Agrupamento de Escolas de Mondim de Basto

2019/2020

PHP

Redes de Comunicação - M5

Curso TGPSI

Realizado Por:

Ricardo Rodrigues Nº!3

11ºD

Cofinanciado por:

Conteúdo

Introdução

Na temática deste trabalho irei falar sobre o que é o PHP, como o podemos instalar, explicarei também como se escrevem algumas síntaxe. Tentarei explicar com se trabalha em PHP e como isso nos pode ser útil.

O que é PHP?

O PHP é uma linguagem de script criada para comunicações do lado do servidor. Consequentemente, ela é capaz de lidar com várias funções de backend como coletar formulários de dados, gerenciar arquivos do servidor, modificar bases de dados e muito mais.

Apesar do PHP ser considerado uma linguagem de scripts de propósito geral, ela é mais usada para desenvolvimento de sites Dinámicos. Isso acontece por causa de um de seus recursos mais notáveis: a habilidade de ser integrado num arquivo HTML.

O que é necessário instalar para começarmos a utilizar o PHP?

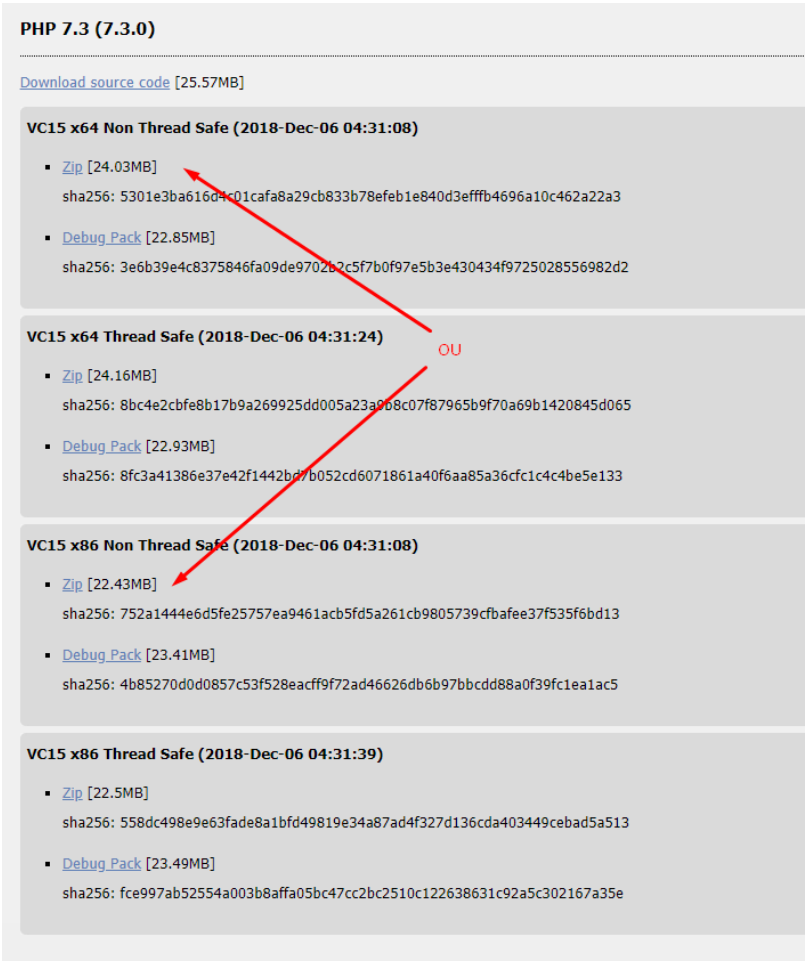
Para começarmos a utilizar o PHP Em primeiro lugar temos de instalar o Wamp, depois seguremos 3 passos:

