

ENTÃO ME AJUDE A
ATUALIZAR, **PHP7** EU
QUERO É VOCÊ



Olá!



Meu nome é Raquel

- ▶ Evangelista do PHP
- ▶ Entusiasta de Comunidades
- ▶ Amante de Pizza de Calabresa
- ▶ Desenvolvedora Web - Reobote Soluções Web
- ▶ Graduanda de Sistemas para Internet
- ▶ Líder projeto Casulo - GE Faculdade Evolução
- ▶ Participante das Comunidades PHP com Rapadura, Frontend CE e Darkmira Tour
- ▶ Representante Arretadas do PHP



O que vamos ver aqui?

- ▶ Você sabe o que é refatorar?
- ▶ Mas, porque refatorar?
É realmente necessário atualizar versão?
- ▶ PHP7 é maravilhando, não sei porque ainda tá de fora
- ▶ Dicas arretadas



Versão principal	Versão intermediária	Data de lançamento	Notas
1	1.0.0	1995-06-08	Oficialmente chamado de "Personal Home Page Tools (PHP Tools)" (Ferramentas para página pessoal). Este foi o primeiro uso para o nome "PHP".
2	2.0.0	1997-11-01	Considerado pelo seu criador como a "mais rápida e simples ferramenta" para criar páginas dinâmicas para a Web.
3	3.0.0	1998-06-06	O desenvolvimento passou a ser feito por vários desenvolvedores em colaboração. Zeev Suraski e Andi Gutmans reescreveram toda a base do PHP nesta versão.
	3.0.18	2000-10-20	Última versão para do PHP 3.0.x. Unsupported Historical Releases
4	4.0.0	2000-05-22	Foi adicionado um melhor sistema de análise sintática (parser) chamado de motor Zend (Zend engine). ^[14]
	4.1.0	2001-12-10	Introduzidas as 'superglobais' (<code>\$_GET</code> , <code>\$_POST</code> , <code>\$_SESSION</code> , etc.) ^[14]
	4.2.0	2002-04-22	A <code>register_globals</code> passou agora a estar desativada por padrão. Dados recebidos via rede são mais inseridos no escopo de variável global , fechando possíveis brechas de segurança. ^[14]
	4.3.0	2002-12-27	Introduziu sua interface de linha de comando (command-line interface - CLI), para complementar o CGI. ^{[14][15]}
	4.4.0	2005-07-11	Adicionadas as páginas do manual para os script <code>phpize</code> e <code>php-config</code> . ^[14]
	4.4.9	2008-08-07	Melhorias na segurança e correção de bugs. Última versão do PHP 4.4.x. ^{[16][17]}
	5.0.0	2004-07-13	Zend Engine II com um novo modelo de objeto. ^[18]
	5.1.0	2005-11-24	Melhorias na performance com a introdução de variáveis de compilação na reengenharia do motor PHP. ^[18] Adicionada biblioteca <i>PHP Data Objects</i> (PDO) como uma nova interface de acesso aos bancos de dados . ^[19]
	5.2.0	2006-11-02	Habilitado por padrão o filtro de extensões. Suporte JSON nativo. ^[18]
	5.2.16	2010-12-16	Última versão para a série 5.2. ^[20]
	5.2.17	2011-01-06	Correção de vulnerabilidade crítica relacionada a ponto flutuante .



5.3.0	2009-06-30	Suporte a nomes de espaço (<i>namespace</i>), Vinculação de nomes (<i>late static bindings</i>), rótulos de salto de código (goto limitado), clausura nativa, arquivos PHP nativos (<i>phar</i>), coletor de lixo para referências circulares, suporte Windows melhorado, sqlite3 , mysqlnd em substituição a libmysql como biblioteca de extensão de trabalho com MySQL , fileinfo em substituição ao mime_magic para um melhor suporte MIME , extensão de internacionalização, e descontinuidade da extensão ereg .
5.3.1	2009-11-19	Mais de 100 correções de problemas , dentre eles algumas falhas de segurança.
5.3.2	2010-03-04	Grande número de correção de bugs.
5.3.3	2010-07-22	Principalmente correções de bugs e segurança; FPM SAPI .
5.3.4	2010-12-10	Principalmente correções de bugs e segurança; FPM SAPI melhorado.
5.3.5	2011-01-06	Conserto de erro crítico relacionado a ponto flutuante .
5.3.6	2011-03-10	Mais de 60 correções de bug reportados em versões anteriores.
5.3.7	2011-08-18	Esta versão focou-se na melhoria da estabilidade da série PHP 5.3.x com mais de 90 correções de bug, algumas relacionadas também à segurança.
5.3.8	2011-08-23	Esta versão corrigiu dois problemas introduzidos na versão PHP 5.3.7.
5.3.9	2012-01-10	Esta versão focou-se na melhoria da estabilidade da série PHP 5.3.x.
5.3.10	2012-02-02	Corrigida execução remota arbitrária reportada por Stefan Esser, CVE-2012-0830.
5.3.13	2012-05-08	Corrigida vulnerabilidade nas instalações utilizando CGI.
5.3.14	2012-06-06	bugs corrigidos .
5.3.15	2012-07-19	bugs corrigidos .
5.3.16	2012-08-16	bugs corrigidos .
5.3.17	2012-09-13	bugs corrigidos .
5.3.21	2013-01-17	bugs corrigidos .




