Overview » MVC » laminas-mvc » Quick Start

Quick Start

On this page

In this example, `/hello/world?message=welcome` will display a page containing the message provided through the URL. This requires several steps:

- Create a controller to obtain the message from the URL, and pass it as a variable to the view
- Create a view to display a page containing the message
- Create a route to match the URL to the controller

Throughout this example, we will work in the `Application` module of the skeleton. The relevant files are located under `module/Application`.

Install the Laminas MVC Skeleton Application

To get started, install the skeleton application via Composer.

If you have not yet done so, install Composer. Once you have, use the `create-project` command to create a new application:

```
$ composer create-project -s dev laminas/laminas-mvc-skeleton my-application
```

Create a Controller

Laminas MVC has several base controller classes for you to start with:

- `AbstractActionController` matches routes to methods within this class. For example, if you had a route with the "list" action, the "listAction" method will be called.
- `AbstractRestfulController` determines the HTTP method from the request, and calls a method according to that. For example, a `POST` HTTP method will call the `update()` method in the class.
- You can also create custom controllers by implementing `Laminas\Stdlib\DispatchableInterface`.

Learn more about controllers in the chapter on controllers.

We will use the `AbstractActionController` base controller. Create the file `src/Application/Controller/HelloController.php`. Add the following code:

```php
<?php
namespace Application\Controller;

use Laminas\Mvc\Controller\AbstractActionController;
use Laminas\View\Model\ViewModel;

// Create an action controller.
class HelloController extends AbstractActionController
{
    // Define an action "world".
    public function worldAction()
    {
        // Get "message" from the query parameters.
        // In production code, it's a good idea to sanitize user input.
        $message = $this->params()->fromQuery('message', 'hello');

        // Pass variables to the view.
        return new ViewModel(['message' => $message]);
    }
}
```

By default, this controller will render the view script located in `view/application/hello/world.phtml`. You can customize this behavior. Learn more about views in the laminas-view documentation.

Create a View Script

Create the file `view/application/hello/world.phtml` and add the following code:

```html
<h1>Greetings!</h1>

<p>You said "<?php echo $this->escapeHtml($message) ?>".</p>
```

> **Escaping output**
>
> The method `escapeHtml()` is a view helper, and it's designed to help mitigate XSS attacks. Never trust user input. If you are at all uncertain about the source of a variable in your view, escape it using one of the view helpers, depending on the type of data.

Create a Route

Routes determine which controller to call based on the URI and other information from the request.

Configure a route and a controller in `module/Application/config/module.config.php`:

```
return [
    'router' => [
        'routes' => [
            // Route name: used to generate links, among other things.
            'hello-world' => [
                'type' => Laminas\Router\Http\Literal::class, // exact match of URI path
                'options' => [
                    'route' => '/hello/world', // URI path
                    'defaults' => [
                        'controller' => Application\Controller\HelloController::class, // unique name
                        'action'     => 'world',
                    ],
                ],
            ],
        ],
    ],
    'controllers' => [
        // Tell the application how to instantiate our controller class
        'factories' => [
            Application\Controller\HelloController::class =>
Laminas\ServiceManager\Factory\InvokableFactory::class,
        ],
    ],
];
```

When the URI path of the request matches `/hello/world`, the specified controller and action will be used. The controller name must be present under the `controllers` key in the configuration. The associated class will then be instantiated and invoked.

Learn more about routing in the laminas-router documentation.

Test it Out!

Create a new vhost pointing its document root to the `public` directory of your application, and fire it up in a browser. You should see the default homepage template of laminas-mvc-skeleton.

Append the path `/hello/world` to your URL and load the page. You should now get the following content:

```
<h1>Greetings!</h1>

<p>You said "hello".</p>
```

Append `?message=welcome` to your URL. You should now get:

```
<h1>Greetings!</h1>

<p>You said "welcome".</p>
```

Congratulations! You've created your first Laminas MVC controller!

Learn More

- Creating custom modules
- Controllers
- Views
- Routing

Previous
Introduction

Next
Default Services

Laminas

Laminas Project The new foundation for the community-supported, open source continuation of Zend Framework

Laminas Components and MVC Components and MVC for enterprise applications

Mezzio PSR-15 middleware in minutes

Laminas API Tools Build RESTful APIs in minutes

Support

License

Copyright