1. Fazer o download do PHP e configura-lo no Windows.

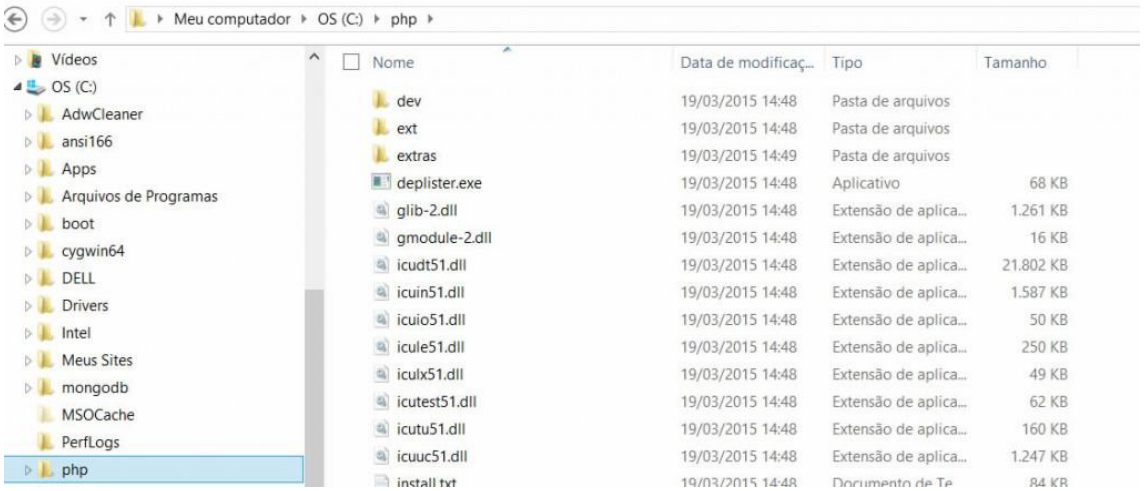
Aceda ao link "http://php.net/downloads.php" e faça o download do PHP para o Windows. Pode fazer o download da versão que quiser desde que seja maior que a 5.4. É aconselhado a fazer o download da versão 7.1 ou superior, pois a versão 5 tem o suporte finalizado desde 18 de dezembro de 2018.

Escolha a plataforma do PHP em relação ao seu computador, x86 ou x64.

Outro detalhe importante é escolher entre as versões *Thread Safe(TS)* ou *Non Thread Safe(NTS)*. Que diz respeito a questões internas do PHP, se for para desenvolvimento escolher a *Non Thread Safe(NTS)*.



Após o download, estrair o arquivo *.zip* na partição principal do seu computador e renomeie a pasta para simplesmente *php*. Veja:



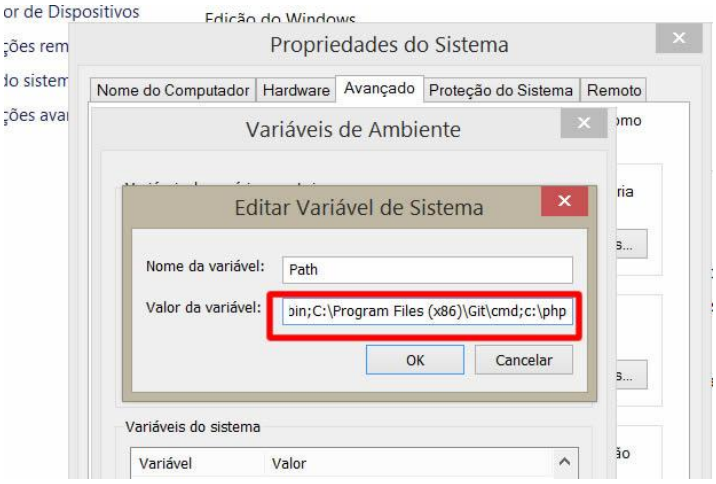
Faça a instalação do Microsoft Visual C++ (o PHP precisa dele para ser executado). Observa-se que em cada versão do PHP disponível para download, existe um *VC11 ou VC14 ou V15*… Isto indica qual versão do Microsoft Visual C++ deve ser instalado. O link de download está logo à esquerda.

