COMP2203: Application Scripting

Session DA6

# Customising Our Page Template

David Argles

ECS
Electronics & Computer Science
University of Southampton

Haven
Consulting
david@argles.org

# Session DA6 Outline:

- From last time...
- Problem!
- Page attributes
- Changing our page attribute
- Commenting our template

Haven
Consulting
david@argles.org

# We had: an HTML5 Page

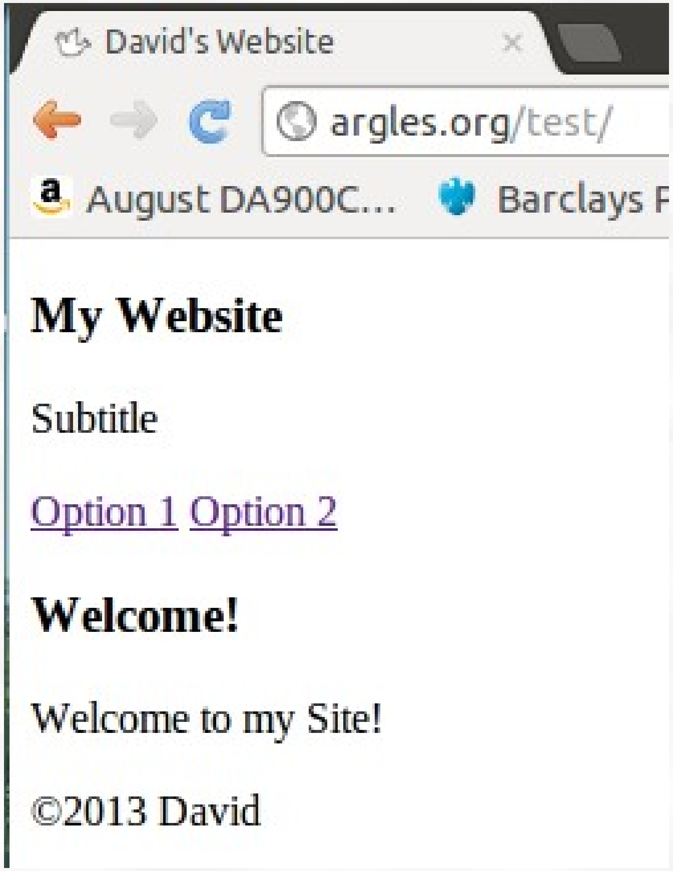
```
<!DOCTYPE html>
<html>
  <head>
    <title>David's Website</title>
  </head>

  <body>
   <container>
    <header>
      <h3>My Website</h3>
      <p>Subtitle</p>
    </header>

    <nav>
      <a href="">Option 1</a>
      <a href="">Option 2</a>
    </nav>

    <section>
      <h3>Welcome!</h3>
      <p>Welcome to my Site!</p>
    </section>

    <footer>
      <p>&copy;2013 David</p>
    </footer>
   </container>
  </body>
</html>
```



