

詳説 「参照」 PHP の参照を完全に理解する

nsfisis (いまむら)

PHPerKaigi 2023

自己紹介

nsfisis (いまむら)

@ デジタルサーカス株式会社

アジェンダ

1. 参照の不思議クイズ
2. PHP 処理系のソースを読む
 - (a) zval と zend_reference
 - (b) 参照代入
3. クイズの解説
4. まとめ

参照の不思議クイズ

参照の不思議クイズ : Q1

```
$x = 1;  
$y =& $x;  
$y = 42;  
echo "x = $x\n";  
// => ???  
echo "y = $y\n";  
// => ???
```

参照の不思議クイズ : Q1

```
$x = 1;  
$y =& $x;  
$y = 42;  
echo "x = $x\n";  
// => 42  
echo "y = $y\n";  
// => 42
```

参照の不思議クイズ : Q2

```
$x = 1;  
$y =& $x;  
$z = $y;  
$z = 42;  
echo "x = $x\n";  
// => ???  
echo "y = $y\n";  
// => ???  
echo "z = $z\n";  
// => ???
```

参照の不思議クイズ : Q2

```
$x = 1;  
$y =& $x;  
$z = $y;  
$z = 42;  
echo "x = $x\n";  
// => 1  
echo "y = $y\n";  
// => 1  
echo "z = $z\n";  
// => 42
```


参照の不思議クイズ : Q3

```
$xs = [1, 2];  
$x =& $xs[0];  
$x = 42;  
echo "x = $x\n";  
// => ???  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [???, ???]
```

参照の不思議クイズ : Q3

```
$xs = [1, 2];  
$x =& $xs[0];  
$x = 42;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]
```

参照の不思議クイズ : Q4

```
$xs = [1, 2];  
$x =& $xs[0];  
$ys = $xs;  
$x = 42;  
$ys[1] = 3;  
echo "x = $x\n";  
// => ???  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [???, ???]  
echo "ys = [$ys[0], $ys[1]]\n";  
// => [???, ???]
```

参照の不思議クイズ : Q4

```
$xs = [1, 2];  
$x =& $xs[0];  
$ys = $xs;  
$x = 42;  
$ys[1] = 3;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]  
echo "ys = [$ys[0], $ys[1]]\n";  
// => [42, 3]
```

PHP 処理系のソースを読む

はじめに

- PHP v8.2.3 ([GitHub](#))
- ソースコードは発表向けに改変しています
 - 本スライドに掲載したコード片には、以下のライセンスが適用されます。
 - <https://github.com/php/php-src/blob/php-8.2.3/LICENSE>
- C 言語としては不正確な説明を話したり載せたりすることがあります

zval と zend_reference

zval とは

PHP の「値」全般

- 整数 (0、42、57)
- 浮動小数点数 (3.14159265)
- 文字列 ("Hello, World!")
- 配列 ([1, 2, 3])
- クラス (new \Exception)

など

zval の定義

```
struct zval {  
    zend_value value;      /* 値本体 */  
    uint32_t   type_info; /* 型情報 */  
  
    union {  
        /* その他メタデータ (省略) */  
    } u2;  
};
```

Zend/zend_types.h#L315-L340

zend_value の定義

```
union zend_value {  
    zend_long        lval;        /* 整数 */  
    double           dval;        /* 浮動小数点数 */  
    zend_string      *str;        /* 文字列 */  
    zend_array       *arr;        /* 配列 */  
    zend_object      *obj;        /* オブジェクト */  
    zend_resource     *res;        /* リソース */  
    zend_reference    *ref;        /* 参照 */  
    /* (略) */  
};
```

Zend/zend_types.h#L295-L313

zend_value の定義

```
union zend_value {  
    zend_long        lval;        /* 整数 */  
    double           dval;        /* 浮動小数点数 */  
    zend_string      *str;        /* 文字列 */  
    zend_array       *arr;        /* 配列 */  
    zend_object      *obj;        /* オブジェクト */  
    zend_resource     *res;        /* リソース */  
    zend_reference    *ref;        /* 参照 */  
    /* (略) */  
};
```

