

Desenvolvimento web: PHP Orientado à Objetos



Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Público-Alvo

- Alunos de cursos voltados para tecnologia:
 - Sistemas de Informação
 - Ciência da Computação
 - etc...
- Pessoas que já possuem conhecimento prévio de lógica de programação e HTML.



Objetivo

- O objetivo desse curso é ensinar os conceitos básicos da linguagem PHP, proporcionando aos alunos conhecimento suficiente para iniciarem seus projetos de programação orientada à objeto utilizando esta linguagem.
- Com a base adquirida nesse curso, a aprendizagem autodidata de outros recursos PHP se tornará muito mais fácil.



Leituras Recomendadas



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vega
www.mr-bin.blogspot.com

Conteúdo Programático

- Módulo I -
Introdução ao Mundo PHP



- Módulo II -
A Linguagem PHP



- Módulo III -
PHP OO - Programação Orientada à
Objetos com PHP



- Módulo IV -
Solução para os dados voláteis:
Sessão



- Módulo I - *Introdução ao Mundo PHP*

***Desenvolvimento web:
PHP Orientado à Objetos***



Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Vida de programador!



<http://www.youtube.com/watch?v=apREUmNp9Ec&feature=related>



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Alguém se identificou???

*Após o momento de descontração,
agora vamos trabalhar!*



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

You aprender a pensar no curso?



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Não!..Você não vai aprender a pensar aqui!

- Durante o curso, você vai aprender a sintaxe do PHP e a lógica dos recursos que ele oferece, **porém isso não garante que você irá fazer bons softwares** com esse conhecimento.
- Todos nós sabemos a gramática da língua portuguesa e como se escreve, porém nem todos nós temos o dom de escrever boas redações. **Com linguagens de programação é a mesma coisa!**
- No português, a prática da escrita traz melhoras consideráveis na qualidade das redações escritas por quem exercita. A prática da programação também traz melhoras de raciocínio e qualidade de software dos programadores!

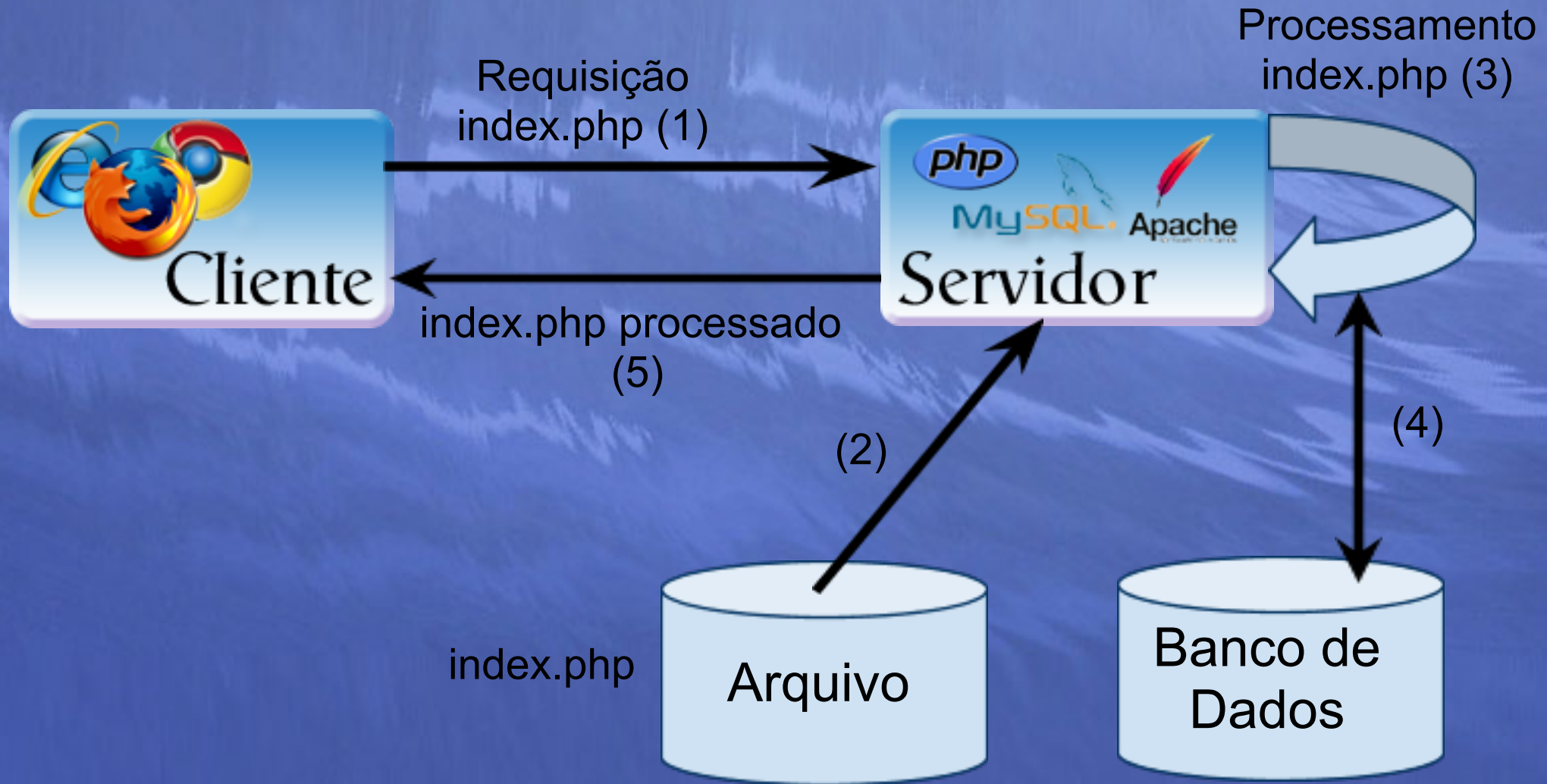


O que é Programação Server-Side?

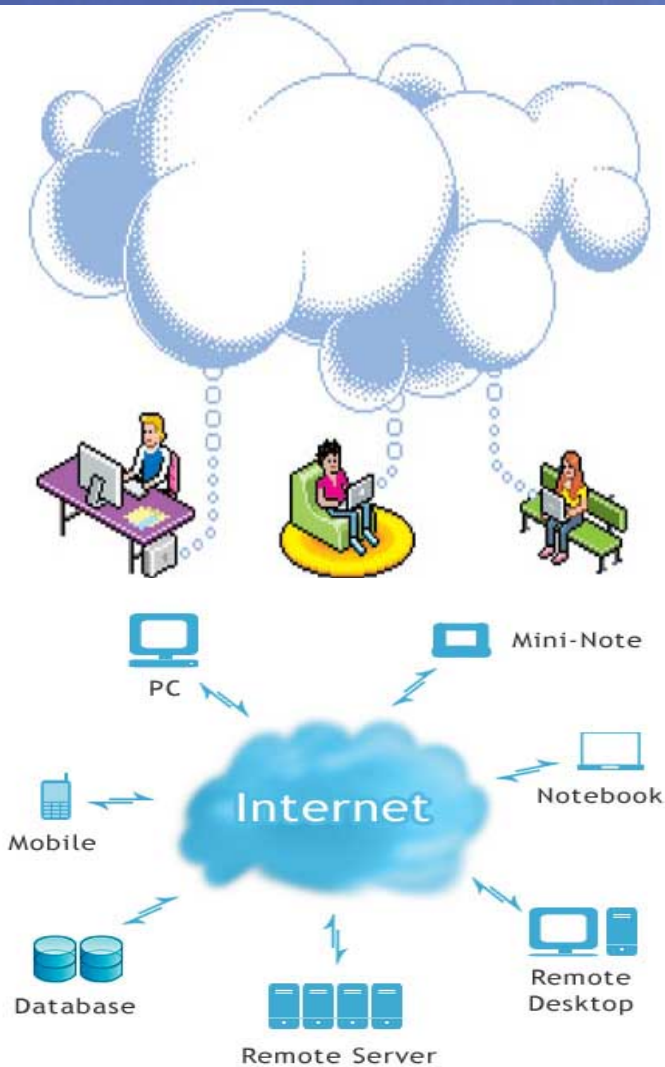
- PHP é uma linguagem de programação interpretada pelos servidores na internet (server-side).
- O Servidor recebe as requisições dos clientes, processa os scripts PHP e retorna para os clientes o PHP já processado em forma de HTML que é visualizado no navegador(browser).
- O cliente **NUNCA** conseguirá ver o código PHP.