**My Website**

Subtitle

Option 1 Option 2

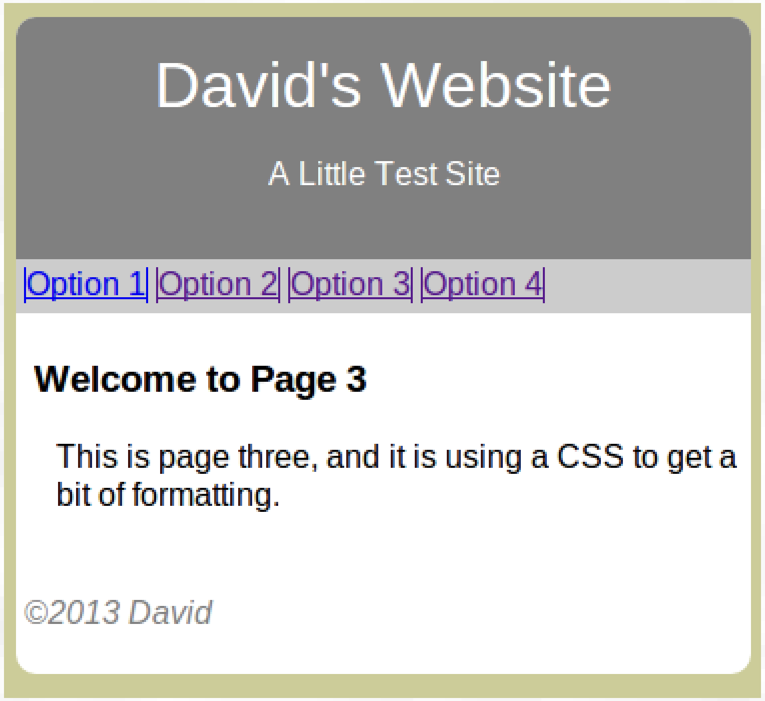
**Welcome!**

Welcome to my Site!

©2013 David

# We added: some CSS

```
...comments...
body{font-family:verdana,arial,sans-serif;...etc...}
container{...some stuff...}
header{
  background-color:gray;
  color:white;
  padding:1em;
  text-align:center;}
header h{font-size:2em;padding:5px;}
nav{
  background-color:#ccc;
  padding:0.3em;}
nav a{
  border-left:solid;
  border-right:solid;
  border-width:0.1em;}
section{padding:0.3em;}
section h3{padding-left:5px;}
section p{padding-left:1em;}
footer{
  font-style:italic;
  color:gray;}
...more comments...
```



David's Website

A Little Test Site

Option 1 | Option 2 | Option 3 | Option 4

### Welcome to Page 3

This is page three, and it is using a CSS to get a bit of formatting.

©2013 David

# We Created a Page Class

```
<?php
class htmlPage
{
  public function streamTop()
  {
?>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <link rel="stylesheet" type="text/css"
        href="library/normal.css">
    <title>David's Website</title>
  </head>

  <body>
    <container>
      <header>
        <h3>My Website</h3>
        <p>Subtitle</p>
      </header>
```

```
      <nav>
        <a href="index.html">Home</a>
        <a href="page2.html">Page 2</a>

        <a href="page3.html">Page 3</a>
        <a href="page4.html">Page 4</a>
      </nav>

    <section>
<?php
  }

  public function streamBottom()
  {
?>
    </section>

    <footer>
      <p>&copy;2013 David</p>
    </footer>
   </container>
  </body>
</html>
<?php
  }
}
```

# And We Made a Page Template

```
<?php
  require(“library/htmlPage.php”);
  $page = new htmlPage();
  $page->streamTop();
?>
      <!-- Page content follows -->
      <h3>Welcome!</h3>
      <p>Welcome to my Site!</p>
      <!-- Now back to the template -->
<?php
  $page->streamBottom();
?>
```

– Note: the green stuff is the page template; the black stuff is what changes from page to page

# But there's a problem!

- It might all look OK, but we have the same text in every page <title>
- It ought to change from page to page
- It's in the HTML class which is predefined
- But we can change that!

# Page Attributes

- We need to create a page attribute that gives the <title> of the page
- We *could* make it so that the attribute can be accessed from the calling page
- But it's better to make it only accessible via a method in our page object

# Creating a Page Attribute

- First, we need to create a suitable attribute in our htmlPage class:

```
<?php
class htmlPage
{
  protected $title = “Default Page Title”;

  public function streamTop()
  {
?>
<!DOCTYPE html>
    ...etc..
<?php
  }

  public function streamBottom()
  {
?>
    ...more HTML streaming stuff...
<?php
  }
}
```

# Changing a Page Attribute

- Now we need a method for changing that attribute:

```
<?php
class htmlPage
{
  protected $title = “Default Page Title”;

  public function setTitle($newTitle)
  {
    $this->title = $newTitle;
    return;
  }

  public function streamTop()
  {
?>
<!DOCTYPE html>
    ...etc..
}
```

