Apostila de
PHP

Todo material desenvolvido pela Coordenadoria do Governo
Eletrônico é resultado de um processo coletivo de produção,
que se iniciou em 2001 e que é permanente.
Agradecemos a todos que colaboraram e que queiram contribuir.

CGE COORDENADORIA DO GOVERNO ELETRÔNICO
Equipe de Treinamento Técnico

Aparecido Quesada
Adriana Tosta
Eder Moura Dourado
Simone Leal dos Santos
Thyago Akira de Morais Ribeiro
Yuri Robinson de Souza

Contato
treinamento_cge@prefeitura.sp.gov.br
telecentros@prefeitura.sp.gov.br

PALÁCIO DO ANHANGABAÚ
VIADUTO DO CHÁ Nº 15
CEP 01002-000 SÃO PAULO
TEL: 3113-8938 FAX 3113-8939

Índice

O que é PHP?

A abreviação PHP vem de ‘’Hypertext PreProcessor’’, que é uma linguagem de programação de código aberto muito utilizada na criação de scripts, que são executados no servidor Web para a manipulação de páginas HTML.

Criar um site com banco de dados se torna uma tarefa muito simples com o PHP. Os bancos de dados atualmente suportados pelo PHP são: Adabas D, dBase, mSQL, InterBase, SyBase, Empress, MySQL, Velocis, FilePro, Oracle, dbm, Informix e Psotgress.

A diferença do PHP para outras linguagens semelhantes, como o JavaScript, é que o código PHP é executado no servidor, sendo enviado para o cliente apenas HTML puro.

Apesar de ser mais utilizado em aplicativos para a Web, o PHP também suporta programação no modo texto e aplicações gráficas para serem executadas em interfaces gráficas com o PHP-GTK.

História

O PHP foi criado em 1994 por Rasmus Lerdorf, que inicialmente o utilizava em sua home page pessoal (Personal Home Page, foi o primeiro nome dado à linguagem). Em 1995 ele passou a ser utilizado por outras pessoas e foi reescrito com novos recursos, sendo renomeado para Personal Home Page Tools/FI (Form Interpreter).

Dois anos depois, o PHP deixou de ser um projeto pessoal de Rasmus Lerdorf e passou a ser desenvolvido por uma equipe de colaboradores, que lançou a versão 3 da linguagem. Atualmente o uso do PHP 4 vem crescendo numa velocidade incrível, e já está sendo desenvolvida a versão 5 do PHP.

Vantagens

O PHP possui inúmeras vantagens, que você verá a seguir:

. É uma linguagem de fácil aprendizado;
. Tem performance e estabilidade excelentes;
. Seu código é aberto e seu uso é livre e gratuito. É possível alterá-lo na medida da necessidade de cada usuário;
. Tem suporte para os principais servidores Web, e suporte nativo para o servidor Web Apache;
. Suporta conexões com os bancos de dados mais utilizados do mercado, como por exemplo MySQL, PostgreSQL, Oracle entre outros.
. É multiplataforma. Tem suporte aos sistemas operacionais mais utilizados no mercado.
. Segurança: O usuário não consegue ver o código PHP, somente o HTML. Isto é importante quando se está trabalhando com senhas.

Comunicação Cliente x Servidor Web

Quando é digitado um endereço no navegador para acessar uma página na Internet, o que acontece é uma requisição do cliente (navegador) ao servidor Web. O servidor processa essa requisição e retorna uma resposta ao cliente, que por sua vez interpreta o código retornado e formata a página para a sua visualização. Esse procedimento acontece em todas as requisições feitas pelo navegador.

Portanto programar para a web pode ser considerado como um jogo que consiste em receber os dados do usuário, processá-los e enviar a resposta dinâmica. Uma vez enviada a resposta, é encerrado o contato entre o servidor e o cliente.

Neste curso o servidor Web que você usará, será o Apache.

Visualização:

Configuração

No curso você usará como sistema operacional a distribuição Debian GNU/Linux 3.0, e os pacotes necessários para o andamento do curso já se encontram instalados e configurados.

Caso queira utilizar esta apostila num outro local siga os requisitos e instruções abaixo:

Hardware:

PC 386 ou superior
16 MB de memória RAM
100 MB livres em disco (PHP + MySQL + Apache)

Software:

Distribuição Linux de sua preferência.
Apache 1.3 ou acima.
MySQL 4.0 ou acima.
PHP 4 .0 ou acima.

Atualmente muitas distribuições Linux já possuem o ambiente PHP+MySQL+APACHE instalados, porém, caso não estejam disponíveis é possível instalar e configurar o ambiente manualmente, seguindo os passos abaixo.

Instalação do MySQL

A instalação do MySQL é simples. Basta seguir corretamente os passos à seguir:

1- Baixe o MySQL no site www.mysql.com/downloads.

2- Em um terminal, entre como root e crie um grupo de usuários chamado mysql.
#groupadd mysql

3- Adicione um novo usuário mysql relacionado ao grupo mysql.
#adduser -g mysql mysql

4- Vá para o diretório onde será feita a instalação (em geral / usr/local).

5- Descompacte e extraia o pacote do MySQL.
#tar -xvzf /<diretorio>/mysql-VERSAO-OS.tar.gz

em que <diretorio> é o local em que está o arquivo tar e VERSAO-OS é a versão e o sistema operacional do pacote (um exemplo seria /usr/src/gz/mysql-3.23.35-pc-linux-gnu-i686.tar.gz).

6- Altere o nome do diretório de mysql-VERSAO-OS para mysql.
#mv mysql-VERSAO-OS mysql

7- Vá para o diretório mysql recém criado.
#cd mysql

8- Execute o script de instalaçao do banco de dados.
#scripts/mysql_install_db

9- Altere os donos e os grupos dos diretórios do MySQL.
#chown -R root /usr/local/mysql
#chown -R mysql /usr/local/mysql/var
#chgrp -R mysql /usr/local/mysql
#chown -R root /usr/local/mysql/bin

10- Inicialize o MySQL.
#bin/safe_mysqld -user=mysql &

Maiores detalhes sobre instalação e versões podem ser obtidos no site do MySQL (www.mysql.com) na seção Documentation.

INSTALAÇÃO DO APACHE E DO PHP

1- Baixe os arquivos dos sites www.apache.org e www.php.net.

2- Vá para o diretório onde serão descompactados os arquivos (/usr/src).

#cd /usr/src

3- Descompacte o apache e o PHP.
#tar -xvzf apache_1.3.x.tar.gz
#tar -xvzf php-4.x.x.tar.gz

4- Vá para o diretório do Apache e configure-o.

```
#cd apache_1.3.x
#./configure -prefix=/www
```

5- Vá agora para o diretório do PHP e proceda à instalação.

```
#cd ../php-x.x.x
#./configure -with-mysql -with-apache=../apache_1.3.x —
enable-tracks-vars -enable-trans-sid
```

Para uma lista completa de opções de configuração visite o site do PHP (www.php.net) e veja a documentação.