Demonstrando programa Server-Side



Isso te faz lembrar alguma coisa?



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

PHP é computação nas nuvens!

- Todo sistema desenvolvido em PHP se enquadra no novo conceito de computação nas nuvens.
- Nesse novo conceito computacional, os dados e softwares migram dos computadores e servidores locais para as nuvens de informação, que são milhares de servidores em cluster espalhados por diversos pontos geográficos.
- Algumas das vantagens da utilização do PHP se confundem com as vantagens da computação nas nuvens.



Vantagens da utilização do PHP

- PHP é software livre!
- Portabilidade (Independente de Sistema Operacional ou equipamento)
- Mobilidade (Acesso aos dados de qualquer lugar!)
- Os softwares desenvolvidos são leves na visão do cliente.
- Possui uma das maiores comunidades de programadores da internet:
 - iMasters - <http://forum.imasters.uol.com.br/>
 - HTMLSTAFF - <http://www.htmlstaff.org/>
 - **PHP MG...**



Grupo de desenvolvedores PHP de Minas Gerais

- **Site do Grupo:** <http://www.phpmg.com>
- **Lista do Google Groups:** <http://groups.google.com/group/phpmg>
- **Comunidade do Orkut:** <http://www.orkut.com.br/Main#Community.aspx?cmm=26992151>



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Ambiente WAMP

- Para programar em PHP no ambiente Windows, as tecnologias mais utilizadas atualmente e que precisam estarem instaladas no computador do programador são:
 - **W**indows
 - **A**pache
 - **M**ySQL
 - **P**HP
- WAMP é um sistema indicado para os usuários que não têm instalado no sistema nenhuma dessas tecnologias (o Windows já deve estar instalado!), já que ele realiza uma instalação completa e desde o zero.
 - **DOWNLOAD:** <http://www.wampserver.com/en/download.php>
- Todos os seus programas PHP devem ser salvos na pasta chamada "WWW" do WAMP.



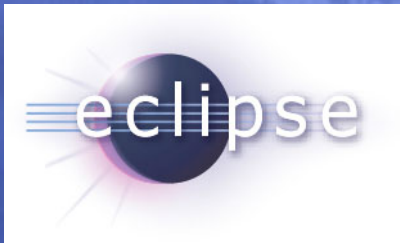
Ferramentas para desenvolvimento



■ *NotePad*



- **NotePad++** : <http://www.baixaki.com.br/download/notepad-.htm>



- **Eclipse**: <http://www.eclipse.org/pdt/>



- **Zend Studio**: <http://www.zend.com/en/products/studio/downloads>



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Hello World!



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Hello World!

<?php

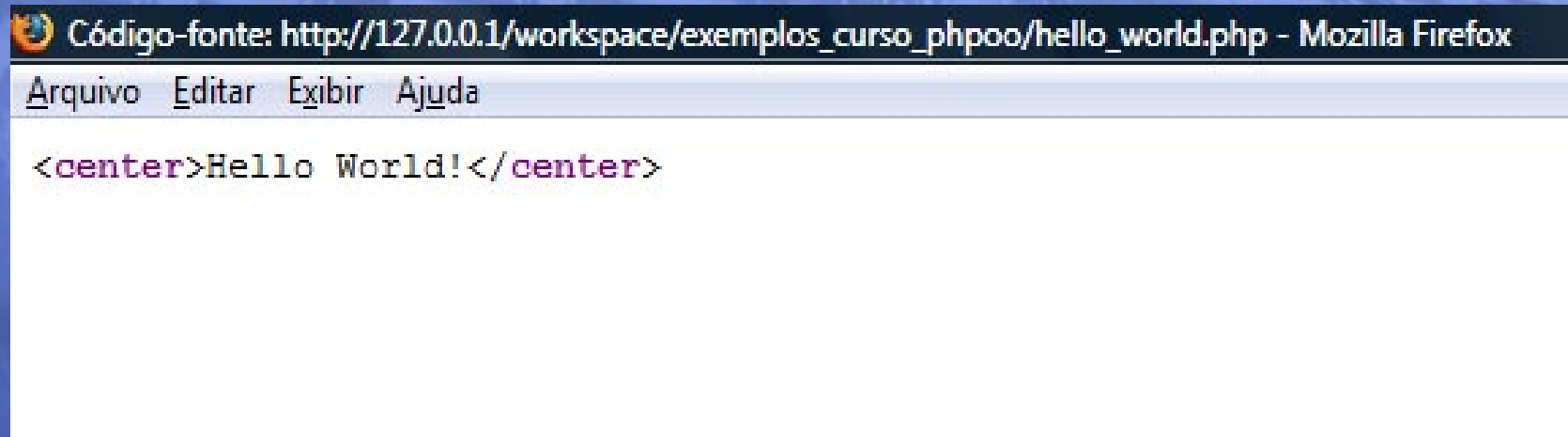
echo("<center>Hello World!</center>");

?>



Cadê o PHP??

- Já no navegador visualizando o seu Hello World!, entre na opção "exibir código-fonte" (Ctrl+U no Firefox).



The screenshot shows a Mozilla Firefox browser window. The title bar reads "Código-fonte: http://127.0.0.1/workspace/exemplos_curso_phpoo/hello_world.php - Mozilla Firefox". The menu bar includes "Arquivo", "Editar", "Exibir", and "Ajuda". The main content area displays the HTML source code: `<center>Hello World!</center>`.

- Você deve ter reparado que só apareceu códigos HTML. Isso acontece porque o servidor já processou o código PHP quando você fez a requisição de abrir a página e devolveu ao cliente (browser) somente o HTML de visualização.



- Módulo II - *A Linguagem PHP*

***Desenvolvimento web:
PHP Orientado à Objetos***



Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

PHP escreve HTML!



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Conceitos básicos da Linguagem

- Todo programador PHP deve conhecer muito bem as Tags HTML, pois todas as partes visuais das aplicações PHP são feitas pelo HTML.
- O PHP escreve HTML e também pode ser escrito misturado com o HTML.
- Como o PHP tem a capacidade de escrever HTML, ele pode ser usado para "formatar" a apresentação do PHP, dando cores, formas e posições na tela.
 - **Exemplo prático:** Desenvolver uma página que escreve uma tabela HTML utilizando o PHP.



Exemplo Tabela HTML com PHP

```
<?php //arquivo "tabela_com_php.php" ?>
<html>
<head><title>Tabela HTML com PHP</title></head>
<body>
<?php
    echo("<table align='center' border='1'>
        <tr>
            <th colspan='2' align='center'>Tabela HTML com PHP</th>
        </tr>
        <tr>
            <td align='center'>Lucas</td><td align='center'>Vegi</td>
        </tr>");
?>
</body>
</html>
```



Tipos de dados

- PHP suporta vários tipos de dados, dentre eles os principais são:
 - Boolean
 - Integer
 - Float
 - String
 - Array
 - Objeto (Veremos no módulo III)
- A tipagem em PHP é dinâmica, ou seja, em PHP **não é necessário especificar os tipos das variáveis**. É possível atribuir qualquer valor a qualquer variável, pois o interpretador PHP faz a alteração do tipo da variável de forma automática e transparente.
- Em PHP todas as variáveis são antecedidas de **\$** (cifrão ou cash).



Variáveis em PHP

- Exemplo de declaração de variável em PHP:
 - **\$nome_da_variavel;**
- No *exemplo prático* a seguir, veremos o uso de variáveis com o PHP fazendo a interpretação automática de tipos.