5.4.0	2012-03-01	Suporte à Trait , suporte a uma versão mais curta na sintaxe de vetores. Items removidos: <code>register_globals</code> , <code>safe_mode</code> , <code>allow_call_time_pass_reference</code> , <code>session_register()</code> , <code>session_unregister()</code> and <code>session_is_registered()</code> . Servidor web embutido. ^[21] Várias melhorias nas funcionalidades já existentes e na performance. Redução dos requerimentos de memória.
5.4.1	2012-04-26	correções de bugs .
5.4.2	2012-05-03	Pacote de segurança para corrigir vulnerabilidade em chamadas PHP-CGI.
5.4.3	2012-05-08	Correção de vulnerabilidade relacionada às instalações baseadas em CGI e correção em vulnerabilidade de <i>buffer overflow</i> na função <code>apache_request_headers()</code> .
5.4.4	2012-06-06	correções de bugs .
5.4.5	2012-07-19	correções de bugs .
5.4.6	2012-08-16	correções de bugs .
5.4.7	2012-09-13	correções de bugs .
5.4.11	2013-01-17	correções de bugs .
5.5.0	2013-06-20	correções de bugs .
5.6.9	2015-05-14	correções de bugs .
7.0.0	2015-12-03	Após muitos anos de desenvolvimento, finalmente a linguagem foi lançada, porém com uma performance surpreendente. A nova versão não trouxe apenas melhorias em performance, mas também novas funcionalidades, além de implementar e fortificar novos recursos na orientação a objetos.
7.1.0	2016-12-01	correções de bugs [1] .
7.2.3	2018-03-01	correções de bugs .
7.3.3	2019-03-07	correções de bugs .



- 6 de Junho de 2019: PHP 7.4 Alpha 1
- 18 de Julho de 2019: PHP 7.4 Beta 1 – Congelamento de recursos
- 28 de Novembro de 2019: Lançamento do PHP 7.4 GA



- 
- ▶ Eu não preciso acompanhar as novas versões, quando o sistema dá problema, eu sei como resolver
 - ▶ Eu não vejo problema com versões antigas, há tantos sistemas no ar que estão assim
 - ▶ Mas, o servidor ainda aceita a versão do meu sistema
 - ▶ Necessidade de atualizar... É melhor criar um novo projeto!



PHP5

Afastar-se do ext / mysql não é apenas sobre segurança, mas também sobre o acesso a todos os recursos do banco de dados MySQL.

O ext / mysql foi criado para o MySQL 3.23 e só recebeu muito poucas adições desde então, mantendo a compatibilidade com esta versão antiga, o que dificulta a manutenção do código. Do alto da minha cabeça, faltando recursos que não suportam, o ext / mysql inclui:

- * Procedimentos armazenados (não pode lidar com vários conjuntos de resultados)
- * Declarações preparadas
- * Criptografia (SSL)
- * Compressão
- * Suporte completo a Charset
- * ...

Portanto, sair do ext / mysql é uma coisa boa.



```
// consulta vulnerável a injeção de SQL
$query = mysql_query("SELECT * FROM alunos WHERE id = $id")

// consulta parametrizada, variável $id é higienizada
$query = mysqli_query("SELECT * FROM alunos WHERE id = ?")
$query->bind_param("i", $id);
```

PHP5

/* mensagem de erro */

Warning: Wrong parameter count for mysql() in

/home/plasacom/public_html/zxx/xxx.php on line 18

Warning: mysql_fetch_object(): supplied argument is not a valid MySQL result resource in /home/plasacom/public_html/zxx/xxx.php on line 19

ao colocar este código:

```
<?php
```

```
error_reporting(0);
```

```
?>
```

com esse código no início da página você consegue desabilitar os erros...

```
error_reporting(E_WARNING);
```

```
1 <?php
2 /* Erro intencional de arquivo */
3 $my_file = @file ('arquivo_nao_existente') or die("Falha abrindo arquivo: '$p
4
5 // Isto funciona para qualquer expressão, não apenas para funções:
6 $valor = @$carrinho[$produto];
7 // você não receberá nenhum aviso se a chave $produto não existir.
8 ?>
```



