PHP7

نویسنده:پیروز جنابی

آموزش وراهنمای ارتقا

مقدمه و پیشگفتار نویسنده

با توجه به پیشرفت روز افزون سیستم های کامپیوتری و بستر اینترنت ، استفاده نکردن از این سیستم ها غیر ممکن است .امروزه برنامه های تحت وب جایگاه بسیار بالایی در بین کاربران دارد و همه ما بطور مستقیم و یا غیر مستقیم با این برنامه ها سر کار داریم . php یکی از زبان های برنامه نویسی وب هست که قدرت بسیار زیادی دارد واز لحاظ خیلی ها از جمله خود من قوی ترین زبان برنامه نویسی تحت وب می باشد .

از این رو سعی بر آن آوردیم مرجعی بر تغییر بزرگی که در این زمینه در پیش داریم بر آوریم . این تغییر بزرگ که به طور کامل در این کتاب مورد بررسی قرار گرفته بیشتر برروی سرعت و مواردی است که بهینه سازی آن موجب راحتی کاربر می شود شما با مطالعه این مقاله می توانید برنامه هایتان را طبق نسخه جدید پی اچ پی سازگار نمایید و از خیلی از اتفاقات بزرگ جلوگیری نمایید.

ما در این مقاله پی اچ پی را از اول آموزش نمی دهیم این مقاله برای کاربران حرفه ای که مسلط به این زبان می باشد تهیه شده و برای کسانی که می خواهند این زبان برنامه نویسی را شروع کنند پیشنهاد نمی شود .

و اما اگر بخواهیم نسخه جدیدPHP که نسخه 7 می باشد را بررسی کنیم این نسخه نسبت به نسخه قبلی تغییرات زیادی داشته ولی نگران نباشید طبق اطلاعات منابع موجود وآزمایشات این شرکت کلیه برنامه های قبلی بر روی این سیستم جواب می دهد و تغییر عمده آن روی سرعت عملیاتها می باشد پس همانطور که از اسم ان که دارای عدد 7 می باشد و این عدد نماد انقلابی بزرگ در همه ی عرصه هاست انتظار می رود به زودی و با انتشار نسخه نهایی این نسخه ، این زبان برنامه نویسی انقلاب بزرگی در اینترنت بوجود آورد. در حال حاضر که این کتاب را می نویسیم نسخه بتا آن منتشر شده و هنوز نسخه نهایی آن منتشر نشده به همین دلیل ممکن است تغییرات دیگری داشته باشد که سعی می کنیم این تغییرات را در وب سایتمان WWW.PIERO.IR انتشار دهیم.

و همچنین تشکر می کنیم بابت انتخاب این مقاله و امیدواریم به بزرگی خود کاستی ها و اشتباهات ما را ببزیرید و ما را در این راستا همراهی نمایید .

این کتاب به صورت کاملا رایگان انتشار یافته است و شما می توانید آن را به صورت رایگان انتشار دهید ولی خواهشمندیم آن را تغییر ندهید و در صورت بروز هرگونه مشکل لطفا با ایمیل زیر ارتباط برقرار نمایید.

INFO@PIERO.IR

با تشکر گروه متخصصین پیرو

نویسنده و مترجم : پیروز جنابی

ویراستار : رویا زمانی

تاریخ انتشار : آذر 94

فهرست

روز اول :آشـنـایـی

درس اول :نصب php7

نصب بروی لینوکس

```
sudo apt-get install python-software-properties
sudo add-apt-repository ppa:ondrej/php-7.0
```

نسخه قبلی پی اچ پی را پاک کرده و کدهای زیر را در ترمینال اجرا می نماییم

```
sudo apt-get update
sudo apt-get purge php5-fpm
sudo apt-get install php7.0 php7.0-fpm php7.0-mysql
sudo apt-get --purge autoremove
```

ویا از کد های زیر استفاده کنید(ابونتو)

```
sudo add-apt-repository ppa:ondrej/php-7.0
sudo apt-get update
sudo apt-get install php7.0
```