Exemplo das variáveis dinâmicas

<?php

\$teste; //declara a variável

\$teste = 1;
echo("Valor inteiro: ".\$teste."
"); //atribui um valor integer para ela
//imprime o valor integer

\$teste = \$teste + 3.6;
echo("Valor float: ".\$teste."
"); //atribui valor float
//imprime o valor float

\$teste = true; //atribui valor boolean
echo("Valor Boolean: ".\$teste."
"); //imprime valor boolean

\$teste = "Agora eu sou uma String!";
echo("Valor String: ".\$teste."
"); //atribui valor string
//imprime valor string

?>



Entrada e saída de dados

- Toda entrada de dados em PHP ocorre a partir de formulários HTML.
- Esses formulários podem utilizar dois métodos de envio de dados:
 - POST
 - GET
- Toda saída de dados em PHP, como já visto anteriormente, é feita a partir da função **echo()**;
 - **Exemplo prático:** Formulário HTML enviando dados por POST e GET



Exemplo formulário HTML com POST

```
<html><!-- Nome do arquivo exemplo_form.php -->
<head><title>Exemplo com formulários</title></head>
<body>
  <form action="exemplo_form.php" method="post" name="exemplo">
    Nome:      <input type="text" name="nome" size="30"><br>
    Mensagem:  <input type="text" name="mensagem" size="30"><br>
               <input type="submit" name="enviar" value="Enviar"><br>
  </form>
  <?php
    echo("Nome: ".$_POST[nome]."<br>");
    echo("Mensagem: ".$_POST[mensagem]."<br>");
  ?>
</body>
</html>
```



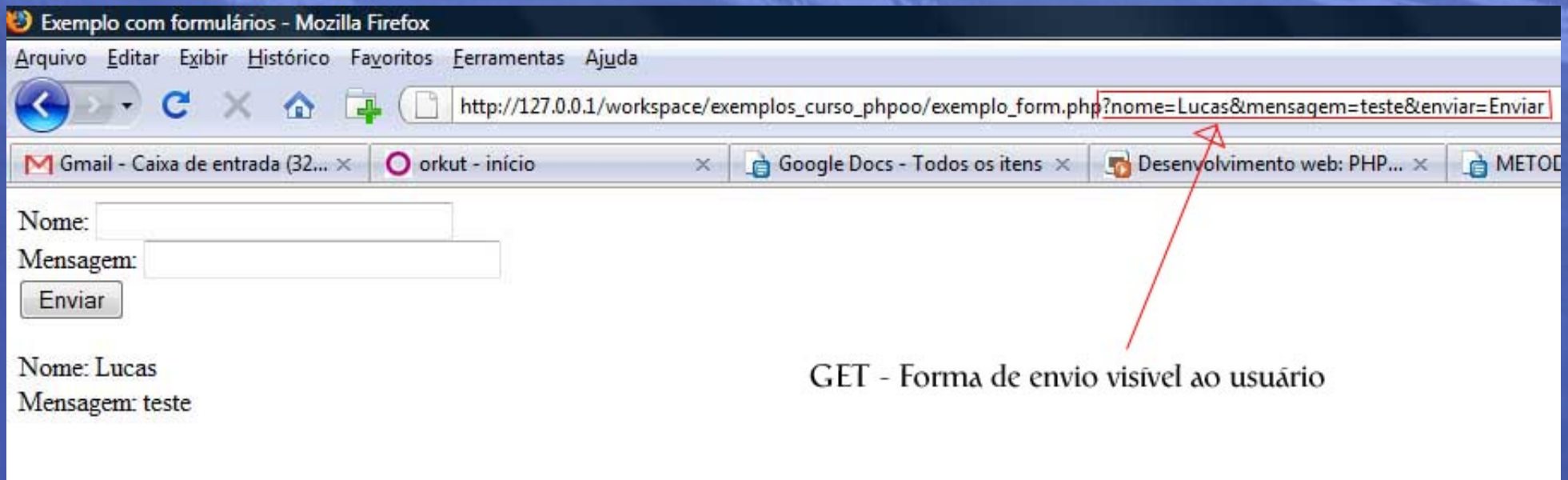
Exemplo formulário HTML com GET

```
<html><!-- Nome do arquivo exemplo_form.php -->
<head><title>Exemplo com formulários</title></head>
<body>
  <form action="exemplo_form.php" method="get" name="exemplo">
    Nome:      <input type="text" name="nome" size="30"><br>
    Mensagem:  <input type="text" name="mensagem" size="30"><br>
               <input type="submit" name="enviar" value="Enviar"><br>
  </form>
  <?php
    echo("Nome: ".$_GET[nome]."<br>");
    echo("Mensagem: ".$_GET[mensagem]."<br>");
  ?>
</body>
</html>
```



GET x POST

- Basicamente a maior diferença entre POST e GET, é o fato de que o método POST suporta enviar uma **maior** quantidade de dados e de forma **transparente** ao usuário.
- Já o método GET suporta enviar uma **menor** quantidade de dados e os envia de uma forma **visível** ao usuário.



Array em PHP

- Um array pode ser iniciado em PHP da seguinte forma:
 - `$vetor = array(17,53,89,5);`
- A partir do exemplo acima, se fosse feito um **`echo($vetor[1]);`** o valor impresso na tela seria **53**.
- Arrays também podem receber valores sem necessariamente terem sido iniciados antes. Ex:
 - `$vet[0] = "Valor atribuido";`
 - `$vet[10] = "Outro valor atribuido";`
- Uma diferença dos arrays em PHP com relação a outras linguagens de programação são os **Arrays Associativos**.



Arrays Associativos

- A maior diferença entre os arrays associativos e os arrays tradicionais, é que **os índices dos arrays associativos podem ser valores não numéricos escolhidos pelo programador.**
- Declaração de array associativo:
 - `$vet_notas = array("joão" => 8,"maria" => 9,"josé" => 6);`
- Considerando o array associativo do exemplo acima, se fosse feito um **`echo($vet_notas["maria"]);`** o valor impresso na tela seria **9**.



Palavras reservadas em PHP

and	or	xor	__FILE__
__LINE__	array()	as	break
class	const	continue	declare
die()	do	echo()	else
empty()	enddeclare	endfor	endif
endswitch	endwhile	eval()	extends
for	foreach	function	global
include()	include_once()	isset()	list()
print()	require()	require_once()	return()
switch	unset()	use	var
__FUNCTION__	__CLASS__	_METHOD_	final (no PHP 5)
interfaca(no PHP 5)	implements (no PHP 5)	instanceof(no PHP 5)	public (no PHP 5)
protected (no PHP 5)	abstract (no PHP 5)	clone (no PHP 5)	try (no PHP 5)
throw (no PHP 5)	cfunction (no PHP 4 somente)	old_function (no PHP 4 somente)	this (no PHP 5)
__NAMESPACE__ (no PHP 5.3)	namespace (no PHP 5.3)	goto (no PHP 6 somente)	__DIR__ (no PHP 5.3)
exception (no PHP 5)	if	catch (no PHP 5)	endforeach
case	new	final (no PHP 5)	exit()
default	static	php_user_filter(no PHP 5)	
elseif	while	private (no PHP 5)	



Operadores aritméticos

- + adição
- subtração
- * multiplicação
- / divisão
- % módulo
- = atribuição



Operadores relacionais e lógicos

Relacionais	Lógicos
< menor que	&& and e (conjunção)
<= menor ou igual a	or ou (disjunção)
> maior que	xor ou exclusivo
>= maior ou igual a	! não (negação)
== igual a	
!= diferente de	



Operadores de atribuição

+= adição

-= subtração

***=** multiplicação

/= divisão

%= módulo

.= concatenação

= atribuição

- $\$c = \$c + 3;$ é equivalente a $\$c += 3;$

- $\$c = \$c - 6;$ é equivalente a $\$c -= 6;$

- $\$c = \$c * 3;$ é equivalente a $\$c *= 3;$

- $\$c = \$c / 5;$ é equivalente a $\$c /= 5;$

- $\$c = \$c \% 2;$ é equivalente a $\$c \% = 2;$

- $\$c = \$c . "teste";$ é equivalente a $\$c .= "teste";$



Procedência de operadores

Operador	Descrição
(mais alta)	
! + -	não lógico, mais unário, menos unário
* / %	multiplicação, divisão, módulo
+ -	adição, subtração
< <= >= >	desigualdades
== !=	igual, diferente
&& and	e lógico
or xor	ou lógico , ou exclusivo
=	atribuição
(mais baixa)	



Incremento e decremento

- Assim como na linguagem C/C++ e Java, os incrementos e decrementos em PHP são feitos pelos operadores **++** e **--** respectivamente.

