# PHP UK CONFERENCE

15th & 16th February 2024

www.phpconference.co.uk

## Dave Liddament

Elevating Legacy: A Case Study on the migration from Laravel 4 to 9

The Brewery, London, UK

@daveliddament

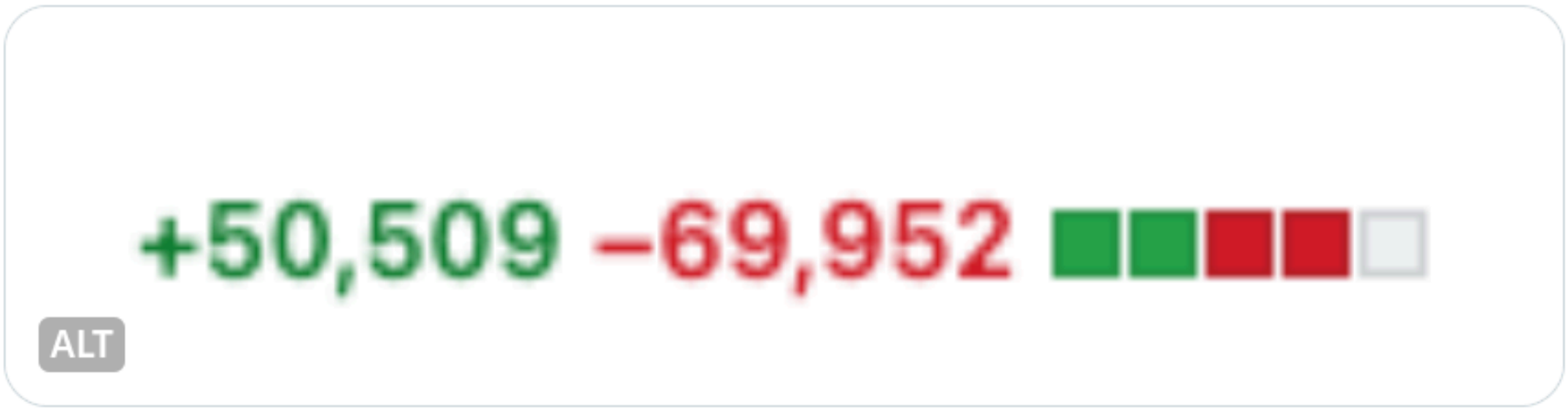
**Dave Liddament**
@DaveLiddament

Finally completed an upgrade from #laravel 4.2 to 9.

To celebrate Laravel have released version 10!

Remember your tests folks. This upgrade wouldn't have been possible, or as smooth, without tests.
Also @rectorphp is pretty handy for this.

I must write a blog/talk about it.




5:25 PM · Feb 24, 2023 · **3,813** Views

Writing a Custom Rector Rule

Thoughts

Execute

Code Prehab

Plan

@daveliddament

Writing a Custom Rector Rule

Thoughts

Execute

Code Prehab

Plan

@daveliddament

Photo by Antoine GIRET on Unsplash

@daveliddament

# Add tombstones

```
final class PriceCalculator
{
 public function calculate(): void
 {
    TombstoneReporter::trigger(
      'PriceCalculator::calculate');

    // Rest of method's code
  }
}
```

# Trigger tombstones

triggered_tombstones.json

```
[
    'PriceCalculator::calculate',
    'PriceController::index',
    'PriceRepository::getItems,
    ...
];
```

Photo by Christin Hume on Unsplash

@daveliddament

# Remove triggered tombstones

```
final class PriceCalculator
{
 public function calculate(): void
 {
    TombstoneReporter::trigger(
      'PriceCalculator::calculate');

    // Rest of method's code
  }
}
```

# Delete code

```
“REMOVE: Product listing page”

“REMOVE: 2 for 1 discount code”
```

Photo by Claudio Schwarz on Unsplash

@daveliddament

# Increase type coverage

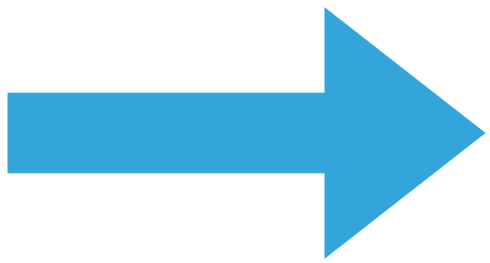
```
function getNames($users)
{

   $names = [];
   foreach($users as $user) {
      $names[] = $user->getUser();
   }

   return join(‘,’, $names);
}
```

**0% type coverage**

```php
/** @array<int,User> $users */
function getNames(array $users): string
{

  $names = [];
  foreach($users as $user) {
    $names[] = $user->getUser();
  }
  return join(‘,’, $names);

}
```

**100% type coverage**

# User

getUser  getUsername

# 0% type coverage

```
function getNames($users)
{

   $names = [];
   foreach($users as $user) {
      $names[] = $user->getUser();
   }

   return join(‘,’, $names);
}
```

# 100% type coverage

```
/** @array<int,User> $users */
function getNames(array $users): string
{

   $names = [];
   foreach($users as $user) {
     $names[] = $user->getUser();
   }
   return join(‘,’, $names);

}
```

# 100% type coverage

```
/** @array<int,User> $users */
function getNames(array $users): string
{

   $names = [];
   foreach($users as $user) {
     $names[] = $user->getUsername();
   }
   return join(‘,’, $names);

}
```

