   statement;
   statement;
};
```

Do something in here to change the condition (unless you want it to continue ∞)

## while loop

```
$today="Monday";

while($today<>"Friday"){
print "<p>Today is ".$today."</p>";
if (rand(1,10)>7){
    $today="Friday";
    };
};

print "Final value of today is ".$today;
```

If the condition is initially false, the loop doesn't execute at all (i.e. 0 times)

## do .. while loop

The structure of a do .. while statement is:

```
do
    statement;
while (condition)
```

Will loop at least once

or with many statements -

```
do {
    statement;
    statement;
    statement;
    statement;
}
while (condition);
```

Do something in here to change the condition (unless you want it to continue ∞)

## do .. while loop

```
do {
$i=rand(1,10);
print "<p>This time i had the value $i</p>";
} while ($i<8);
```

As condition is at the bottom of the structure the loop must execute at least once

## arrays

- indexed arrays
- associative arrays
- useful functions
- iterating over an array

Much of this material is explained in PHP programming 2nd Ed. Chap 5

## associative arrays

Uses labels to index the cells

$favTV

| Dan | Barry | Phil | Walter |
|---|---|---|---|
| West Wing | Mad men | Horizon | Desperate Housewives |

```
print $favTV['Barry'];
```

Mad men

## associative arrays

Uses labels to index the cells

$favTV

| Dan | Barry | Phil | Walter |
|---|---|---|---|
| West Wing | Mad men | Horizon | Desperate Housewives |

```
print $favTV['Dan'];
```

West Wing

## associative arrays

### Uses labels to index the cells

$favTV

| Dan | Barry | Phil | Walter |
|---|---|---|---|
| West Wing | Mad men | Horizon | Desperate Housewives |

```
$favTV['Walter']="ER"
```

## associative arrays

### Uses labels to index the cells

$favTV

| Dan | Barry | Phil | Walter |
|---|---|---|---|
| West Wing | Mad men | Horizon | ER |

```
$favTV['Walter']="ER"
```

## associative arrays

### Use simple assignment to create the array

```
$favTV["Dan"]="West Wing";
$favTV["Barry"]="Mad men";
$favTV["Phil"]="Horizon";
```

$favTV

| Dan | Barry | Phil |
|---|---|---|
| West Wing | Mad men | Horizon |

## associative arrays

### If all the values are known in advance, use the reserved word array

```
$favTV = array (
         "Dan"=>"WestWing",
         "Barry"=>"Mad men",
         "Phil"=>"Horizon"
         );
```

$favTV

| Dan | Barry | Phil |
|---|---|---|
| West Wing | Mad men | Horizon |

## useful functions

### See appropriate references for full definitions of these

| functions | explanation |
|---|---|
| count() | no of array cells |
| array_pad() | create an array with the same value |
| array-slice() | extract part of an array |
| array_keys() | returns an array of the keys |
| array_key_exists() | see if a key exists |

## Processing an associative array

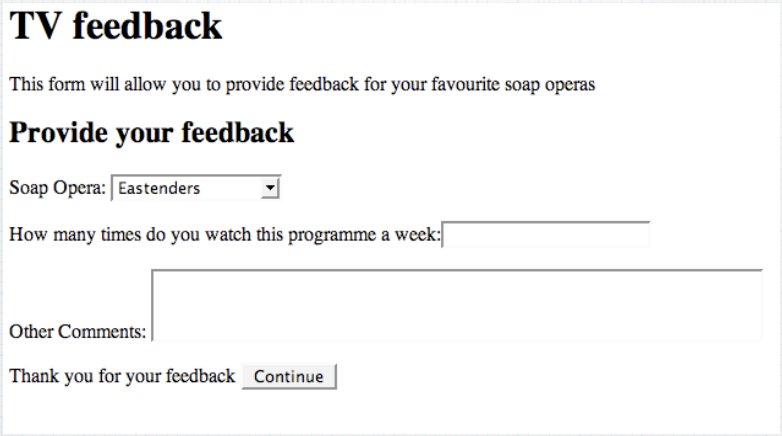
### Use a foreach loop to iterate over the cells

$favTV

| Dan | Barry | Phil |
|---|---|---|
| West Wing | Mad men | Horizon |

```
foreach ($favTV as $person=>$programme){
  print $person." likes ".$programme;
  );
```

## Web forms




## Web forms

### Two ways of passing information between pages

| | |
|---|---|
| get | form information is passed through the URL |
| post | form information is embedded in the HTTP stream |

## Web forms