- **Exemplo:**

<?php

```
$c = 6;  
$b = $c++;  
echo($b); //imprime 6  
echo($c); //imprime 7
```

?>

- **Exemplo 2:**

<?php

```
$c = 6;  
$b = --$c;  
echo($b); //imprime 5  
echo($c); //imprime 5
```

?>



Estruturas de controle de fluxo

- As estruturas de controle de fluxo do PHP são **idênticas às do C/C++**.
- Em nosso curso serão estudadas as estruturas mais usadas do PHP, embora essas não sejam as únicas existentes:
 - ***Estruturas de seleção***
 - if
 - switch
 - ***Estruturas de repetição***
 - while
 - for



Estrutura de seleção - IF

<?php

```
if( condição )  
{
```

```
    //códigos executados quando atendida a condição
```

```
    //códigos executados quando atendida a condição
```

```
    //códigos executados quando atendida a condição
```

```
}
```

```
else
```

```
{
```

```
    //códigos executados quando não atendida a condição
```

```
    //códigos executados quando não atendida a condição
```

```
}
```

?>

■ Exemplo prático com IF



Exemplo comando IF

```
<html>
<head><title>Exemplo IF - Curso Desenvolvimento Web: PHP orientado à objetos</title></head>
<body>
  <form action="exemplo_if.php" method="post" name="teste_if">
    Dado 1:<input type="text" name="dado1" size="10"><br>
    Dado 2:<input type="text" name="dado2" size="10"><br>
    <input type="submit" value="Calcular">
  </form>
  <?php
    if($_POST[dado1] != "" && $_POST[dado2] != "")    //testa se o usuário digitou os dados
    {
      $total = $_POST[dado1] + $_POST[dado2];        //soma os dados digitados
      echo("O total da sua soma é: ".$total."<br>");    //imprime o total

      if($total > 15)
      {
        echo("O total da sua soma é MAIOR que 15!<br>");
      }
      else
      {
        echo("O total da sua soma é MENOR que 15!<br>");
      }
    } //end if
  ?>
</body>
</html> <!-- nome do arquivo exemplo_if.php -->
```



Estrutura de seleção - SWITCH

<?php

```
switch ($variavel)
```

```
{
```

```
    case valor1:
```

```
        //códigos executados quando $variavel igual valor1
```

```
        break;
```

```
    case valor2:
```

```
        //códigos executados quando $variavel igual valor2
```

```
        break;
```

```
    default:
```

```
        //códigos executados quando $variavel diferente de valor1 e valor2
```

```
}
```

?>

■ Exemplo prático com SWITCH



Exemplo comando SWITCH

```
<html>
<head><title>Exemplo SWITCH - Curso Desenvolvimento Web: PHP orientado à objetos</title></head>
<body>
  <form action="exemplo_switch.php" method="post" name="teste_switch">
    Dado:<input type="text" name="dado" size="10"><br>
    <input type="submit" value="Enviar">
  </form>
  <?php
    if($_POST[dado] != "")          //testa se o usuário digitou o dado
    {
      switch($_POST[dado])          //a variável de parâmetro é o dado digitado
      {
        case "Lucas":
          echo("Dado digitado: ".$_POST[dado].". Esse cara é gente boa!<br>");
          break;

        case "PHP":
          echo("Dado digitado: ".$_POST[dado].". Essa linguagem é massa!<br>");
          break;

        default:
          echo("Dado digitado: ".$_POST[dado].". Dado não previsto!<br>");
      }
    }
  <?>
</body>
</html> <!-- nome do arquivo exemplo_switch.php -->
```



Estrutura de repetição - WHILE

```
<?php
while( condição )
{
    //códigos executados enquanto atendida a condição
    //códigos executados enquanto atendida a condição
    //códigos executados enquanto atendida a condição
}
?>
```

■ Exemplo prático com WHILE



Exemplo comando WHILE

```
<html>
<head><title>Exemplo WHILE - Curso Desenvolvimento Web: PHP orientado à
objetos</title></head>
<body>
  <form action="exemplo_while.php" method="post" name="teste_while">
    Dado:<input type="text" name="dado" size="10"><br>
    <input type="submit" value="Enviar">
  </form>
  <?php
    if($_POST[dado] != "")          //testa se o usuário digitou o dado
    {
      $aux = $_POST[dado];          //variável auxiliar recebe o valor digitado

      while($aux <= 50)
      {
        echo("Variável auxiliar: ".$aux."<br>");
        $aux += 2;                  //soma mais 2 ao valor atual da variável auxiliar
      }
    }
  }
  <?>
</body>
</html> <!-- nome do arquivo exemplo_while.php -->
```



Estrutura de repetição - FOR

```
<?php
```

```
for( inicia o contador ; condição ; incrementa ou decrementa contador )
```

```
{
```

```
    //códigos executados enquanto atendida a condição
```

```
    //códigos executados enquanto atendida a condição
```

```
    //códigos executados enquanto atendida a condição
```

```
}
```

```
?>
```

■ Exemplo prático com FOR




```

<html>
<head><title>Exemplo FOR - Curso Desenvolvimento Web: PHP orientado à objetos</title></head>
<body>
  <form action="exemplo_for.php" method="post" name="teste_for">
    Nº de linhas:<input type="text" name="linhas" size="10"><br>
    <input type="submit" value="Enviar">
  </form>
  <?php
    if($_POST[linhas] != "")      //testa se o usuário digitou o dado
    {
      $aux = $_POST[linhas];      //variável auxiliar recebe o valor digitado
      echo("<table border='1'>
        <tr>
          <th colspan='2' bgcolor='#bbbbbb'>Tabela Dinâmica</th>
        </tr>");
      //inicia uma tabela HTML com cabeçalho
      for($i = 0; $i < $aux; $i++)
      {
        if($i % 2 == 0)           //testa se o valor do contador é par
        {
          $cor = "#ffffff";       //atribui valor hexadecimal de cor
        }
        else
        {
          $cor = "#bbbbbb";       //atribui valor hexadecimal de cor
        }
        echo("<tr><td bgcolor='$cor'>$i</td><td bgcolor='$cor'>$i</td></tr>");
        //escreve linha da tabela com 2
        células
      }
      //end for
      echo("</table>"); //encerra tabela HTML
    }
    //end if
  <?>
</body>
</html> <!-- nome do arquivo exemplo_for.php -->

```

Exemplo comando FOR



Funções em PHP

- Assim como em praticamente todas as linguagens de programação, PHP tem suporte a criação de funções.
- Uma função é uma “caixa preta” que realiza uma tarefa sempre que ela for chamada.
- A sintaxe da criação das funções é **idêntica à do C/C++**
- Uma função pode retornar ou não valores, podendo assim ser moldada pela necessidade do programador.
- Os parâmetros das funções podem ser passados por valor ou referência.



Criação de Funções em PHP

- Função com passagem de parâmetro por valor:

```
function nome_da_funcao ($parametro1, $parametro2)
{
    //comandos executados pela função
    return valor_retornado;
}
```

- Função com passagem de parâmetro por referência:

```
function nome_da_funcao (&$parametro1, &$parametro2)
{
    //comandos executados pela função
    return valor_retornado;
}
```



Chamada de função em PHP

<?php

```
function soma($a, $b)
{
    $total = $a + $b;
    return $total;
}
```

```
$valor1 = 2;
$valor2 = 3;
```

```
$resultado = soma($valor1,$valor2);    //chama a função soma
echo($resultado);                      //imprime 5 na tela
```

?>



Praticando funções em PHP

- Para visualizar melhor a utilização das funções em PHP, vamos fazer um **exemplo prático** envolvendo duas funções, uma utilizando passagem de parâmetros por valor e outra utilizando a passagem de parâmetro por referência.