## 100% automated!

# Tests

Photo by Pascal Meier on Unsplash

@daveliddament

# Code Prehab

Remove dead code

Increase Type coverage

Increase test coverage

Writing a Custom Rector Rule

Thoughts

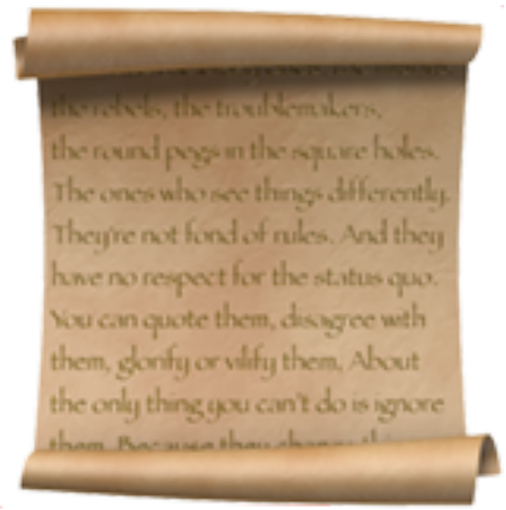
Execute

Code Prehab

Plan

@daveliddament

# SHIFT




@daveliddament


https://laravelshift.com/

@daveliddament

https://getrector.com/

# 359 Rules (and counting)

## RenameMethodRector

Turns method names to new ones.

### 🔧 configure it!

- class: `Rector\Renaming\Rector\MethodCall\RenameMethodRector`

```
 $someObject = new SomeExampleClass;
-$someObject->oldMethod();
+$someObject->newMethod();
```

## ChangeIfElseValueAssignToEarlyReturnRector

Change if/else value to early return

- class: `Rector\EarlyReturn\Rector\If_\ChangeIfElseValueAssignToEarlyReturnRector`

```
class SomeClass
{
    public function run()
    {
        if ($this->hasDocBlock($tokens, $index)) {
-            $docToken = $tokens[$this->getDocBlockIndex($tokens, $index)];
-        } else {
-            $docToken = null;
+            return $tokens[$this->getDocBlockIndex($tokens, $index)];
        }
-
-        return $docToken;
+        return null;
    }
}
```

```
composer require --dev rector/rector
```

```
vendor/bin/rector
```

```
No "rector.php" config found. Should we
generate it for you? [yes]:
 > yes
```

```
 [OK] The config is added now. Re-run command
to make Rector do the work!
```

```php
<?php

use Rector\Config\RectorConfig;
use Rector\Renaming\Rector\MethodCall\RenameMethodRector;
use Rector\Renaming\ValueObject\MethodCallRename;

return RectorConfig::configure()
    ->withPaths([
        __DIR__ . '/src',
    ])
->withConfiguredRule(
    RenameMethodRector::class,
    [
        new MethodCallRename(
            App\Framework\Model::class,
            'belongsTo',
            ‘belongs')
    ]
);
```

```
 7/7 [============================] 100%
2 files with changes
====================

1) src/Framework/Model.php:5

    ---------- begin diff ----------
@@ @@

 abstract class Model
 {
-    public function belongsTo(string $model): BelongsTo
+    public function belongs(string $model): BelongsTo
     {
         return new BelongsTo($model);
     }
    ----------- end diff -----------

Applied rules:
 * RenameMethodRector
```

```
2) src/Models/Car.php:10

    ---------- begin diff ----------
@@ @@
 {
     public function user(): BelongsTo
     {
-        return $this->belongsTo('User');
+        return $this->belongs('User');
     }

     public function manufacturer(): BelongsTo
     {
-        return $this->belongsTo(Manufacturer::class);
+        return $this->belongs(Manufacturer::class);
     }
 }
    ---------- end diff ----------

Applied rules:
 * RenameMethodRector
```

# Rulesets:  A set of rules

Rector: PHP Language upgrades

https://github.com/rectorphp/rector-symfony

https://github.com/driftingly/rector-laravel

### Renamed "Swift" Methods

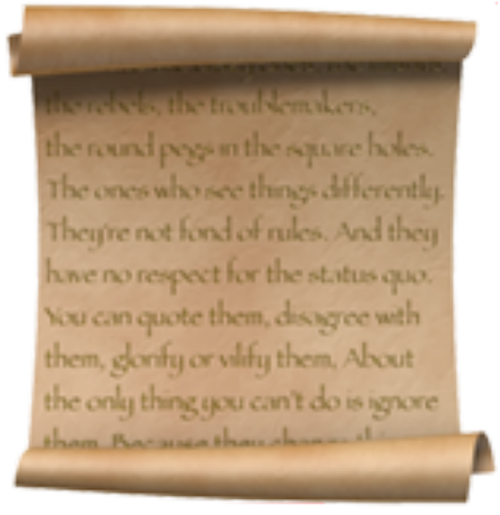
Various SwiftMailer related methods, some of which were undocumented, have been renamed to their Symfony Mailer counterparts. For example, the `withSwiftMessage` method has been renamed to `withSymfonyMessage`:

https://laravel.com/docs/9.x/upgrade#symfony-mailer

---

```php
$rectorConfig>ruleWithConfiguration(
RenameMethodRector::class,
[
    new MethodCallRename(
        'Illuminate\Mail\Message',
        'withSwiftMessage',
        'withSymfonyMessage'),
]);
```

https://github.com/driftingly/rector-laravel/blob/main/config/sets/laravel90.php#L89




