Controllers and Routing

tutorial #ci-normal01

James L. Parry
B.C. Institute of Technology

Tutorial Goals

This tutorial will walk you through some common controller and routing challenges.

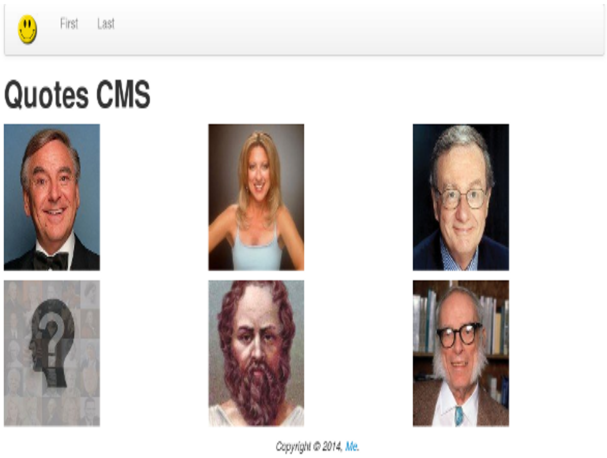
The goal is to fix the broken starter webapp, as directed.

That can't be so hard, can it? <-- said with a straight face.

Suggestion: you may want to skim the slideshow first, before working your way through it.

Preparation

I have prepared a starter webapp – a mini "content management system". It is incomplete and the pieces are not properly connected, and you need to fix these, as described following.



The Starter Webapp

The "Quotes" starter webapp uses Bootstrap and jQuery, simply because they make it so easy to have a not-ugly-though-the-same-as-everyone-else's looking site.

The Quotes webapp looks beautiful, I have to admit, but none of the links work, apart from the logo taking you back to the homepage :(

Your job is to fix these broken links, per the directions that follow :)

What Comes With the Starter?

The starter app uses a base controller, `application/core/MY_Controller`, which uses `application/views/_template` to provide the shell and layout of rendered webpages. The app uses a "view templating" strategy, where a controller is expected to set the name of its view fragment, and any other data, as `$this->data` parameters, and to then call the inherited `render()` method.

Data for the starter app is provided in the `application/models/Quotes` model, which has a mock database of quotations. No RDB is needed to run this app.

Where Do I Start?

There is a view fragment template in `application/views/justone.php`. This is meant to display just a single quote.

Each of the broken links needs to resolve to the appropriate controller, or method in a controller, which extracts the relevant quote data from the model, and builds the view parameters for `justone` to present it.

Solve the logic for the first broken link that you fix, and you can re-use the technique for all the others.

The logic? Inside the appropriate method of the appropriate controller, the handling of a request should be to retrieve the appropriate record from the `quotes` model, add its fields to the appropriate elements of the `$this->data` parameter array, and then call your inherited `render` method.

What Needs Fixing?

You might use the table below as a checklist. The next few slides provide helpful details.

Link	Location	Routing Strategy
first	Navbar	Normal routing
last	Navbar	Normal routing
sleep	Homepage	Wildcard routing
lock/em/up	Homepage	Wildcard routing
show/3	Homepage	Wildcard routing
dunno	Homepage	Callback routing
wise/bingo	Homepage	Regular expression routing
comp4711/wisdom	Homepage	Regular expression routing

Suggested Process

Fix the problems in sequence, working your way down the list.

For each of these, I suggest you determine the controller responsible for handling the request, the method inside that controller, and then the necessary routing.

How will you know if your "fix" worked? Clicking on an author on the homepage will lead to the featured quote from him/her.

Helpful Hint

A helpful hint... if you retrieve a record from your model, and you want to add its key/value pairs to the view parameters array, you could have a bunch of assignments (`$this->data['who']= ...`), but there is an easier way:

```
$record = $this->quotes->whatever(...);
$this->data = array_merge($this->data, $record);
```

The above references a "whatever" method inside your "quotes" object, which comes from autoloading `application/models/Quote.php`.

`array_merge` is a function built-in to PHP, which merges an associative array (the second parameter) into another (the first parameter), and returns the merged resulting associative array.

#1 First

Link	first
Location	Menu navbar, first link after the logo
Expected result	Bob Monkhouse's quote (he is the first of the six)
Routing Strategy	Normal routing
Required	No routing rule
How to Fix It	A suitable **new controller** in controllers/
Hint	application/controllers/First::index()

#2 Last

Link	last
Location	Menu navbar, last link after the logo
Expected result	Isaac Asimov's quote (he is the last of the six)
Routing Strategy	Normal routing
Required	No routing rule; should lead to a subfolder
How to Fix It	A suitable **new controller** in controllers/last
Hint	application/controllers/last/Welcome::index()

#3 Sleep

Link	sleep
Location	Top left author image
Expected result	Bob Monkhouse's quote
Routing Strategy	Wildcard routing
Required	routing rule, leading to first/zzz
How to Fix It	Remapped to the "zzz" method inside the "First" controller
Hint	application/controllers/First::zzz()

#4 Lockemup

Link	lock/em/up
Location	Top middle author image
Expected result	Elaine Boosler's quote
Routing Strategy	Wildcard routing
Required	Routing rule leasing to welcome/shucks
How to Fix It	Remapped, using wildcards (lock / * / *), i.e. the "shucks" method inside the "Welcome" controller
Hint	application/controllers/Welcome::shucks()

#5 Show

Link	show/3
Location	Top right author image
Expected result	Mark Russell's quote
Routing Strategy	Wildcard routing
Required	Routing rule, leading to first/gimme/3), i.e. the "gimme" method inside the "First" controller, with that method defined to take a parameter
How to Fix It	Remapped using wildcard (show / digit
Hint	application/controllers/First::gimme(id)

#6 Dunno

Link	dunno
Location	Lower left author image
Expected result	Anonymous quote?
Routing Strategy	Callback routing
Required	Routing rule which returns a suitable image instead of a web page.
How to Fix It	Surprise! have a callback function that sets a suitable mime-type header and then returns a random SFW image that you have added to the data folder

Hint	Strictly inside the routing configuration, something like ...
	`if (!file_exists($source)) { show_404($source); }`
	`$this->load->helper('file');`
	`$mimeType = get_mime_by_extension($source);`
	`header("Content-type: " . $mimeType);`
	`header('Content-Disposition: inline');`
	`readfile($source);`
	`die();`

#7 Bingo

Link	wise/bingo
Location	Lower middle author image
Expected result	Socrates' quote
Routing Strategy	Regular expression routing
Required	Routing rule, leading to /bingo (**new controller**)
How to Fix It	Remapped using regular expression (4 letters / bingo)
Hint	application/controllers/Bingo::index()

#8 Wisdom

Link	comp4711/wisdom
Location	Lower right author image
Expected result	Isaac Asimov's quote
Routing Strategy	Regular expression routing
Required	Routing rule, leading to wise/bingo
How to Fix It	Remapped using regular expression (comp + digits / *)
Hint	application/controllers/Wise::bingo()

Are We Done Yet?

When you are done, nothing will **look** different on the homepage, but the links will work properly :)

Congratulations!

You have completed tutorial #ci-normal01: Controllers and Routing

If you would take a minute to provide some feedback, we would appreciate it!

The next activity in sequence is: There is nothing further in this course

You can use your browser's back button to return to the page you were on before starting this activity, or you can jump directly to the course homepage, organizer, or reference page.