ویا از کد های زیر استفاده کنید(دبیان)

```
deb http://packages.dotdeb.org <distribution> all
deb-src http://packages.dotdeb.org <distribution> all
Add the GPG key:
wget https://www.dotdeb.org/dotdeb.gpg
sudo apt-key add dotdeb.gpg
Install PHP 7:
sudo apt-get update
sudo apt-get install php7.0
```

و یا از کد های زیر استفاده کنید (cent os)

```
rpm      -Uvh      https://dl.fedoraproject.org/pub/epel/epel-release-latest-
7.noarch.rpm
rpm -Uvh https://mirror.webtatic.com/yum/el7/webtatic-release.rpm
yum install php70w
```

و همچنین برای نصب بروی مک mac os x

```
brew tap homebrew/dupes
brew tap homebrew/versions
brew tap homebrew/homebrew-php
brew install php70
```

جهت نصب بروی نسخه ویندوز برنامه wamp را دانلود کرده و نصب نمایید

ویا به آدرس زیر مراجعه نمایید.

http://windows.
php.net/download#php-7.0

درس دوم :اجرای یک برنامه ساده و یاد آوری

```
<?php
//کلیه برنامه ها با ساختار  <?php شروع می شوند//

// ساختار آرایه بدین گونه می باشد
$foo = array('Apple', 'Banana', 'Pear');
echo $foo[0]; // چاپ Apple
echo $foo[1]; // چاپ Banana
echo $foo[2]; // چاپ Pear
// انتساب و مقدار دهی $name[index] = value; syntax
$foo[3] = 'Kiwi';
echo $foo[3]; // will output Kiwi
// اضاغه کردن مقدار به آرایه
$foo[] = 'Chestnut';
//کلیه برنامه ها با ساختار  ?> اتمام می یابد//
?> //
```

درس سوم: نکات نوع اسکالر

پی اچ پی به شما امکان می دهد نوع دریافتی ارگمان را تعریف کنید به مثالهای زیر توجه کنید:

```
// بدون نوع
function getNextWeekday($date) { /*...*/ }
// با نوع(PHP 5):
function getNextWeekday(DateTime $date) { /*...*/ }
:
// با نوع (PHP 7):
function getNextWeekday(int $date) { /*...*/ }
```

این نکات در ابتدا فقط به کلاس ها و رابط محدود بود، اما به زودی گسترش یافت و اجازه می دهد تاآرایه قابل فراخوانی باشد. PHP 7 این امر بیشتر گسترش داده و انواع اسکالر مانند int و float، رشته و بولی را هم فراهم ساخته .

حالت چک کردن

سیستم نوع انعطاف پذیر PHP یکی از محبوب ترین ویژگی های آن است، اجازه می دهد رشته عددی مورد نظر شما مورد استفاده قرار گیرد به عنوان عدد صحیح و بالعکس. این سنت در PHP 7 ادامه دارد .ولی شما این امکان را دارید برای تعریف مانند برنامه های دیگر ، به مثال زیر توجه فرمایید.

```
float floor ( float $value )
```

هنگامی که شما در رشته عددی یا اعداد صحیح در PHP تعریف می کنید آن را به یک floatتبدیل می کند . این رفتار به سادگی به فضای کاربری توابع تعمیم داده شده است.

جدول زیر نشان می دهد چه مقدارهای می تواند به چه مقدار هایی تبدیل شود.

Type declaration	int	float	string	bool	object
int	yes	yes	yes	yes	no
float	yes	yes	yes	yes	no
string	yes	yes	yes	yes	yes
bool	yes	yes	yes	yes	no
object	yes	yes	yes	yes	no

حالت جدید

PHP 7 دارای حالت جدید "strick" است که با قرار دادن کد تعریف آن در بالای اسکریپت فعال می شود مانند:

```php
<?php
declare(strict_types=1);
function welcome(string $name) {
echo 'Hello ' . $name;
}
welcome('World'); // Prints: Hello World
```

توجه فرمایید این تعریف در بالای کد قرار گیرد.