@daveliddament

Laravel

Search

Prologue

- Release Notes
- Upgrade Guide
- Contribution Guide

# Upgrade Guide

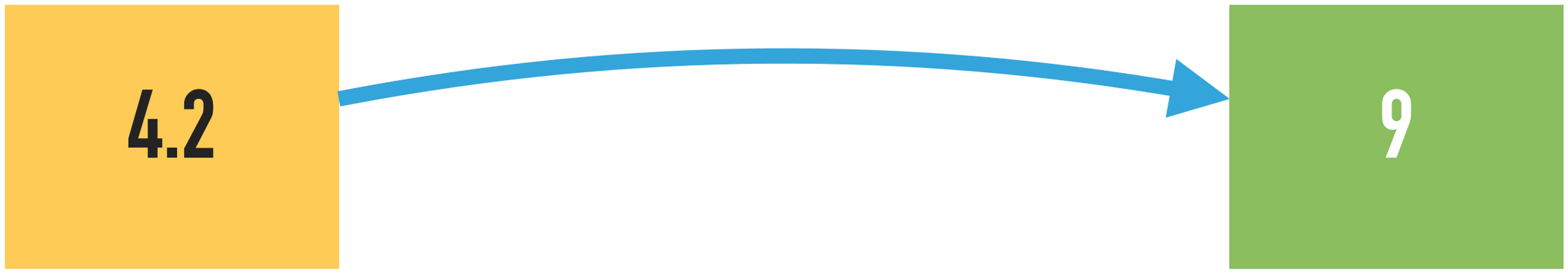
## # Upgrading to 10.0 from 9.x

Symfony

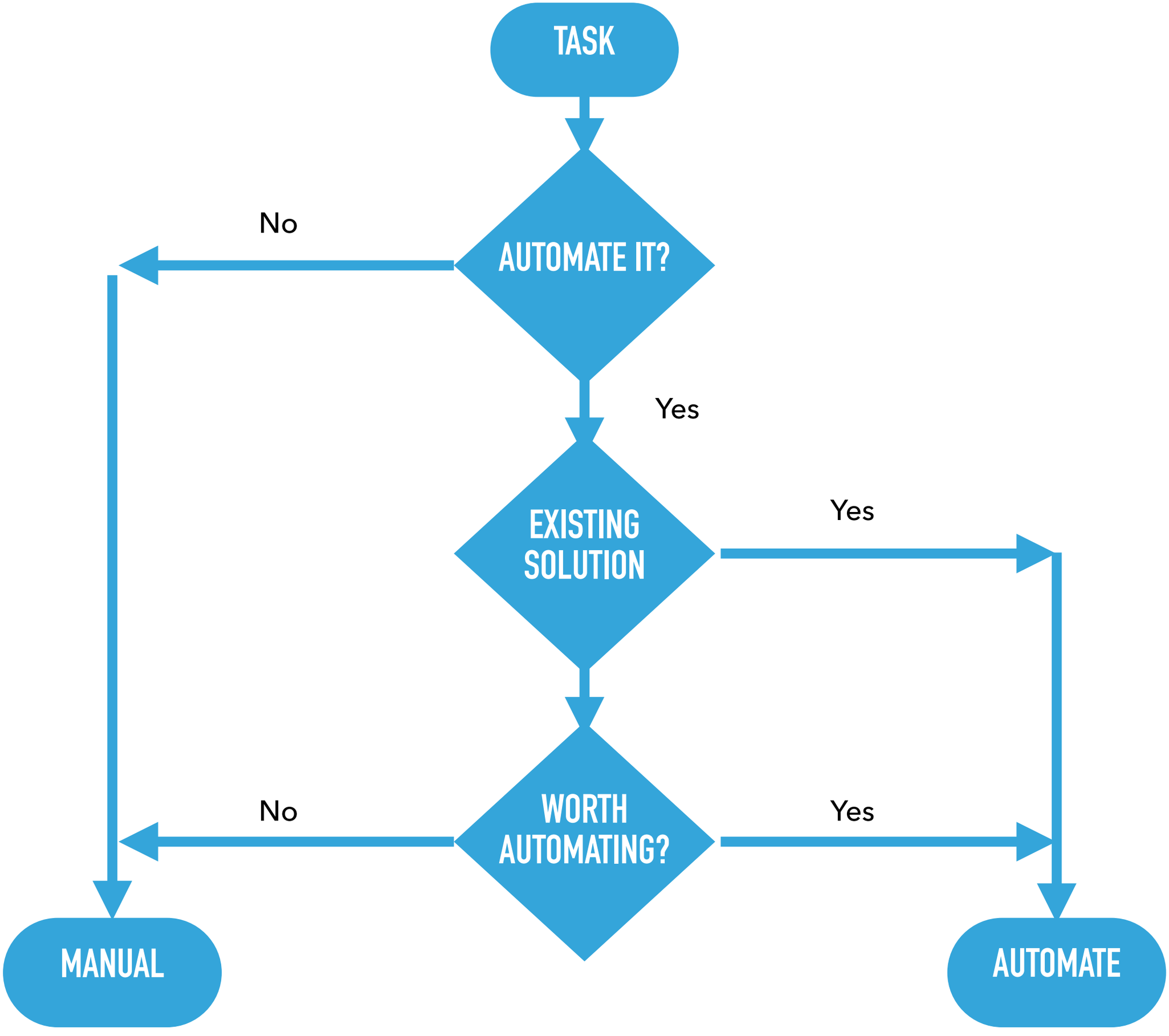
Upgrade to highest minor version → Fix all the deprecation warnings → Upgrade major version → Run flex recipes

https://symfony.com/doc/current/setup/upgrade_major.html

@daveliddament




- [x] Laravel validators
- [x] Symfony validators
- [x] E2E tests
- [x] Initial controller (probably login one)
- [x] Controllers and middleware
- [x] Remaining commands
- [x] Unit tests
- [x] Multiple environments (currently only test environment is setup)
  Includes ansible updates
- [x] Full testings
- [x] Merge in changes done since this branch started
- [x] PHP 8.1 upgrade


