

Intermediate PHP

Bradley Holt & Matthew Weier O'Phinney

Arrays



Photo by AJ Cann

- ⦿ Associate values to keys
- ⦿ Keys can be integers (enumerative arrays) or strings (associative arrays)
- ⦿ Values can be primitive types, objects, or other arrays (multidimensional arrays)

```
<?php
$post = array(
    'id'                => 1234,
    'title'             => 'Intermediate PHP',
    'updated'           => new DateTime(
        '2010-12-16T18:00:00-05:00'
    ),
    'draft'              => false,
    'priority'           => 0.8,
    'categories'         => array('PHP', 'BTv')
);

```

Reading, Writing, and Appending Arrays

```
<?php
$post = array(
    'id'                => 1234,
    'title'             => 'Intermediate PHP',
    // ...
);
echo $post['title']; // Intermediate PHP
```

```
<?php
$post = array(
    'id'                => 1234,
    'title'             => 'Intermediate PHP',
    // ...
);
$post['title'] = 'PHP 201';
```

```
<?php
$post = array(
    'id'                => 1234,
    'title'              => 'Intermediate PHP',
    'updated'            => new DateTime(
        '2010-12-16T18:00:00-05:00'
    ),
    'draft'              => false,
    'priority'           => 0.8,
    'categories'         => array('PHP', 'BTv')
);
$post['summary'] = 'Arrays, functions, and
objects';
```

```
<?php
$post = array(
    // ...
    'categories' => array('PHP', 'BTv')
);
$post['categories'][] = 'Programming';
print_r($post['categories']);
/*
Array
(
    [0] => PHP
    [1] => BTv
    [2] => Programming
)
*/
```

```
<?php
$post = array(
    // ...
    'categories' => array('PHP', 'BTv')
);
$post['categories'][1] = 'Burlington';
print_r($post['categories']);
/*
Array
(
    [0] => PHP
    [1] => Burlington
)
*/
```

Iterating Over Arrays

```
<?php
$post = array(
    // ...
    'categories' => array('PHP', 'BTv')
);
foreach ($post['categories'] as $v) {
    echo $v . PHP_EOL;
}
/*
PHP
BTv
*/
```

```
<?php
$post = array(
    // ...
    'categories' => array('PHP', 'BTv')
);
foreach ($post['categories'] as $k => $v) {
    echo $k . ':' . $v . PHP_EOL;
}
/*
0: PHP
1: BTv
*/
```

```
<?php
$post = array(
    'id'                => 1234,
    'title'             => 'Intermediate PHP',
    // ...
);
foreach ($post as $k => $v) {
    echo $k . ':' . $v . PHP_EOL;
}
/*
id: 1234
title: Intermediate PHP
...
*/
```

Implode and Explode

```
<?php
$post = array(
    // ...
    'categories' => array('PHP', 'BTv')
);
echo implode(', ', $post['categories']);
// PHP, BTv
```

Implode returns a string, not an array.

```
<?php
$post = array(
    // ...
    'categories' =>
        explode( ', ', 'PHP, BTV')
);
print_r($post['categories']);
/*
Array
(
    [0] => PHP
    [1] => BTV
)
*/
```

Explode returns an array.

Array Key Exists,
In Array, and Array Keys

```
<?php
$post = array(
    'id'                => 1234,
    // ...
    'categories'        => array('PHP', 'BTv')
);
if (array_key_exists('categories', $post))
{
    echo implode(', ', $post['categories']);
} else {
    echo 'Uncategorized';
}
```

```
<?php
$post = array(
    // ...
    'categories' => array('PHP', 'BTv')
);
if (in_array('PHP', $post['categories'])) {
    echo 'PHP: Hypertext Preprocessor';
}
```

```
<?php
$posts = array(
    1233 => array(/* ... */),
    1234 => array(/* ... */),
);
print_r(array_keys($posts));
/*
Array
(
    [0] => 1233
    [1] => 1234
)
*/
```

Sorting Arrays

```
<?php
$post = array(
    // ...
    'categories' => array('PHP', 'BTv')
);
sort($post['categories']);
print_r($post['categories']);
/*
Array
(
    [0] => BTv
    [1] => PHP
)
*/
```

Sorting Function Attributes

- ⦿ Some sort by value, others by key
- ⦿ Some maintain key association, others do not
- ⦿ Some sort low to high, others high to low
- ⦿ Some are case sensitive, some are not
- ⦿ See:
<http://www.php.net/manual/en/array.sorting.php>