どれが入っているのかをどう区別する？

zval の定義

```
struct zval {  
    zend_value value;      /* 値本体 */  
    uint32_t   type_info; /* 型情報 */  
  
    union {  
        /* その他メタデータ (省略) */  
    } u2;  
};
```

type_info が型情報を保持している

この値を見て、zend_value に何が入っているかを区別する

いわゆる "tagged-union"

PHP の型

```
#define IS_UNDEF          0 /* 不明、未初期化 */
#define IS_NULL           1 /* null */
#define IS_FALSE          2 /* false */
#define IS_TRUE           3 /* true */
#define IS_LONG           4 /* 整数 */
#define IS_DOUBLE          5 /* 浮動小数点数 */
#define IS_STRING          6 /* 文字列 */
#define IS_ARRAY           7 /* 配列 */
#define IS_OBJECT          8 /* オブジェクト */
#define IS_RESOURCE        9 /* リソース */
#define IS_REFERENCE      10 /* 参照 */
```

参照は `IS_REFERENCE`

内部的には、独立した型として実装されている

PHP の型

```
union zend_value {  
    zend_long        lval;        /* 整数 */  
    double            dval;        /* 浮動小数点数 */  
    zend_string       *str;        /* 文字列 */  
    zend_array        *arr;        /* 配列 */  
    zend_object       *obj;        /* オブジェクト */  
    zend_resource      *res;        /* リソース */  
    zend_reference     *ref;        /* 参照 */  
    /* (略) */  
};
```

zend_reference はどんなデータ構造か？

zend_reference の定義

```
struct zend_reference {  
    uint32_t    refcount; /* 参照カウント */  
    uint32_t    type_info; /* (説明略) */  
    zval        val;      /* 指している値 */  
    zend_property_info_source_list sources; /* (説明略) */  
};
```

[Zend/zend_types.h#L537-L541](#)

参照カウント : 同じ値への参照がどれだけあるか？

具体例

```
$x = 1;
```

\$x: zval

type_info: IS_LONG

value: zend_value

1

具体例

```
$x = 1;  
$y =& $x;
```

\$x: zval

type_info: ???

value: zend_value

???

\$y: zval

type_info: ???

value: zend_value

???

参照代入

zend_assign_to_variable_reference

```
void zend_assign_to_variable_reference(  
    zval *lhs, zval *rhs  
) {  
    ZVAL_NEW_REF(rhs);  
  
    rhs->value->refcount++;  
  
    lhs->value = rhs->value;  
    lhs->type_info = IS_REFERENCE;  
}
```

\$lhs =& \$rhs;

\$lhs= 左边

\$rhs= 右边

Zend/zend_execute.c#L533-L557

ZVAL_NEW_REF

```
ZVAL_NEW_REF(rhs)
    zend_reference *ref = new zend_reference();
    ref->refcount = 1;
    /* rhs の中に入っている値を ref にコピー */
    ZVAL_COPY_VALUE(&ref->val, rhs);
    rhs->value = ref;
    rhs->type_info = IS_REFERENCE;
```

Zend/zend_types.h#L1077-L1086

rhs の中身を参照でラップする

zend_assign_to_variable_reference

```
void zend_assign_to_variable_reference(  
    zval *lhs, zval *rhs  
) {  
    ZVAL_NEW_REF(rhs);  
  
    rhs->value->refcount++;  
  
    lhs->value = rhs->value;  
    lhs->type_info = IS_REFERENCE;  
}
```

具体例

```
$rhs = 1;  
$lhs =& $rhs;
```

```
ZVAL_NEW_REF(rhs);
```

```
rhs->value->refcount++;
```

```
lhs->value = rhs->value;  
lhs->type_info = IS_REFERENCE;
```

\$rhs: zval

type_info: IS_LONG

value: zend_value

1

具体例

```
$rhs = 1;  
$lhs =& $rhs;
```

```
/* ZVAL_NEW_REF */  
zend_reference *ref =  
    new zend_reference();  
ref->refcount = 1;  
ZVAL_COPY_VALUE(&ref->val, rhs);  
rhs->value = ref;  
rhs->type_info = IS_REFERENCE;
```