TASK
AUTOMATE IT?
No
Yes
EXISTING SOLUTION
Yes
WORTH AUTOMATING?
No
Yes
MANUAL
AUTOMATE

Development

UPGRADE

Development

@daveliddament

# Smaller deploys would be better*

| | Framework | PHP | PHPUnit |
|---|---|---|---|
| Current deployed versions | 12 | 8.1 | 8 |
| Current supported versions | 12 | 8.0, 8.1, 8.2 | 8, 9 |
| New supported versions | 13 | 8.2, 8.3 | 9, 10 |

| | Framework | PHP | PHPUnit |
|---|---|---|---|
| Current deployed versions | 12 | 8.1 | 8 |
| Current supported versions | 12 | 8.0, 8.1, 8.2 | 8, 9 |
| New supported versions | 13 | 8.2, 8.3 | 9, 10 |

Development

UPGRADE

Development

Development

UPGRADE

Development

Dev

MOVING CODE

Dev

UPGRADE

Development

WRITE AUTOMATION SCRIPTS

@daveliddament

# Code Prehab

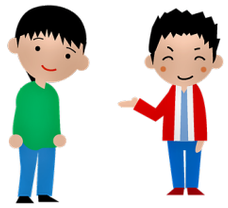
# Remove dead code

# Increase Type coverage

# Increase test coverage

# Plan

SHIFT



- [x] Laravel validators
- [x] Symfony validators
- [x] E2E tests
- [x] Initial controller (probably login one)
- [x] Controllers and middleware
- [x] Remaining commands
- [x] Unit tests
- [x] Multiple environments (currently only test environment is setup)
  Includes ansible updates
- [x] Full testings
- [x] Merge in changes done since this branch started
- [x] PHP 8.1 upgrade

@daveliddament

Writing a Custom
Rector Rule

Thoughts

Execute

Code Prehab

Plan

@daveliddament

# Tips

- Be systematic; do one type of change at once. (Both manual and automation).
  - Note other changes you need to make.
- Use git commits (even if you later squash them)
- Upgrades: fix the deprecations, you don’t need to (immediately) use the new features.
- Do the minimum number of changes between possible release points

# Writing a Custom Rector Rule


@daveliddament

Old

```
class Car extends Model
{
  public function user()
  {
    return $this->belongsTo('User');
  }
}
```

New

```
class Car extends Model
{
  public function user()
  {
    return $this->belongsTo(App\Models\User::class);
  }
}
```

# No Change

```
class Car extends Model
{
  public function user()
  {
    return $this->belongsTo(User::class);
  }
}
```

# No Change

```
class CarService
{
  public function user()
  {
    return $this->belongsTo('User');
  }
}
```

```
$ vendor/bin/rector custom-rule

 What is the name of the rule class (e.g. "LegacyCallToDbalMethodCall")?:
 > FixModelMappingsRector

Generated files
===============

 * utils/rector/src/Rector/FixModelMappingsRector.php
 * utils/rector/tests/Rector/FixModelMappingsRector/Fixture/some_class.php.inc
 * utils/rector/tests/Rector/FixModelMappingsRector/config/configured_rule.php
 * utils/rector/tests/Rector/FixModelMappingsRector/FixModelMappingsRectorTest.php

 [OK] Base for the "FixModelMappingsRector" rule was created. Now you can fill the
missing parts

 [OK] We also update composer.json autoload-dev, to load Rector rules. Now run
"composer dump-autoload" to update paths
```

@daveliddament

**Old**

```php
<?php
namespace Utils\Rector\Tests\Rector\FixModelMappingsRector\Fixture;

use App\Framework\BelongsTo;
use App\Framework\Model;

final class Car1 extends Model
{
    public function user(): BelongsTo
    {
        return $this->belongsTo('User');
    }
}
```

**New**

```php
<?php
namespace Utils\Rector\Tests\Rector\FixModelMappingsRector\Fixture;

use App\Framework\BelongsTo;
use App\Framework\Model;

final class Car1 extends Model
{
    public function user(): BelongsTo
    {
        return $this->belongsTo(App\Models\User::class);
    }
}
```

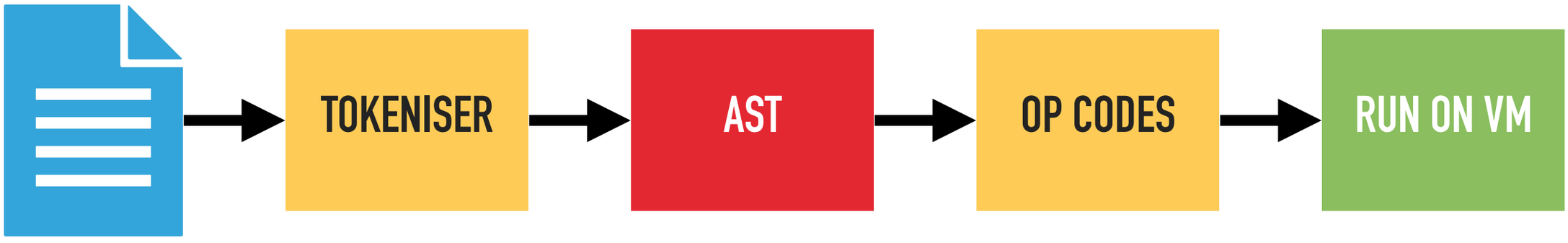
**No change**

```php
<?php
namespace Utils\Rector\Tests\Rector\FixModelMappingsRector\Fixture;

use App\Framework\BelongsTo;
use App\Framework\Model;

final class Car1 extends Model
{
    public function user(): BelongsTo
    {
        return $this->belongsTo(App\Models\User::class);
    }
}
```