6- Compile e instale o PHP.

```
#make
#make install
```

7- Retorne ao diretório do Apache e configure-o para o PHP, executando então a compilação e a instalação do Apache com PHP.

```
#cd ../apache_1.3.x
#./configure -activate-module=src/modules/php4/libphp4.a
#make
#make install
```

8- Volte ao diretório do PHP e copie o arquivo php.ini.

```
#cd ../php-x.x.x
#cp php.ini-dist /usr/local/lib/php.ini
```

9- Vá até o diretório de configuração do apache e altere o arquivo httpd.conf, inserindo a seguinte linha:

```
Add type application/x-httpd-php    .php
```

10- Por fim, inicialize o Apache.

```
#cd /www/bin
#./apachectl start
```

Após seguir essas instruções todo o ambiente estará instalado.

Teste de Funcionamento

Você precisa fazer um teste para saber se o PHP está em funcionamento. Em um editor de textos qualquer, digite o código abaixo e salve como "teste.php" no diretório habilitado para o servidor Apache (pergunte ao seu instrutor em que local salvar seu documento). Depois abra o Mozilla e digite o endereço onde está o arquivo.

Ex:
http://localhost/pastahabilitada/teste.php.

```
<?php
phpinfo();
?>
```

Deverá surgir a configuração atual do PHP. Veja a figura abaixo.

Sintaxe Básica do PHP

O PHP tem uma sintaxe muito simples, o que facilita muito a organização dos programas a serem desenvolvidos. Outro ponto interessante que você verá é que os códigos em PHP são embutidos no HTML, facilitando muito a análise de possíveis erros nos programas desenvolvidos. A seguir, exemplos da sintaxe do PHP:

<?php	<?	<script language=''PHP''>
...	...	...
...	...	...
?>	?>	</script>

Tudo que estiver delimitado por <? e ?> será processado no servidor. O navegador cliente receberá apenas o resultado do processamento.

Entre cada instrução em PHP é necessário utilizar o ponto-e-vírgula para finalizá-la (caso contrário, ocorrerão erros na execução do script). Na última instrução do bloco de script não é necessário o uso do ponto-e-vírgula, mas por questões estéticas recomenda-se o uso sempre. Tome sempre bastante cuidado para não esquecer de finalizar as instruções com o ponto-e-vírgula.

Um script PHP geralmente tem como resultado uma página html, ou algum outro texto. Para gerar esse resultado, deve ser utilizada uma função de impressão. A função echo envia ao navegador cliente o conteúdo tal como a forma abaixo:

echo “Texto à ser enviado ao navegador cliente”;

O texto dentro das aspas duplas será enviado ao navegador do cliente.

Exemplo de código PHP:

```
<html>
<head>
<title>Script PHP</title>
</head>
<body>
<?php
  echo “Olá, eu sou um script PHP!!!”;
?>
</body>
</html>
```

Organizando os seus Programas

Quando você começa a codificar a ansiedade de terminar o programa é grande. Porém, à medida que esse código cresce, se você não tomar certas providências, ficará totalmente perdido. Uma boa prática é inserir comentários em seus programas. Outra maneira é criar pequenos módulos (includes, explicados mais adiante) com os conteúdos que você sempre utiliza. Depois você insere esses módulos apenas no momento que for usar.

COMENTÁRIOS

Há dois tipos de comentários no PHP, de uma linha e com mais de uma linha.

A) COMENTÁRIOS DE UMA LINHA

Marca como comentário até o final da linha ou até o final do bloco de comandos PHP.

Pode ser delimitado pelo caracter # ou por duas barras (//). Veja o exemplo abaixo:

```
<? echo teste;   # Isto é um comentário.
   echo teste;   // É similar ao anterior.
?>
```

B) COMENTÁRIOS COM MAIS DE UMA LINHA

Tem como delimitadores os caracteres /* para o início do bloco e */ para o final do comentário. Se o delimitador de final de código do PHP (?>) estiver dentro de um comentário, não será reconhecido pelo interpretador. Veja os exemplos abaixo:

```
<?
   echo teste;  /* Isto é um comentário com mais
de uma linha, mas não funciona corretamente ?>
*/

<?
   echo teste;  /* Isto é um comentário com mais
de uma linha que funciona corretamente
*/
?>
```

EXERCICÍOS

Veja no exercício a seguir como o PHP interege com o HTML.

Utilize um editor de texto puro (como o gedit ou o kedit), para digitar e testar o código abaixo. Salve na pasta indicada pelo seu instrutor com o nome de “ecoando.php”, abra um navegador e digite o endereço http://localhost/<endereço>/ecoando.php

Obs: onde estiver escrito <endereço> digite o local indicado pelo seu instrutor.

```
<html>
<title>Exercicio 1</title>
<body>
A primeira linha e <u>normal</u>. <br>

<? echo " A segunda linha ja e escrita pelo
<b>PHP</b>." ; ?>

<br>
Nisto, vem uma <font color=#FF0000>terceira mais
 complexa</font>... <br>

<? echo " E logo a <font color=#00FF00>seguir</font>,
a quarta <font color=#0000FF>aindamais</font>
complicada!"
 ; ?><br>

</body>
</html>
```

Variáveis

As variáveis do PHP sempre começam com $ e são declaradas quanto ao tipo (inteiro, string, etc) no momento em que é atribuído o seu valor, não sendo necessário indicar o nome e tipo da variável como em outras linguagens.

Trabalhar com variáveis em PHP é uma atividade simples, como você verá a seguir:

Não é necessário declarar as variáveis;

Para definir as variáveis, é necessário apenas colocar como primeiro caracter o '$', juntamente com a string referente ao nome da variável, e esta string deve começar com uma letra ou o caracter '_'.

O PHP é case sensitive, isto é, '$a' é diferente de '$A'. É aconselhável utilizar os nomes de variáveis com letras minúsculas, por causa das variáveis pré-definidas da linguagem, que são declaradas com maiúsculas;

Tipos de Dados Suportados

INTEGER (INTEIRO)

É utilizado para números inteiros. Veja como declarar uma variável do tipo inteiro:

```
$curso =  1000;      // número inteiro positivo
$curso = -1000;      // número inteiro negativo
```

DOUBLE (DUPLA PRECISÃO)

É utilizado para números reais, podendo fazer cálculos com grande precisão. Veja os exemplos abaixo:

```
$curso = 1.050;    // O ponto é o separador decimal
$curso = 52e3;     // Notação científica (equivale a 52000)
```

STRINGS

É utilizado para strings de caracteres. As strings podem ser delimitadas de duas maneiras:

Com aspas duplas (‘’ ‘’), todas as variáveis dentro da string serão resolvidas.

```
<?
$curso1 = 20;
$curso2 = ‘’curso1 é igual a $curso1’’;
echo $curso2;
// Ficará: $curso2 = ‘’curso1 é igual a 20’’
?>
```

Com o uso de apóstrofos (‘ ‘), a string permanece como aparece, sem substituições.

```
<?
$curso1 = 20;
$curso2 = ‘curso1 não é igual a $curso1’;
echo $curso2;
// Ficará: $curso2 = ‘’curso1 não é igual a $curso1’’
?>
```

BOOLEANO

É utilizado para valores verdadeiros (True) ou falsos (False).