Para o PHP funcionar corretamente, é preciso ativar as suas configurações padrões. Dentro da pasta descompactada no C: ; temos vários arquivos, o executável do PHP (php.exe) e dois arquivos de configurações: *php.ini-development* e *php.ini-production*. Estes dois arquivos apresentam configurações para rodar o PHP em modo de produção ou em modo desenvolvimento que é adequado para se desenvolver aplicações PHP, pois com isso ativa-se o controle de erros e debug da aplicação. Portanto, dá se um novo nome o arquivo *php.ini-development* para somente *php.ini.* O executável do PHP sempre carregará as configurações do arquivo *php.ini.*

Configurar-se o PHP para funcionar em qualquer momento no terminal.

Abra-se o *Painel de Controle*, em *Sistema*, selecione a guia *Avançado,* depois clique em *Variáveis de ambiente* no rodapé da janela. Na seção *Variáveis do sistema*, selecione-se *Path*, agora clique se em Editar, em *Valor da variável*, vamos até o final do campo de texto, agora iremos colocar o caminho onde o nosso PHP está, acrescente antes um ; (ponto e vírgula) para finalizar os caminhos anteriores e coloque *c:\php*, então, ficará assim: *….;c:\php.*

Confirme tudo e o PHP está configurado.



Agora configurar-se o arquivo de *hosts* para apontar o nome *localhost* para *127.0.0.1*.

Abra-se o menu iniciar e selesiona-se no seu editor de texto com o botão direito do rato e clique em *Executar como administrador*. No menu *Abrir* segue-se o caminho C:*Windows**System32**drivers**etc*. Abra-se o arquivo *hosts*, se ele não aparecer, selecione *Todos os arquivos* para ele ser mostrado na janela.

Agora verifica-se, se existe a linha *127.0.0.1 localhost*, se existir está pronto, senão acrescente-a ao final do arquivo, salve e feche o programa.

```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windc
#
# This file contains the mappings of IP addresses to host names.
# entry should be kept on an individual line. The IP address shc
# be placed in the first column followed by the corresponding hc
# The IP address and the host name should be separated by at lea
# space.
#
# Additionally, comments (such as these) may be inserted on indi
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
#       127.0.0.1       localhost
#       ::1             localhost

127.0.0.1       localhost
```

Para Finalizar reinicia-se o PC.

2. Usar o servidor embutido do PHP.

Para utilizar um serbidor tem de abrir o terminal de prefênria e digitar:

```
C:\Users\argen\Desktop
λ php -v
PHP 7.2.4 (cli) (built: Mar 28 2018 04:26:56) ( NTS MSVC15 (Visual C++ 2017) x64 )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.2.0, Copyright (c) 1998-2018 Zend Technologies
```

Logo em seguida de ver uma mensagem com a versão do PHP instalada então correu tudo bem.

Até a versão do PHP 5.3 tínhamos que ter um servidor web (Apache, Nginx, etc) a parte instalado para rodar aplicações no browser, mas, a partir da versão 5.4 foi introduzido um servidor embutido no PHP. Assim, pode-se iniciar um servidor a qualquer momento e em qualquer pasta, e ainda ter múltiplos servidores a processar simultaneamente em portas diferentes. Isto para o desenvolvimento é uma “mão na roda”.

Para iniciar o servidor embutido, na raiz de qualquer pasta de seu projeto PHP, faça:

```
C:\Users\argen\Desktop
λ php -S localhost:8080
PHP 7.2.4 Development Server started at Tue Dec 18 17:08:54 2018
Listening on http://localhost:8080
Document root is C:\Users\argen\Desktop
Press Ctrl-C to quit.
```

Só isto já iniciará o servidor embutido no projeto, neste caso só resta entrar no browser *localhost:8080/index.php* e sua aplicação será renderizada! Então, para usar o servidor embutido é só digitar no terminal: php -S localhost:(porta que esteja livre: 8080,8081,9999,etc).