منابع

- RFC: Scalar Type Declarations[27]
- PHP Documentation: Type Hinting[28]

درس چهارم:مقایسه ترکیبی

در php7یک نوع جدید مقایسه ای داریم <=> که در صورت استفاده حالتهای زیر را بر می گرداند.

0 اگر هر دو عبارت برابر هستند

1 اگر در سمت چپ بیشتر باشد

-1 اگر سمت راست بیشتر باشد

و این عملگر همانند strcmp عمل می کند با این تفاوت که همه چیز را مقایسه می کند به مثال های زیر توجه کنید

```php
// Integers
echo 1 <=> 1; // 0
echo 1 <=> 2; // -1
echo 2 <=> 1; // 1
// Floats
echo 1.5 <=> 1.5; // 0
echo 1.5 <=> 2.5; // -1
echo 2.5 <=> 1.5; // 1
// Strings
echo "a" <=> "a"; // 0
echo "a" <=> "aa"; // -1
echo "zz" <=> "aa"; // 1
// Arrays
echo [] <=> []; // 0
echo [1, 2, 3] <=> [1, 2, 3]; // 0
echo [1, 2, 3] <=> []; // 1
echo [1, 2, 3] <=> [1, 2, 1]; // 1
echo [1, 2, 3] <=> [1, 2, 4]; // -1
// Objects
$a = (object) ["a" => "b"];
$b = (object) ["a" => "b"];
echo $a <=> $b; // 0
$a = (object) ["a" => "b"];
$b = (object) ["a" => "c"];
echo $a <=> $b; // -1
$a = (object) ["a" => "c"];
$b = (object) ["a" => "b"];
echo $a <=> $b; // 1
// only values are compared
$a = (object) ["a" => "b"];
$b = (object) ["b" => "b"];
echo $a <=> $b; // 0
```

واین عملگر می تواند کمک زیادی در عملیات مرتب سازی انجام دهد لطفا به مثال زیر توجه کنید

```
class Spaceship {
public $name;
public $maxSpeed;
public function __construct($name, $maxSpeed) {
$this->name = $name;
$this->maxSpeed = $maxSpeed;
}
}
$spaceships = [
new Spaceship('Rebel Transport', 20),
new Spaceship('Millenium Falcon', 80),
new Spaceship('X-Wing Starfighter', 80),
new Spaceship('TIE Bomber', 60),
new Spaceship('TIE Fighter', 100),
new Spaceship('Imperial Star Destroyer', 60),
];
// Sort the spaceships by name (in ascending order)
usort($spaceships, function ($ship1, $ship2) {
return $ship1->name <=> $ship2->name;
});
echo $spaceships[0]->name; // "Imperial Star Destroyer"
// Sort the spaceships by speed (in descending order)
// Notice how we switch the position of $ship1 and $ship2
usort($spaceships, function ($ship1, $ship2) {
return $ship2->maxSpeed <=> $ship1->maxSpeed;
});
echo $spaceships[0]->name; // "TIE Fighter"
```

با این عملگر شما بسیار از پیچیدگی برنامه کم کرده اید

منابع

- RFC: Combined Comparison (Spaceship) Operator[31]
- PHP Manual: Comparison Operators[32]

درس پنجم: یونیکد ها

عدم پشتیبانی پی اچ پی از یونیکدهای بومی می تواند همه چیز را دشوار سازد .کتابخانه های مثل iconv ویا mbstring کار می کنند با رشته ها ، ولی هیچ مکانیزم ساده ای نیست برای تبدیل به یونیکدهای مختلف بدون استفاده از html و json نیست به طور مثال:

```
$char = html_entity_decode('&#x2603', 0, 'UTF-8');
$char = mb_convert_encoding('&#x2603', 'UTF-8', 'HTML-ENTITIES');
$char = json_decode('"\\u2603"');
```

ولی php7 به گونهای خاص این کار را انجام می دهد به مثال زیر توجه کنید

```
$char = "\u{2603}";
```

این قابلیت تحول عظیمی است و کارها را بسیار راحتر می کند

```
echo "\u{202E}This is backwards"; // displays: sdrawkcab si sihT

echo "\u{58}"; // "X"
echo "\u{0058}"; // "X"
```

و بهتر است از حالت مقابل استفاده کنیم :{1F427}\u در واقع باستفاده از **{} آکولاد** که قابل فهم تر باشد.