ARRAY

É utilizado para armazenar vários itens de dados do mesmo tipo.

OBJETO

É utilizado para armazenar instâncias de classes.

Caracteres de Escape

Os caracteres de escape começam com uma barra invertida (\) e são colocados dentro das strings. Internamente eles são substituídos pelos caracteres reais e pelas ações que esses caracteres simbolizam.

\n	Nova linha. Desce para a linha de baixo
\r	Retorno de carro (semelhante a \n). Coloca o cursor no começo da linha.
\t	Tabulação horizontal. Pula para a próxima tabulação.
\\	Barra Invertida. Substitui por uma barra invertida.
\$	Cifrão. Substitui por um cifrão.
\''	Aspas. Substitui por aspas.

EXERCÍCIOS:

No exercício a seguir é mostrada a forma como o PHP utiliza variáveis.

Utilizando um editor de texto puro salve o arquivo na pasta indicada pelo seu instrutor com o nome de “variaveis.php”, abra um navegador e digite o endereço http://localhost/<endereço>/variaveis.php

```
<html>
<title>Exercicio 2</title>
<body>

Exemplo de utilizacao de variaveis: <br><br>

<?
$inteiro=10;
$real=20.0;
$caracter= 'V';
$cor1= "#FF0000";
$cor2= "#0000FF";
?>

<font color= <? echo $cor1 ?>> A variavel $inteiro tem
o valor <? echo $inteiro ?>.
</font> <br>

<font color= <? echo $cor2 ?>> A variavel $real tem
o valor <? echo $real ?>.
</font> <br>

<font color= <? echo $cor1 ?>> O caracter escolhido
e o <? echo $caracter ?>.
</font> <br>

</body>
</html>
```

Operadores

ARITMÉTICOS

O PHP possui todos os operadores aritméticos.
Os principais são:

OPERADOR	SIGNIFICADO
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
%	Resto de divisão

STRING

O PHP possui um único operador de string, que é o operador de concatenação ‘’.’’ . Veja o exemplo abaixo:

```
<?
$x = ‘’Alô ‘’;
$s = $x.’’Mundo’’;
echo($s.’’\n’’);       // Imprimirá ‘’Alô Mundo’’ “
?>
```

ATRIBUIÇÃO

O único operador de atribuição do PHP é o ‘’=’’. Este, combinado com os operadores aritméticos e de string, pode reduzir o tamanho do código.

OPERADOR	SIGNIFICADO
=	Atribuição simples.
+=	Atribuição com adição.
-=	Atribuição com subtração.
*=	Atribuição com Multiplicação.
/=	Atribuição com divisão.
%=	Atribuição com módulo.
.=	Atribuição com concatenação.

Veja o exemplo abaixo.

```
<?
$curso = 7;
$curso += 2;   //($curso fica com o valor 9)
echo $curso;
?>
```

Veja outro exemplo utilizando a atribuição com módulo (resto de divisão):

```
<?
$resto = 11;
$resto %= 2;   //($resto fica com o valor 1)
echo $resto;
?>
```

No exemplo acima inicialmente a variável $resto possuí valor 11.
A operação " $resto %=2; " é equivalente à operação " $resto=$resto%2; " , isto é, as duas calculam o resto da divisão da variável $resto por 2.

Incrementação: Podem ser utilizados de duas formas: antes ou depois da variável. O incremento de uma variável soma 1 unidade à uma variável e armazena o resultado na mesma. O decremento subtraí uma variável em 1 unidade. Quando utilizado antes, retorna o valor da variável antes de incrementá-la ou decrementá-la. Quando utilizado depois, retorna o valor da variável já incrementado ou decrementado.

++ incremento
— decremento

Exemplos:

```
$a = $b = 10; // $a e $b recebem o valor 10
$c = $a++; // $c recebe 10 e $a passa a ter 11
$d = ++$b; // $d recebe 11, valor de $b já incrementado
$d++; // $d recebe + 1 unidade e passa a valer 12
```

LÓGICOS

Os operadores lógicos trabalham com os valores completos, utilizando TRUE ou FALSE.

OPERADOR	SIGNIFICADO	EXEMPLO
and	''e'' lógico	$a and $b
or	''ou'' lógico	$a or $b
!	Não (inversão)	!$b
&&	''e'' lógico	$a && $b
||	''ou'' lógico	$a || $b

COMPARAÇÃO

Os operadores de comparação são usados para comparar dois valores.

==	igual a
!=	diferente de
<>	Diferente de
<	menor que
>	maior que
<=	menor ou igual a
>=	maior ou igual a

Estruturas de Controle

If: O comando if testa a condição passada e executa o bloco de código caso o valor retornado da condição seja verdadeiro. Veja o exemplo abaixo:

```
$a = 1;

if ($a == 1)
{
...
...
...
}
```

Caso a condição passada retorne um valor falso, e seja necessário executar um bloco de código diferente, utiliza-se a instrução **else**. Veja o exemplo abaixo:

```
<?
$a = 1;
$b = 2;
if ($a > $b)
   {
    echo ''$a > $b'';
   }
else
   {
    echo ''$b > $a'';
   }
?>
```

Ainda existe a instrução **elseif**, para situações onde precisa-se verificar mais que uma condição. Veja o exemplo abaixo:

```
<?
$a = 1;
$b = 2;
$c = 3;
if ($a > $b)
  {
    echo “$a e&acute; maior que $b”;
  }
elseif ($a > $c)
  {
    echo “$a e&acute; maior que $c”;
   }
else
   {
    echo “$a e&acute; menor que $b e $c”;
   }
?>
```

switch: O comando switch atua de maneira semelhante a uma série de comandos if na mesma expressão. Freqüentemente, o programador pode comparar uma variável com diversos valores, e executar um código diferente dependendo de qual valor é igual ao da variável.

Quando isso for necessário, deve-se usar o comando switch. O exemplo seguinte mostra dois trechos de código que fazem a mesma coisa, sendo que o primeiro utiliza uma série de ifs e o segundo utiliza switch:

```
if ($i == 0)
   print i é igual a zero;
elseif ($i == 1)
   print i é igual a um;
elseif ($i == 2)
   print i é igual a dois;
}
switch ($i) {
case 0:
     print i é igual a zero;
     break;
case 1:
     print i é igual a um;
     break;
case 2:
     print i é igual a dois;
     break;
}
```

Expressão Condicional: Existe um operador de seleção que é ternário. Funciona assim:

(expressao1)?(expressao2):(expressao3)

O interpretador PHP avalia a primeira expressão. Se ela for verdadeira, a expressão retorna o valor de expressão2. Senão, retorna o valor de expressão3.