Síntaxe básica em PHP.

```php
<?php
    $value = 'Jesus';

    // This is a simple if statement
    if( isset( $value ) )
    {
        print $value;
    }

    print '<br />';

    // This is an alternative
    isset( $value ) AND print( $value );
?>
```

Comentário em PHP.

```php
<?php
   echo 'Isto é um teste'; // Estilo de comentário de uma linha em  c++
   /* Este é um comentário de múltiplas linhas
      ainda outra linha de comentário */
   echo 'Isto é ainda outro teste';
   echo 'Um teste final'; # Este é um comentário de uma linha no estilo shell
?>
```

Exemplo de uma declaração de variáveis em PHP.

```php
<?php
$var = 'Bob';
$Var = 'Joe';
echo "$var, $Var";      // exibe "Bob, Joe"

$4site = 'not yet';     // inválido; começa com um número
$_4site = 'not yet';    // válido; começa com um sublinhado
$täyte = 'mansikka';    // válido; 'ä' é um caracter ASCII (extendido) 228
?>
```

Exemplo de Echo and Print.

Encho

```php
<?php
echo "Hello World";
echo "This spans
multiple lines. The newlines will be
output as well";
echo "This spans\nmultiple lines. The newlines will be\noutput as well.";
echo "Escaping characters is done \"Like this\".";
// You can use variables inside of an echo statement
$foo = "foobar";
$bar = "barbaz";
echo "foo is $foo"; // foo is foobar
// You can also use arrays
$baz = array("value" => "foo");
echo "this is {$baz['value']} !"; // this is foo !
// Using single quotes will print the variable name, not the value
echo 'foo is $foo'; // foo is $foo
// If you are not using any other characters, you can just echo variables
echo $foo;          // foobar
echo $foo,$bar;     // foobarbarbaz
// Some people prefer passing multiple parameters to echo over concatenation.
echo 'This ', 'string ', 'was ', 'made ', 'with multiple parameters.', chr(10);
echo 'This ' . 'string ' . 'was ' . 'made ' . 'with concatenation.' . "\n";
echo <<<END
This uses the "here document" syntax to output
multiple lines with $variable interpolation. Note
that the here document terminator must appear on a
line with just a semicolon. no extra whitespace!
END;
// Because echo does not behave like a function, the following code is invalid.
($some_var) ? echo 'true' : echo 'false';
// However, the following examples will work:
($some_var) ? print 'true' : print 'false'; // print is also a construct, but
                                            // it behaves like a function, so
                                            // it may be used in this context.
echo $some_var ? 'true': 'false'; // changing the statement around
?>
```

Print

```php
<?php
print("Hello World");

print "print() also works without parentheses.";

print "This spans
multiple lines. The newlines will be
output as well";

print "This spans\nmultiple lines. The newlines will be\noutput as well.";
```

```
print "escaping characters is done \"Like this\".";

// You can use variables inside a print statement
$foo = "foobar";
$bar = "barbaz";

print "foo is $foo"; // foo is foobar

// You can also use arrays
$bar = array("value" => "foo");

print "this is {$bar['value']} !"; // this is foo !

// Using single quotes will print the variable name, not the value
print 'foo is $foo'; // foo is $foo

// If you are not using any other characters, you can just print variables
print $foo;          // foobar

print <<<END
This uses the "here document" syntax to output
multiple lines with $variable interpolation. Note
that the here document terminator must appear on a
line with just a semicolon no extra whitespace!
END;
?>
```

