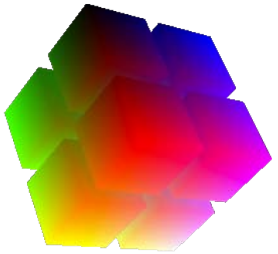
バイナリ処理に SplFixedArray を使った感想

“よや” <yoya@awm.jp>

2015/10/09

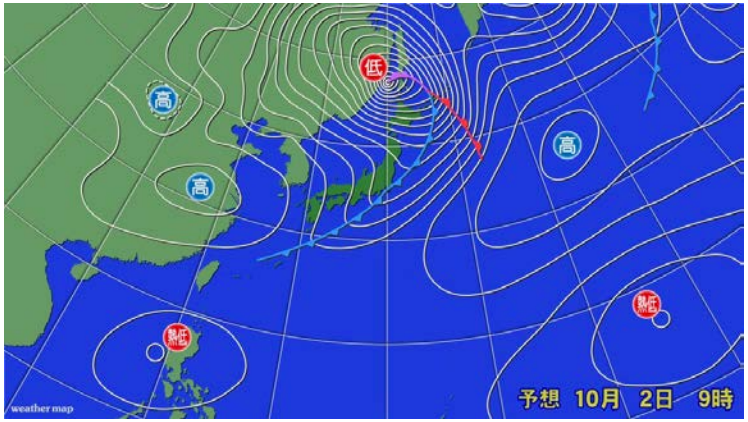
自己紹介



- よやのプロフィール
  - https://osdn.jp/users/yoya/
- PHPでバイナリ編集するのが趣味です
  - https://github.com/yoya/IO_Bit (Bit単位編集)
  - https://github.com/yoya/IO_MIDI (MIDI 編集)
- PHP 拡張も簡単なのなら作れます
  - https://osdn.jp/projects/swfed/ (SWF Editor)
- (蛇足) ImageMagick のストーキングしてます
  - http://d.hatena.ne.jp/yoya/searchdiary?word=ImageMagick

phpcon2015 未参加の理由

- 熊本旅行してました ＞＜;



爆弾低気圧
楽しい！

おはなし目次

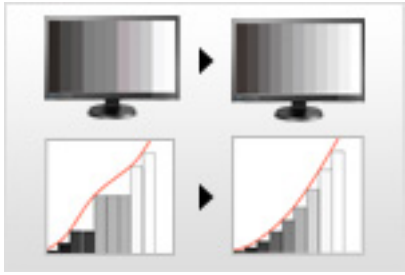
- SplFixedArray とは？
- 使おうとした経緯
- 使ってみた
- メリット
- デメリット
- 注意点
- 改善案

SplFixedArray とは？

- PHP5.3 で導入された新しい配列クラス
  - PHP5.2 はメンテ終わったので使って良さそう(?)
- 従来の Array と何が違うのか
  - 数字でしかアクセス出来ない
    - 連想配列機能をとっぱらった (文字列をキーに出来ない)
  - 配列のサイズ(要素数)が固定 (あとで増やせない)
- メリット
  - メモリを節約できる
    - メモリ使用量が半分弱になる
- デメリット
  - いろいろ (今日のネタ)

使おうとした経緯 (1/2)

- JPEG の ICC カラープロファイルをバイナリを分解していた
  - https://github.com/yoya/IO_ICC
- カラーマネジメントはここ参考
  - http://w3.kcua.ac.jp/~fujiwara/infosci/colorman/color_manage.html
  - 要するにデジカメやモニタの機種によって、実際の色とRGB値の対応が違うのを補正する仕組み



http://www.eizo.co.jp/eizolibrary/index4.html

使おうとした経緯 (2/2)

- ICC プロファイルは補正データで BYTE 配列がいっぱい
  - PHP の array はメモリを沢山使うので配列のサイズが大きいと大変
- メモリ使いすぎて溢れた！
  - array の代わりに SplFixedArray を使うとメモリ節約できるらしいので試してみる

使ってみた (1/2)

- これだけでメモリが半分に？

```php
<?php
$init_mem = memory_get_usage();
$arr = array();
for ($i = 0 ; $i < 1024 ; $i++) {
    $arr[] = 255;
}
echo memory_get_usage() - $init_mem.PHP_EOL;
?>

実行結果: 148024
```

```php
<?php
$init_mem = memory_get_usage();
$arr = new SplFixedarray(1024);
for ($i = 0 ; $i < 1024 ; $i++) {
    $arr[$i] = 255;
}
echo memory_get_usage() - $init_mem.PHP_EOL;
?>

実行結果: 57968
```