منابع

- RFC: Unicode Codepoint Escape Syntax [3] [3]

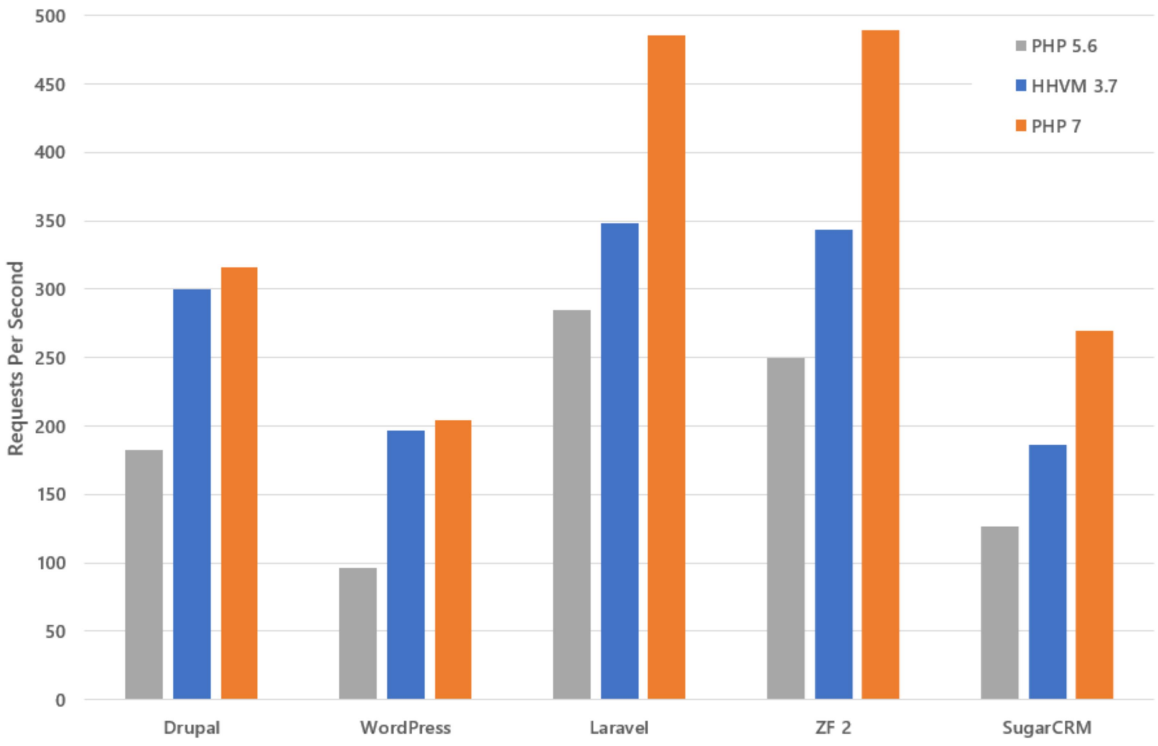
روز دوم: تغییرات زبان وارتقاء

در **php7امکانات** زیادی اضافه شده که با یادگیری آنها شما می توانید خیلی سریع تر و قوی تر برنامه نویسی کنید

درس ششم: کارایی

Php7در مقایسه با نسخه ی قبلی خود بسیار سریع تر و قوی تر می باشد به نمودار زیر توجه فرمایید.

(Zend Performance Team)



تغییرات ناسازگار

اگر شما با نسخه پیشرفته php (5.5,5,6) کار کرده باشید بیشتر کدهای شما به خوبی جواب می دهد در واقع php7تمامی کد ها و پروژه های قبلی شما را سازگار می کند وحتی شاید بهتر و پر سرعت تر ولی در اینجا بعضی از موارد را متذکر می شویم که شما در پروژه هایتان استفاده کنید برای جلوگیری از بروز مشکل در پروژه هایتان.