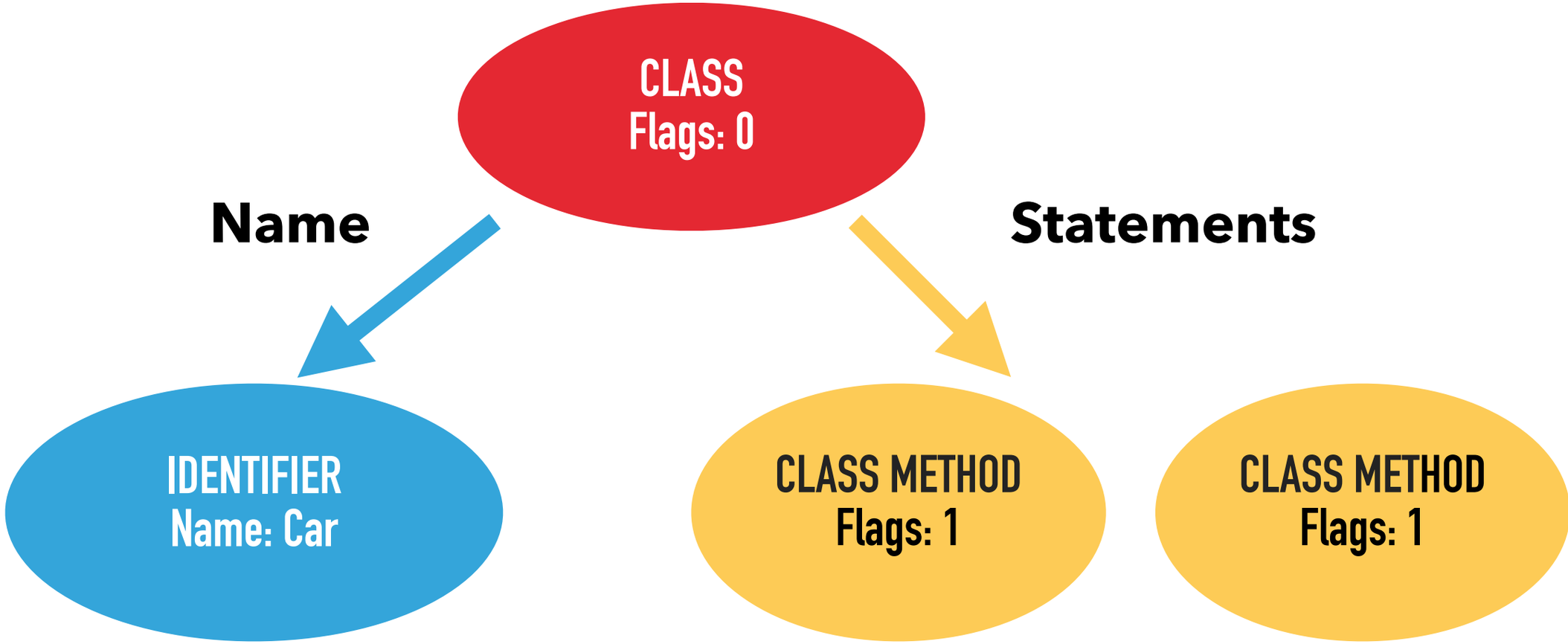
```
FixModelMappingsRector extends AbstractRector
{

    public function getRuleDefinition(): RuleDefinition

    public function getNodeTypes(): array

    public function refactor(Node $node): ?Node
}
```

```php
public function getRuleDefinition(): RuleDefinition
{
    return new RuleDefinition(
        "Fix Model mappings",
        [
            new CodeSample(
                '$this->belongsTo("User");',
                '$this->belongsTo(User::class);',
            ),
        ]
    );
}
```