While: Este comando é utilizado para realizar laços de repetições condicionais. Ele executa o bloco de código enquanto a condição passada for verdadeira, e caso a condição inicial que foi passada se torne falsa, o bloco

não será executado. Veja o exemplo abaixo:

```
<?
echo “While”.”<br>”;
$a = 1;
while ($a <= 10)
 {
  echo “N&uacute;mero: “.$a++.”<br>”;
 }
?>
```

Do ... While: Este comando tem a mesma idéia que o comando while, porém, seu teste de condição é feito no final do bloco de código. Veja o exemplo abaixo:

```
<?
echo “Do...While”.”<br>”;
$c = 0;
do
  {
   echo “N&uacute;mero: “.++$c. “<br>”;
  } while ($c < 10);
?>
```

For: Como nos outros comandos que realizam laços condicionais, o comando for também precisa de uma condição para ser testada a cada laço realizado, porém, este comando necessita de mais dois parâmetros, que seriam a declaração da variável contadora e a instrução de incremento.

Veja o exemplo abaixo:

```
<?
echo “For...Next”.”<br>”;
for ($a=0; $a<=10; $a++)
    {
      echo “N&uacute;mero: “.$a.”<br>”;
    }
?>
```

EXERCICÍOS:

O exercício abaixo demonstra a utilização das estruturas de controle if, switch e while. Salve seu arquivo com o nome de “verifica.php” e digite o endereço http://localhost/<endereço>/verifica.php

```
<html>
<title>verifica.php</title>
<body>
<?
//Condicao de texto em italico
if ($italico== “on”)
{
$italico_abre= “<i>”;
$italico_fecha= “</i>”;
}
else
{
$italico_abre= “”;
$italico_fecha= “”;
}
```

```
//Escolha do alinhamento para texto em italico
switch($alinhamento)
{
//Texto com alinhamento a esquerda
case “esquerda”:
$alinhar= “left”;
break;
//Texto com alinhamento central
case “centro”:
$alinhar= “center”;
break;
//Texto com alinhamento a direita
case “direita”:
$alinhar= “right”;
break;
}
//Identificacao do numero de repeticoes
if ($vezes<1 or $vezes>30)
{
echo “Valor invalido, introduza um numero entre 1 e 30.<br>”;
}
else
{
//Ciclo para repetir o numero de vezes pretendido
$repetir=1;
while ( $repetir<=$vezes)
{
echo “<h$tamanho_hx align=$alinhar>$italico_abre<
font color=$cor> $repetir- $texto </font>$italico_fecha</
h$tamanho_hx>”;
$repetir++;
} //fecha o while
}
?>
</body>
</html>
```

Quebra de Fluxo

Break: O comando break pode ser utilizado em comandos de laços condicionais e sua função é parar imediatamente o fluxo do bloco de código. Veja o exemplo abaixo:

```
<?
$a = 20;
while ($a > 0)
      {
       if ($a == 5)
         {
           echo “N&uacute;mero inv&aacute;lido!”;
      break;
          }
        echo “N&uacute;mero  “.$a.”<br>”;
       $a—;
      }
?>
```

Continue: O comando continue também funciona dentro dos laços condicionais, porém, não pára o fluxo do bloco de código, e sim, volta para o início dele. Veja o exemplo abaixo:

```
<?
for ($a=0;$a<10;$a++)
   {
    if ($a == 5)
      {
        continue;
      }
    else
     {
        echo “Número: “.$a.”<br>”;
     }
   }
?>
```

Include

A função include coloca o conteúdo de um outro arquivo, com ou sem código em PHP, substituindo pelo novo código. O código do arquivo incluído é processado em tempo de execução, permitindo assim, que sejam usados ''includes'' dentro de estruturas de controle como for e while. Veja o exemplo abaixo.

Programa: Teste.php

```
<?
echo ''Teste A'';
echo ''Teste B'';
include ''Externo.php'';
echo ''Teste D'';
?>
```

Programa: Externo.php

```
<?
echo ''Teste C'';
?>
```

O programa com o comando include ficaria da seguinte forma:

```
<?
echo ''Teste A'';
echo ''Teste B'';
echo ''Teste C'';
echo ''Teste D'';
?>
```

O comando **include** é muito utilizado quando você quer definir funções ou variáveis que serão utilizadas com freqüência em várias páginas de um site.

Funções

Funções são pequenas rotinas de código que realizam tarefas específicas, auxiliando o programador a deixar seus programas mais organizados. Qualquer tarefa a ser executada dentro de um programa pode ser uma função, desde um simples comando de impressão até tarefas mais complexas. Veja o exemplo abaixo:

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

Onde:
soma é o nome da função.
$a e **$b** são argumentos.
$c é o valor retornado.

A instrução **return** é opcional. Toda função pode opcionalmente retornar um valor, ou simplesmente executar os comandos e não retornar valor algum. As funções podem retornar inteiros, strings, doubles, etc.

Não é possível que uma função retorne mais de um valor, mas é permitido fazer com que uma função retorne um valor composto, como listas ou arrays.

Qualquer código PHP válido pode estar contido no interior de uma função.

Exemplo de chamada de uma função:

```
<?
$a=12;
$b=14;
echo 'Exemplo de uso de funções em PHP';
soma($a,$b);
function soma($a, $b)
    {
      $c = $a + $b;
      return $c;
    }
echo "<br>$a<br>$b";
?>
```

No PHP há uma série de funções já prontas, bastando apenas fazermos uma chamada à essas funções. No exemplo abaixo há um exemplo de uso de uma dessas funções. Echo é uma dessas funções.

No site www.php.net você encontra um guia de todas as funções existentes no PHP e no final dessa apostila são mostradas algumas dessas funções e exemplos de como utilizá-las.

Exemplo de função do PHP:

```
<?
echo 'Exemplo de uso de funções em PHP';
$nome='FulAno DOs Santos';
$nome=strtolower($nome);
echo $nome;
?>
```

A variável *nome* possuí inicialmente a string "FulAno DOs Santos". Após uso da função strtolower a variável passa a ter valor "fulano dos santos", pois essa função transforma todos os caracteres maiúsculos em minúsculos.

Passagem de Argumentos

É possível passar argumentos para uma função PHP, podendo-se passar argumentos tanto por valor como por referência.

PASSAGEM DE ARGUMENTOS POR REFERÊNCIA

Como **padrão**, o PHP passa parâmetros por valor, o que **não** possibilita que as alterações feitas pela função na variável permaneçam após o término da função. Para que as funções alterem o valor das variáveis definitivamente, usa-se a passagem de argumentos por **referência**, que consiste em colocar o sinal **&** antes do $. Veja o exemplo abaixo:

```
function arg_ref(&$variável)
    {
      $variável = ‘’valor’’;
    }
```

Veja mais exemplos de passagem de parâmetros:

```
function mais5($numero) {
    $numero += 5;
}
$a = 3;
mais5($a); //$a continua valendo 3
```

No exemplo acima, como a passagem de parâmetros é por valor, a função mais5 é inútil, já que após a execução sair da função o valor anterior da variável é recuperado. Se a passagem de valor fosse feita por referência, a variável $a teria 8 como valor.