Exemplo Prático com funções



Exemplo de Funções PHP

```
<html>
<head><title>Exemplo FUNÇÕES - Curso Desenvolvimento Web: PHP orientado à objetos</title></head>
<body>
  <form action="exemplo_funcoes.php" method="post" name="teste_funcoes">
    Dado:<input type="text" name="dado" size="10"><br>
    <input type="submit" value="Enviar">
  </form>
  <?php
    function parametro_valor($parametro)
    {
      $parametro = "Seu valor foi alterado!";
    }
    function parametro_referencia(&$parametro)
    {
      $parametro = "Seu valor foi alterado!";
    }

    if($_POST[dado] != "")          //testa se o usuário digitou o dado
    {
      $aux = $_POST[dado];          //variável auxiliar recebe o valor digitado
      parametro_valor($aux);
      echo("Valor da variável \$aux depois de chamar a função parametro_valor( ): $aux<br>");
      parametro_referencia($aux);
      echo("Valor da variável \$aux depois de chamar a função parametro_referencia( ): $aux<br>");
    }
  <?>
</body>
</html> <!-- nome do arquivo exemplo_funcoes.php -->
```



Funções próprias do PHP

- Assim como em outras linguagens de programação, o PHP possui uma vasta quantidade de funções próprias da linguagem para executar funcionalidades diversas. Em nosso curso veremos apenas algumas dessas funções.
 - ***Funções do PHP que serão vistas no curso:***
 - `date()`;
 - `is_numeric()`;
 - `strlen()`;
 - `substr()`;
 - `explode()`;



Função date()

- A função date(); do PHP retorna data ou hora local(do servidor), depende dos parâmetros que ela recebe na chamada;
- A função date(); recebe como parâmetros valores String pré-determinados e a partir desses parâmetros ela retorna valores correspondentes.

- **Exemplo:**

```
<?php
```

```
$dt = date("d/m/Y"); //variavel $dt recebe o retorno da função  
echo($dt); //apresentará na tela a data atual ex: 03/06/2009
```

```
?>
```



Alguns parâmetros para a função date()

Caractere de format	Descrição	Exemplo de valores retornados
Day	---	---
<i>d</i>	Dia do mês, 2 dígitos com preenchimento de zero	01 até 31
<i>D</i>	Uma representação textual de um dia, três letras	Mon até Sun
<i>j</i>	Dia do mês sem preenchimento de zero	1 até 31
<i>l</i> ('L' minúsculo)	A representação textual completa do dia da semana	Sunday até Saturday
Mês	---	---
<i>F</i>	Um representação completa de um mês, como January ou March	January até December
<i>m</i>	Representação numérica de um mês, com leading zeros	01 a 12
<i>M</i>	Uma representação textual curta de um mês, três letras	Jan a Dec
<i>n</i>	Representação numérica de um mês, sem leading zeros	1 a 12
Year	---	---
<i>Y</i>	Uma representação de ano completa, 4 dígitos	Exemplos: 1999 ou 2003
<i>y</i>	Uma representação do ano com dois dígitos	Exemplos: 99 ou 03

Lista completa de parâmetros: http://www.php.net/manual/pt_BR/function.date.php



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Função `is_numeric()`

- Essa função recebe como parâmetro uma variável e retorna **TRUE** se ela for um número ou **FALSE** se ela for uma string.

- *Exemplo:*

<?php

```
$aux = "Sou uma string!";           //variável recebe um valor String
if(is_numeric($aux) == true)
{
    echo("Ela é um número!");
}
else
{
    echo("Ela NÃO é um número!");
}
//NESSE CASO SERÁ IMPRESSO "Ela NÃO é um número!"
```

?>



Função strlen()

- Essa função recebe como parâmetro uma variável do tipo String e retorna o número de caracteres dela (tamanho).

- **Exemplo:**

```
<?php
```

```
$aux = "Lucas";           //variável recebe valor tipo String
```

```
echo(strlen($aux));       //será impresso 5;
```

```
?>
```



Função substr()

- Essa função recebe 3 parâmetros e retorna parte da String especificada.
 - O primeiro parâmetro que ela recebe é a String;
 - O segundo parâmetro é o valor referente ao início da parte desejada da String;
 - O terceiro parâmetro é referente ao número de caracteres da parte desejada da String;

- **Exemplo**

```
<?php
```

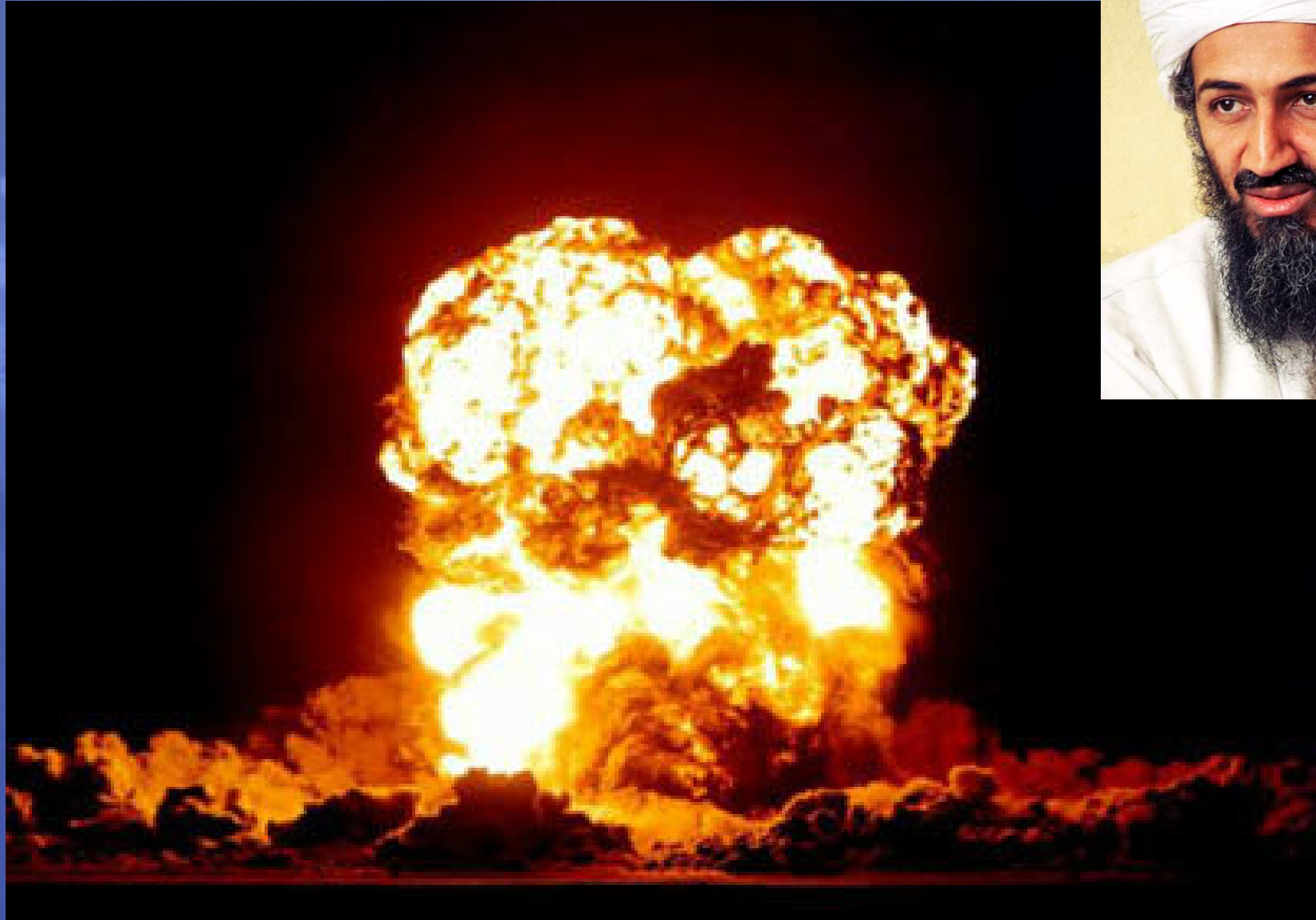
```
$aux = "Curso PHP!";  
echo( substr($aux, 0, 4) );
```

```
?>
```

```
//variável recebe valor String  
//será impresso Curso
```



Função explode()



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Função explode()

