Assets

Frontend Asset Management

Anonymous · 09/09/2026

Frontend Asset Management

Derafu Foundation includes a pre-configured asset build system using Vite. This document explains how to work with frontend assets in your project.

Overview

The asset management system handles:

- CSS compilation.
- JavaScript bundling.
- Image optimization.
- Source maps generation.
- Asset versioning.

Directory Structure

```
project_dir/
├── assets/
│   ├── css/
│   │   └── app.css           # Main CSS file.
│   ├── img/                  # Image files.
│   └── js/
│       ├── app.js            # Main JavaScript entry point.
│       └── images.js         # Image import handling.
├── public/
│   └── static/               # Output directory (generated).
│       ├── css/
│       ├── img/
│       └── js/
└── vite.config.js            # Vite configuration.
```

Configuration

The Vite configuration is defined in `vite.config.js`. Key features include:

- Multiple entry points: `app.js`, `app.css`, and `images.js`.
- Output directory: `public/static/`.
- Custom file naming with `.min` suffix.
- Image optimization using `vite-plugin-sharp`.

Entry Points

- `app.js`: Main JavaScript code.
- `app.css`: Main CSS styles.
- `images.js`: Used for processing and importing images.

Image Optimization

Images are automatically optimized during the build process using the Sharp image processing library. The configuration includes:

- PNG compression: Quality 85%, maximum compression level.
- JPEG compression: Quality 85%, progressive encoding.
- Automatic resizing of large images to a maximum of 2000×2000 pixels.

Usage

Adding Assets

1. **CSS**: Add your CSS files to the `assets/css/` directory.
2. **JavaScript**: Add your JavaScript files to the `assets/js/` directory.
3. **Images**: Add your images to the `assets/img/` directory.

Importing Assets

CSS

In your main `app.css` file:

```
/* Import other CSS files */
@import './components/buttons.css';
```

JavaScript

In your JavaScript files:

```
// Import other JavaScript files.
import './components/slider.js';

// Import CSS from JavaScript.
```

```
import '../css/components/modal.css';

// Import images.
import logo from '../img/logo.png';
```

Images

For processing multiple images at once, use the `images.js` file:

```
// Import all images in a directory.
import.meta.glob('../img/**/*');
```

Building Assets

To build assets for production:

```
npm run build
```

This will:

1. Compile and bundle all assets.
2. Optimize images.
3. Generate minified files with source maps.
4. Output everything to the `public/static/` directory.

Using Built Assets

In your HTML/Twig templates, reference the built assets:

```
<link rel="stylesheet" href="https://www.derafu.dev/static/css/app.min.css">
<script src="/static/js/app.min.js"></script>
<img src="/static/img/logo.png">
```

Best Practices

1. **Organize by component**: Group related CSS, JS, and images by feature or component.
2. **Optimize images before adding**: While the build process optimizes images, starting with optimized images is better.
3. **Use CSS imports**: Keep CSS modular by using imports.
4. **Lazy load images**: For image-heavy pages, consider implementing lazy loading.
5. **Watch file size**: Monitor the size of your built assets to ensure optimal loading times.

Advanced Configuration

The Vite configuration can be extended to support:

- CSS preprocessors (Sass, Less, etc.).
- TypeScript.
- Additional plugins for specific needs.
- Custom output paths and formats.

To modify the configuration, edit the `vite.config.js` file as needed (remember to deactivate the `Derafu\\Foundation\\Installer::copyFiles` from the `post-install-cmd` and `post-update-cmd` scripts in your `composer.json` file).

Last updated on 09/09/2026