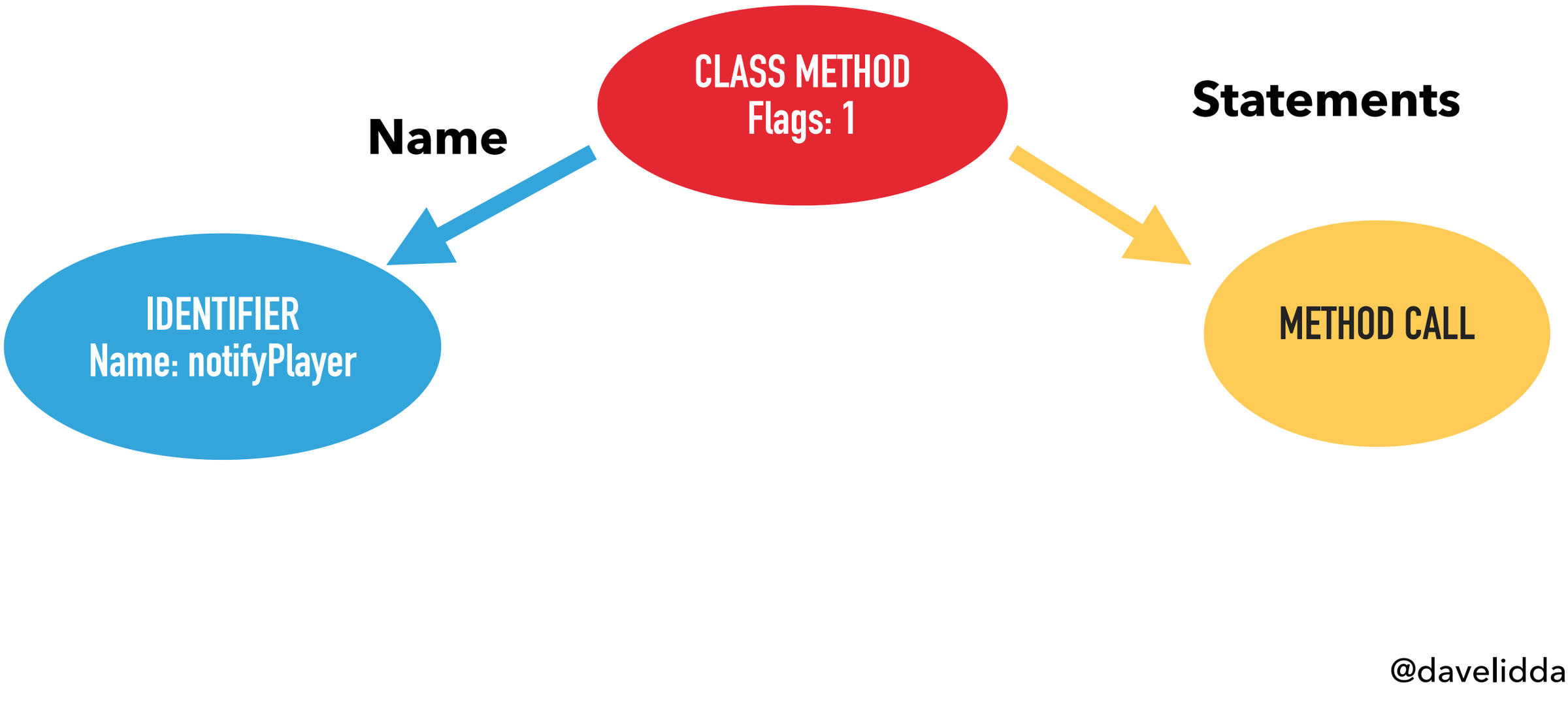
```
FixModelMappingsRector extends AbstractRector
{

    public function getRuleDefinition(): RuleDefinition

    public function getNodeTypes(): array

    public function refactor(Node $node): ?Node
}
```


TOKENISER
AST
OP CODES
RUN ON VM

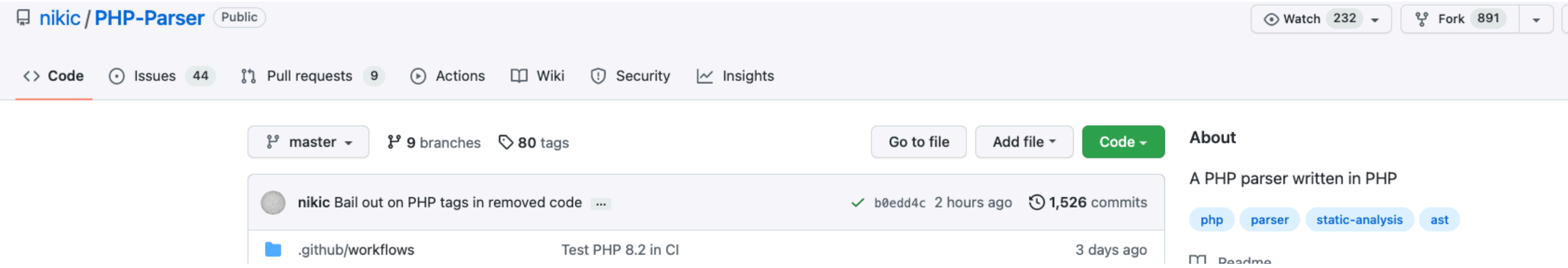
```
class Car
{
    public function users() {…}

    public function anotherMethod() {…}
}
```



```
public function user ()
{
    return $this->belongsTo('User');
}
```



# https://github.com/nikic/PHP-Parser




- PHP code can be represented by an AST
- Different types of Node
- Nodes contain information
- Each type of node has different information

CLASS
Flags: 0

IDENTIFIER
Name: PersonNotifier

PROPERTY
Flags: 4

CLASS METHOD
Flags: 1

CLASS METHOD
Flags: 1

IDENTIFIER
Name: notifyPlayer

PARAM
ByRef: false
Variadic: false

METHOD CALL

IDENTIFIER
Name: string

VALUE
ByRef: false
Variadic: false

```
$this->belongsTo('User');

$this->belongsTo(App\Models\User::class);
```

```
class MethodCall extends \PhpParser\Node\Expr\CallLike
{
    /** @var Expr Variable holding object */
    public $var;

    /** @var Identifier|Expr Method name */
    public $name;

    /** @var array<Arg|VariadicPlaceholder> Arguments */
    public $args;

    // Rest of class …
```

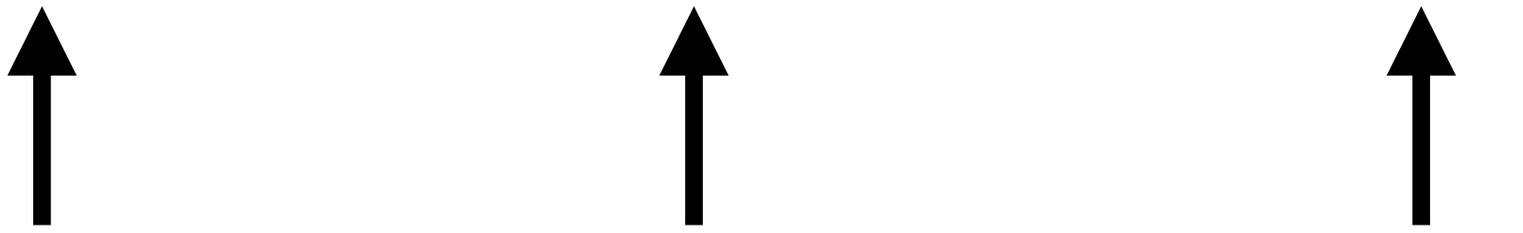
---

```
$this -> belongsTo ('User');
```

```
public function getNodes(): array
{
    return [MethodCall::class];
}
```

```
public function refactor(Node $node): ?Node
{

}
```

```
$this->belongsTo('User');
```