# Changing a Page Attribute

- That's fine, but it won't do anything unless we use this attribute in streamTop():

```
public function streamTop()
   {
?>
<!DOCTYPE html>
 ...etc...
    <title>David's Website</title>
  </head>
 ...etc...
```

```
public function streamTop()
   {
?>
<!DOCTYPE html>
 ...etc...
 <!php echo(“<title>”.$this->title.”</title>”); ?>
  </head>
 ...etc...
```

# Changing the Page Attribute

– Now all we have to do is set the page title before calling streamTop() in our webpage:

```
<?php
  require(“library/htmlPage.php”);
  $page = new htmlPage();
  $page->setTitle(“Home: David's Site”);
  $page->streamTop();
?>
      <!-- Page content follows -->
      <h3>Welcome!</h3>
      <p>Welcome to my Site!</p>
      <!-- Now back to the template -->
<?php
  $page->streamBottom();
?>
```

# But Actually...

- **Now we've got into attributes, there's a better way of doing our page object**
  - Our page object ought really to contain all the HTML to form our page
  - We should use attributes to customise an individual page – including the main page content
  - The object itself should return the entire HTML stream for the page

# Constructing a True Page Object

- For now, we have two attributes that change
  - The <title>
  - The <section> content
- So we need methods that allow us to set the <title> and <section> content:

# Constructing the Page Object

```
<?php
class htmlPage
{
  protected $title = “Default Page Title”;
  protected $content = “Default Page Content”;

  public function setTitle($newTitle)
  {
    $this->title = $newTitle;
    return;
  }

  public function setContent($newContent)
  {
    $this->content = $newContent;
    return;
  }

}
```

# Using Our Object

- Now we need to use this new approach in our page:

```
<?php
  require(“library/htmlPage.php”);
  $page = new htmlPage();
  $page->setTitle(“Home: David's Site”);
  $page->setContent(“
      <h3>Welcome!</h3>
      <p>Welcome to my Site!</p>
”);
?>
```

# Streaming the Page Content

- But we still have to stream the final page HTML:

```
    public function HTMLstream()
    {
      $output=”
<html>
  <head>
    ...etc...
    <title><?php echo($this->title); ?></title>
  </head>

  <body>
    ...etc...
      <nav>
        <a href="page1.php">Home</a>
        ...etc...
      </nav>

    <section>
      <?php echo($this->content); ?>
    </section>

    ...etc...
</html>”;
      return $output;
    }
```

# Streaming the Page Content

- And we need to call this from our individual pages:

```
<?php
  require(“library/htmlPage.php”);
  $page = new htmlPage();
  $page->setTitle(“Home: David's Site”);
  $page->setContent(“
      <h3>Welcome!</h3>
      <p>Welcome to my Site!</p>
“);
  echo($page->HTMLstream());
?>
```

# Using Our Template

- And that's it!
- Now, just copy the template for as many pages as you need (as before)
- Then remember to change the <title> attribute as well as the page content
- We can use this approach to change other attributes of our page template if required

# Commenting the Template

- We ought to add comments to our template page

```
<?php
/**
 * template.php is a template page for our website
 *
 * It calls our class, htmlPage, sets the title for our page, & streams
 * the boilerplate code around our page content.
 *
 * @author David Argles <d.argles@gmx.com>
 * @version 15-08-2013, 1:41h
 * @copyright 2013 Haven Consulting
 */
  require(“library/htmlPage.php”);
  $page = new htmlPage();
  // In the next line, change the parameter to suit the page
  $page->setTitle(“Template Page: David's Site”);
  $page->setContent(“
      <!-- Page content follows -->
      <h3>Welcome!</h3>
      <p>Welcome to my Site!</p>
“);
  echo($page->HTMLstream());
?>
/**---------------------------------------------
 *            End of Code
 *---------------------------------------------
 */
```

# Break

- The notes pages give a link to the "step4" directory with an example site built this way
- It also has a few notes on how to use this approach to make a website template that can be reused :-)
- In effect, we've built our own little website framework (sort of a mini-Zend/mini-Bootstrap all in one)