\$rhs: zval

type_info: IS_LONG
value: zend_value

1

具体例

```
$rhs = 1;  
$lhs =& $rhs;
```

```
/* ZVAL_NEW_REF */  
zend_reference *ref =  
    new zend_reference();  
ref->refcount = 1;  
ZVAL_COPY_VALUE(&ref->val, rhs);  
rhs->value = ref;  
rhs->type_info = IS_REFERENCE;
```

\$rhs: zval

type_info: IS_LONG
value: zend_value

1

zend_reference

refcount: 1
val: zval

<uninitialized>

具体例

```
$rhs = 1;  
$lhs =& $rhs;
```

```
/* ZVAL_NEW_REF */  
zend_reference *ref =  
    new zend_reference();  
ref->refcount = 1;  
ZVAL_COPY_VALUE(&ref->val, rhs);  
rhs->value = ref;  
rhs->type_info = IS_REFERENCE;
```

\$rhs: zval

type_info: IS_LONG
value: zend_value

1

zend_reference

refcount: 1
val: zval

type_info: IS_LONG
value: zend_value

1

具体例

```
$rhs = 1;  
$lhs =& $rhs;
```

```
/* ZVAL_NEW_REF */  
zend_reference *ref =  
    new zend_reference();  
ref->refcount = 1;  
ZVAL_COPY_VALUE(&ref->val, rhs);  
rhs->value = ref;  
rhs->type_info = IS_REFERENCE;
```

\$rhs: zval

type_info: IS_REFERENCE
value: zend_value



zend_reference

refcount: 1
val: zval

type_info: IS_LONG
value: zend_value

1

具体例

```
$rhs = 1;  
$lhs =& $rhs;
```

```
ZVAL_NEW_REF(rhs);
```

```
rhs->value->refcount++;
```

```
lhs->value = rhs->value;  
lhs->type_info = IS_REFERENCE;
```

\$rhs: zval

type_info: IS_REFERENCE
value: zend_value



zend_reference

refcount: 1
val: zval

type_info: IS_LONG
value: zend_value

1

具体例

```
$rhs = 1;  
$lhs =& $rhs;
```

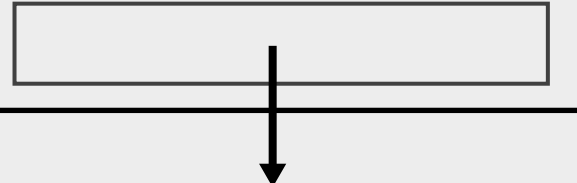
```
ZVAL_NEW_REF(rhs);
```

```
rhs->value->refcount++;
```

```
lhs->value = rhs->value;  
lhs->type_info = IS_REFERENCE;
```

\$rhs: zval

type_info: IS_REFERENCE
value: zend_value

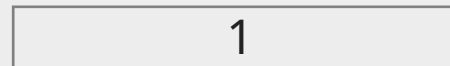


```
graph TD; A[ ] --> B[ ];
```

zend_reference

refcount: 2
val: zval

type_info: IS_LONG
value: zend_value



```
graph TD; A[1];
```

具体例

```
$rhs = 1;  
$lhs =& $rhs;
```

```
ZVAL_NEW_REF(r
```

```
rhs->value->refcount++;
```

```
lhs->value = rhs->value;  
lhs->type_info = IS_REFERENCE;
```