PHP5

```
<script language="php">  
    echo 'alguns editores (como o FrontPage) não  
        suportam processar instruções com tags assim';  
</script>
```

```
<% echo 'Você também pode utilizar tags no estilo ASP'; %>  
<%= $variable; %> é um atalho para <% echo $variable; %>
```



PHP5

```
<?php
$senha = md5($_POST['senha']);

// Adicionando a função md5($STRING), a sua senha esta criptografada!
?>
```



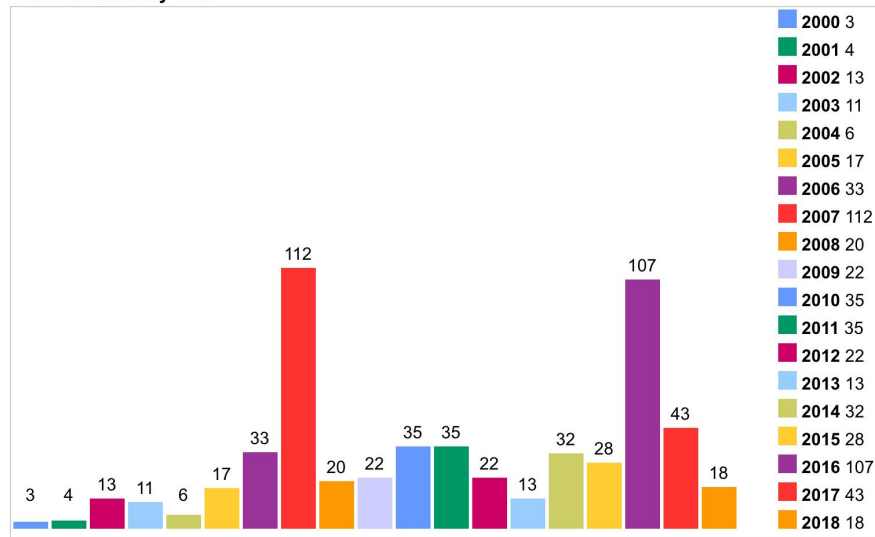
É muita, treta!

PHP5

- ▶ Falta de padronização
- ▶ Processamento lento
- ▶ Precariedade na segurança



Vulnerabilities By Year

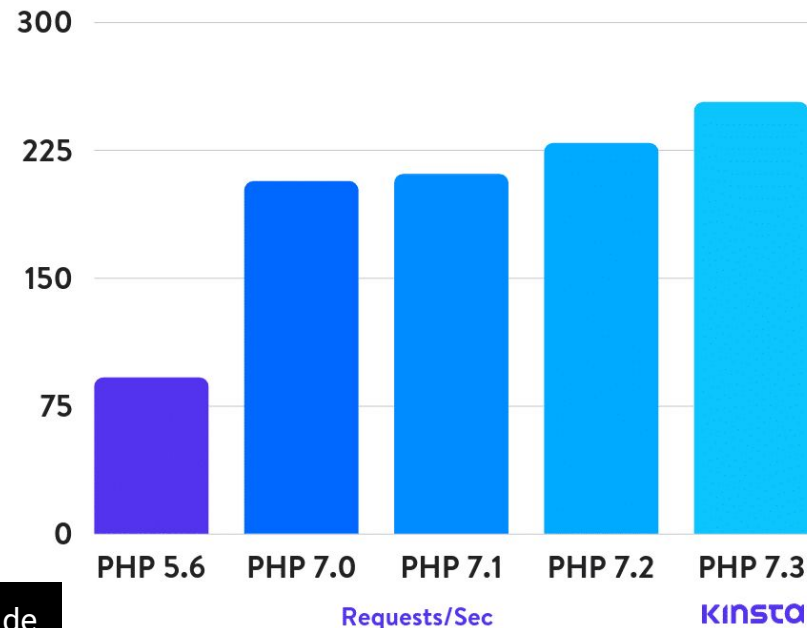


Vulnerabilidades de segurança do PHP por ano



Nós rodamos nossos próprios benchmarks de desempenho PHP com PHP 7.3. Vimos que o WordPress 5.0 no PHP 7.3 poderia executar quase três vezes mais transações (pedidos) por segundo em comparação com o PHP 5.6.