O que ocorre normalmente é que ao ser chamada uma função, o interpretador salva todo o escopo atual, ou seja, os conteúdos

das variáveis. Se uma dessas variáveis for passada como parâmetro, seu conteúdo fica preservado, pois a função irá trabalhar na verdade com uma cópia da variável. Porém, se a passagem de parâmetros for feita por referência, toda alteração que a função realizar no valor passado como parâmetro afetará a variável que o contém.

Há duas maneiras de fazer com que uma função tenha parâmetros passados por referência: indicando isso na declaração da função, o que faz com que a pasagem de parâmetros sempre seja assim; e também na própria chamada da função. Nos dois casos utiliza-se o modificador "&". Vejamos um exemplo que ilustra os dois casos:

```
function mais5(&$num1, $num2) {
  $num1 += 5;
  $num2 += 5;
}
$a = $b = 1;
mais5($a, $b); /* Neste caso, só $num1 terá seu valor
alterado, pois a passagem por referência está
definida na declaração
da função. */
mais5($a, &$b); /* Aqui as duas variáveis terão seus
valores alterados. */
```

ARGUMENTOS COM VALORES PRÉ-DEFINIDOS (DEFAULT)

Em PHP é possível ter valores default para argumentos de funções, ou seja, valores que serão assumidos em caso de nada ser passado no lugar do argumento. Quando algum parâmetro é declarado desta maneira, a passagem do mesmo na chamada da função torna-se opcional.

```
function teste($vivas = “testando”) {
  echo $vivas;
}
teste(); // imprime “testando”
teste(“outro teste”); // imprime “outro teste”
```

É bom lembrar que quando a função tem mais de um parâmetro, o que tem valor default deve ser declarado por último:

```
function teste($figura = circulo, $cor) {
  echo “a figura é um “, $figura, “ de cor “ $cor;
}
teste(azul);
/* A função não vai funcionar da maneira esperada,
ocorrendo um erro no interpretador.
A declaração correta é: */
function teste2($cor, $figura = circulo) {
echo “a figura é um “, $figura, “ de cor “ $cor;
}
teste2(azul);
/* Aqui a funcao funciona da maneira esperada,
ou seja, imprime o texto: “a figura é um círculo
de cor azul” */
```

Escopo das Variáveis

O escopo de uma variável define aonde ela pode ser utilizada. No PHP as variáveis podem ser Globais, Locais e Estáticas. Na grande maioria dos casos todas as variáveis têm escopo global. As variáveis globais são declaradas em qualquer parte do código PHP e podem ser usadas em qualquer parte desse código à partir de sua declaração.

VARIÁVEIS GLOBAIS

Para se usar variáveis globais dentro de uma função, elas devem ser declaradas, dentro da mesma função, utilizando-se a palavra global.

Exemplo:

```
$curso = ''PHP''

function mostra()
{
    global $curso;
   echo $curso;
 }

mostra();
```

VARIÁVEIS LOCAIS

As variáveis locais são aquelas declaradas dentro de uma função e que fazem parte somente daquela função, não podendo ser usadas fora daquela função. Veja o exemplo abiaxo:

Exemplo:

```
function mostra()
{
  $x = 15;
  echo $x;
}
echo $x;     //Não imprimirá nada já que $x não é global
mostra();    //Imprimirá 15
```

VARIÁVEIS ESTÁTICAS

As variáveis estáticas são uma adaptação das variáveis locais para que o valor atribuído a elas dentro da função não seja perdido quando a mesma função terminar.

```
function incrementa()
{
  static $x = 0;
  echo $x;
  $x++;
}
```

No exemplo acima, cada vez que a função incrementa() for executada, a variável estática **$x** será impressa e depois incrementada, sem que, ao término da função, seu valor seja perdido. As variáveis estáticas são essenciais quando usadas em funções recursivas, que são funções que chamam a si mesmas.

EXERCICÍOS:

No exercício que você verá agora existem bons exemplos do uso de funções. Com ele, sua idéia de como as funções trabalham nos scripts PHP ficará mais clara. Após digitar seu script salve seu arquivo com o nome de “funcoes.php”. Depois teste seu script no navegador.

```
<html>
<title>Exercicio</title>
<body>
<?
// calcula a soma de duas variaveis
function soma($a,$b)
{
```

```
$total=$a+$b;
return ($total);
}
//calcula a subtracao de duas variaveis
function subtracao($a,$b)
{
$total=$a-$b;
return ($total);
}
//calcula a multiplicacao de duas variaveis
function multiplicacao($a,$b)
{
$total=$a*$b;
return ($total);
}
//calcula a divisao de duas variaveis
function divisao($a,$b)
{
$total=$a/$b;
return ($total);
}

// valores das variaveis
$a=3;
$b=2;
//saída da soma
$soma=soma($a,$b);
echo “A soma de $a com $b é $soma ! <br>”;
//saída da subtracao
$sub=subtracao($a,$b);
echo “A diferença entre $a e $b é de $sub !44<br>”;
//saída da multiplicacao
$multi=multiplicacao($a,$b);
echo “O produto entre $a e $b é de $multi !<br>” ;
//saída da divisao
```

```
$div=divisao($a,$b);
echo “A divisão de $a por $b é $div !<br>”;
?>
</body>
</html>
```

Arrays

Array é um tipo de variável que possui seu conteúdo agrupado por **índices**, como um vetor ou um dicionário. Estes índices podem ser de qualquer tipo suportado pelo PHP. Pode-se criar um array usando as funções **list()** ou **array()**, ou pode-se atribuir explicitamente o valor de cada elemento. Também é possível criar uma array, simplesmente adicionando-se valores ao array. Veja o exemplo abaixo:

```
<?
$array[0] = “Curso”;
$array[1] = “PHP”;
$array[“MPB”] = “Gilberto Gil”;
echo $array[0].”<br>”;
echo $array[1].”<br>”;
echo $array[“MPB”].”<br>”;
?>
```

Listas

As listas são utilizadas em PHP para realizar várias atribuições, como por exemplo, atribuir valores de uma array para variáveis, como mostra o exemplo a seguir:

```
<?
list($a, $b, $c) = array(0 => ‘’a’’, 1 => ‘’b’’, 2 => ‘’c’’);
?>
```

O programa anterior atribuirá simultaneamente e respectivamente os valores do array às variáveis passadas como parâmetros para o comando list. É muito importante lembrar que só serão passadas ao comando list os elementos do array que possuírem os índices com valores inteiros e não negativos.

