

PHPStan の力で Algebraic Data Types を 実現する

nsfisis (いまむら)

第 160 回 PHP 勉強会@東京

自己紹介

nsfisis (いまむら)



@ デジタルサーカス株式会社

Algebraic Data Type

代数的データ型

すごい enum 型

ADT の例

```
// 注: これは架空の文法です
enum JsonValue {
    case Null;
    case Boolean(bool);
    case Number(int|float);
    case String(string);
    case Array(array);
    case Object(array);
}
```

ADT の例 (今回扱うもの)

```
// 注: これは架空の文法です
enum OptionalInt {
    case None;
    case Some(int);
}
```

実現したいもの

```
// OptionalInt を返す関数
function f() { /* 略 */ }

$x = f();
if (/* $x が値を持っているかどうか判定する */) {
    echo /* $x が内部に持っている int の値を「安全に」取り出す */;
}
```

どうやって実現するか？

Array shape 編

Object shape 編

Type assertion 編

PHPStan 魔改造編

Array shape 編

Array shape

```
$none = [  
  'has_value' => false,  
];  
$some = [  
  'has_value' => true,  
  'value' => 42,  
];
```

Array shape

```
/** @var array{has_value: false} $none */  
$none = [  
    'has_value' => false,  
];  
/** @var array{has_value: true, value: int} $some */  
$some = [  
    'has_value' => true,  
    'value' => 42,  
];
```

Array shape

```
/** @return (array{has_value: false}  
          |array{has_value: true, value: int}) */  
function f(): array { /* 略 */ }
```

Array shape

```
/** @return (array{has_value: false}
           |array{has_value: true, value: int}) */
function f(): array { /* 略 */ }

$x = f();
if ($x['has_value']) {
    // PHPStan は、$x['value'] が int であることを認識できる
    echo $x['value'];
}
```

Object shape 編

Object shape

```
/** @var object{has_value: false} $none */  
$none = (object)[  
    'has_value' => false,  
];  
/** @var object{has_value: true, value: int} $some */  
$some = (object)[  
    'has_value' => true,  
    'value' => 42,  
];
```

Object shape

```
/** @return (object{has_value: false}
           |object{has_value: true, value: int}) */
function f(): stdClass { /* 略 */; }

$x = f();
if ($x->has_value) {
    // 型が絞りこまれない！
    echo $x->value;
}
```


Type assertion 編

Type assertion

```
/** @phpstan-assert-if-true int $a */  
/** @phpstan-assert-if-false !int $a */  
function check_is_int(mixed $a): bool {  
    return is_int($a);  
}  
  
$a = hogehoge();  
if (check_is_int($a)) {  
    PHPStan\dumpType($a);  
    // => int  
}
```

Type assertion

```
abstract class OptionalInt {  
    /**  
     * @phpstan-assert-if-true OptionalIntSome $this  
     * @phpstan-assert-if-false OptionalIntNone $this  
     */  
    public function hasValue(): bool {  
        return $this instanceof OptionalIntSome;  
    }  
}  
class OptionalIntNone extends OptionalInt {}  
class OptionalIntSome extends OptionalInt {  
    public function __construct(public int $value) {}  
}
```

Type assertion

```
$x = f();  
if ($x->hasValue()) {  
    echo $x->value;  
}
```

Type assertion

```
abstract class OptionalInt {  
    /**  
     * @phpstan-assert-if-true OptionalIntSome $this  
     * @phpstan-assert-if-false OptionalIntNone $this  
     */  
    public function hasValue(): bool {  
        return $this instanceof OptionalIntSome;  
    }  
}  
class OptionalIntNone extends OptionalInt {}  
class OptionalIntSome extends OptionalInt {  
    public function __construct(public int $value) {}  
}
```

ここまでのまとめ

Array shape ではうまく動く

Object shape では動かない

Type assertion では動くが課題アリ

Object shape でも型を絞り込みたい！

PHPStan 魔改造編

PHPStan 魔改造

```
if ($x['has_value']) {  
    // $x['has_value'] の型が truthy に確定したことで、  
    // $x に HasOffsetValueType('has_value', truthy)  
    // という型が付く  
    echo $x['value'];  
}
```


PHPStan 魔改造

```
if ($x['has_value']) {  
    // $x['has_value'] の型が truthy に確定したことで、  
    // $x に HasOffsetValueType('has_value', truthy)  
    // という型が付く  
    echo $x['value'];  
}
```

```
// (array{has_value: false}  
//   |array{has_value: true, value: int})  
// と  
// HasOffsetValueType('has_value', truthy)  
// とを合成する
```

HasOffsetValueType を付ける

HasOffsetValueType と array shape を合成する

HasPropertyValueType を付ける

HasPropertyValueType と object shape を合成する

PHPStan と array shape を使った ADT もどき
Object shape での型の絞り込みは未実装
現在鋭意実装中

3/7-9: PHPerKaigi 2024

3/15-16: Ya8 2024

4/13: PHP カンファレンス小田原 2024