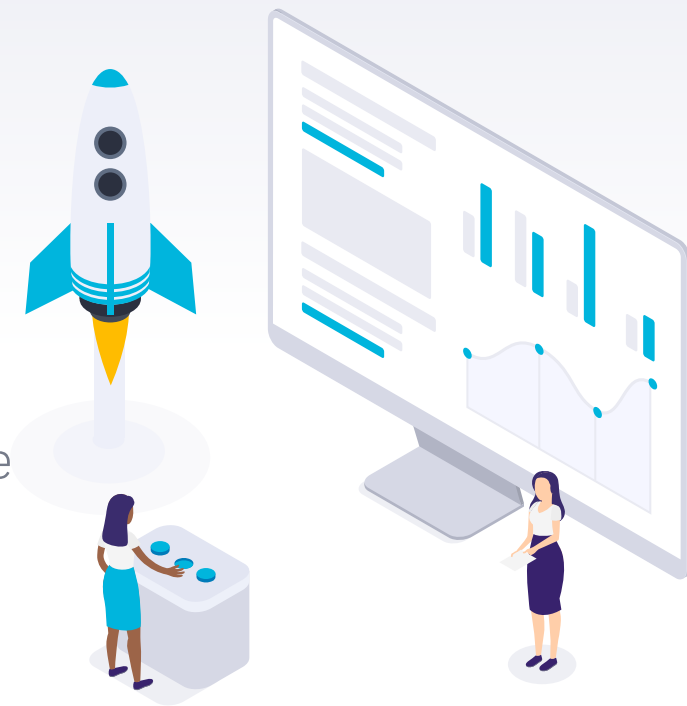
WordPress 5.0



Referências do WordPress 5.0 PHP

1 Refatorar

Alterar a estrutura interna de um software de forma a torná-lo mais fácil de perceber sem modificar o seu comportamento observável



A mudança é muito dolorosa?
Como posso começar?



Benefícios

- ▶ Legibilidade
- ▶ Compreensão
- ▶ Melhor Organização
- ▶ Futura Manutenção
- ▶ Otimização



▶ Perguntas que se deve fazer

- ▶ Meu código antigo funciona, mas, será que funciona bem?
- ▶ Posso reutilizar meu código em um novo projeto?
- ▶ Outra pessoa consegue ler e entender meu script?



PHP7

Linda caixinha de surpresas,
removeu o que atrapalhava, trouxe
o que a gente precisava



O homi é brabo

► Desempenho fantástico

- ▶ Mysql removido!
- ▶ Construtores do PHP 4 obsoletos removidos!
- ▶ Algumas alternativas de declaração de documento PHP foram removidas



O homi é brabo

Constantes pré-definidas

PDO::FETCH_ASSOC ([integer](#))

Specifies that the fetch method shall return each row as an array indexed by column name as returned in the corresponding result set. If the result set contains multiple columns with the same name,

PDO::FETCH_ASSOC returns only a single value per column name.

PDO::FETCH_NAMED ([integer](#))

Specifies that the fetch method shall return each row as an array indexed by column name as returned in the corresponding result set. If the result set contains multiple columns with the same name,

PDO::FETCH_NAMED returns an array of values per column name.

PDO::FETCH_NUM ([integer](#))

Specifies that the fetch method shall return each row as an array indexed by column number as returned in the corresponding result set, starting at column 0.

PDO::FETCH_BOTH ([integer](#))

Specifies that the fetch method shall return each row as an array indexed by both column name and number as returned in the corresponding result set, starting at column 0.

PDO::FETCH_OBJ ([integer](#))

Specifies that the fetch method shall return each row as an object with property names that correspond to the column names returned in the result set.



O homi é brabo

```
<?php
$dbh = new PDO('mysql:host=localhost;dbname=test', $user, $pass);
// use the connection here
$sth = $dbh->query('SELECT * FROM foo');
```

```
<?php
$pdo = new PDO('pgsql:host=192.168.137.1;port=5432;dbname=anydb', 'anyuser', 'pw');
sleep(5);
$stmt = $pdo->prepare('SELECT * FROM sometable');
$stmt->execute();
$pdo = null;
```



O homi é brabo

```
public function commit()
{
    if (!--$this->transactionCounter) {
        return parent::commit();
    }
    return $this->transactionCounter >= 0;
}
```

```
class Database extends PDO
{

    protected $transactionCount = 0;

    public function beginTransaction()
    {
        if (!$this->transactionCounter++) {
            return parent::beginTransaction();
        }
        $this->exec('SAVEPOINT trans'.$this->transactionCounter);
        return $this->transactionCounter >= 0;
    }
}
```

```
public function rollback()
{
    if (--$this->transactionCounter) {
        $this->exec('ROLLBACK TO trans'.$this->transactionCounter + 1);
        return true;
    }
    return parent::rollback();
}
```