EXERCICÍOS:

O exercício abaixo mostra a utilização de vetores e listas. Salve seu arquivo com o nome de “listas.php” na pasta habilitada e digite o endereço http://localhost/<endereço>/listas.php no seu navegador para testá-lo.

```
<html>
<title>Exercicio</title>
<body>
<?
$meses=array( “Janeiro”, “Fevereiro”, “Março”, “Abril”,
“Maio”, “Junho”, “Julho”, “Agosto”, “Setembro”, “Outubro”,
“Novembro”, “Dezembro”);
?>
Ciclo While
<br><br>
<?
$mes=0;
while ($mes<=11)
{
echo “Mês: $meses[$mes] <br>”;
$mes++ ;
}
?>
<br><br><br>
Ciclo For
<br><br>
<?
```

```
for ($mes=0; $mes<=11; $mes++)
{
echo “Mês: $meses[$mes] <br>”;
}
?>
</body>
</html>
```

Estabelecendo conexões entre PHP e MySQL

Para acessar bases de dados num servidor MySQL, é necessário antes estabelecer uma conexão. Para isso, deve ser utilizado o comando mysql_connect.

int mysql_connect(string [host[:porta]] , string [login] , string [senha]);

O valor de retorno é um inteiro que identifica a conexão, ou falso se a conexão falhar. Antes de tentar estabelecer uma conexão, o interpretador PHP verifica se já existe uma conexão estabelecida com o mesmo host, o mesmo login e a mesma senha. Se existir, o identificador desta conexão é retornado. Senão, uma nova conexão é criada.

Uma conexão estabelecida com o comando mysql_connect é encerrada ao final da execução do script. Para encerrá-la antes disso deve ser utilizado o comando mysql_close, que tem a seguinte assinatura:

int mysql_close(int [identificador da conexão]);

Se o identificador não for fornecido, a última conexão estabelecida será encerrada.

SELECIONANDO A BASE DE DADOS

Depois de estabelecida a conexão, é preciso selecionar a base de dados a ser utilizada, através do comando mysql_select_db, que segue o seguinte modelo:

```
int mysql_select_db(string base, int [conexao] );
```

Novamente, se o identificador da conexão não for fornecido, a última conexão estabelecida será utilizada.

REALIZANDO CONSULTAS

Para executar consultas SQL no MySQL, utiliza-se o comando mysql_query, que tem a seguinte assinatura:

```
int mysql_query(string query, int [conexao] );
```

Onde query é a expressão SQL a ser executada, sem o ponto-e-vírgula no final, e conexao é o identificador da conexão a ser utilizada. A consulta será executada na base de dados selecionada pelo comando mysql_select_db.

É bom lembrar que uma consulta não significa apenas um comando SELECT. A consulta pode conter qualquer comando SQL aceito pelo banco.

O valor de retorno é falso se a expressão SQL for incorreta, e diferente de zero se for correta. No caso de uma expressão SELECT, as linhas retornadas são armazenadas numa memória de resultados, e o valor de retorno é o identificador do resultado. Alguns comandos podem ser realizados com esse resultado:

APAGANDO O RESULTADO

int mysql_free_result(int result);

O comando mysql_free-result deve ser utilizado para apagar da memória o resultado indicado.

NÚMERO DE LINHAS

int mysql_num_rows(int result);

O comando mysql_num_rows retorna o número de linhas contidas num resultado.

UTILIZANDO OS RESULTADOS

Existem diversas maneiras de ler os resultados de uma query SELECT. As mais comuns serão vistas a seguir:

int mysql_result(int result, int linha, mixed [campo]);

Retorna o conteúdo de uma célula da tabela de resultados.
result é o identificador do resultado;
linha é o número da linha, iniciado por 0;
campo é uma string com o nome do campo, ou um número correspondente ao número da coluna. Se foi utilizado um alias na consulta, este deve ser utilizado no comando mysql_result.

Este comando deve ser utilizado apenas para resultados pequenos. Quando o volume de dados for maior, é recomendado utilizar um dos métodos a seguir:

array mysql_fetch_array(int result);

Lê uma linha do resultado e devolve um array, cujos índices

são os nomes dos campos. A execução seguinte do mesmo comando lerá a próxima linha, até chegar ao final do resultado.

array mysql_fetch_row(int result);

Semelhante ao comando anterior, com a diferença que os índices do array são numéricos, iniciando pelo 0 (zero).

Projeto

Você irá desenvolver um site dinâmico utilizando a linguagem PHP e o servidor de banco de dados MySQL. O objetivo desse site será a localização de funcionários nos Telecentros. Fazendo-se uma busca pelo nome do funcionário, serão disponibilizadas informações como: cargo, telefone e e-mail. O site permitirá inclusão, consulta, alteração e exclusão de dados.

1 - CRIAÇÃO DA BASE DE DADOS E TABELAS

Utilizando o servidor de banco de dados MySQL, o primeiro passo será definir a base de dados e as tabelas em que você guardará as informações. Você pode criar a estrutura de dados diretamente no MySQL, seguindo o roteiro abaixo:

Roteiro:

1. Abra uma janela de terminal, no modo texto.
2. Digite o comando: **mysql -u <usuario> -p** (pergunte ao instrutor qual usuário e senha você deve utilizar).
3. Crie uma base de dados no MySQL, que conterá as tabelas a serem utilizadas no projeto. O comando é: **create database telecentros;**

Onde **''telecentros''** é o nome do banco de dados.

4. Depois de criada, acesse a base de dados com o seguinte comando: **use telecentros;**

Agora você já pode pensar nas tabelas que serão necessárias neste projeto. Para facilitar, você irá utilizar uma única tabela, chamada **alunos**.

Tipos de Campos

varchar(N)	É um campo texto variável de no máximo N caracteres.
integer	É um inteiro padrão.
char(N)	É um campo texto com exatamente N caracteres.
text	É um campo texto com no máximo 65535 caracteres.
date	É um campo data no formato ''AAAA-MM-DD''.
Not null	Significa que o campo não pode ser nulo.
Primary key	Significa que é um campo chave.

Para criar essa tabela, você utilizará o seguinte comando:

```
create table alunos (
nome varchar(50) not null primary key,
cargo varchar(40) not null,
telefone varchar(10) not null,
email varchar(40),
);
```

Dicas:

OPERAÇÃO	COMANDO
Para adicionar ou excluir campos da tabela, depois que ela foi criada: Supondo que você quisesse excluir o campo cargo da tabela funcionarios:	alter table funcionarios drop column cargo;
Supondo que você quisesse adicionar novamente o campo cargo na tabela funcionarios:	alter table funcionarios add column cargo;
Para visualizar as bases de dados existentes:	show databases;
Para visualizar as tabelas pertencentes a uma base de dados:	use base de dados; show tables;
Para visualizar os campos de uma tabela:	desc tabela;
Para visualizar todos os registros de uma tabela:	select * from tabela;

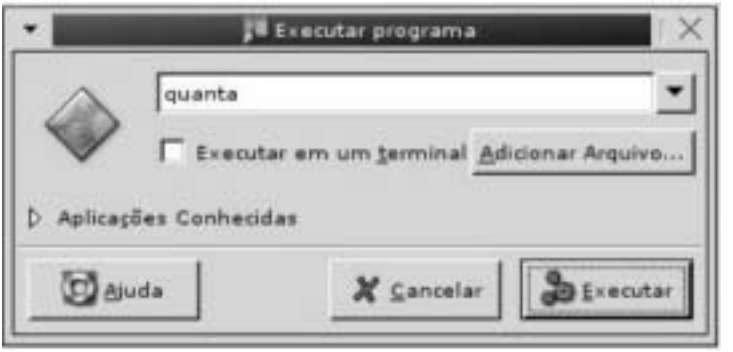
2 - CRIAÇÃO DA HOME PAGE DO SITE

A página principal (homepage) do site será bastante simples e trará um menu com as opções de inclusão, consulta, alteração e exclusão. As opções terão links para seus respectivos módulos.