- var is an object that extends Model
- name is belongsTo
- 1st argument is a string

```
public function refactor(Node $node): ?Node
{
    // Is var an object that extends Model?

    $modelType = new ObjectType(Model::class);
    if (!$this->isObjectType($node->var, $modelType)) {
        return null;
    }
```

```
// Is name belongsTo

if (!$this->isName($node->name, 'belongsTo')) {
    return null;
}
```

```
// Is 1st argument a string?

$arg = $node->args[0] ?? null;
if (! $arg instanceof Arg) {
    return null;
}

if (! $arg->value instanceof String_) {
    return null;
}
```

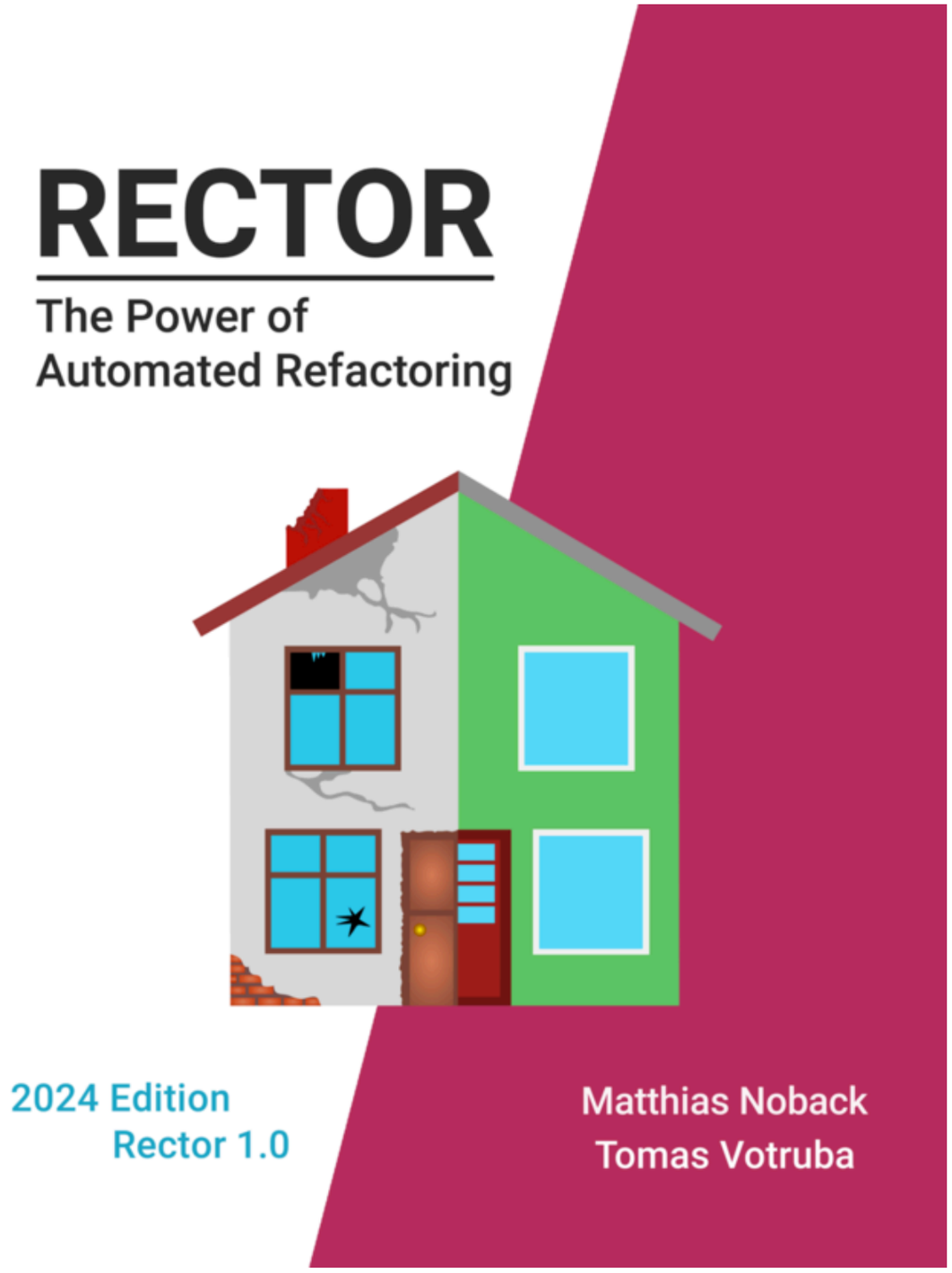
```
// Return updated node

$fqcn = 'App\\Models\\' . $arg->value->value;

$arg->value = new ClassConstFetch(
                        new Name($fqcn),
                        new Identifier("class"));

    return $node;
}
```

