Index of /~jharvard/S75-Sections/7-02_MVC

<table>
<thead>
<tr>
<th>Name</th>
<th>Last modified</th>
<th>Size</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent Directory</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01_MVC_intro.php</td>
<td>01-Jul-2012 16:53</td>
<td>3.2K</td>
<td></td>
</tr>
<tr>
<td>02_unfinished_MVC_example.php</td>
<td>02-Jul-2012 07:47</td>
<td>209</td>
<td></td>
</tr>
<tr>
<td>03_MVC_no_link_functionality.php</td>
<td>02-Jul-2012 07:47</td>
<td>209</td>
<td></td>
</tr>
<tr>
<td>04_MVC_with_links.php</td>
<td>02-Jul-2012 08:32</td>
<td>209</td>
<td></td>
</tr>
<tr>
<td>05_note_about_permalinks.php</td>
<td>02-Jul-2012 20:35</td>
<td>1.3K</td>
<td></td>
</tr>
<tr>
<td>app02/</td>
<td>02-Jul-2012 05:33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>app03/</td>
<td>01-Jul-2012 10:06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>app04/</td>
<td>02-Jul-2012 08:30</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Apache/2.2.22 (Fedora) Server at 192.168.119.128 Port 80
**Intro to MVC**

Web Developers call dynamic websites **web applications** because from the coder’s perspective, **dynamic websites are applications that run on a web server**.

Modern applications follow **design patterns**. One very common design pattern is Model, View, Controller, or MVC.

To follow MVC when writing an application, you **separate your code** according to the function of the code:

- **Model** code stores the application data and *models* your application’s domain. For example, if you were writing a blogging application, the Model code would determine what data needs to be stored about a Blog Post, and take responsibility for storing and retrieving Blog Post data.
- **View** code produces anything *viewable* by the site visitor. For example, in a blogging application, the View code would turn a Blog Post’s data into HTML and CSS, which would then be rendered in the site viewer’s web browser and determine how the site appears to the visitor.
- **Controller** code receives all of site visitor’s clicks and other input, and uses them to help the user *control* your application. For example, in a blogging site, if a user clicks on ‘next post’, the controller will receive that click, fetch the appropriate Model from the model code, send and send it to the View to turn it into the HTML that the site viewer’s browser will render for the Site Viewer’s viewing pleasure.

**Posted By:** P. Myer Nore

**Posted On:** 7/2/2012
Web Developers call dynamic websites **web applications** because from the coder's perspective, **dynamic websites are applications that run on a web server**.

Modern applications follow **design patterns**. One very common design pattern is Model, View, Controller, or MVC.

To follow MVC when writing an application, you **separate your code** according to the function of the code:

- **Model** code stores the application data and models your application's domain. For example, if you were writing a blogging application, the Model code would determine what data needs to be stored about a Blog Post, and take responsibility for storing and retrieving Blog Post data.

- **View** code produces anything *viewable* by the site visitor. For example, in a blogging application, the View code would turn a Blog Post's data into HTML and CSS, which would then be rendered in the site viewer's web browser and determine how the site appears to the visitor.

- **Controller** code receives all of site visitor's clicks and other input, and uses them to help the user control your application. For example, in a blogging site, if a user clicks on 'next post', the controller will receive that click, fetch the appropriate Model from the model code, send and send it to the View to turn it into the HTML that the site visitor's browser will render for the Site Viewer's viewing pleasure.
Post Title: Intro to MVC

lorem ipsum

hipster ipsum
Post Title: Intro to MVC

Intro to MVC

Web Developers call dynamic websites web applications because from the coder's perspective, dynamic websites are applications that run on a web server.

Modern applications follow design patterns. One very common design pattern is Model, View, Controller, or MVC.

To follow MVC when writing an application, you separate your code according to the function of the code:

- **Model** code stores the application data and models your application's domain. For example, if you were writing a blogging application, the Model code would determine what data needs to be stored about a Blog Post, and take responsibility for storing and retrieving Blog Post data.

- **View** code produces anything viewable by the site visitor. For example, in a blogging application, the View code would turn a Blog Post's data into HTML and CSS, which would then be rendered in the site viewer's web browser and determine how the site appears to the visitor.

- **Controller** code receives all of site visitor's clicks and other input, and uses them to help the user control your application. For example, in a blogging site, if a user clicks on 'next post', the controller will receive that click, fetch the appropriate Model from the model code, send and send it to the View to turn it into the HTML that the site viewer's browser will render for the Site Viewer's viewing pleasure.

P. Myer Nore

Tue, 03 Jun 2003 09:39:21 GMT
Intro to MVC

Web Developers call dynamic websites web applications because from the coder's perspective, dynamic websites are applications that run on a web server.

Modern applications follow design patterns. One very common design pattern is Model, View, Controller, or MVC.

To follow MVC when writing an application, you separate your code according to the function of the code:

- **Model** code stores the application data and models your application's domain. For example, if you were writing a blogging application, the Model code would determine what data needs to be stored about a Blog Post, and take responsibility for storing and retrieving Blog Post data.
- **View** code produces anything viewable by the site visitor. For example, in a blogging application, the View code would turn a Blog Post's data into HTML and CSS, which would then be rendered in the site viewer's web browser and determine how the site appears to the visitor.
- **Controller** code receives all of site visitor's clicks and other input, and uses them to help the user control your application. For example, in a blogging site, if a user clicks on 'next post', the controller will receive that click, fetch the appropriate Model from the model code, send and send it to the View to turn it into the HTML that the site viewer's browser will render for the Site Viewer's viewing pleasure.

P. Myer Nore

Tue, 03 Jun 2003 09:39:21 GMT
On Generating Links in PHP

When thinking about printing out links to the screen using PHP, you have to be careful. There are a number of interesting problems to think about here. One possible innovation we could introduce to our blogging system at this point would be to make it so people could pull up an article by its title in the url. Consider the following url:

http://www.example.com/index.php?pageTitle=Correct Usage of <pre>

This url has several problems. If you wanted to link to the page title (as is customary with blogs) you would have to think ahead to achieve the effect:

Correct Usage of <pre>

<a href='http://localhost/php%2Fcreated%2Fpage%2Furl.php?pageTitle=Correct+Usage+of+%3Cpre%3E'>Correct Usage of &lt;pre&gt;</a>


P. Myer Nore

Tue, 20 May 2003 08:56:02 GMT