O homi é brabo

```
<?php
$sth = $dbh->prepare("SELECT name, colour FROM fruit");
$sth->execute();

/* Fetch all of the remaining rows in the result set */
print("Fetch all of the remaining rows in the result set:\n");
$result = $sth->fetchAll();
print_r($result);
?>
```

Fetch all of the remaining rows in the result set:

```
Array
(
    [0] => Array
        (
            [name] => apple
            [0] => apple
            [colour] => red
            [1] => red
        )

    [1] => Array
        (
            [name] => pear
            [0] => pear
            [colour] => green
            [1] => green
        )

    [2] => Array
        (
            [name] => watermelon
            [0] => watermelon
            [colour] => pink
            [1] => pink
        )
)
```



O homi é brabo

- ▶ Erros fatais podem ser transformados em exceções

Throwable does not work on PHP 5.x.

To catch both exceptions and errors in PHP 5.x and 7, add a catch block for Exception AFTER catching Throwable first.

Once PHP 5.x support is no longer needed, the block catching Exception can be removed.

```
try
{
    // Code that may throw an Exception or Error.
}
catch (Throwable $t)
{
    // Executed only in PHP 7, will not match in PHP 5
}
catch (Exception $e)
{
    // Executed only in PHP 5, will not be reached in PHP 7
}
```



O homi é brabo

- ▶ Erros fatais podem ser transformados em exceções

Throwable does not work on PHP 5.x.

To catch both exceptions and errors in PHP 5.x and 7, add a catch block for Exception AFTER catching Throwable first.

Once PHP 5.x support is no longer needed, the block catching Exception can be removed.

```
try
{
    // Code that may throw an Exception or Error.
}
catch (Throwable $t)
{
    // Executed only in PHP 7, will not match in PHP 5
}
catch (Exception $e)
{
    // Executed only in PHP 5, will not be reached in PHP 7
}
```



O homi é brabo

- ▶ Erros fatais podem ser transformados em exceções

```
<?php
function inverse($x) {
    if (!$x) {
        throw new Exception('Divisão por zero.');
```

```
    }
    return 1/$x;
}

try {
    echo inverse(5) . "\n";
    echo inverse(0) . "\n";
} catch (Exception $e) {
    echo 'Exceção capturada: ', $e->getMessage(), "\n";
}

// Execução continua
echo "Olá mundo\n";
?>
```

0.2

Exceção capturada: Divisão por zero.
Olá mundo



O homi é brabo

- ▶ **Libsodium é o poder!**

Argon2 é uma função de derivação de chave que foi selecionada como vencedora do concurso de Hashing de senhas em julho de 2015. Foi projetada por Alex Biryukov , Daniel Dinu e Dmitry Khovratovich, da Universidade do Luxemburgo . A implementação de referência do Argon2 é liberada sob uma licença Creative Commons CC0 (ou seja, domínio público) ou a Apache License 2.0.



O homi é brabo

- ▶ **Libsodium é o poder!**

A biblioteca de criptografia de sódio (libsodium) é uma biblioteca de software moderna e fácil de usar para criptografia, descriptografia, assinaturas, hash de senha e muito mais.



O homi é brabo



- ▶ **Libsodium é o poder!**

Criptografia :

um subconjunto de ciência da computação que se concentra na comunicação segura.

Chave :

Na criptografia, uma chave é uma informação que determina a saída de um algoritmo criptográfico.

Nonce :

Um número que deve ser usado apenas uma vez (ou seja, para uma determinada chave ou conjunto de chaves).

Funções hash criptográficas (hashes):

uma transformação unidirecional determinística de dados de comprimento variável em uma saída de tamanho fixo - por si só, uma função hash não usa uma chave.

Criptografia de chave secreta :

algoritmos e protocolos criptográficos em que os dois participantes compartilham a mesma chave secreta.

Criptografia de chave pública :

algoritmos e protocolos criptográficos em que cada participante possui uma chave privada e uma chave pública relacionada.

O homi é brabo

- ▶ Libsodium é o poder!



```
$password = 'iHaveGoodPass';
$storePassword = sodium_crypto_pwhash_str($password, SODIUM_CRYPTOPWHASH_OPSLIMIT_INTERACTIVE,
    SODIUM_CRYPTOPWHASH_MEMLIMIT_INTERACTIVE);

sodium_crypto_pwhash_str_verify($storePassword, $password);

// Let's check stored hash
print_r($storePassword);
// $argon2id$v=19$m=65536,t=2,p=1$J0+Q1YfNg0Hqzt8MSSMsXA$vHq/WYrz2tKJS5XIBiZF7xFcr+N6V3J/ncFJiNPi4uY
```

```
if(sodium_crypto_pwhash_str_verify($storePassword, $password)) {
    echo "Password is correct";
    // Continue code like... Auth::login($user, true);
} else {
    echo "Password incorrect!";
    // Continue code like... redirect()->guest(route('login'));
}
```