---

**User::class** is a ClassConstFetch Node in the AST

```php
public function refactor(Node $node): ?Node
{
    // Is var an object that extends Model?
    $modelType = new ObjectType(Model::class);
    if (!$this->isObjectType($node->var, $modelType)) {
        return null;
    }

    // Is name belongsTo
    if (!$this->isName($node->name, 'belongsTo')) {
        return null;
    }

    // Is 1st argument a string?
    $arg = $node->args[0] ?? null;
    if (! $arg instanceof Arg) {
        return null;
    }
    if (! $arg->value instanceof String_) {
        return null;
    }

    // Return updated node
    $fqcn = 'App\\Models\\' . $arg->value->value;
    $arg->value = new ClassConstFetch(new Name($fqcn),new Identifier("class"));

    return $node;
}
```

https://github.com/DaveLiddament/rector-rule-demo-update-belongs-to




@daveliddament

# Code Prehab

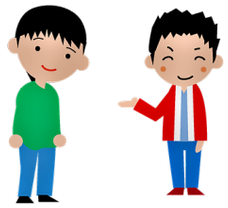
## Remove dead code

## Increase Type coverage

## Increase test coverage

# Plan

SHIFT



- [x] Laravel validators
- [x] Symfony validators
- [x] E2E tests
- [x] Initial controller (probably login one)
- [x] Controllers and middleware
- [x] Remaining commands
- [x] Unit tests
- [x] Multiple environments (currently only test environment is setup)
  Includes ansible updates
- [x] Full testings
- [x] Merge in changes done since this branch started
- [x] PHP 8.1 upgrade

# Execute

## Small focused steps

## Minimum to upgrade

@daveliddament

Writing a Custom Rector Rule

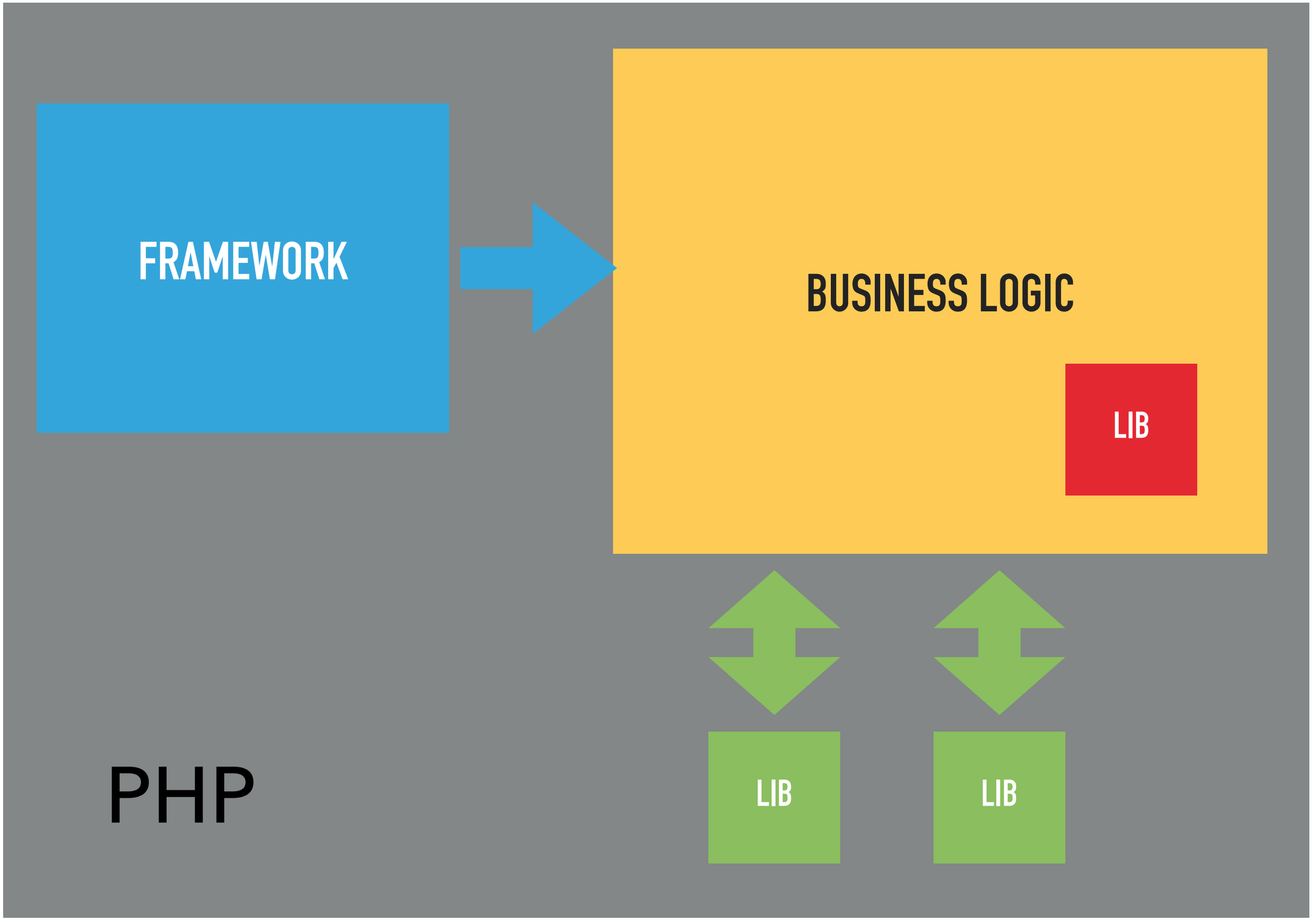
Thoughts

Execute

Code Prehab

Plan

@daveliddament




@daveliddament

# Dave Liddament

@daveliddament