

# CodeIgniter - Using View layouts and templates

Topics : [Codeigniter](#)

Written on [February 29, 2024](#)

In CodeIgniter, you can implement view layouts and templates to create consistent and reusable layouts for your web application. Layouts and templates allow you to define a common structure for your views, including headers, footers, navigation menus, and other shared elements. Here's how you can use view layouts and templates in CodeIgniter:

## 1. Creating Layouts:

- Create a layout file that defines the common structure for your views. This layout typically contains HTML markup for the header, footer, and other shared elements.
- You can include placeholders or markers within the layout file to indicate where the content of individual views should be inserted.

Example layout file (layout.php):

```
<!DOCTYPE html>
<html>
<head>
    <title><?php echo $title; ?></title>
</head>
<body>
    <header>
        <!-- Header content -->
    </header>

    <nav>
        <!-- Navigation menu -->
    </nav>

    <main>
        <!-- Content placeholder -->
        <?php echo $content; ?>
    </main>

    <footer>
        <!-- Footer content -->
    </footer>
```

```
</body>
</html>
```

## 2. Creating Views:

- Create individual view files for your application pages or sections.
- Each view file contains the specific content to be displayed within the layout.

Example view file (home.php):

```
<section>
    <h1>Welcome to My Website</h1>
    <p>This is the homepage content.</p>
</section>
```

## 3. Loading Views with Layouts:

- In your controller method, load the layout and views using the `$this->load->view()` method.
- Pass the layout file and view data as parameters to the `load->view()` method.

Example controller method:

```
public function index() {
    // Load layout with view content
    $data['title'] = 'Home';
    $data['content'] = $this->load->view('home', '', TRUE); // Load view
content into a variable
    $this->load->view('layout', $data); // Pass layout and data to
load->view() method
}
```

## 4. Passing Data to Views:

- You can pass data to views as an associative array when loading the view.
- In the example above, `$data['title']` is passed to the layout file to set the page title, and `$data['content']` contains the content of the specific view.

## 5. Reusing Layouts and Templates:

- By using layouts and templates, you can create a consistent look and feel across multiple pages of your web application.
- You can reuse layouts and templates for different views, reducing duplication of code and promoting maintainability.

## 6. Advanced Techniques:

- You can create multiple layouts for different sections of your application or for different user roles.
- Use view partials or template inheritance to further modularize your views and layouts.

ARYATECHNO