

Libraries and Helpers, Part (b)

tutorial #normal05b

James L. Parry
B.C. Institute of Technology

Tutorial Goals

This tutorial is meant to give you some practice using and extending a library and a helper, as described in lesson 6.

This tutorial part (#5b) deals only with adding a new quotation. It is a continuation of tutorial #5, and assumes you have completed that.

Revisions & Notes

•

What Needs Doing?

1. Enable the supplied bolding hook (#5)
2. Add a viewer rating widget (#5)
3. Randomize the quote presented on the homepage (#5)
4. Add a maintenance controller, presenting an ordered list of the quotes
5. Add the ability to add a new quote
6. Add the ability to edit a quote (#5c)
7. Add the ability to delete a quote (#5c)
8. Add the ability to fake admin rights (#5c)
9. Add an on-page editing for an admin (#5c)
10. Return to the page we were on after editing a quote (#5c)

The Starting Point

The webapp is functional when you start, with the homepage shown to the right.

It has a simple menubar on the top, with the smiley face linked back to the homepage, and the "View 'Em" menu link taking you to the "viewer" page.

The quote shown has been chosen randomly.



4. ADD A MAINTENANCE CONTROLLER

We want to add quotation maintenance to our webapp

We will add an Admin controller to do this.

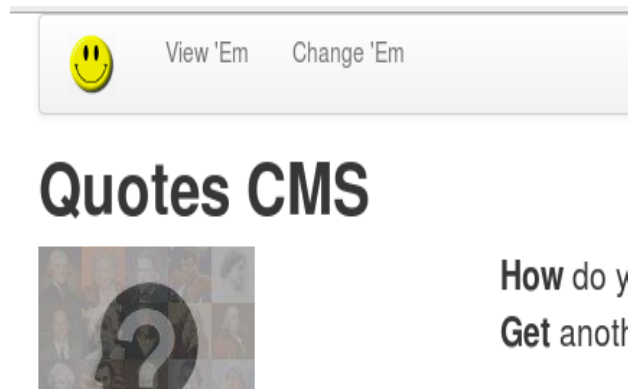
Its default view will be a list of the quotes, with provision to edit or delete any one of them, and the ability to add a new quotation.

Update Our Navbar Menu

This is easy. Add an entry for our Admin controller to the menu_choices array in applicaiton/config/config.php, and a (broken) link for our planned maintenance controller will now be on the top of every page.

```
'menudata' => array(
    array('name' => "View 'Em",
    'link' => '/viewer'),
    array('name' => "Change 'Em",
    'link' => '/admin'),
)
```

In a later tutorial, we will restrict access to the maintenance, to authorized users only.



Start With a Basic Controller

Let's add our Admin controller. I suggest copying the Welcome controller, and removing what we don't need.

The starting point should then look like...

```
class Admin extends Application
{
    function __construct() {
        parent::__construct();
    }

    function index() {
        $this->data['pagebody'] =
        'homepage';
        $this->render();
    }
}
```

Oh No - It Broke!

If we check the Admin page, it looks broken, because we are just using the homepage, without supplying any data to show.



Main Admin View

Let's make a main view for maintenance. This can be as simple as the contacts list from an earlier tutorial, but using the field names appropriate to the Quotes model.

application/views/admin_list would look like...

```
<table cols="" border="0">
  <tr>
    <th>ID</th>
    <th>Author</th>
    <th>Mugshot</th>
    <th>Their Saying</th>
  </tr>
  {quotes}
  <tr>
    <td>{id}</td>
    <td>{who}</td>
    <td>{mug}</td>
    <td>{what}</td>
  </tr>
  {/quotes}
</table>
```

Use the Main Admin View

The `index()` method of our Admin controller should use this view instead of the homepage, and it should pass the collection of quotations data as a view parameter.

It should also set the page title properly :)


```
function index() {
    $this->data['title'] =
    'Quotations Maintenance';
    $this->data['quotes'] =
    $this->quotes->all();
    $this->data['pagebody'] =
    'admin_list';
    $this->render();
}
```

5. ADD QUOTE ADDITION

Let's add a link to the maintenance list, linked to a new method in our Admin controller, `add()`.

