Custom Registrar

Creating Custom Rule Registrar

Anonymous · 09/09/2026

Creating Custom Rule Registrar

For better organization, you can create a custom rule registrar:

```
use Derafu\DataProcessor\Contract\RuleRegistrarInterface;
use Derafu\DataProcessor\Contract\RuleRegistryInterface;

final class CustomRuleRegistrar implements RuleRegistrarInterface
{
    public function register(RuleRegistryInterface $registry): void
    {
        // Register caster rules.
        $registry
            ->addCasterRule('custom_type_1', CustomType1Rule::class)
            ->addCasterRule('custom_type_2', CustomType2Rule::class);

        // Register transform rules.
        $registry
            ->addTransformerRule('custom_transform_1', CustomTransform1Rule::class)
            ->addTransformerRule('custom_transform_2', CustomTransform2Rule::class);

        // Register sanitizer rules.
        $registry
            ->addSanitizerRule('custom_sanitize_1', CustomSanitize1Rule::class)
            ->addSanitizerRule('custom_sanitize_2', CustomSanitize2Rule::class);

        // Register validator rules.
        $registry
            ->addValidatorRule('custom_validate_1', CustomValidate1Rule::class)
            ->addValidatorRule('custom_validate_2', CustomValidate2Rule::class);
    }
}

// Usage.
$processor = ProcessorFactory::create(
    new CustomRuleRegistrar(),
    withDefaultRules: false // Create without default rules using the factory.
);
```

This allows you to:

- Create domain-specific rule sets.
- Override default rules with custom implementations.
- Group related rules together.
- Control which rules are available in your application.

Last updated on 09/09/2026