- Ela divide uma String em partes, retornando um array, sendo que cada posição desse array contém uma parte da String dividida.
- Ela recebe dois parâmetros:
 - O primeiro é o delimitador responsável pela divisão;
 - O segundo é a String que vai ser dividida;
- **Exemplo:**

<?php

```
$dt = "03/06/2009";           //variável recebe um valor String
$vet = explode("/", $dt);
echo ($vet[0]."<br>");          //imprime "03"
echo ($vet[1]."<br>");          //imprime "06"
echo ($vet[2]."<br>");          //imprime "2009"
```

?>



Inclusão de arquivos

- Em PHP é possível usar funções e variáveis contidas em arquivos diferentes, para isso basta incluir o arquivo onde está a função ou variável, no arquivo atual onde elas serão usadas.
- Essa técnica funciona como se fosse a inclusão de bibliotecas de código.
- Para fazer a inclusão de arquivos, o PHP oferece 4 funções especiais:
 - include;
 - include_once;
 - require;
 - require_once;



Inclusão de arquivos - **INCLUDE_ONCE**

- No nosso curso usaremos basicamente somente a função de inclusão **include_once** para evitar erros.
- Se você tentar incluir duas ou mais vezes um mesmo arquivo, é gerado um erro no interpretador PHP, pois é como se você tentasse incluir a mesma biblioteca de códigos mais de uma vez.
- O include_once é inteligente com relação a isso e evita esse erro, pois antes dele fazer a inclusão, ele testa se o arquivo que você está tentando incluir já está incluído. Se ele já estiver incluído, o include_once não irá incluí-lo novamente, mas caso não tiver incluído, ele o incluirá.

■ **Exemplo prático include_once**



Exemplo com *INCLUDE_ONCE*

<?php

//nome do arquivo "exemplo_include_once.php"

//ESSE ARQUIVO SERÁ INCLUIDO NO ARQUIVO exemplo_include_once_imprime.php

```
function imprime_data()
{
    $dt = date("d/m/Y");
    return $dt;
}
```

?>

<?php

//nome do arquivo "exemplo_include_once_imprime.php"

include_once 'exemplo_include_once.php'; //é passado o caminho do arquivo a ser incluído

echo(imprime_data()); //chamada da função que é declarada o arquivo que foi incluído

?>



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

- Módulo III -
PHP OO - Programação
Orientada à Objetos com PHP

Desenvolvimento web:
PHP Orientado à Objetos



Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Introdução a orientação à objetos

- A orientação à objetos é uma maneira de programar que trata de modelar os processos de programação de uma maneira próxima à realidade: tratando a cada componente de um programa como um **objeto real**, com suas respectivas características e funcionalidades.
- Um software orientado à objetos é composto basicamente de:
 - Classes
 - Métodos (construtor, set, get...)
 - Atributos
 - Objetos



Introdução a orientação à objetos

- **Exemplo de classe PHP:**

- Uma **classe pessoa** possui os **atributos** :
 - nome
 - cpf
 - rg
 - telefone
 - endereço
 - email
- Uma **classe pessoa** possui os **métodos** :
 - construtor
 - SETs
 - GETs
 - cadastrar
 - altera
 - exclui



PHP Orientado à Objetos

- A partir da versão 5, a orientação à objetos do PHP ficou praticamente completa, não deixando nada a desejar com relação as linguagens de programação voltadas para desktop.
- Dentre as funcionalidades que ela suporta, podemos citar Sobrescrita de função, Sobrecarga de função, Herança, Polimorfismo e Interfaces.
- Assim como no Java, PHP OO não suporta herança múltipla. A solução que os programadores fazem para compensar isso é a utilização de interfaces.



PHP Orientado à Objetos

- No nosso curso, a parte de orientação à objeto do PHP que será trabalhada, é composta basicamente de:
 - Criação de classe
 - Atributos
 - Métodos construtor
 - Encapsulamento
 - Modificadores de acesso
 - private
 - public
 - protected
 - Métodos Set
 - Métodos Get
 - Criação de objetos
 - Chamada de métodos
 - Herança



Criação de classes

- Exemplo de criação de **classe usuário**

```
<?php
class usuario
{
    //atributos da função
    //métodos da função
    ...
    ...
    ...
}
```

??>



Atributos

- Exemplo de criação de **atributos da classe usuário**

<?php

```
class usuario
```

```
{
```

```
    $idUsuario;        //atributo
```

```
    $nome;              //atributo
```

```
    $login;             //atributo
```

```
    $senha;             //atributo
```

```
    //métodos da função
```

```
}
```

?>



Métodos construtores

- Em PHP, os métodos construtores podem ter **dois** possíveis nomes:
 - mesmo nome da classe
 - `__construct`
- Exemplo de criação do **método construtor da classe usuário**

<?php

```
class usuario
```

```
{
```

```
    $idUsuario;    //atributo
```

```
    $nome;         //atributo
```

```
    $login;        //atributo
```

```
    $senha;        //atributo
```

```
function __construct() //criação do método construtor
```

```
{
```

```
}
```

```
//métodos da função...
```

```
}
```

?>



Encapsulamento

- É um processo no qual se ocultam as características internas de um objeto àqueles elementos que não têm porque conhece-las. Os **modificadores de acesso** servem para indicar as permissões que terão outros objetos para acessar a seus métodos e atributos.
- **Modificadores de acesso em PHP:**
 - public
 - private
 - protected



Modificadores de Acesso PHP

- **public :**

- É o modificador de acesso padrão, ou seja, quando não se define o modificador de acesso de um método ou atributo, esse é interpretado pelo PHP como public. É o modificador *mais permissivo*, pois os métodos ou atributos public podem ser acessados em qualquer parte do programa.

- **private :**

- É o modificador de acesso *mais restritivo*. Um atributo ou método private, só pode ser acessado pelo próprio objeto.

- **protected :**

- É um modificador de *restrição média*. Um atributo ou método protected, pode ser acessado pelo próprio objeto e por objetos das classes filhas (que o herdou).



Modificadores de Acesso PHP - Exemplo

- Exemplo de **encapsulamento** (uso de **modificadores de acesso**)

<?php

```
class usuario
```

```
{
```

```
    private $idUsuario;        //atributo privado
```

```
    private $nome;            //atributo privado
```

```
    private $login;           //atributo privado
```

```
    private $senha;           //atributo privado
```

```
    public function __construct() //criação do método construtor público
```

```
{
```

```
}
```

```
    //métodos da função...
```

```
}
```

?>



Métodos SET

- Cada **atributo private** de uma classe, deve possuir um **método public SET** correspondente para que possam ser atribuídos valores a ele através desse método.

<?php

```
class usuario
{
    private $idUsuario;    //atributo privado
    private $nome;         //atributo privado
    private $login;        //atributo privado
    private $senha;        //atributo privado

    public function __construct()    //criação do método construtor público
    {

    }

    public function setNome($n)
    {
        $this->nome = $n;    //atributo $nome recebe o valor referente a $n
    }
}
```

?>



Métodos GET

- Cada **atributo private** de uma classe, deve possuir um **método public GET** correspondente para que possam ser recuperados seus valores através desse método.

