FREE eBook

LEARNING

php-7

Free unaffiliated eBook created from
Stack Overflow contributors.

#php-7

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: php-7

It is an unofficial and free php-7 ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official php-7.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with php-7

Remarks

This section provides an overview of what php-7 is, and why a developer might want to use it.

It should also mention any large subjects within php-7, and link out to the related topics. Since the Documentation for php-7 is new, you may need to create initial versions of those related topics.

Examples

Installation or Setup

Detailed instructions on getting php-7 set up or installed.

PHP-7, better and new.

After php 5.4. 5.5 and 5.6 this new mayor update came. The update comes with many new programming features, techniques and ways of writing code. Installing PHP 7 could be done in multiple ways.

To install it for a localhost development like WAMP or XAMPP either check for software updates from their end and see if they come with the new PHP 7, If not you could always download a new PHP version. Note that by now PHP 7.1.4 is already released in its early state. If you've downloaded the new version don't forget to implement it the right way so that you're able to use it. Doing this could be established by following the following tutorial for WAMP: tutorial and the following turorial for XAMPP: tutorial

Update a servers PHP version
If you have your own server to host your websites on updating PHP could be established in multiple different ways. From a `sudo apt-get update && sudo apt-get upgrade` command it could work. But a better way is to download the new package with for linux

```
sudo apt-get install python-software-properties software-properties-common
sudo LC_ALL=C.UTF-8 add-apt-repository ppa:ondrej/php
sudo apt-get update
```

When you've updated PHP you're able to use many new functions and features. Next to that there are also speed improvements that came with this update.

Now you only need to check if PHP-7 had been installed by running the following command:

```
php -v
```

The output should be something like this.

```
PHP 7.1.2 (cli) (built: Feb 14 2017 21:38:43) ( ZTS MSVC14 (Visual C++ 2015) x86)
Copyright (c) 1997-2017 The PHP Group
Zend Engine v3.1.0, Copyright (c) 1998-2017 Zend Technologies
```

Read Getting started with php-7 online: https://riptutorial.com/php-7/topic/9686/getting-started-with-php-7

Chapter 2: Anonymous class

Introduction

Anonymous classes are useful when simple, one-off objects need to be created. They can be used in place of a full class definition.

They can everything a normal class can: pass arguments through to their constructors, extend other classes, implement interfaces, use traits.

Anonymous classes are assigned a name by the engine, This name has to be regarded an implementation detail, which should not be relied upon.

Examples

Simple in-place data wrapper

```
interface IArrayWrapper {
    public function getProperties(): array;
    public function has(string $name): bool;
    public function __toString();
    // ...
};

/**
 * Lightweight in-place data wrapper.
 * Demonstrates usage of anonymous class in conjunction with interface.
 *
 * Provides some basic functionality for managing array data in OO style.
 * Can be used as a wrapper for API request/response data etc.
 * Converts data to JSON with simple `(string)` cast.
 */
new class($data) implements IArrayWrapper
{
    /** @var array */
    private $data;

    public function __construct(array $data)
    {
        $this->data = $data;
    }

    public function getProperties(): array
    {
        return is_array($this->data) ? array_keys($this->data) : [] ;
    }

    public function has(string $name): bool
    {
        return (bool)($this->data[$name] ?? false);
    }

    public function get(string $name)
```

```
    {
        return $this->data[$name] ?? null;
    }

    public function __isset($name)
    {
        return $this->has($name);
    }

    public function __get($name)
    {
        return $this->get($name);
    }

    public function __toString()
    {
        return json_encode($this->data);
    }
};
```

Usage

Assume our `$data` as follows and class is stored in `$cls` variable:

```
$data = ['a' => 'b', 'c' => 'd', 'e' => 5];
```

```
$cls->a; // b
$cls->b; // null
$cls->e; // 5
isset($cls->a); // true
isset($cls->b); // false
$cls->has('a'); // true
$cls->has('b'); // false
$cls->getProperties(); // Array([0] => a [1] => c [2] => e)
(string)$cls; // {"a":"b","c":"d","e":5}
$cls instanceof IArrayWrapper; // true
```

Read Anonymous class online: https://riptutorial.com/php-7/topic/9781/anonymous-class

Chapter 3: Null Coalesce Operator

Introduction

The `null` coalescing operator (`??`) has been added as syntactic sugar for the common case of needing to use a ternary in conjunction with `isset()`.

It returns its first operand if it exists and is not `NULL`; otherwise it returns its second operand.

Examples

General usage

```
// Fetches the value of $_GET['id'] and returns 0 if it does not exist.
$id = $_GET['id'] ?? 0;
// This is equivalent to:
$id = isset($_GET['id']) ? $_GET['id'] : 0;

// Coalescing can be chained: this will return the first defined value out of
// $_GET['id'], $_POST['id'], and 0.
$id = $_GET['id'] ?? $_POST['id'] ?? 0;
```

Read Null Coalesce Operator online: https://riptutorial.com/php-7/topic/9715/null-coalesce-operator

Chapter 4: Spaceship operator

Introduction

The spaceship operator is used for comparing two expressions. For example, $a <=> $b returns -1, 0 or 1 when $a is respectively less than, equal to, or greater than $b. Comparisons are performed according to PHP's usual type comparison rules.

Examples

Generic numerical example

Generic example in a form of `$a <=> $b` matrix.

```
0 <=> 1; // -1 (left operand less than right, right is greater)
0 <=> 0; // 0 (operands are equal)
1 <=> 0; // 1 (left operand greater than right, left is greater)
1 <=> 1; // 0 (operands are equal)
```

```
╔═══════╦═══╦═══╗
║ $a/$b ║ 0 ║ 1 ║
╠═══════╬═══╬═══╣
║   0   ║ 0 ║-1 ║
║   1   ║ 1 ║ 0 ║
╚═══════╩═══╩═══╝
```

$a - leftmost column, **$b** - topmost row

Sorting a list of numbers

```
$array = [1, 0, 5, 9, 3, 7, 6, 8, 4, 2];

usort($array, function (int $a, int $b): int {
    return $a <=> $b;
});

print_r($array);
```

```
Array
(
    [0] => 0
    [1] => 1
    [2] => 2
    [3] => 3
    [4] => 4
    [5] => 5
    [6] => 6
    [7] => 7
    [8] => 8
```

```
    [9] => 9
)
```

Simple Example

```
$a = 5;
$b = 10;

$a <=> $a; // 0, because $a == $a
$a <=> $b; // -1, because $a < $b
$b <=> $a; // 1, because $b > $a
```

Read Spaceship operator online: https://riptutorial.com/php-7/topic/9716/spaceship-operator

Credits

S. No	Chapters	Contributors
1	Getting started with php-7	Community, Deathstorm, E_p
2	Anonymous class	Paul T. Rawkeen
3	Null Coalesce Operator	Paul T. Rawkeen
4	Spaceship operator	Paul T. Rawkeen, Yahya Uddin