\$lhs: zval

type_info: IS_REFERENCE
value: zend_value

\$rhs: zval

type_info: IS_REFERENCE
value: zend_value

zend_reference

refcount: 2
val: zval

type_info: IS_LONG
value: zend_value

1

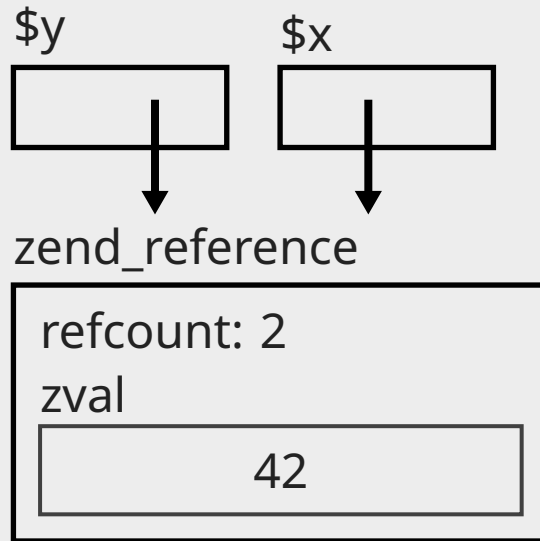
クイズの解説

クイズの解説 : Q1

```
$x = 1;  
$y =& $x;  
$y = 42;  
echo "x = $x\n";  
// => 42  
echo "y = $y\n";  
// => 42
```

クイズの解説 : Q1

```
$x = 1;  
$y =& $x;  
$y = 42;  
echo "x = $x\n";  
// => 42  
echo "y = $y\n";  
// => 42
```

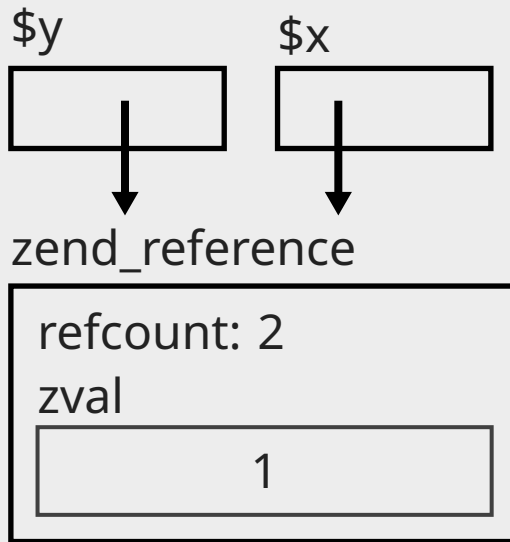


クイズの解説 : Q2

```
$x = 1;  
$y =& $x;  
$z = $y;  
$z = 42;  
echo "x = $x\n";  
// => 1  
echo "y = $y\n";  
// => 1  
echo "z = $z\n";  
// => 42
```

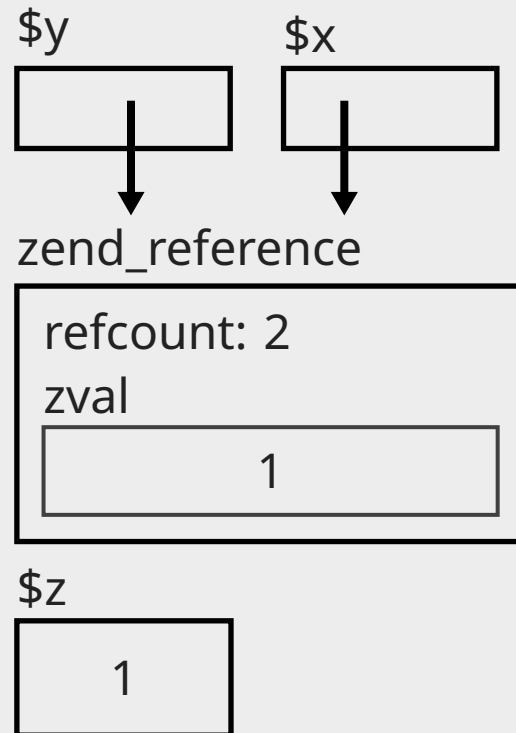
クイズの解説 : Q2

```
$x = 1;  
$y =& $x;  
$z = $y;  
$z = 42;  
echo "x = $x\n";  
// => 1  
echo "y = $y\n";  
// => 1  
echo "z = $z\n";  
// => 42
```