Sorting Functions

- ⦿ array_multisort()
- ⦿ asort()
- ⦿ arsort()
- ⦿ krsort()
- ⦿ ksort()
- ⦿ natcasesort()
- ⦿ natsort()
- ⦿ rsort()
- ⦿ shuffle()
- ⦿ sort()
- ⦿ uasort()
- ⦿ uksort()
- ⦿ usort()

Stacks and Queues



Photo by Phil Cooper



Photo by Thomas W

Array Push and
Array Pop (stack)

```
<?php
$post = array(
    // ...
    'categories' => array('PHP')
);
array_push($post['categories'], 'BTv');
echo array_pop($post['categories']);
// BTv
print_r($post['categories']);
/*
Array
(
    [0] => PHP
)
*/
```

Array Unshift and
Array Pop (queue)

```
<?php
$post = array(
    // ...
    'categories' => array('BTv')
);
array_unshift($post['categories'], 'PHP');
print_r($post['categories']);
/*
Array
(
    [0] => PHP
    [1] => BTv
)
*/
echo array_pop($post['categories']);
// BTv
```

Functions

Internal Functions

```
<?php
print_r(get_defined_functions());
/*
Array
(
    [internal] => Array
        (
            [0] => zend_version
            [1] => func_num_args
            // ...
        )
    [user] => Array
        (
        )
)
*/
```

```
<?php
$functions = get_defined_functions();
echo count($functions['internal']);
// 1857
```

```
<?php
$name = 'Marcus Börger';
if (function_exists('mb_strtolower')) {
    echo mb_strtolower($name, 'UTF-8');
    // marcus börger
} else {
    echo strtolower($name);
    // marcus b?rger
}
```

User-Defined Functions

Rules

- Any valid PHP code is allowed inside functions, including other functions and class definitions
- Function names are case-insensitive
- Once defined, a function cannot be undefined

```
<?php
function nextId($posts)
{
    return max(array_keys($posts)) + 1;
}
$posts = array(
    1233 => array(/* ... */),
    1234 => array(/* ... */),
);
echo nextId($posts); // 1235
```

Type Hinting

```
<?php
function nextId(array $posts)
{
    return max(array_keys($posts)) + 1;
}
$posts = array(
    1233 => array(/* ... */),
    1234 => array(/* ... */),
);
echo nextId($posts); // 1235
echo nextId(1234);
// Argument 1 passed to nextId() must be an
array, integer given
```

Multiple Arguments

```
<?php
function isPublished(DateTime $published,
$draft)
{
    if ($draft) { return false; }
    $now = new DateTime();
    return $now >= $published;
}
$published = new DateTime
('2010-12-16T18:00:00-05:00');
$draft = false;
var_dump(isPublished($published, $draft));
// bool(true)
```

Default Arguments

```
<?php
function isPublished(DateTime $published,
$draft, $now = false)
{
    if ($draft) { return false; }
    $now = $now ? $now : new DateTime();
    return $now >= $published;
}
$published = new DateTime
('2010-12-16T18:00:00-05:00');
$draft = false;
$now = new DateTime
('2010-12-16T17:59:59-05:00');
var_dump(isPublished($published, $draft,
$now));
// bool(false)
```

Function Overloading
(not really)

```
<?php
function nextId($arg1)
{
    switch (true) {
        case is_array($arg1):
            return max(array_keys($arg1)) + 1;
        case is_int($arg1):
            return $arg1 + 1;
    }
}
$posts = array(
    1233 => array(/* ... */),
    1234 => array(/* ... */),
);
echo nextId($posts); // 1235
echo nextId(1234); // 1235
throw new InvalidArgumentException();
```

Variable Number of Arguments

```
<?php
function mostRecent()
{
    $max = false;
    foreach (func_get_args() as $arg) {
        $max = $arg > $max ? $arg : $max;
    }
    return $max;
}
$mostRecent = mostRecent(
    new DateTime('2010-12-14T18:00:00'),
    new DateTime('2010-12-16T18:00:00'),
    new DateTime('2010-12-15T18:00:00')
);
// 2010-12-16T18:00:00
```

Objects

Questions?

Thank You

Bradley Holt & Matthew Weier O'Phinney

License

Intermediate PHP is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License.