This will look cheesy, but the link can be styled nicely later.


```
...
    {/quotes}
</table>
<a href='/admin/add'>Add a new
quotation</a>
```


[View 'Em](#)
[Change 'Em](#)

Quotations Maintenance

ID	Author	Mugshot	Their Saying
1	Bob Monkhous	bob-monkhous-150x150.jpg	When I die, I want to go peacefully like my grandfather did-in his sleep. Not yelling and screaming like the passengers in his car.
2	Elayne Boosler	elayne-boosler-150x150.jpg	I have six locks on my door all in a row. When I go out, I lock every other one. I figure no matter how long somebody stands there picking the locks, they are always locking three.
3	Mark Russell	bob-monkhous-150x150.jpg	The scientific theory I like best is that the rings of Saturn are composed entirely of lost airline luggage.
4	Anonymous	Anonymous-150x150.jpg	How do you get a sweet little 80-year-old lady to say the F word? Get another sweet little 80-year-old lady to yell "BINGO!"
5	Socrates	socrates-150x150.jpg	By all means, marry. If you get a good wife, you'll become happy; if you get a bad one, you'll become a philosopher.
6	Isaac Asimov	isaac-asimov-150x150.jpg	Those people who think they know everything are a great annoyance to those of us who do.

Copyright © 2014-2015. [Me](#).


[View 'Em](#)
[Change 'Em](#)

Quotations Maintenance

ID	Author	Mugshot	
1	Bob Monkhous	bob-monkhous-150x150.jpg	When I die, I want to go peacefully like my c passengers in his car.
2	Elayne Boosler	elayne-boosler-150x150.jpg	I have six locks on my door all in a row. Whi somebody stands there picking the locks, th
3	Mark Russell	bob-monkhous-150x150.jpg	The scientific theory I like best is that the rin
4	Anonymous	Anonymous-150x150.jpg	How do you get a sweet little 80-year-old lai to yell "BINGO!"
5	Socrates	socrates-150x150.jpg	By all means, marry. If you get a good wife, philosopher.
6	Isaac Asimov	isaac-asimov-150x150.jpg	Those people who think they know everythi

[Add a new quotation](#)

Handling Quote Addition

Generally speaking, to handle adding a new quotation, we

1. Create an empty record, with default/empty fields
2. Present it to the user to fill in
3. Submit the completed form
4. If any errors, redisplay the form, with messages
5. If no errors, add the new quote.
6. Redisplay the list of quotes

Quote Addition - The Form

We need a form for maintaining a single quote. Let's call it `application/views/quote_edit.php`

We could start with plain HTML, shown right, but that would look so bad that we couldn't show our face professionally for months.

```
<form action="/admin/confirm">
  <label for="id">ID#</label>
  <input type="text" id="id"
    name="id"></input>
  ...
</form>
```

Quote Addition - Prettier Form

We could add some Bootstrap styling, like the justone view!

Hmmm - that would be better, but still ugly HTML.

```
<div class="row">
  <form action="/admin/confirm">
    <label for="id">ID#</label>
    <input type="text" id="id" name="id"></input>
    ...
  </form>
</div>
```

Formfields Helper to the Rescue

This is a great opportunity to exploit the `formfields` helper talked about in class last week.

Let's plan to create styled fields, one for each column in the `Quotes` model.

Our `quote_edit` view:

```
<div class="row">
  <form action="/admin/confirm">
    {fid}
    {fwho}
    {fmug}
    {fwhat}
  </form>
</div>
```

Handle the New Quote Request

We can get down to serious work in our controller now.

Add an `add` method, that we created a link to in our admin quotations list.

Because we planned the steps, all this method needs to do is create a new record, and hand off to the presenter method.

```
Admin::add()

// Add a new quotation
function add() {
  $quote =
$this->quotes->create();
  $this->present($quote);
}
```

Construct the Form

Our presenter method can build the form from the supplied record, using the formfields helper.

Admin::present:

```
// Present a quotation for
adding/editing
function present($quote) {
    $this->data['fid'] =
makeTextField('ID#', 'id',
$quote->id);
    $this->data['fwho'] =
makeTextField('Author', 'who',
$quote->who);
    $this->data['fmug'] =
makeTextField('Picture', 'mug',
$quote->mug);
    $this->data['fwhat'] =
makeTextArea('The Quote',
'what', $quote->what);
    $this->data['pagebody'] =
'quote_edit';
    $this->render();
}
```

Helper Oops

When we click on the "Add a new quotation" link, our webapp blows up :(

We forgot to load the helper!

Fatal error: Call to undefined function makeTextField() in /data/WORK/pub7/htdocs/winter2015-lab05/application/controllers/Admin.php on line 33

A PHP Error was encountered

Severity: Error

Message: Call to undefined function makeTextField()

Filename: controllers/Admin.php

Line Number: 33

Helper Fix

Add a line to our Admin's constructor to fix this.

```
$this->load->helper('formfields');
```

We can work with this, and it looks a lot better than a plain HTML form!

[View 'Em](#)[Change 'Em](#)

Quotes CMS

ID#

Author

Picture

The Quote

Handle Form Submission

We need a submit button to trigger form processing.