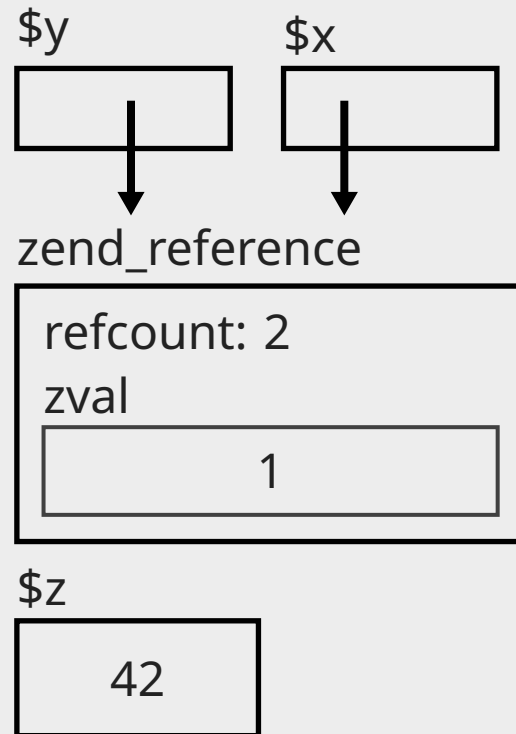
クイズの解説 : Q2

```
$x = 1;  
$y =& $x;  
$z = $y;  
$z = 42;  
echo "x = $x\n";  
// => 1  
echo "y = $y\n";  
// => 1  
echo "z = $z\n";  
// => 42
```



クイズの解説 : Q2

```
$x = 1;  
$y =& $x;  
$z = $y;  
$z = 42;  
echo "x = $x\n";  
// => 1  
echo "y = $y\n";  
// => 1  
echo "z = $z\n";  
// => 42
```

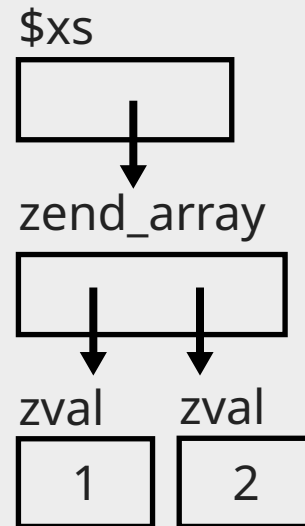


クイズの解説 : Q3

```
$xs = [1, 2];  
$x =& $xs[0];  
$x = 42;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]
```

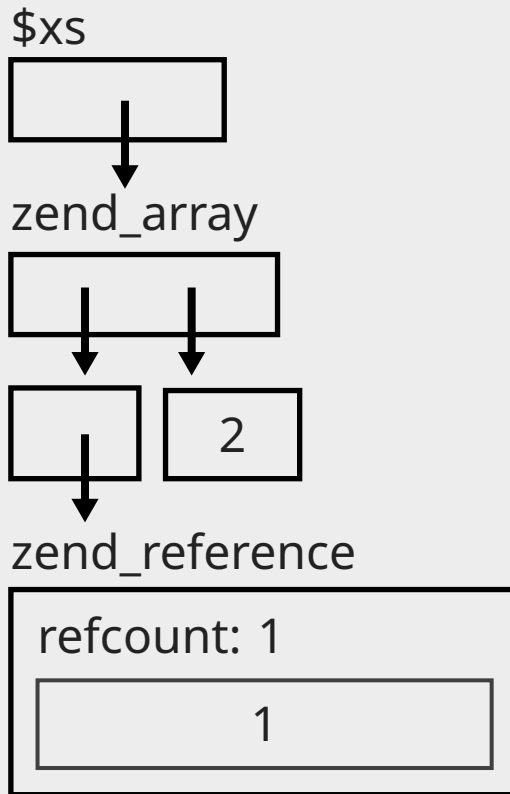
クイズの解説 : Q3

```
$xs = [1, 2];  
$x =& $xs[0];  
$x = 42;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]
```