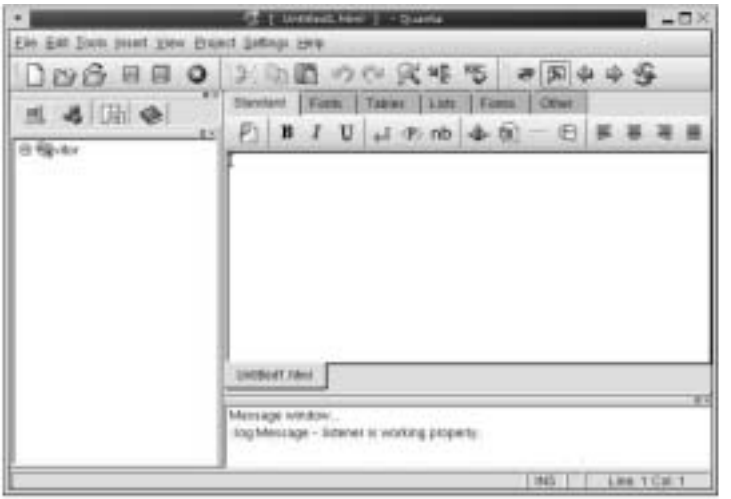
Essa página inicial terá a extensão html e será criada utilizando-se o Quanta Plus. O Quanta Plus é uma ferramenta de desenvolvimento para Web.

Roteiro:

1.Pressione as teclas **<Alt> + <F2>**.
2.Digite: **quanta**. Veja a figura abaixo.



3.Clique no botão **Executar**. Será exibida a tela do Quanta.



4.Clique no menu **File**.
5.Aponte na opção **New**.

Digite as seguintes opções de menu: Inclusão, Consulta, Alteração, Exclusão e Sair.

Transforme as opções do menu em links e direcione para seus respectivos endereços:
inclusao.html, consulta.html, alteracao.html, exclusao.html, e sair.html.
Salve a página como index.html no diretório indicado pelo instrutor.

3 - MÓDULO DE INCLUSÃO

Você irá criar a página para o formulário de inclusão. Você irá digitar o código utilizando um editor de texto qualquer. Esse arquivo terá extensão html.

3.1) FORMULÁRIO INCLUSAO.HTML

Após digitar o código, salve-o e teste-o. Veja a figura abaixo:

Quando o formulário for submetido, dará um erro, alertando que o programa **inclusao.php**, para o qual você está encaminhando os dados não existe. Então, você precisa criá-lo. Observe que a próxima página não terá mais a extensão **html** e sim **php**, pois o código vai conter programação PHP.

Dica:

Quando o servidor recebe a requisição de uma página HTML, ele apenas envia a página requisitada. Por outro lado, quando a requisição é de uma página com extensão PHP, o servidor processa o código antes de enviá-la. Pode-se combinar os códigos HTML e PHP.

3.2) PROGRAMA INCLUSÃO.PHP

O programa "inclusao.php" vai tratar os dados recebidos através do formulário, incluindo-os no banco de dados.

Dicas:

No código referente ao programa "inclusão.php", você utilizará algumas funções do PHP:

Trim: retira os espaços em branco de uma variável.

A expressão **or die** pode ser usada como uma alternativa para o **if/else**.

```
<html>
<head>
<title>Inclusao.php</title>
</head>
<body bgcolor="#FFFFFF">
<?php

// Tira os espaços em branco das variáveis recebidas
pelo formulário
$nome = trim($nome);
$cargo = trim($cargo);
$telefone = trim($telefone);
$email = trim($email);

echo ("<p><center><img src=\"telecentro.gif\" width=\"640\"
height=\"44\"></center></p>");

// Consiste Nome
if (empty($nome) || empty($telefone) || empty($cargo))
{
```

```
     echo (“<font color=\”#FF0000\”>
<b>Campo(s) obrigat&oacute;rio(s) n&atilde;o
preenchido(s)</
b></font>”);
    echo (“
     <table width=\”640\” border=\”0\” cellspacing=\”0\”
align=\”center\”>
    <tr>
      <td>
<p><b>Formulário de Inclusão: <br>
            </b></p>
            <form method=\”post\” action=\”inclusao.php\”>
             <p>Nome completo:
              <input type=\”text\” name=\”nome\” value=\
”$nome\” size=\”25\”
maxlength=\”50\”>
            </p>
             <p>Cargo: <input type=\”text\” name=\”cargo\”
value=\”$cargo\”
size=\”40\” maxlength=\”40\”> </p>
              <p>Telefone:
               <input type=\”text\” name=\”telefone\”
value=\”$telefone\”
maxlength=\”10\” size=\”10\”>
            </p>
             <p>E-mail:
               <input type=\”text\” name=\”email\” value=\”$email\”
size=\”25\” maxlength=\”40\”>
            </p>
           <p>
              <input type=\”submit\” name=\”Submit\”
value=\”Enviar\”>
               <center> <b> <a href=\”index.html\”>Home</a> </b>
</center>
            </p>
```

```
            </form>
        </td>
      </tr>
      <tr>
        <td> </td>
      </tr>
     </table>
“);
}
else {

// Inclui os dados recebidos do formulário na tabela de
alunos

// Cria uma conexão com o servidor MySQL passando
host, username e senha
      $conec = mysql_connect (“localhost”,”root”,
”telecentros”) or die
      (“Falha na conexão com o banco de dados”);

// Declaração SQL
      $declar = “INSERT into alunos values (‘$nome’,
$cargo’,
‘$telefone’, ‘$email’)”;
// Roda a query e trata o resultado

      if (mysql_db_query (“telecentros”, $declar, $conec)) {
         echo (“<BR><BR>”);
          echo (“<center> <b> <font size = 4> Inclusão
Efetuada </font> </b>
         </center>”);
         echo (“<BR>”);
          echo (“<center> <b> <a href=\”inclusao.html\”>
Voltar</a> </b>
</center>”);
```

```
    }
    else {
        echo (“<BR><BR>”);
     echo (“<center> <b> <font size = 4> Erro - Inclus&atilde;
o n&atildeo Efetuada
</font>

        </b> </center>”);
        echo (“<BR>”);
         echo (“<center> <b> <a href=\”inclusao.html\”>
Voltar</a> </b>
</center>”);
    }

// Fecha a conexão com o servidor MySQL (Opcional)
mysql_close ($conec);
}
?>
</body>
</html>
```

3.3) TESTANDO O MÓDULO DE INCLUSÃO

Roteiro:

1. Abra o Mozilla e digite o seguinte endereço do site:
http://localhost/<endereço>/index.html
onde <endereço> será o local indicado pelo instrutor.

