Zend Framework Cheatsheet

A quick reference guide for Zend Framework, covering essential components, configurations, and best practices.

Module Management

Module Structure

A Zend Framework module typically includes the following directories:

- `config/` : Configuration files (module.config.php)
- `src/` : Source code (controllers, models, forms, etc.)
- `view/` : View scripts
- `Module.php` : Module class

Module Configuration

`module.config.php`	Main configuration file for the module. Defines routes, controllers, services, and view configurations.
`Module.php`	Module class that defines the `getConfig()` method to merge module-specific configurations.
Loading Modules	Modules are typically loaded in the `config/application.config.php` file under the `modules` key.

Example: module.config.php

```
return [
    'controllers' => [
        'factories' => [
            'MyModule\Controller\Index' =>
'MyModule\Factory\IndexControllerFactory',
        ],
    ],
    'router' => [
        'routes' => [
            'mymodule' => [
                'type'    => 'Literal',
                'options' => [
                    'route'    => '/mymodule',
                    'defaults' => [
                        'controller' =>
'MyModule\Controller\Index',
                        'action'     =>
'index',
                    ],
                ],
            ],
        ],
    ],
    'view_manager' => [
        'template_path_stack' => [
            'mymodule' => __DIR__ .
'/../view',
        ],
    ],
];
```

Controllers and Actions

Controller Basics

Controllers handle incoming requests and return responses.

- Extend `AbstractActionController`.
- Define action methods (e.g., `indexAction()`).
- Retrieve request parameters using `$this->params()`.

Action Methods

`indexAction()`	Default action method. Typically displays a list of resources or a homepage.
`createAction()`	Handles the creation of a new resource, typically via a form submission.
`updateAction()`	Handles the updating of an existing resource, typically via a form submission.
`deleteAction()`	Handles the deletion of a resource.

Example: Controller

```
namespace MyModule\Controller;

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

class IndexController extends
AbstractActionController
{
    public function indexAction()
    {
        return new ViewModel();
    }
}
```

View Layer

View Scripts

View scripts are PHP files that generate the HTML output.

- Located in the `view/` directory of a module.
- Use the `.phtml` extension.
- Access variables passed from the controller.

View Helpers

`$this->url()`	Generates a URL based on a route name.
`$this->form()`	Renders a form.
`$this->escapeHtml()`	Escapes HTML entities in a string.
`$this->translate()`	Translates a string using the translation service.

Example: View Script

```
<h1>Welcome!</h1>

<p>This is a sample view script.</p>

<a href="<?php echo $this->url('mymodule'); ?
>">MyModule Home</a>
```

Layouts

Layouts provide a consistent structure for multiple pages. Configure the layout in the `view_manager` section of your `module.config.php` or `application.config.php`. Set the default layout with `'template' => 'layout/layout'`.

Forms and Input Filters

Form Basics

Forms are used to handle user input.

- Extend `Zend\Form\Form`.
- Add elements with appropriate input types.
- Validate input using input filters.

Input Filters

`Zend\InputFilter\InputFilter`	Used to define validation rules for form elements.
Validators	Examples: `StringLength`, `EmailAddress`, `NotEmpty`
Filters	Examples: `StringTrim`, `StripTags`, `ToInt`

Example: Form

```
namespace MyModule\Form;

use Zend\Form\Form;
use Zend\Form\Element;

class MyForm extends Form
{
    public function __construct($name = null,
$options = [])
    {
        parent::__construct($name, $options);

        $this->add([new
Element\Text('username')]);
        $this->add([new
Element\Email('email')]);
        $this->add([new
Element\Submit('submit', ['label' =>
'Submit'])]);
    }
}
```