نویسنده و مترجم : پیروز جنابی

این موارد مستقیما ار منبع php7 که در تاریخ 7 نوامبر 2015 منتشر شده است آمده است که شما می توانید با ورود به لیک زیر آن را ببینید. https://github.com/php/php-src/blob/PHP-7.0.0/UPGRADING

تغییرات زبان

دسترسی متغیر

تجزیه چپ به راست

```
$$foo['bar']['baz']  // interpreted as ($$foo)['bar']['baz']
$foo->$bar['baz']  // interpreted as ($foo->$bar)['baz']
$foo->$bar['baz']()  // interpreted as ($foo->$bar)['baz']()
$Foo::$bar['baz']()  // interpreted as (Foo::$bar)['baz']()
```

برای اجرا در حالت قبل باید از اکولاد {} به صورت صریح و روشن استفاده کرد

```
${$foo['bar']['baz']}
$foo->{$bar['baz']}
$foo->{$bar['baz']}()
Foo::{$bar['baz']}()
```

متغیر های سراسری

برای استفاده از متغیر های سراسری باید از حالت زیر استفاده کرد:

```
global ${$foo->bar};
```

رفتار موثر بر پرانتز

پرانتزها در اطراف متغیر یا تابع از هر گونه نفوذ جلوگیری می کند. به عنوان مثال کد زیر ، که در آن نتیجه یک فراخوانی تابع است .

```
function getArray() { return [1, 2, 3]; }
$last = array_pop(getArray());
// Strict Standards: Only variables should be passed by reference
$last = array_pop((getArray()));
// Strict Standards: Only variables should be passed by reference
```

در نسخه حال strick standards هیچ توجی به بپرانتز نمی کند.

با انتساب از مرجع (by-refrence)

آرایه ها و عناصر شی که در طول برنامه ساخته می شود بوسیله مبدان ارجاع داده می شود به مثال زیر توجه فرمایید.

```
$array = [];
$array["a"] =& $array["b"];
$array["b"] = 1;
var_dump($array);
```

خروجی : ["a"=1, "b"= 1], در حالی که در نسخه قبلی به این صورت بود ["b"=1, "a"= 1];

مرجع

- https://wiki.php.net/rfc/uniform_variable_syntax
- https://wiki.php.net/rfc/abstract_syntax_tree

LIST()

سفارش تخصیص متغیر

list () دیگر متغیرها را در جهت معکوس اختصاص دهید. برای مثال

```
list($array[], $array[], $array[]) = [1, 2, 3];
var_dump($array);
```

خروجی [1, 2, 3] == $array در نسخه قبل .[3, 2, 1]

نکته : توجه فرمایید فقط ترتیب انتساب تغییر کرده است ولی روش انتساب به همان گونه است.

```
list($a, $b, $c) = [1, 2, 3];
// $a = 1; $b = 2; $c = 3;
```

انتساب لیست های خالی

انتساب به لیستهای خالی دیگر به صورت های زیر معتبر نیست

```
list() = $a;
list(,,) = $a;
list($x, list(), $y) = $a;
```

لیستها دیگر رشته های غیر متراکم را پشتیبانی نمی کند

```
$string = "xy";
list($x, $y) = $string;
```

در حال حاضر درون متغیر x و y هیچ مقداری قرار نمی دهد در صورتی که در نسخه قبلی $x=”x” , $y=”y”.

علاوه بر این لیست های تضمین شده اند برای کار با آرایه ها و اشیا

```
list($a, $b) = (object) new ArrayObject([0, 1]);
```

در این نسخه $a==0,$b==1 در حالی که در نسخه قبلی مقادیر خالی را بر می گرداند

منابع:

https://wiki.php.net/rfc/abstract_syntax_tree#changes_to_list
https://wiki.php.net/rfc/fix_list_behavior_inconsistency

FOREACH()

تعامل با اشاره گر آرایه داخلی

تکرار با حلقه forech()دیگر تاثیری بر آرایه داخلی ندارد به مثال زیر توجه فرمایید:

```
$array = [0, 1, 2];
foreach ($array as &$val) {
var_dump(current($array));
}
```

خروجی در حال حاضر int(0) ولی در نسخه قبلی int(1) و int(2) و bool(false).