- 半分以下！ 少し物足りないけど、まぁ凄い
- array を片っ端から new SplFixedArray に書き換えた

使ってみた (slice が無い)

- array_slice が使えない

```
<?php
$arr = new SplFixedarray(10);
var_dump(array_slice($arr, 2, 3));

?>
// 実行結果
PHP Warning:  array_slice() expects parameter 1 to be array, obj\
ect given in /Users/yoya/prog/php/arr_test_slice.php on line 3
NULL
```

- (array) で誤魔化すか自分で実装する

```
<?php
$arr = new SplFixedarray(10);
//var_dump(array_slice($arr, 2, 3));
var_dump(array_slice((array)$arr, 2, 3));

?>
// 実行結果
array(3) {
  [0]=>
```

使ってみた (join が無い)

- join (implode)が使えない

```php
<?php
$arr = new SplFixedarray(10);
var_dump(join("", $arr));

?>
// 実行結果
PHP Warning:  join(): Invalid arguments passed in /Users/yoya/pr\
og/php/arr_test_join.php on line 3
NULL
```

- (array) で誤魔化すか自分で実装する

```php
<?php
$arr = new SplFixedarray(10);
// var_dump(join("", $arr));
var_dump(join("", (array)$arr));

?>
// 実行結果
string(0) ""
```

使ってみた (shuffle が無い)

- shuffle が使えない

```php
<?php
$arr = new SplFixedarray(10);
shuffle($arr);

?>
// 実行結果
PHP Warning:  shuffle() expects parameter 1 to be array, object \
given in /Users/yoya/prog/php/arr_test_shuffle.php on line 3
```

- (array) で誤魔化すか自分で実装する

```php
<?php
$arr = new SplFixedarray(10);
$arr2 = (array) $arr;
shuffle($arr2);

?>
// 実行結果
```

(array) で誤魔化さず実装した

- https://github.com/yoya/IO_ICC/blob/master/IO/ICC/FixedArray.php

```php
<?php

class IO_ICC_FixedArray extends SplFixedArray {
    // ref) https://github.com/devi/Salt/blob/master/FieldElement.php
    function slice($offset, $length) {
        $origsize = $this->getSize();
        if ($offset < 0) {
            $offset += $origsize;
        }
        if (($length === 0) || ($origsize  < $offset + $length)) {
            $length =  $origsize - $offset;
        }
        $slice = new IO_ICC_FixedArray($length);
        $j = $offset;
        for ($i = 0 ; $i < $length ; ++$i) {
            $slice[$i] = $this->offsetGet($j);
            $j++;
        }
        return $slice;
    }
    function join($glue) {
        $arr = $this->toArray();
        return join($glue, $arr);
    }
    function shuffle() {
        $size = $this->getSize();
        $newArr = new IO_ICC_FixedArray($size);
        $arr = (array) $this;
        shuffle($arr);
        for ($i = 0 ; $i < $size ; $i++) {
            $this->offsetSet($i, $arr[$i]);
        }
    }
}
```

is_array に引っかからない

```php
<?php
$arr1 = array();
$arr2 = new SplFixedarray(10);
var_dump(is_array($arr1));  // => true
var_dump(is_object($arr1)); // => false
var_dump(get_class($arr1)); // => (Warning)
var_dump(is_array($arr2));  // => false
var_dump(is_object($arr2)); // => true
var_dump(get_class($arr2)); // => "SplFixedArray"
?>
// 実行結果
bool(true)
bool(false)
PHP Warning:  get_class() expects parameter 1 to be object, arr\
ay given in /Users/yoya/prog/php/arr_test_isarray.php on line 6
bool(false)
bool(false)
bool(true)
string(13) "SplFixedArray"
```

ツリーを辿って改変

- よくやりますよね！

```php
<?php
$arr1 = [[1, 2, 3], [7, 8, 9]]; // array

adding_walktree($arr1, 1);
print_r($arr1);

function adding_walktree(&$arr, $delta) {
    if (is_array($arr)) {
        foreach ($arr as $idx => $dummy) {
            adding_walktree($arr[$idx], $delta);
        }
    } else {
        $arr += $delta;
    }
}
```

```
Array
(
    [0] => Array
        (
            [0] => 2
            [1] => 3
            [2] => 4
        )

    [1] => Array
        (
            [0] => 8
            [1] => 9
            [2] => 10
        )

)
```