```
<form name="form1" action="responseget.php" method="get">
<p>Soap Opera:
    <select name="soapname">
      <option value="EE">Eastenders</option>
      <option value="CS">Coronation Street</option>
      <option value="EMD">Emmerdale</option>
    </select>
</p>
<p>How many times do you watch this programme a week:<input type="text" name="timesaweek"></p>

<p>Other Comments: <textarea name="comments" cols="60" rows="3"></textarea></p>
<p>Thank you for your feedback <input type="submit" name="continue" value="Continue" /></p>
</form>
```

## method="GET"

1. The form values are passed in the URL using name = value pairs

```
responseget.php?soapname=EE&timesaweek=3&comments=Its
+depressing&continue=Continue
```

Note how special encoding has to occur for spaces, =, & etc.

Called URL encoding and done automatically by the browser

## $_GET associative array

PHP automatically creates an associative array with the passed URL values

```
responseget.php?soapname=EE&timesaweek=3&comments=Its
+depressing&continue=Continue
```

$_GET

| soapname | timesaweek | comments | continue |
|---|---|---|---|
| EE | 3 | its depressing | Continue |

## Web forms

responseget.php page

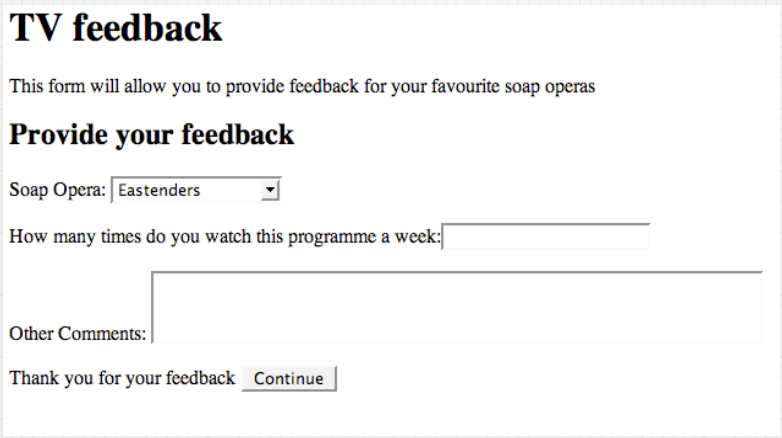
```
$soapname=$_GET['soapname'];
$timesaweek =$_GET['timesaweek'];
$comments=$_GET['comments'];

print "The soap watched is ";
switch ($soapname) {
    case EE:
        print "Eastenders<br />";
        break;
    case CS:
        print "Coronation Street<br />";
        break;
    default:
        print "Emmerdale<br />";
        break;
};

print "The amount watched was ".$timesaweek."<br />";
print "Comments ".$comments."<br />";
```

We can use the _GET associative array to get at the form selections

## Web forms - POST example




Looks the same in the browser

## Web forms

```
<form name="form1" action="responsepost.php" method="post">
<p>Soap Opera:
    <select name="soapname">
      <option value="EE">Eastenders</option>
      <option value="CS">Coronation Street</option>
      <option value="EMD">Emmerdale</option>
    </select>
</p>
<p>How many times do you watch this programme a week:<input type="text" name="timesaweek"></p>

<p>Other Comments: <textarea name="comments" cols="60" rows="3"></textarea></p>
<p>Thank you for your feedback <input type="submit" name="continue" value="Continue" /></p>
</form>
```

Small change here

## $_GET associative array

PHP automatically creates an associative array with the passed URL values from the HTTP stream

$_POST

| soapname | timesaweek | comments | continue |
|---|---|---|---|
| EE | 3 | its depressing | Continue |

## Web forms

### responsepost.php page

```
$soapname=$_POST['soapname'];
$timesaweek =$_POST['timesaweek'];
$comments=$_POST['comments'];

print "The soap watched is ";
switch ($soapname) {
    case EE:
        print "Eastenders<br />";
        break;
    case CS:
        print "Coronation Street<br />";
        break;
    default:
        print "Emmerdale<br />";
        break;
};

print "The amount watched was ".$timesaweek."<br />";
print "Comments ".$comments."<br />";
```

Uses the _POST array

name-value pairs INSIDE the http stream

## User defined functions

### Useful for pieces of code that we use to reuse

Pass values into the function here

```
function name(args){

...code for function

return arg;
}
```

Return value out of the function here

ONLY ONE!

## Example

### simple multiplication

```
function multiply($number1, $number2){
$result=$number1*$number2;
return($result);
};
```

```
$num1=3;
$num2=10;

$answer=multiply($num1,$num2);
```