Design Principles

Design Principles

Anonymous · 09/09/2026

Design Principles

The Derafu HTTP component provides a lightweight, standards-compliant approach to HTTP request handling. By following established PHP interoperability standards (PSRs) and sound architectural principles, it enables the development of robust, maintainable web applications with minimal overhead.

The Derafu HTTP is designed with several key principles in mind:

PSR Compliance

- Implements PSR-7 for HTTP messages.
- Follows PSR-11 for container interoperability.
- Supports PSR-15 for middleware.

Separation of Concerns

Each component has a single, well-defined responsibility:

- Runtime manages the application lifecycle.
- Kernel coordinates processing.
- Handlers implement business logic.
- Container manages dependencies.

Flexibility

The design allows for:

- Custom request handlers.
- Middleware integration.
- Extensible container configuration.
- Multiple runtime environments.

Testability

The clear separation of components and dependency injection make unit testing straightforward:

- Mock the container for handler tests.
- Create test requests easily.
- Validate responses without invoking the full stack.

Implementation Guidelines

When working with the Derafu HTTP component:

1. **Define Request Handlers** for different routes or endpoints.
2. **Configure the Container** with your application services.
3. **Set up your routes** to map URLs to handlers.
4. **Extend the base classes** as needed for custom functionality.

The architecture is designed to be minimal yet powerful, allowing developers to focus on the business logic rather than HTTP processing details.

Last updated on 09/09/2026