تکرار آرایه بوسیله مقدار

هنگامی که تکراری را بسته به مقدار (by-value) توسط حلق forech() انجام می شود یک کپی از ان آرایه گرفته شده و عملیاتها بر روی آن انجام می شود و تکرار حلقه هیچ تاثیری بروی آرایه ندارد به مثال زیر توجه فرمایید:

```
$array = [0, 1, 2];
$ref =& $array; // Necessary to trigger the old behavior
foreach ($array as $val) {
var_dump($val);
unset($array[1]);
}
```

خروجی (0 1 2), در حالی که در نسخه قبلی گزینه دوم را چاپ نمی کرد و خروجی بدین گونه بود (0 2).

تکرار حلقه بوسیله ارجاع به مرجع (by-refrence)

در حلقه تکرار (by-refrence)، تغییرات در آرایه برای تاثیر گذاری ادامه خواهد داشت بنابر این PHP یک تغییر خوب دارد ، حفظ موقعیت صحیح در تعدادی از موارد. به عنوان مثال

```
$array = [0];
foreach ($array as &$val) {
var_dump($val);
$array[1] = 1;
}
```

خروجی int(0) , int(1)ولی در نسخه قبلی خروجی int(0)

تکرار اشیا

تکرار ساده (non-Traversable) اشیاء با مقدار و یا با مرجع مانند رفتار با مرجع است.

این منطبق بر رفتار قبلی نسخه قبلی ولی موقعیت دقیق تردارد.

مدیریت ذکر شده در نقطه قبلی.

تکرار از اشیاء گذرپذیر بدون تغییر باقی می ماند.

منبع :

- https://wiki.php.net/rfc/php7_foreach

دسترسی عددی

لیترال مبنای هشت نامعتبر

لیترال مبنای هشت نامعتبر (شامل رقم بزرگتر از 7) در حال حاضر خطای کامپایل تولید می کند. به مثال زیر توجه فرمایید

```
$i = 0781; // 8 is not a valid octal digit!
```

پیش از این ، رقم نامعتبر به سادگی نادیده گرفته می شد. به این ترتیب $i مساوی با 7 می شد، چرا که دو رقم آخردر آرامش دور انداخته می شد.

تغییر بیتی منفی

تغییر شیفتی در اعداد منفی در حال حاضر خطای ArithmeticError می دهد به مثال زیر توجه نمایید:

```
var_dump(1 >> -1);
// ArithmeticError: Bit shift by negative number
```

تغیر شیفتی به سمت چپ

تغییر شیفتی به سمت چپ همیشه جوابش صفر است: به مثال توجه کنید

```
var_dump(1 << 64); // int(0)
```

در نسخه قبلی بسته به پردازشگر خروجی می داد به طور مثال در (x64 , x86) خروجی 1 می باشد.

شیفت به سمت راست

تغییر شیفت به سمت راست

در شیفت به سمت راست هم بسته به علامت عدد مقدار های 0 ویا -1 می دهد به مثال زیر توجه فر مایید:

```
var_dump(1 >> 64); // int(0)
var_dump(-1 >> 64); // int(-1)
```

منبع:

- https://wiki.php.net/rfc/integer_semantics

دسترسی رشته ای

رشته که شامل اعداد هگزادسیمال است از این به بعد در نظر گرفته می شود و نیازی رفتار ویژه عددی وجود ندارد. برخی از نمونه های رفتار جدید:

```
var_dump("0x123" == "291"); // bool(false) (previously true)
var_dump(is_numeric("0x123")); // bool(false) (previously true)
var_dump("0xe" + "0x1"); // int(0) (previously 16)
var_dump(substr("foo", "0x1")); // string(3) "foo" (previously "oo")
// Notice: A non well formed numeric value encountered
```