Exemplos de Tipos de Dados

1. String

```
<?php
$x = "Hello world!";
$y = 'Hello world!';

echo $x;
echo "<br>";
echo $y;
?>
```

2. Integer

```
<?php
$x = 5985;
var_dump($x);
?>
```

3. Float

```
<?php
$x = 10.365;
var_dump($x);
?>
```

4. Boolean

```
$x = true;
$y = false;
```

5. Array

```
<?php
$cars = array("Volvo","BMW","Toyota");
var_dump($cars);
?>
```

6. Object

```
<?php
class Car {
    function Car() {
        $this->model = "VW";
    }
}

// create an object
$herbie = new Car();

// show object properties
echo $herbie->model;
?>
```

7. NULL Value

```
<?php
$x = "Hello world!";
$x = null;
var_dump($x);
?>
```

8. Resource

The special resource type is not an actual data type. It is the storing of a reference to functions and resources external to PHP.
A common example of using the resource data type is a database call.
We will not talk about the resource type here, since it is an advanced topic.

Exemplo de Strings

```
<?php
echo 'isto é uma string comum';

echo 'Você pode incluir novas linhas em strings,
dessa maneira que estará
tudo bem';

// Imprime: Arnold disse uma vez: "I'll be back"
echo 'Arnold disse uma vez: "I\'ll be back"';
```

```php
// Imprime: Você tem certeza em apagar C:\*.*?
echo 'Você tem certeza em apagar C:\\*.*?';

// Imprime: Você tem certeza em apagar C:\*.*?
echo 'Você tem certeza em apagar C:\*.*?';

// Imprime: Isto não será substituido: \n uma nova linha
echo 'Isto não será substituido: \n uma nova linha';

// Imprime: Variáveis $também não $expandem
echo 'Variáveis $também não $expandem';
?>
```

Funções das Strings

- strlen() - Return the Length of a String
- str_word_count() - Count Words in a String
- strrev() - Reverse a String
- strpos() - Search For a Text Within a String
- str_replace() - Replace Text Within a String

PHP Constants

a) create PHP constants

To create a constant, use the define() function.

Syntax

define(*name*, *value*, *case-insensitive*)

Parameters:

name: Specifies the name of the constant

value: Specifies the value of the constant

case-insensitive: Specifies whether the constant name should be case-insensitive. Default is false

PHP Operators

a) Lista de todas as tabelas

Precedência de Operadores

Exemplo	Nome	Resultado
+$a	Identidade	Conversão de $a para int ou float conforme apropriado.
-$a	Negação	Oposto de $a.
$a + $b	Adição	Soma de $a e $b.
$a - $b	Subtração	Diferença entre $a e $b.
$a * $b	Multiplicação	Produto de $a e $b.
$a / $b	Divisão	Quociente de $a e $b.
$a % $b	Módulo	Resto de $a dividido por $b.
$a ** $b	Exponencial	Resultado de $a elevado a $b. Introduzido no PHP 5.6.

Operadores Aritméticos

Associação	Operadores	Informação Adicional
não associativo	*clone new*	clone e new
esquerda	*[*	array()
direita	**	aritmética
direita	++ -- ~ *(int) (float) (string) (array) (object) (bool)* @	types e incremento/decremento
não associativo	*instanceof*	tipos
direita	*!*	lógicos
esquerda	* / %	aritmética
esquerda	+ - .	aritmética e string
esquerda	<< >>	bits
não associativo	< <= > >=	comparação
não associativo	== != === !== <> <=>	comparação
esquerda	&	bits e referências
esquerda	^	bits
esquerda	|	bits
esquerda	&&	lógicos
esquerda	||	lógicos
direita	??	comparação
esquerda	? :	ternário
direita	= += -= *= **= /= .= %= &= |= ^= <<= >>=	atribuição
esquerda	*and*	lógicos
esquerda	*xor*	lógicos
esquerda	*or*	lógicos