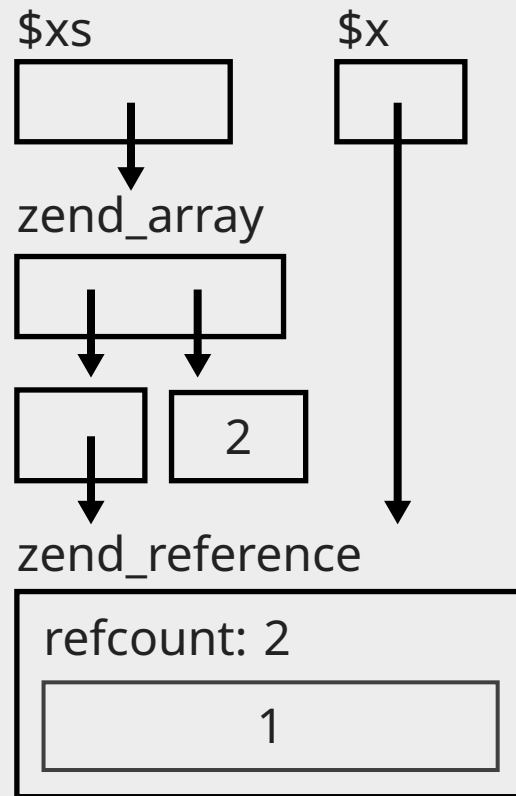
クイズの解説 : Q3

```
$xs = [1, 2];  
$x =& $xs[0];  
$x = 42;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]
```



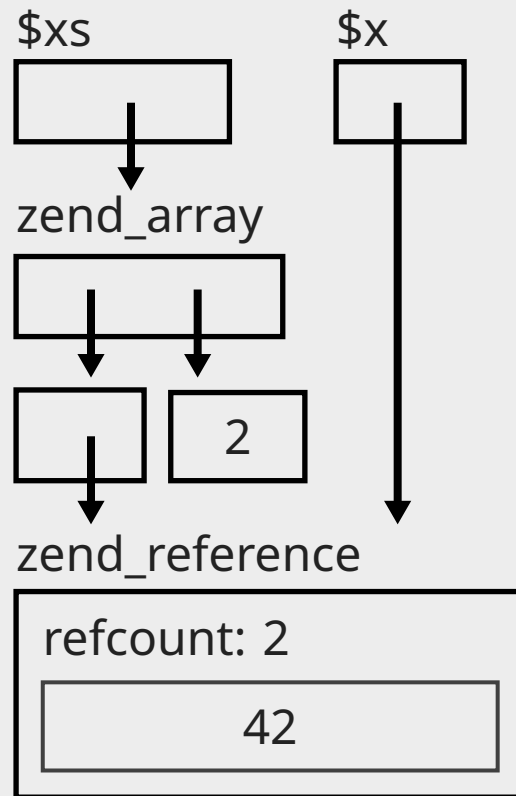
クイズの解説 : Q3

```
$xs = [1, 2];  
$x =& $xs[0];  
$x = 42;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]
```



クイズの解説 : Q3

```
$xs = [1, 2];  
$x =& $xs[0];  
$x = 42;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]
```

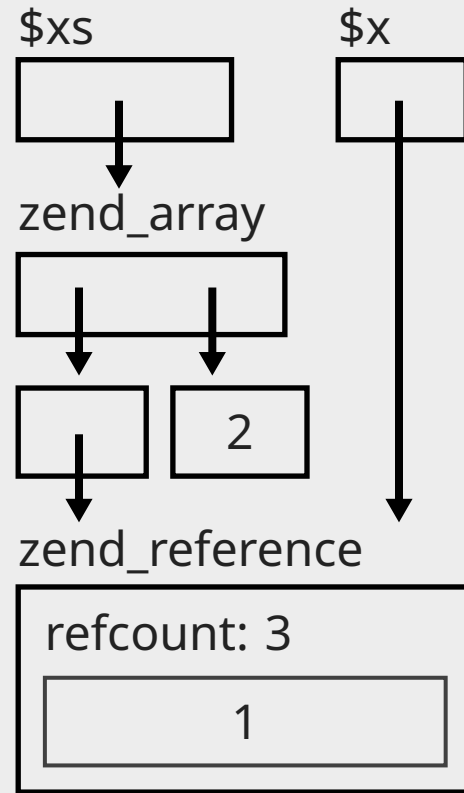


クイズの解説 : Q4

```
$xs = [1, 2];  
$x =& $xs[0];  
$ys = $xs;  
$x = 42;  
$ys[1] = 3;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]  
echo "ys = [$ys[0], $ys[1]]\n";  
// => [42, 3]
```