<?php

```
class usuario
{
    private $idUsuario;    //atributo privado
    private $nome;         //atributo privado
    private $login;        //atributo privado
    private $senha;        //atributo privado

    public function __construct()    //criação do método construtor público
    {

    }

    public function getNome()
    {
        return $this->nome;    //retorna valor do atributo $nome
    }
}
```

?>



Criação de objetos

Exemplo de criação de objeto da **classe usuario**:

```
<?php
```

```
//nome do arquivo cria_objeto.php
```

```
include_once "usuario.class.php"; //inclui classe usuario
```

```
$objeto = new usuario(); //cria objeto da classe pessoa
```

```
?>
```



Chamada de métodos

Exemplo de chamada de métodos da **classe** **usuario**:

```
<?php
```

```
//nome do arquivo cria_objeto.php
```

```
include_once "usuario.class.php"; //inclui classe usuario
```

```
$objeto = new usuario(); //cria objeto da classe pessoa
```

```
$aux = "Lucas Vegi"; //variável auxiliar recebe nome
```

```
$objeto->setNome($aux); //chama método setNome
```

```
echo( $objeto->getNome() ); //chama método getNome
```

```
?>
```



Herança em PHP

- Com a Herança, as classes podem herdar as características (atributos e métodos) de outras, de modo que se podem fazer **classes especializadas**, baseadas em outras mais gerais.
- *Exemplo de uma classe herdando outra:*

<?php

```
include_once "usuario.class.php"; //código da classe pai deve ser incluído na filha
class admin extends usuario      //classe admin herda da classe usuario
{
    private $setor;
    public function __construct()
    {
        parent::__construct();    //chama construtor da classe pai
    }
}
```

?>



Herança em PHP

- Exemplo de chamada de métodos da **classe admin** que foram **herdados da classe usuario**:

<?php

```
//nome do arquivo cria_objeto_heranca.php
```

```
include_once "admin.class.php"; //inclui classe admin
```

```
$objeto = new admin(); //cria objeto da classe admin
```

```
$aux = "Lucas Vegi"; //variável auxiliar recebe nome
```

```
$objeto->setNome($aux); //chama método setNome
```

```
echo( $objeto->getNome() ); //chama método getNome
```

?>



Exemplo de uma aplicação em PHP OO

- A partir de agora começaremos a desenvolver uma aplicação chamada Blog' IS.
- Essa aplicação basicamente tem a funcionalidade de um blog onde um usuário poderá postar mensagens.
- Até o término desse curso, essa aplicação será aperfeiçoada, ganhando cada vez mais funcionalidades.

Mãos a obra!



<?php

```
class post //arquivo post.class.php
{
    private $titulo;
    private $texto;
    private $momento;

    public function __construct()
    {

    }

    public function getTitulo()
    {
        return $this->titulo;
    }

    public function getTexto()
    {
        return $this->texto;
    }
}
```



//continuação da classe.....

```
public function getMomento()
{
    return $this->momento;
}

public function setTitulo($tit)
{
    $this->titulo = $tit;
}

public function setTexto($tex)
{
    $this->texto = $tex;
}

public function setMomento()
{
    $this->momento = date("H:i:s");
}
?>
```


<?php

//arquivo home.php

```
include_once 'post.class.php';
session_start();
echo("
<html>
<head><title>Blog'IS - Information System</title></head>
<body>");
echo("
<table align='center' border='0' width='780' cellspacing='5'>
  <tr>
    <td align='center' colspan='2'><a href='home.php'><img src='blogis.jpg'
border='0'/></a></td>
  </tr>
  <tr>
    <td align='center' bgcolor='#86a7f6'><font face='verdana' size='3' color='#FFFFFF'><b>Seja
bem vindo</b></font></td>
  </tr>
  <tr>
    <td align='center' colspan='2'><form action="" method='POST'>
  <tr>
    <td align='center'><input type='text' name='titulo'></td>
  </tr>
  <tr>
    <td align='center'><textarea name='texto'></textarea></td>
  </tr>
```



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

```

<tr>
    <td align='center'><input type='submit' value='POSTAR!'></td>
</tr> </form>
");
if(isset($_POST['titulo']) && isset($_POST['texto']))
{
    $postagem = new post();
    $postagem->setTitulo($_POST['titulo']);
    $postagem->setTexto($_POST['texto']);
    $postagem->setMomento();
    echo("<tr>
        <td align='center' bgcolor='#86a7f6'><font face='verdana' size='3'
color='#FFFFFF'><b>Postagens</b></font></td>
    </tr>");
    echo("<table border=0 align='center' cellspacing='0' cellpadding='0'>");
    echo("    <tr><td align='center'><font size='5' face='arial'><b><u>".$_postagem->getTitulo().
</u></b></font><br><br></td></tr>
        <tr><td align='center'><font size='3' face='arial'>".$_postagem->getTexto().
</font></td></tr>
        <tr><td align='center'><font size='2' face='arial'>horário - ".$_postagem-
>getMomento()."</font><br><hr><br><br></td></tr>
        </table>
    ");
}
echo("</table></body></html>");
?>

```



- Módulo IV -
***Solução para os dados
voláteis: Sessão***

***Desenvolvimento web:
PHP Orientado à Objetos***



Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Dados são mais voláteis em PHP

- Em PHP, assim que um script é executado pelo servidor e retornado para o cliente em forma de HTML, ***todos os valores contidos nas variáveis e objetos são perdidos***, pois eles são retirados da memória RAM dos servidores.
- Isso se deve ao fato de que um servidor web executa ao mesmo tempo milhares de scripts oriundos de toda parte do mundo, portanto eles devem ***otimizar o uso de memória RAM***, evitando o desperdício.
- *Tudo bem, mas e se eu quiser manter meus dados na memória ram dos servidores por mais tempo???*



Uso de SESSÃO

- Sessões são **vetores associativos** especiais do PHP que reservam um espaço na memória RAM dos servidores e assim **tornam os dados menos voláteis**.
- Os dados mantidos em uma sessão permanecem na memória RAM dos servidores enquanto o navegador do cliente(browser) estiver aberto ou enquanto a sessão não expirar por inatividade ou por vencer sua vida útil.



Iniciando SESSÃO

- Pra começar a trabalhar com uma sessão, você precisa primeiro aprender uma coisa. **Iniciar a sessão**. Mesmo que você ainda não saiba o que vai colocar nela. Vamos por partes.

```
<?php
```

```
    session_start();           //inicia a sessão
```

```
?>
```



Tempo de vida de uma SESSÃO

- Um sessão, por padrão tem o **TTL** (time to live ou tempo de vida) de **180 minutos**. Se você precisa por algum motivo que sua sessão expire em menor ou maior tempo, use a função **session_cache_expire(minutos);**

<?php

```
session_start();           //inicia a sessão  
session_cache_expire(10); //sessão dura 10 minutos
```

?>



Registrando uma SESSÃO

- Digamos que você quer manter algum dado na sessão, para posterior uso.

Vamos para um exemplo prático.

```
<?php
```

```
session_start();           //inicia a sessão  
$nome = "Lucas Vegi";      //variável recebe valor string
```

```
$_SESSION['nome'] = $nome;  //registra sessão
```

```
echo("Valor da sessão: " . $_SESSION['nome']); //imprime valor
```

```
?>
```



Verificando se uma SESSÃO existe

- O PHP oferece um método especial para verificar se uma sessão foi registrada, esse método chama-se **isset()** e retorna um valor **boolean**.

<?php

```
session_start();           //inicia a sessão
```

```
if( isset($_SESSION['nome']) == true) //testa se sessão existe
{
    echo("Sessão existe!");
}
else{ echo("Sessão não existe!"); }
```

?>



Destruindo uma SESSÃO

- Agora que você sabe como registrar, expirar, verificar se existe, vamos **destruir** uma sessão. Para isso usaremos o método especial do PHP chamado **unset()**

<?php

```
session_start();           //inicia a sessão
```

```
unset($_SESSION['nome']);   //destroi a sessão
```

?>



Implementação do Blog'IS com sessão

- A mesma aplicação desenvolvida anteriormente, agora será aperfeiçoada consideravelmente com a utilização de sessão e dessa forma ficará muito mais fácil entender esses conceitos.
- O sistema será composto das classes:
 - post
 - usuario
- O sistema se comportará da seguinte forma:
 - **Caso o usuário estiver logado:**
 - Ele deve mostrar o nome do usuário logado,
 - Mostra a opção de postar conteúdo
 - Mostra postagens já realizadas
 - **Caso contrário:**
 - Mostra a opção "cadastrar".