The easiest way is to use the formfield helper for this, adding a snippet of code inside our `Admin::present()`, just before the `render()` call.

```
$this->data['fsubmit'] =  
makeSubmitButton('Process  
Quote', "Click here to validate  
the quotation data",  
'btn-success');
```

You also need to add `{fsubmit}` to your quote editing form. The result should look like...



The screenshot shows a web interface for a 'Quotes CMS'. At the top, there is a navigation bar with a yellow smiley face icon and two links: 'View 'Em' and 'Change 'Em'. Below this, the title 'Quotes CMS' is displayed in a large, bold, black font. The form consists of several input fields: 'ID#' (a text box with a small icon on the right), 'Author' (a text box), 'Picture' (a text box), and 'The Quote' (a larger text area). At the bottom of the form, there is a green button labeled 'Process Quote'.

Fix the Form Method

If you click on the "Process Quote" button, we get a 404 error :(

Oh - that is because we didn't add the confirm method :)

But the format of the URL looks funny -

 winter2015.local/admin/confirm?id=&who=&mug=&what=

We need to specify the POST method in the opening form element in the `quote_edit` view...

```
<form action="/admin/confirm" method="post">
```

Submission Handling

We need our `Admin::confirm()` method in order to go further. We can start with an empty method, shown below. Clicking the form's submit button will now lead to a blank page, because we haven't told it to do anything.

```
// process a quotation edit
function confirm() {

}
```

Extract Submitted Record

We need to extract the submitted fields, to use as a quotations record for validation or database updates. Note the first statement - creating an empty quote object to populate.

You might be tempted to use the PHP `$_POST` array, but then you need to check if anticipated fields are actually part of the submission.

You might be tempted to use `$this->input->post()` to get all the fields at once, but you may not want them all, and they might not all be handled the same, by the time we are done.

```
function confirm() {
    $record =
$this->quotes->create();
    // Extract submitted fields
    $record->id =
$this->input->post('id');
    $record->who =
$this->input->post('who');
    $record->mug =
$this->input->post('mug');
    $record->what =
$this->input->post('what');
    ...
}
```

Trivial Handling

Ignoring any error checking, our handling method needs to either add a new quote or update an existing one in our model, and then redisplay the admin page. We can determine which to do by checking if the "ID" field has a value ... it should be blank for a new quote, and contain the quote # for an existing one

```
...  
// Save stuff  
if (empty($record->id)) $this->quotes->add($record);  
else $this->quotes->update($record);  
redirect('/admin');
```

Try Unsafe Adding

It would be easy to trigger an error in the quotes addition, but if we are careful...

Trigger the new quote addition, and provide some silly new quote. I cited Yoda, shown right.

Quotes CMS

ID#

Author


Picture

The Quote

Unsafe Adding Result

On submission, we can see the new quotation when the confirm method send us back to the admin page.

We are not done yet! Still to tackle are protecting the ID field, validation, remembering field values, and image uploading.

 [View 'Em](#) [Change 'Em](#)

Quotations Maintenance

ID	Author	Mugshot	Their Saying
1	Bob Monkhouse	bob-monkhouse-150x150.jpg	When I die, I want to go peacefully like my grandfather did-i passengers in his car.
2	Elayne Boosler	elayne-boosler-150x150.jpg	I have six locks on my door all in a row. When I go out, I lock somebody stands there picking the locks, they are always lo
3	Mark Russell	bob-monkhouse-150x150.jpg	The scientific theory I like best is that the rings of Saturn are
4	Anonymous	Anonymous-150x150.jpg	How do you get a sweet little 80-year-old lady to say the F w to yell "BINGO!"
5	Socrates	socrates-150x150.jpg	By all means, marry. If you get a good wife, you'll become h philosopher.
6	Isaac Asimov	isaac-asimov-150x150.jpg	Those people who think they know everything are a great ar
7	Yoda		Do or do not. There is no try


[Add a new quotation](#)

Copyright © 2014-2015, [Me](#).

We Need to Step Up Our Game

On submission, we can see the new quotation when the confirm method send us back to the admin page.

We are not done yet! Still to tackle are protecting the ID field, validation, remembering field values, and image uploading.