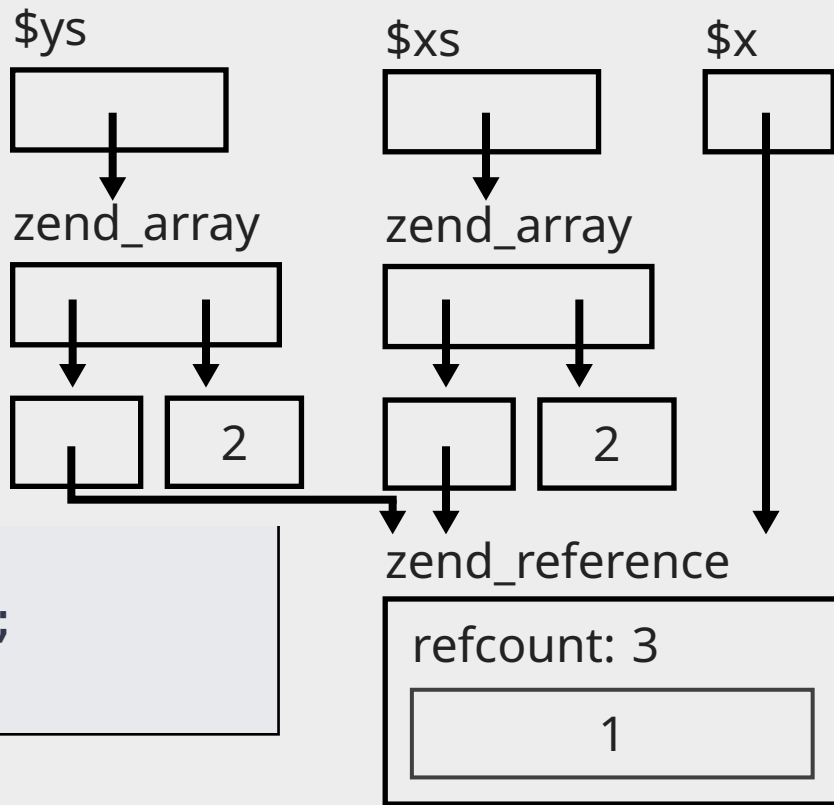
クイズの解説 : Q4

```
$xs = [1, 2];  
$x =& $xs[0];  
$ys = $xs;  
$x = 42;  
$ys[1] = 3;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]  
echo "ys = [$ys[0], $ys[1]]\n";  
// => [42, 3]
```



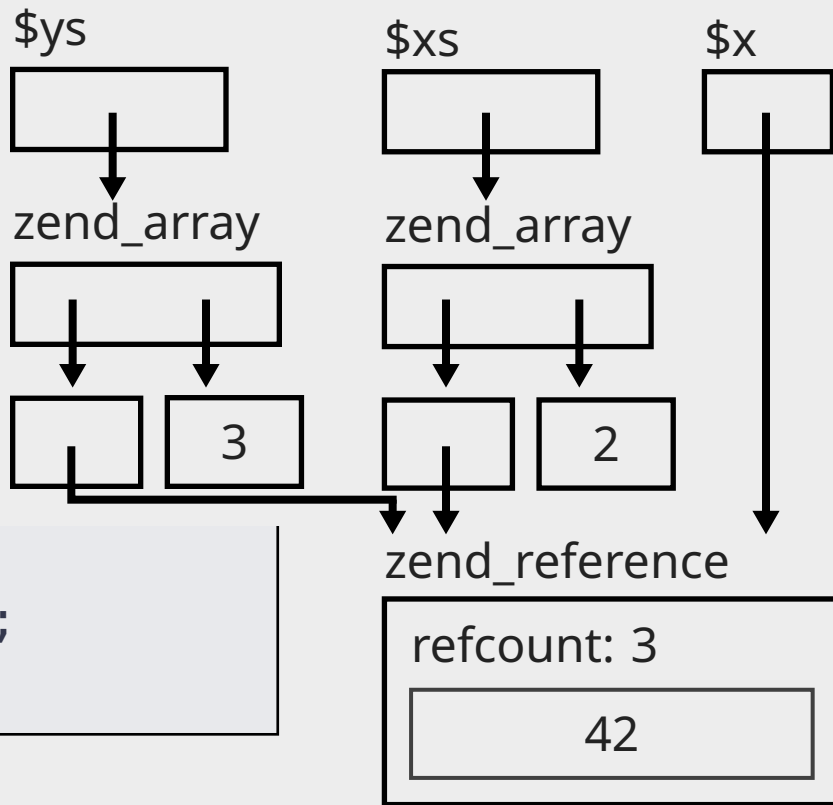
クイズの解説 : Q4

```
$xs = [1, 2];  
$x =& $xs[0];  
$ys = $xs;  
$x = 42;  
$ys[1] = 3;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]  
echo "ys = [$ys[0], $ys[1]]\n";  
// => [42, 3]
```