از این به بعد تابع filter_var() می تواند چک کند که آیا رشته ما دارای اعداد هگزا دسیمال هست یا خیر

```
$str = "0xffff";
$int = filter_var($str, FILTER_VALIDATE_INT, FILTER_FLAG_ALLOW_HEX);
if (false === $int) {
throw new Exception("Invalid integer!");
}
var_dump($int); // int(65535)
```

موارد حذف شده

- حذف کد های asp(<%)و اسکریپت های (<script language=php>) php

منبع https://wiki.php.net/rfc/remove_-alternative_php_tags

- حذف پشتیبانی برای اختصاص دادن مقادیر جدید by-refrence
- حذف پشتیبانی برای صدا زدن متد های غیر استاتیک از $this ناسازگار جهت اطلاعات بیشتر : https://wiki.php.net/rfc/incompat_ctx.
- پشتیبانی حذف برای style # در فایل های ini. و جایگزین شدن style-;
- $HTTP_RAW_POST_DAT دیگر قابل استفاده نیست

تغییرات استاندارد در کتابخانه

- substr() مقدار خالی را بر می گرداند وقتی استباه باشد ولی در نسخه جدید مقدار falseرا بر می گرداند
- call_user_method()وcall_user_method_array() دیگر وجود ندارد
- ob_start() دیگر یک خطای E_RECOVERABLE_ERROR را بر می گرداند
- الگوریتم مرتب سازی داخلی بهبود یافته است
- تابع dl() حذف شده
- setcookie() برای ایجاد مقادیر خالی خطا می دهد و دیگر کوکی با مقدار خالی نمی سازد

تغییرات دیگر

Curl

- غیر فعال سازی متد CURLOPT_SAFE_UPLOAD حذف شده همه ی بار گزاری ها باید از curl_file / CURLFile استفاده نمایند.

Date

- $is_dst از پارامتر های mktimeوgmmktime حذف شده است.

DBA

- dba_delete() مقدار خالی را در صورت عدم یافت بر می گرداند.

GMP

- نیاز دارد به کتابخانه نسخه 4.2 به بعد.
- gmp_setbit()وgmp_clrbit() مقدار flaseرا بر می گرداند در موارد اشتباه .

libxml

- اضافه شدن LIBXML_BIGLINES و کار می کند با libxml 2.9.0 و اضافه می کند پشتیبانی numbers >16- bit برای گزارش خطا.

Mcrypt

- حذف mcrypt_generic_endو جایگزینی mcrypt_generic_deinit().
- حذف توابع mcrypt_ecb(), mcrypt_cbc(), mcrypt_cfb() and mcrypt_ofb()

Session

- session_start() می تواند یک ازایه را به عنوان تنظیمات بگیرئ به عنوان مثال
  session. cache_limiter=private تنظیم می کند['cache_limiter'‘private’]
- توابع validate_sid(), update_timestamp() که شناسه session را تحلیل می کند.
- توابع validateSid(), updateTimestamp() شناسه دسترسی به SISSION را بروز رسانی می کند.
- session.lazy_write(default=On) در تنظیمات پیش فرض پی اچ پی فعال است و این گزینه باعث می شود شما فقط وقتی می توانید SESSION را مقدار دهی کنید که آن را آپدیت کرده باشید.

Opcache

- حذف opcache.load_comments از تنظیمات.

PCRE

- حذف PREG_REPLACE_EVAL و جایگزین شدن با preg_replace_callback().

PDO_pgsql

- حذف ویژگی PGSQL_ATTR_DISABLE_NATIVE_PREPARED_STATEMENT از
  ATTR_EMULATE_PREPARES

Standard

- حذف شدن setlocale(). و جایگزینی با LC_*.
- حذف شدن set_magic_quotes_runtime() و جایگزینی با magic_quotes_runtime().

JSON

- خروجی تابع json_decode با یک آرگومان برار خالی می باشد .

Stream

- حذف set_socket_blocking() و جایگزینی آن با stream_set_blocking()

XSL

- حذف `xsl.security_prefs` در فایل تنظیمات و جایگزینی آن با `XsltProcessor::setSecurityPrefs()`

با تشکر از اینکه وقت با ارزشتون را در اختیار ما قرار دادید