<?php

class post

```
{
    private $titulo;
    private $texto;
    private $momento;

    public function __construct()
    {

    }

    public function getTitulo()
    {
        return $this->titulo;
    }

    public function getTexto()
    {
        return $this->texto;
    }

    public function getMomento()
    {
        return $this->momento;
    }
}
```

```
public function setTitulo($tit)
{
    $this->titulo = $tit;
}

public function setTexto($tex)
{
    $this->texto = $tex;
}

public function setMomento()
{
    $this->momento = date("H:i:s");
}

public function publicar()
{
    $_SESSION['posts'][$_SESSION['contador_posts']]['titulo'] = $this->titulo;
    $_SESSION['posts'][$_SESSION['contador_posts']]['texto'] = $this->texto;
    $_SESSION['posts'][$_SESSION['contador_posts']]['momento'] = $this->momento;
    $_SESSION['contador_posts']+=1;
}
}
```

//arquivo post.class.php

?>



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com


```
<?php
```

```
class usuario
{
    private
    $nome;
    $email;
    function __construct()
    {
    }

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

    public function setNome($n)
    {
        $this->nome = $n;
    }

    public function getEmail()
    {
        return $this->email;
    }
}
```

```
public function setEmail($n)
```

```
{
    $this->email = $n;    //atributo $email recebe o valor referente a $n
}

public function cadastrar()
{
    session_start();
    $_SESSION['usuario']= $this->nome;
    $_SESSION['email']= $this->email;
    $_SESSION['contador_posts']=0;

    echo "<tr>
        <td align='center' bgcolor='#86a7f6' width='50%'><a href='me-cadastrar.php'
style='text-decoration: none;'><font face='verdana' size='3' color='#FFFFFF'><b>Me
cadastrar!</b></font></a></td>
        <td align='center' bgcolor='#86a7f6' width='50%'><a href='me-logar.php'
style='text-decoration: none;'><font face='verdana' size='3'
color='#FFFFFF'><b>Logar</b></font></a></td>
    </tr>
    <tr>
        <td align='center' colspan='2'><br><br><br><font face='verdana' size='3'
color='#7097f2'><b>Usuário cadastrado com sucesso!</b></font></td>
    </tr>
    <tr>
        <td align='center' colspan='2'><a href='home.php'><font face='verdana'
size='3' color='#0344e1'><b>Faça já os seus POSTs!</b></font></a></td>
    </tr>
    ";
}
} //arquivo usuario.class.php
```

```
?>
```



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

```
<?php
//nome do arquivo me-cadastrar.php

include_once "usuario.class.php";
session_start();
echo("
<html>
<head><title>Blog'IS - Information System</title></head>
<body>");
echo("
<table align='center' border='0' width='780' cellspacing='5'>
  <tr>
    <td align='center' colspan='2'><a href='home.php'><img src='blogis.jpg' border='0'/></a></td>
  </tr>
");

if($_SESSION['usuario'])
{
  session_destroy();
  session_start();
}

if(($_POST['nome']) && ($_POST['email']))
{
  $user = new usuario();
  $user->setNome($_POST['nome']);
  $user->setEmail($_POST['email']);
  $user->cadastrar();
}
```



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

```

else
{
    echo("
        <tr>
            <td align='center' bgcolor='#86a7f6' width='50%'><a href='me-cadastrar.php' style='text-decoration:
none;'><font face='verdana' size='3' color='#FFFFFF'><b>Me cadastrar!</b></font></a></td>
            <td align='center' bgcolor='#86a7f6' width='50%'><a href='#' style='text-decoration: none;'><font
face='verdana' size='3' color='#FFFFFF'><b>Logar</b></font></a></td>
        </tr>
        <tr>
            <form action='' method='POST'>
                <td align='center' colspan='2'><br><br><br><font face='arial' size='2'><b>Nome:</b></font>&nbsp;
&nbsp;<input type='text' size='20' name='nome'></td>
            </tr>
            <tr>
                <td align='center' colspan='2'><font face='arial' size='2'><b>Email:</b></font>&nbsp;&nbsp;<input
type='text' size='20' name='email'></td>
            </tr>
            <tr>
                <td align='center' colspan='2'><input type='submit' value='Cadastre-me'></td>
            </tr>
        </form>
    ");
}
echo("</table></body></html>");
//fim do arquivo me-cadastrar.php
?>

```



<?php

//nome do arquivo home.php

include_once 'post.class.php';

session_start();

echo("

<html>

<head><title>Blog'IS - Information System</title></head>

<body>");

echo("

<table align='center' border='0' width='780' cellpadding='5'>

<tr>

<td align='center' colspan='2'></td>

</tr>

");

if(\$_GET['opcao']=='sair')

{

unset(\$_SESSION);

session_destroy();

}



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

```

if(isset($_SESSION['usuario']))
{
    echo "<tr>
        <td align='center' bgcolor='#86a7f6'><font face='verdana' size='3' color='#FFFFFF'><b>Usuário logado:
</b><i>".$_SESSION['usuario'].</i> - <a href='?opcao=sair'>Sair</a></font></td>
    </tr>
    <form action="" method='POST'>
    <tr>
        <td align='center'><input type='text' name='titulo'></td>
    </tr>
    <tr>
        <td align='center'><textarea name='texto'></textarea></td>
    </tr>
    <tr>
        <td align='center'><input type='submit' value='POSTAR!'></td>
    </tr>
    </form>
    ";
}

if(isset($_POST['titulo']) && isset($_POST['texto']))
{
    $postagem = new post();
    $postagem->setTitulo($_POST['titulo']);
    $postagem->setTexto($_POST['texto']);
    $postagem->setMomento();
    $postagem->publicar();
}

```



```

if(isset($_SESSION['posts']))
{
    echo("<tr>
        <td align='center' bgcolor='#86a7f6'><font face='verdana' size='3'
color='#FFFFFF'><b>Postagens</b></font></td>
    </tr>");
    echo("<table border=0 align='center' cellpadding='0' cellspacing='0'>");
    for($i=0;$i<count($_SESSION['posts']);$i++)
    {
        echo "<tr><td align='center'><font size='5' face='arial'><b><u>".$_SESSION['posts'][$i]['titulo']."
</u></b></font><br><br></td></tr>
        <tr><td align='center'><font size='3' face='arial'>".$_SESSION['posts'][$i]['texto']."</font></td></tr>
        <tr><td align='center'><font size='2' face='arial'>publicado por ".$_SESSION['usuario']." - ".$_SESSION
['posts'][$i]['momento']."</font><br><hr><br><br></td></tr>";
    }
    echo("</table>");
}
else
{
    echo "<tr>
        <td align='center' bgcolor='#86a7f6' width='50%'><a href='me-cadastrar.php' style='text-decoration:
none;'><font face='verdana' size='3' color='#FFFFFF'><b>Me cadastrar!</b></font></a></td>
        <td align='center' bgcolor='#86a7f6' width='50%'><a href='#' style='text-decoration: none;'><font
face='verdana' size='3' color='#FFFFFF'><b>Logar</b></font></a></td>
    </tr>";
}
echo("</table></body></html>");          //fim do arquivo home.php
?>

```



Desenvolvimento web: PHP Orientado à Objeto
 Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Mesmo com o uso de SESSÃO os dados continuam voláteis

- Vocês devem ter observado que mesmo com o uso de sessão, os dados continuam voláteis após o fechamento do navegador.
- Isso se deve ao fato de que a Sessão simula no servidor, uma área memória equivalente à memória RAM desktop, que é volátil. Para resolver essa limitação e guardar os dados em definitivo, somente com o uso de um Banco de Dados.
- Além de guardar os dados em definitivo, o uso do banco de dados proporcionaria que os usuários tivessem acesso a todas as postagens de todos os usuários e não somente às suas.



Dê sua opinião sobre o curso!

- Entrem por favor no link abaixo e respondam algumas poucas perguntas com relação ao curso. Sua contribuição será muito importante!

[http://spreadsheets.google.com/viewform?
formkey=ckx2WIVhaU5Pd2dwX2FOVjFiNEhhYIE6MA..](http://spreadsheets.google.com/viewform?formkey=ckx2WIVhaU5Pd2dwX2FOVjFiNEhhYIE6MA..)



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Contatos

lucasvegi@gmail.com



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com

Referências Bibliográficas

- ALVES, L. L; BITTENCOUT, F. R. PHP: Conceitos Essenciais para Implementação de Aplicações Web. 7 Faces, Itabira, MG, v. 4, p. 193-208, 2003.
- <http://apostilas.fok.com.br/manual-do-php/reserved.php>
- <http://www.php.net>
- <http://www.revistaphp.com.br/artigo.php?id=79>



Desenvolvimento web: PHP Orientado à Objeto
Lucas Francisco da Matta Vegi
www.mr-bin.blogspot.com