 [View 'Em](#) [Change 'Em](#)

Quotations Maintenance

ID	Author	Mugshot	Their Saying
1	Bob Monkhouse	bob-monkhouse-150x150.jpg	When I die, I want to go peacefully like my grandfather did-i passengers in his car.
2	Elayne Boosler	elayne-boosler-150x150.jpg	I have six locks on my door all in a row. When I go out, I lock somebody stands there picking the locks, they are always lo
3	Mark Russell	bob-monkhouse-150x150.jpg	The scientific theory I like best is that the rings of Saturn are
4	Anonymous	Anonymous-150x150.jpg	How do you get a sweet little 80-year-old lady to say the F w to yell "BINGO!"
5	Socrates	socrates-150x150.jpg	By all means, marry. If you get a good wife, you'll become h philosopher.
6	Isaac Asimov	isaac-asimov-150x150.jpg	Those people who think they know everything are a great ar
7	Yoda		Do or do not. There is no try

[Add a new quotation](#)

Copyright © 2014-2015, [Me](#).

Protect the ID Form Field

The formfields helper method that we used, `makeTextField(...)`, has additional parameters, one of which let's us disable a field, preventing it from being edited.

In our `Admin::present()`, change `makeTextField('ID#', 'id', $quote->id);` to `makeTextField('ID#', 'id', $quote->id, "Unique quote identifier, system-assigned", 10, 10, true);` and observe the difference when we ask to add a new quotation.



Quotes CMS

ID#

Unique quote identifier, system-assigned

Author

Picture

The Quote

Process Quote

Explanations

We also added an explanation message, which you can optionally do for the other input fields.

```
In our Admin::present(), change  
makeTextField('ID#', 'id',  
$quote->id); to  
makeTextField('ID#', 'id',  
$quote->id, "Unique quote  
identifier, system-  
assigned", 10, 10, true);  
and observe the difference when we  
ask to add a new quotation.
```

We also added an explanation message, which you can optionally do for the other input fields.



Quotes CMS

ID#

Unique quote identifier, system-assigned

Author

Picture

The Quote

Process Quote

Simple Validation

Let's add some simple validation rules, for instance insisting that an author's name be non-empty, and that a quotation have a minimum length of 20 characters.

You may not have noticed earlier, but our base controller, `Applicaiton`, initializes an `errors` property, to an empty array, on its line 26.

We can exploit that, in our `confirm` method, by adding entries to the `errors` property if there is an error, as shown to the right.

```
// validation
if (empty($record->who))
    $this->errors[] = 'You must
specify an author.';
    if (strlen($record->what) <
20)
        $this->errors[] = 'A
quotation must be at least 20
characters long.';

// Save stuff
...
```

Dealing With Errors

If there are any errors, we want to redisplay the form, with error messages, and without updating the database.

A simple way to do this would be to present the form again, and rely on the `present` method to show any errors.

Redisplaying the form is easy, shown right.

```
// validation
...
// redisplay if any errors
if (count($this->errors) > 0)
{
    $this->present($record);
    return; // make sure we
don't try to save anything
}

// Save stuff
...
```

Displaying Errors

Shown to the right is some code to build a view parameter for any error messages. It isn't the best written, but it is expedient.

It assumes that your `application/config/constants.php` defines the constant `BREAK` with a line like this:
`define('BR', '
');`

```
function present($quote) {
    // format any errors
    $message = '';
    if (count($this->errors) > 0)
    {
        foreach ($this->errors as
        $booboo)
            $message .= $booboo . BR;
    }
    $this->data['message'] =
    $message;
    ...
}
```

Formatting the Error Display

We need to add the view parameter to application/views/quote_edit.php:

```
<div class="row">
  <div class="errors">{message}</div>
  <form action="/admin/confirm" method="post">
```

And we need to add a style rule in assets/css/style.css:

```
.errors {color: red;
font-weight:bold;
font-size:large}
```



Quotes CMS

You must specify an author.

ID#

Unique quote identifier, system-assigned

Author

Picture

The Quote

Process Quote

Remembering Field Values?

At the moment, it looks like field values are being kept when we need to redisplay a form with errors.

The data we are dealing with is simple enough, and all fields for a record are present in the HTML form as well, so we can get away with our current approach.

The better solution is to use a data transfer buffer.

That means using sessions, and storing the record currently being edited or created as a value in the session container.

I am leaving that topic for an upcoming tutorial :-/

Are We Done Yet?

We are as done as I expect you to be for lab 5.

Image uploading and basic processing is not that hard, but I am out of time for this tutorial. I will add that to one of the upcoming ones, perhaps 5C.

The remaining tutorial 5 topics will make it up into the site, but I don't expect you to complete them for lab 5. I am not even sure if I will get them finished before the lab 5 deadline.

I do want them to be ready before you start on Assignment 2 in earnest, as I think there are many techniques that would be useful for you.

Congratulations!

You have completed tutorial #normal05b: Libraries and Helpers, Part (b)

If you would take a minute to [provide some feedback](#), we would appreciate it!

The next activity in sequence is: [normal05c](#) Libraries and Helpers, Part (c)

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.