# Exercise Solutions

## Section 3 Lesson 4. Request Routing

### Exercise 1: Create a Custom Route

Use your IDE to do this exercise.

**Step 1. Create a new module `Training_Routing`**

- Refer to the exercise solutions for Section 2, Lesson 6, Exercise 1 – Configuration XML for instructions on how to create a new module. Adjust the module name so that the module is named `Training_Routing`. After you complete that exercise you will have a skeleton module to work with for this exercise.

**Step 2. Configure a route with the front name `training`**

a) Open the *config.xml* file of the `Training_Routing` module.

b) Add the XML configuration to configure the frontend route `training` inside the `config` node.

```
<frontend>
  <routers>
    <training_routing>
      <use>standard</use>
      <args>
        <module>Training_Routing</module>
        <frontName>training</frontName>
      </args>
    </training_routing>
  </routers>
</frontend>
```

This will add a frontend router named after the class group `training_routing` with the front name `training` mapped to the directory *Training/Routing/controllers/*.

### Step 3. Add a class `Training_Routing_PracticeController`

a) Create the *controllers* directory for the module *app/code/local/Training/Router/controllers/*.

b) Create the file *app/code/local/Training/Router/controllers/PracticeController.php*.

c) Add the PHP class declaration `Training_Router_PracticeController` and extend the class `Mage_Core_Controller_Front_Action`.

```
<?php

class Training_Routing_PracticeController
    extends Mage_Core_Controller_Front_Action
{

}
```

### Step 4. Add an `indexAction()` method

- Add the public function `indexAction()` to the `Training_Routing_PracticeController` class.

```
public function indexAction()
{

}
```

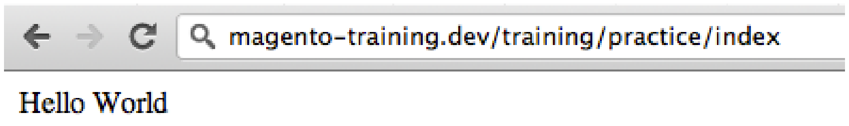
### Step 5. Output `'Hello World'` from it

a) We could simply `echo 'Hello World'` from the method, but then the front controller event `controller_front_send_response_before` will fire after output has already been sent.

The correct way to generate output in a controller class is to fetch an instance of the response object and set the response body.

```
$this->getResponse()->setBody('Hello World');
```

b) Open the browser and call the front name `training` of our module and specify the `practice` controller and the `index` action `/training/practice/index`.



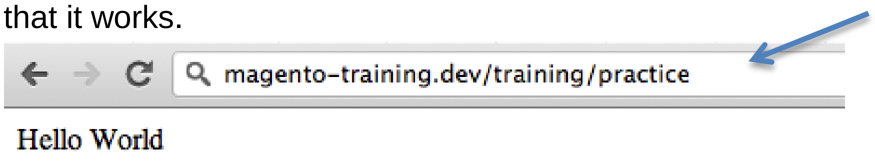
magento-training.dev/training/practice/index

Hello World

You should see the `Hello World` output on an otherwise empty page.

c) If it doesn't work, check that your module configuration is being loaded as described in the exercise solutions for Section 3, Lesson 6, Exercise 1. If that does not resolve the issue, check that the router configuration is correct and that it is contained inside the `<config>` node.

d) Because the index is the default fallback for Magento, the same URL can also be accessed using the request path */training/practice*. Open your browser and check that it works.



magento-training.dev/training/practice

Hello World

## Exercise 2: Override a Controller

You will be extending the `Training_Routing` module from the previous exercise.

### Step 1. Override the `Mage_Customer_AccountController` class

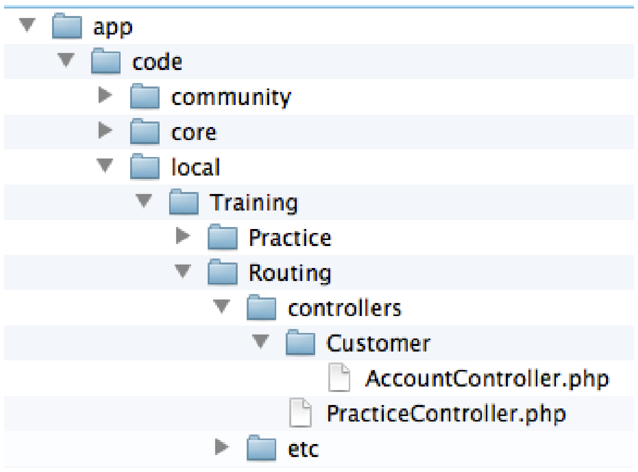
a) Open the *etc/config.xml* file from the `Training_Routing` module.

b) Add the XML to add another module to the list for the `customer` route.

```
<frontend>
  <routers>
    <customer>
      <args>
        <modules>
          <training_routing
          before="Mage_Customer">Training_Routing_Customer</training_routing>
        </modules>
      </args>
    </customer>
  </routers>
</frontend>
```

**Note**: Magento will now first check in the directory *Training/Routing/*controllers*/Customer/* for matching controllers, before looking into *Mage/Customer/controllers/*.

c) Create the file *AccountController.php* inside the *Training/Routing/controllers/Customer/* directory.



```
app
  code
    community
    core
    local
      Training
        Practice
        Routing
          controllers
            Customer
              AccountController.php
            PracticeController.php
          etc
```

d) Add the PHP class declaration `Training_Routing_Customer_AccountController` and extend the class `Mage_Customer_AccountController`.

```php
require_once 'Mage/Customer/controllers/AccountController.php';

class Training_Routing_Customer_AccountController
  extends Mage_Customer_AccountController
{
}
```

**Note:** The file containing the parent class must be included manually because controller class names don't map to the file system, so the autoloader can't include them automatically.