O homi é brabo

- [sodium_add](#) — Add large numbers
- [sodium_base642bin](#) — Description
- [sodium_bin2base64](#) — Description
- [sodium_bin2hex](#) — Encode to hexadecimal
- [sodium_compare](#) — Compare large numbers
- [sodium_crypto_aead_aes256gcm_decrypt](#) — Decrypt
- [sodium_crypto_aead_aes256gcm_encrypt](#) — Encrypt
- [sodium_crypto_aead_aes256gcm_is_available](#) — Check
- [sodium_crypto_aead_aes256gcm_keygen](#) — Get random
- [sodium_crypto_aead_chacha20poly1305_decrypt](#) — Decrypt
- [sodium_crypto_aead_chacha20poly1305_encrypt](#) — Encrypt
- [sodium_crypto_aead_chacha20poly1305_ietf_decrypt](#) — Decrypt
- [sodium_crypto_aead_chacha20poly1305_ietf_encrypt](#) — Encrypt
- [sodium_crypto_aead_chacha20poly1305_ietf_keygen](#) — Get random
- [sodium_crypto_aead_chacha20poly1305_keygen](#) — Get random
- [sodium_crypto_aead_xchacha20poly1305_ietf_decrypt](#) — Decrypt
- [sodium_crypto_aead_xchacha20poly1305_ietf_encrypt](#) — Encrypt
- [sodium_crypto_aead_xchacha20poly1305_ietf_keygen](#) — Get random
- [sodium_crypto_auth_keygen](#) — Get random bytes for
- [sodium_crypto_auth_verify](#) — Verifies that the tag is
- [sodium_crypto_auth](#) — Compute a tag for the message
- [sodium_crypto_box_keypair_from_secretkey_and_publickey](#) — Description
- [sodium_crypto_box_keypair](#) — Randomly generate a key pair
- [sodium_crypto_box_open](#) — Verify and decrypt a ciphertext
- [sodium_crypto_box_publickey_from_secretkey](#) — Derive the public key
- [sodium_crypto_box_publickey](#) — Description
- [sodium_crypto_box_seal_open](#) — Decrypt the ciphertext
- [sodium_crypto_box_seal](#) — Encrypt a message
- [sodium_crypto_box_secretkey](#) — Description

- [sodium_crypto_box_seed_keypair](#) — Deterministically derive the key pair from a single key
- [sodium_crypto_box](#) — Encrypt a message
- [sodium_crypto_generichash_final](#) — Complete the hash
- [sodium_crypto_generichash_init](#) — Initialize a hash
- [sodium_crypto_generichash_keygen](#) — Get random bytes for key
- [sodium_crypto_generichash_update](#) — Add message to a hash
- [sodium_crypto_generichash](#) — Get a hash of the message
- [sodium_crypto_kdf_derive_from_key](#) — Derive a subkey
- [sodium_crypto_kdf_keygen](#) — Get random bytes for key
- [sodium_crypto_kx_client_session_keys](#) — Description
- [sodium_crypto_kx_keypair](#) — Creates a new sodium keypair

- [sodium_crypto_secretstream_xchacha20poly1305_rekey](#) — Description
- [sodium_crypto_shorthash_keygen](#) — Get random bytes for key
- [sodium_crypto_shorthash](#) — Compute a fixed-size fingerprint for the message
- [sodium_crypto_sign_detached](#) — Sign the message
- [sodium_crypto_sign_ed25519_pk_to_curve25519](#) — Convert an Ed25519 public key to a Curve25519 public key
- [sodium_crypto_sign_ed25519_sk_to_curve25519](#) — Convert an Ed25519 secret key to a Curve25519 secret key
- [sodium_crypto_sign_keypair_from_secretkey_and_publickey](#) — Description
- [sodium_crypto_sign_keypair](#) — Randomly generate a secret key and a corresponding public key
- [sodium_crypto_sign_open](#) — Check that the signed message has a valid signature
- [sodium_crypto_sign_publickey_from_secretkey](#) — Extract the public key from the secret key
- [sodium_crypto_sign_publickey](#) — Description
- [sodium_crypto_sign_secretkey](#) — Description
- [sodium_crypto_sign_seed_keypair](#) — Deterministically derive the key pair from a single key
- [sodium_crypto_sign_verify_detached](#) — Verify signature for the message
- [sodium_crypto_sign](#) — Sign a message
- [sodium_crypto_stream_keygen](#) — Get random bytes for key
- [sodium_crypto_stream_xor](#) — Encrypt a message
- [sodium_crypto_stream](#) — Generate a deterministic sequence of bytes from a seed
- [sodium_hex2bin](#) — Decodes a hexadecimally encoded binary string
- [sodium_increment](#) — Increment large number
- [sodium_memcmp](#) — Test for equality in constant-time
- [sodium_memzero](#) — Overwrite buf with zeros
- [sodium_pad](#) — Add padding data
- [sodium_unpad](#) — Remove padding data