クイズの解説 : Q4

```
$xs = [1, 2];  
$x =& $xs[0];  
$ys = $xs;  
$x = 42;  
$ys[1] = 3;  
echo "x = $x\n";  
// => 42  
echo "xs = [$xs[0], $xs[1]]\n";  
// => [42, 2]  
echo "ys = [$ys[0], $ys[1]]\n";  
// => [42, 3]
```



まとめ

- C が読めると世界が広がる
 - PHP、Apache httpd、MySQL 等
- PHP の処理系を気軽に読もう

おまけ

入り切らなかったクイズ

参照の不思議クイズ : Q5

```
$g = 1;
function f(&$x) {
    $x =& $GLOBALS['g'];
}
$y = 0;
f($y);
$y = 42;
echo "y = $y", PHP_EOL;
// => ???
echo "g = $g", PHP_EOL;
// => ???
```

参照の不思議クイズ : Q5

```
$g = 1;
function f(&$x) {
    $x =& $GLOBALS['g'];
}
$y = 0;
f($y);
$y = 42;
echo "y = $y", PHP_EOL;
// => 42
echo "g = $g", PHP_EOL;
// => 1
```

参照の不思議クイズ : Q6

```
class C {  
    public int $x = 1;  
}  
$c = new C();  
$y =& $c->x;  
$y = 'PHPerKaigi';  
// => ???
```

参照の不思議クイズ : Q6

```
class C {  
    public int $x = 1;  
}  
$c = new C();  
$y =& $c->x;  
$y = 'PHPerKaigi';  
// => Fatal error: TypeError
```

入り切らなかったクイズの解説

参照の不思議クイズ : Q5

- 参照は (C における) ポインタでないことを示す例
- `$x` への代入で、`$x` が `$g` と同じものを指すようになる

参照の不思議クイズ：Q6

- プロパティの参照を作ると、本編で説明を省いた `zend_reference` の `sources` に情報が格納され、型チェックがおこなわれる
- 参照を使って `typed property` の型チェックを潜り抜けられないようにするための仕様

曖昧にごまかしていた説明の補足

「A が B を参照する」という言い回しについて

- `$x =& $y` と書いたとき、`$x` と `$y` は同じものを指している (本編の図を参照のこと)
- `$x` が `$y` を指しているわけでも、`$y` が `$x` を指しているわけでもない
- `$x =& $y` と `$y =& $x` はまったく同じ効果を持ち、区別できない
- しかし、本編では「A が B を参照する」や「A が B を指す」という言い方をしている。このあたりの解説を入れる時間がなかった

ポインタの話

- C 言語レベルで、構造体の中に値が埋め込まれているのか、ポインタ経由で間接参照しているのかは、本来厳密に区別する必要がある
- この発表は C 言語の説明を極力排したので、ポインタの解説が入れられなかった
- そのため、ポインタかどうかの区別はごまかし、図では矢印を使っている
- 本当なら矢印が何を意味するのかを説明する必要があるが、入り切らなかった
- 知りたいかたはオリジナルのソースをご覧ください