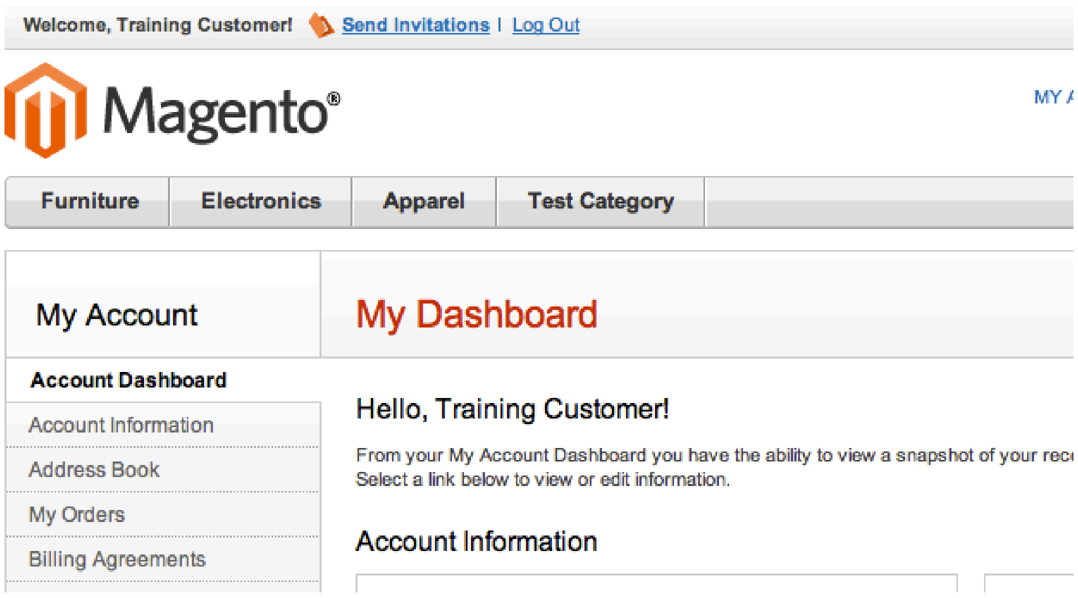
### Step 2. Overload the `loginPostAction()` method

a) Add the `public function loginPostAction()` method. For now call only the parent method, so that the functionality is not broken by the override.

```
class Training_Routing_Customer_AccountController
  extends Mage_Customer_AccountController
{
  public function loginPostAction()
  {
    parent::loginPostAction();
  }
}
```

b) Log in, creating a new customer account if required.

c) Notice that you are redirected to the customer account dashboard after the login.



Hello, Training Customer!

From your My Account Dashboard you have the ability to view a snapshot of your rec
Select a link below to view or edit information.

Account Information

d) Log out of the customer account.

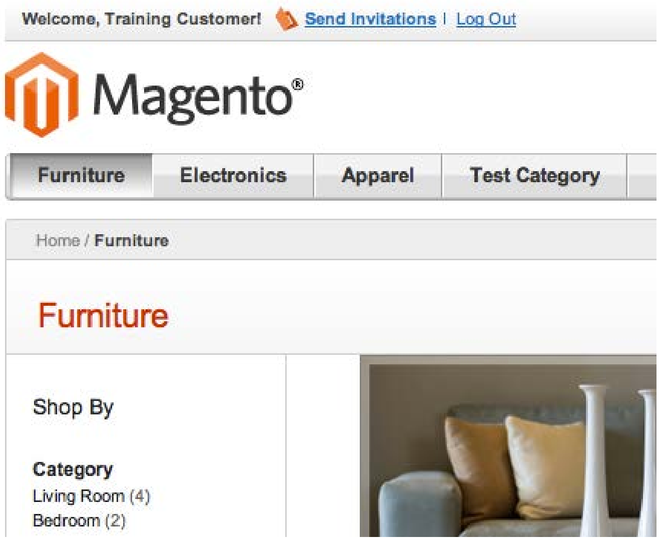
### Step 3. Set the `after_auth_url` property on the customer session model to `'catalog/category/view/id/10'`

a) Update the `loginPostAction()` method so that it sets the `after_auth_url` property of the customer session model before it calls the parent method.

```
public function loginPostAction()
{
  $url = Mage::getUrl('catalog/category/view', array('id' => 10));
  $this->_getSession()->setAfterAuthUrl($url);
  parent::loginPostAction();
}
```

b) Log back in to the customer account.

c) Notice that you are redirected to the furniture category after the login.


Welcome, Training Customer!
Send Invitations
Log Out
Magento
Furniture
Electronics
Apparel
Test Category
Home / Furniture
Furniture
Shop By
Category
Living Room (4)
Bedroom (2)