rd verification string

public key



O homi é brabo

► Indução de Tipos: Scalar Types

ELA NÃO VAI VIRAR JAVA!

Quatro tipos escalares:

- boolean
- integer
- float (número de ponto flutuante, ou também double)
- string

Quatro tipos compostos:

- array
- object
- callable
- iterable
- Mixed
- Number
- callback (também chamado callable)
- Arrayobject
- Void

Pseudotipos

```
<?php
$a_bool = TRUE;    // um booleano
$a_str  = "foo";   // uma string
$a_str2 = 'foo';   // uma string
$an_int = 12;      // um inteiro

echo gettype($a_bool); // mostra: boolean
echo gettype($a_str);  // mostra: string

// Se ele é um inteiro, incrementa-o com quatro
if (is_int($an_int)) {
    $an_int += 4;
}

// Se $bool é uma string, mostre-a
// (não imprime nada)
if (is_string($a_bool)) {
    echo "String: $a_bool";
}
?>
```



O homi é brabo

```
<?php

class Produto
{
    public $nome;
    private $preco;
    private $categoria;

    function __construct(string $nome, float $preco, Categoria $categoria = null)
    {
        $this->nome = $nome;
        $this->preco = $preco;
        $this->categoria = $categoria;
    }

    public function getCategoria()
    {
        return $this->categoria;
    }

    public function setCategoria(Categoria $categoria)
    {
        $this->categoria = $categoria;
    }
}
```



O homi é brabo

```
<?php
```

```
require_once "Categoria.php";
require_once "Produto.php";

$categoria = new Categoria('Farmácia');
$produto = new Produto('Buscopan', 100, $categoria);

echo $produto->nome . ' - ' . $produto->getCategoria()->getNome();
```

```
<?php
```

```
class Categoria
{
    private $nome;

    public function __construct(string $nome)
    {
        $this->nome = $nome;
    }

    public function getNome()
    {
        return $this->nome;
    }
}
```



O homi é brabo

► Operador Spaceship

```
var_dump(2 <=> 3); // retorna -1  
var_dump(2 <=> 2); // retorna 0  
var_dump(2 <=> 1); // retorna 1
```

Ou seja, o operador `<=>` retorna um destes 3 valores:

- retorna -1 quando o primeiro operando é menor que o segundo
- retorna 0 quando os dois operandos são iguais
- retorna 1 quando o segundo operando é maior que o primeiro



O homi é brabo

- ▶ Null Coalesce Operator

Ele faz com que esta linha:

```
$email = $_POST['email'] ?? 'valor padrão';
```

... seja transformada nesta:

```
$email = isset($_POST['email']) ? $_POST['email'] : 'valor padrão';
```



O homi é brabo

- ▶ Classes Anônimas

```
echo (new Outer)->func2()->func3();
```



```
class Outer
{
    private $prop = 1;
    protected $prop2 = 2;

    protected function func1()
    {
        return 3;
    }

    public function func2()
    {
        return new class($this->prop) extends Outer {
            private $prop3;

            public function __construct($prop)
            {
                $this->prop3 = $prop;
            }

            public function func3()
            {
                return $this->prop2 + $this->prop3 + $this->func1();
            }
        };
    }
}
```

O homi é brabo

Recomendações de padrões PHP

► PSR

NUM	TÍTULO	EDITOR	COORDENADOR	PATROCINADOR
1	Padrão Básico de Codificação	Paul M. Jones	N / D	N / D
3	Interface do registrador	Jordi Boggiano	N / D	N / D
4	Padrão de carregamento automático	Paul M. Jones	Phil Sturgeon	Larry Garfield
6	Interface de Cache	Larry Garfield	Paul Dragoonis	Robert Hafner
7	Interface de Mensagem HTTP	Matthew Weier O'Phinney	Beau Simensen	Paul M. Jones
11	Interface de contêiner	David Négrier, Matthieu Napolí	Matthew Weier O'Phinney	Korvin Szanto
12	Guia Completo de Estilo de Codificação	Korvin Szanto	Alexander Makarov	Chris Tankersley
13	Hipermídia Links	Larry Garfield	Matthew Weier O'Phinney	Marc Alexander
14	Dispatcher de Eventos	Larry Garfield	N / D	Cees-Jan Kiewiet
15	Manipuladores HTTP	Woodk Gilk	N / D	Matthew Weier O'Phinney
16	Cache Simples	Paul Dragoonis	Jordi Boggiano	Fabien Potencier
17	Fábricas HTTP	Woodk Gilk	N / D	Matthew Weier O'Phinney
18	Cliente HTTP	Tobias Nyholm	N / D	Sara Golemon