ツリーを辿って改変 (SplTypedArray で挑戦)

- こんな感じに実装したい

```php
<?php
$arr2 = new SplFixedArray(2);
$arr2[0] = new SplFixedArray(3);
$arr2[1] = new SplFixedArray(3);
$arr2[0][0] = 1 ; $arr2[0][1] = 2 ; $arr2[0][2] = 3;
$arr2[1][0] = 7 ; $arr2[1][1] = 8 ; $arr2[1][2] = 9;

adding_walk($arr2, 1);
print_r($arr2);

function adding_walk(&$arr, $delta) {
    if (is_object($arr)) {
        foreach ($arr as $idx => $dummy) {
            adding_walk($arr[$idx], $delta);
        }
    } else {
        $arr += $delta;
    }
}
```

ツリーを辿って改変したかった

- 配列の要素をリファレンスで渡せない

```
Pro:php $ php arr_test_rec2.php
PHP Notice:  Indirect modification of overloaded element of SplFixedArray has no
effect in /Users/yoya/prog/php/arr_test_rec2.php on line 14

Notice: Indirect modification of overloaded element of SplFixedArray has no effec
t in /Users/yoya/prog/php/arr_test_rec2.php on line 14
PHP Notice:  Indirect modification of overloaded element of SplFixedArray has no
effect in /Users/yoya/prog/php/arr_test_rec2.php on line 14

Notice: Indirect modification of overloaded element of SplFixedArray has no effec
t in /Users/yoya/prog/php/arr_test_rec2.php on line 14
PHP Notice:  Indirect modification of overloaded element of SplFixedArray has no
effect in /Users/yoya/prog/php/arr_test_rec2.php on line 14

Notice: Indirect modification of overloaded element of SplFixedArray has no effec
t in /Users/yoya/prog/php/arr_test_rec2.php on line 14
PHP Notice:  Indirect modification of overloaded element of SplFixedArray has no
effect in /Users/yoya/prog/php/arr_test_rec2.php on line 14

Notice: Indirect modification of overloaded element of SplFixedArray has no effec
t in /Users/yoya/prog/php/arr_test_rec2.php on line 14
PHP Notice:  Indirect modification of overloaded element of SplFixedArray has no
effect in /Users/yoya/prog/php/arr_test_rec2.php on line 14

Notice: Indirect modification of overloaded element of SplFixedArray has no effec
```

メリット

- メモリが半分位まで省エネできる
- アクセスも多分早い

デメリット

- array のメソッドが使えない
  - toArray メソッドがあるので最悪、配列にする手もある。(が、それだと性能落ちるので本末転倒)
  - 今回は slice, join, shuffle を自作
- 配列要素の参照渡しが出来ない
  - 回避策見つからないので、誰か知ってたら教えて

教訓

- はじめから SplFixedArray で行くなら良いが、既存のコードの array を書き換える時には注意が必要。
- SplFixedArray でツリー構造を作る時には注意。(配列を入れ子にするとか)
  - あとで辿って書き換える処理が書けないかも。
    - (誰か良い方法を知ってたら教えてください)
  - 木の一番末端 (leaf node) のみ SplFixedArray を使うのが安全っぽい。

改善案 (1/2)

- SplFixedArray は Zval の変数コンテナが並ぶだけなので BYTE 配列で使いたい場合にメモリが巨大に。実データは 1024 バイトなのに 57968も使ってる。

```php
<?php
$init_mem = memory_get_usage();
$arr = new SplFixedarray(1024);
for ($i = 0 ; $i < 1024 ; $i++) {
    $arr[$i] = 255;
}
echo memory_get_usage() - $init_mem.PHP_EOL;
?>

実行結果: 57968
```

- SplTypedArray ください。C言語風配列
  - (PHP だと無理？)

改善案 (2/2)

- 便利メソッド欲しい
  - 無いと、ちょっと戸惑う。敷居の高さ。
  - とりあえず slice だけでもあって良いのでは

ｏ ｏ ｏ

- よく考えたら String でバイナリの容器を作って、get, set のマジックメソッドでギチギチに詰める、MyTypedArray 作れば良いか。。(今度作るかも)

- 以上です