Operadores bit a bit

Exemplo	Nome	Resultado
$a & $b	E (AND)	Os bits que estão ativos tanto em *$a* quanto em *$b* são ativados.
$a | $b	OU (OR inclusivo)	Os bits que estão ativos em *$a* ou em $b são ativados.
$a ^ $b	XOR (OR exclusivo)	Os bits que estão ativos em *$a* ou em *$b*, mas não em ambos, são ativados.
~ $a	NÃO (NOT)	Os bits que estão ativos em *$a* não são ativados, e vice-versa.
$a << $b	Deslocamento à esquerda	Desloca os bits de *$a* $b passos para a esquerda (cada passo significa "multiplica por dois")
$a >> $b	Deslocamento à direita	Desloca os bits de *$a* $b passos para a direita (cada passo significa "divide por dois")

Operadores de Comparação

Exemplo	Nome	Resultado
$a == $b	Igual	Verdadeiro (**TRUE**) se $a é igual a $b.
$a === $b	Idêntico	Verdadeiro (**TRUE**) se $a é igual a $b, e eles são do mesmo tipo.
$a != $b	Diferente	Verdadeiro se $a não é igual a $b.
$a <> $b	Diferente	Verdadeiro se $a não é igual a $b.
$a !== $b	Não idêntico	Verdadeiro de $a não é igual a $b, ou eles não são do mesmo tipo (introduzido no PHP4).
$a < $b	Menor que	Verdadeiro se $a é estritamente menor que $b.
$a > $b	Maior que	Verdadeiro se $a é estritamente maior que $b.
$a <= $b	Menor ou igual	Verdadeiro se $a é menor ou igual a $b.
$a >= $b	Maior ou igual	Verdadeiro se $a é maior ou igual a $b.
$a <=> $b	Spaceship (nave espacial)	Um integer menor que, igual a ou maior que zero quando *$a* é, respectivamente, menor que, igual a ou maior que *$b*. Disponível a partir do PHP 7.

Operadores de Incremento/Decremento

Exemplo	Nome	Efeito
++$a	Pré-incremento	Incrementa $a em um, e então retorna $a.
$a++	Pós-incremento	Retorna $a, e então incrementa $a em um.
--$a	Pré-decremento	Decrementa $a em um, e então retorna $a.
$a--	Pós-decremento	Retorna $a, e então decrementa $a em um.

Operadores Lógicos

Exemplo	Nome	Resultado
$a and $b	E	Verdadeiro (**TRUE**) se tanto $a quanto $b são verdadeiros.
$a or $b	OU	Verdadeiro se $a ou $b são verdadeiros.
$a xor $b	XOR	Verdadeiro se $a ou $b são verdadeiros, mas não ambos.
! $a	NÃO	Verdadeiro se $a não é verdadeiro.
$a && $b	E	Verdadeiro se tanto $a quanto $b são verdadeiros.
$a || $b	OU	Verdadeiro se $a ou $b são verdadeiros.

Operadores de array

Exemplo	Nome	Resultado
$a + $b	União	União de $a e $b.
$a == $b	Igualdade	**TRUE** se $a e $b tem os mesmos pares de chave/valor.
$a === $b	Identidade	**TRUE** se $a e $b tem os mesmos pares de chave/valor na mesma ordem e do mesmo tipo.
$a != $b	Desigualdade	**TRUE** se $a não é igual a $b.
$a <> $b	Desigualdade	**TRUE** se $a não é igual a $b.
$a !== $b	Não identidade	**TRUE** se $a não é identico a $b.

PHP IF... ELSE...ELSEIF

The if...else statement executes some code if a condition is true and another code if that condition is false.

Syntax

```
if (condition) {
    code to be executed if condition is true;
} else {
    code to be executed if condition is false;
}
```

Conclusão

Com a temática deste trabalho fiquei, a saber, mais sobre o PHP, percebi que trabalhar com o PHP não é tão complicado como parece.

Bibliografia

https://blog.schoolofnet.com/como-instalar-o-php-no-windows-do-jeito-certo-e-usar-o-servidor-embutido/