O homi é brabo



► SOLID

SOLID é um acrônimo criado por Michael Feathers, após observar que cinco princípios da orientação a objetos e design de código — Criados por Robert C. Martin (a.k.a. Uncle Bob) e abordados no artigo The Principles of OOD — poderiam se encaixar nesta palavra.

1. Uma classe deve ter um, e somente um, motivo para mudar.
2. Objetos ou entidades devem estar abertos para extensão, mas fechados para modificação
3. Uma classe derivada deve ser substituível por sua classe base.
4. Uma classe não deve ser forçada a implementar interfaces e métodos que não irão utilizar.
5. Dependenda de abstrações e não de implementações.

S.O.L.I.D: Os 5 princípios da POO

1. **S — Single Responsibility Principle** (Princípio da responsabilidade única)
2. **O — Open-Closed Principle** (Princípio Aberto-Fechado)
3. **L — Liskov Substitution Principle** (Princípio da substituição de Liskov)
4. **I — Interface Segregation Principle** (Princípio da Segregação da Interface)
5. **D — Dependency Inversion Principle** (Princípio da inversão da dependência)

“

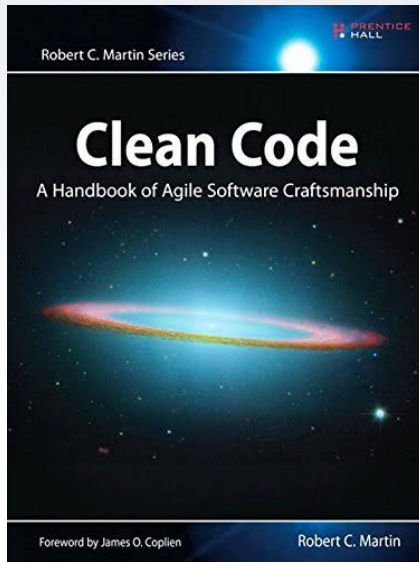
Conclusão

“A arte de ser ora audacioso ora prudente é a arte de vencer”



Dica arretada





<https://devtube.com.br/curso-poo.php>



Referências

- ▶ <http://rberaldo.com.br/tratamento-erros-php/>
- ▶ <https://forum.imasters.com.br/topic/229485-tratamento-de-erros/>
- ▶ <https://forum.imasters.com.br/topic/275542-esconder-os-warnings-sem-alterar-no-phpini/>
- ▶ <https://medium.com/joaorobertopb/o-que-%C3%A9-solid-o-guia-completo-para-voc%C3%AA-entender-os-5-princ%C3%ADpios-da-poo-2b937b3fc530>
- ▶ <https://tableless.com.br/10-novidades-do-php-7/>
- ▶ <https://www.infopedia.pt/dicionarios/lingua-portuguesa/refatorar>
- ▶ <https://kinsta.com/pt/blog/versoes-do-php/>
- ▶ <https://medium.com/joaorobertopb/o-que-%C3%A9-solid-o-guia-completo-para-voc%C3%AA-entender-os-5-princ%C3%ADpios-da-poo-2b937b3fc530>
- ▶ https://www.php.net/manual/pt_BR/book.sodium.php
- ▶ <https://kinsta.com/pt/blog/php-7-4/>
- ▶ <https://paragonie.com/book/pecl-libsodium>
- ▶ https://wiki.php.net/rfc/mysql_deprecation
- ▶ <https://pt.stackoverflow.com/questions/579/por-que-n%C3%A3o-devemos-usar-fun%C3%A7%C3%B5es-do-tipo-mysql-://aldehyf.wordpress.com/2013/11/11/desabilitar-avisos-de-erro-php-warnings/>
- ▶ [://forum.imasters.com.br/topic/534598-ocultar-ou-criptografar-senha-no-banco-de-dados-com-php-mysql/](https://forum.imasters.com.br/topic/534598-ocultar-ou-criptografar-senha-no-banco-de-dados-com-php-mysql/)



Obrigada!

- ▶ Insta: raquelbarra_
- ▶ Twitter: labarraquel
- ▶ Github: rsb12php