2.No menu da página principal, clique na opção **Inclusão**.

3.Deixe os campos do formulário em branco. Clique em **Enviar**. Deverá mostrar uma mensagem de erro. O único campo que não é obrigatório é o e-mail.

4.Preencha o formulário com os dados do funcionário: nome completo, cargo, telefone e e-mail.

5.Clique em **Enviar**. Deverá mostrar a mensagem **"Inclusão Efetuada"**.

6.Volte para a página do formulário e entre com outros dados, só que desta vez entre com um nome que já existe no banco de dados.

7.Clique em **Enviar**. Deverá mostrar a mensagem **"Inclusão não efetuada"**, pois o campo nome é chave e não aceita valores duplicados.

8.Insira 3 funcionários.

4 - MÓDULO DE CONSULTA

Você irá criar a página com o formulário de consulta.

4.1) FORMULÁRIO CONSULTA.HTML

```
<html>
<head> <title>Consulta.html</title>
</head>
<body bgcolor="#FFFFFF">
<table width="640" border="0" cellspacing="0" align=
"center">
    <tr valign="top">
     <td>
          <p><img src="telecentro.gif" width="640"
height="44"></p>
```

```
            <!mensagem>
             <p><b>Formul&aacute;rio de Consulta<br>
             </b></p>
             <form method=”post” action=”consulta.php”>
              <p>Nome completo:
               <input type=”text” name=”nome” size=”25"
maxlength=”50">
             </p>
             <p>
               <input type=”submit” name=”Submit” value=
”Enviar”> </p>
             </form>
              <center> <b> <a href=”index.html”>Home</a> </b>
</center>
      </td>
    </tr>
    <tr>
      <td> </td>
    </tr>
</table>
</body>
</html>
```

Após digitar, salve e teste. Veja a figura abaixo:

Quando o formulário for submetido, dará um erro, alertando que o programa consulta.php, para o qual você está encaminhando os dados não existe. Então, você precisa criá-lo.

4.2) PROGRAMA CONSULTA.PHP

O programa consulta.php vai receber o nome do formulário, pesquisar no banco de dados e mostrar as informações referentes ao aluno.

Dicas:

No código referente ao programa consulta.php, você utilizará mais algumas funções do PHP:

File: Lê um arquivo, retornando o seu conteúdo como um array. Cada linha do arquivo será representada por um elemento do array.

Implode: Armazena todo o conteúdo de um array como uma string, concatena os conteúdos de cada elemento do array em uma string, utilizando ou não um delimitador entre eles.

Str_replace: Vai ler uma string e substituir um determinado valor por outro. No nosso caso, essa função substituirá a expressão "<!mensagem>", contida no código do "consulta.html", por uma mensagem de erro. Portanto, não se esqueça de colocar no html a expressão a ser substituída (como comentário).

mysql_num_rows: obtém o número de registros que retornou do select.

mysql_fetch_row: obtém os campos do registro que retornou do select.

```
<html>
<head>
<title>Consulta.php</title>
</head>
<body bgcolor="#FFFFFF">
<?php
// Tira os espaços em branco das variáveis recebidas pelo
formulário
$nome = trim($nome);
// Consiste Nome
if (empty($nome)) {
$html = file("consulta.html");
$html = implode(" ",$html);
$erro = "<center><font color=\"#FF0000\"> Preencha o
campo<b>Nome</b></font></center>";
$html = str_replace("<!mensagem>",$erro,$html);
echo ($html);
}
else {
echo ("<p><center><img src=\"telecentro.gif\" width=\"640\"
height=\"44\"></center></p>");

// Cria uma conexão com o servidor MySQL
// Parâmetros: host, username, senha
$conec = mysql_connect ("localhost","root","telecentros");

// Declaração do SQL
$declar = "SELECT cargo, telefone, email from alunos
where nome = '$nome'";
// Roda a query e verifica se encontrou registro
$query = mysql_db_query ('telecentros', $declar, $conec)
or die ("Erro no
acesso ao banco");
$achou = mysql_num_rows($query);
```

```
// echo ($achou);
// Se encontrou, guarda as variáveis
if ($achou > 0) {
$row = mysql_fetch_row ($query);
    $cargo = $row[0];
    $telefone = $row[1];
    $email = $row[2];
    echo (“<BR>”);
     echo (“<table width=\”640\” border=\”0\” cellspacing=\”0\”
align=\”center\”> <tr> <td>”);
    echo (“<b> Resultado da Consulta </b>”);
echo (“<BR><BR>”);
       echo (“<b> Nome: </b> $nome <BR>”);
       echo (“<b> Cargo: </b> $cargo <BR>”);
       echo (“<b> Telefone: </b> $telefone <BR>”);
       echo (“<b> E-mail: </b> $email <BR>”);
      echo (“</td> </tr> </table>”);
       echo (“<center> <b> <a href=\”consulta.html\”>Voltar</
a> </b>
</center>”);
}
else {
      echo (“<BR>”);
       echo (“<center> <b> Aluno n&atilde;o cadastrado
</b> </center>”);
      echo (“<BR>”);
       echo (“<center> <b> <a href=\”consulta.html\”>Voltar</
a> </b>
</center>”);
}
}
?>
</body>
</html>
```

4.3) TESTANDO O MÓDULO DE CONSULTA

Roteiro:

1. Abra o Mozilla e digite o seguinte endereço do site:
   http:///localhost/<endereço>/index.html

2.No menu da página principal, clique na opção **Consulta**.

3.Deixe o campo nome do aluno em branco e clique em **Enviar**. Deverá aparecer uma mensagem de erro.

4.Preencha o formulário com um nome de aluno inexistente e clique em enviar. Deverá aparecer a seguinte mensagem “Aluno não cadastrado”.

5.Preencha o formulário com um nome de aluno válido e clique em enviar. Deverá mostrar os dados do aluno.
Veja a figura abaixo:

5 - MÓDULO DE EXCLUSÃO

Você irá criar a página com o formulário de exclusão.

5.1) FORMULÁRIO EXCLUSAO.HTML

```
<html>
<head>
<title>Exclusao.html</title>
<meta http-equiv=”Content-Type” content=”text/
html; charset=iso-8859-1">
</head>
<body bgcolor=”#FFFFFF”>
<table width=”640" border=”0" cellspacing=”0" align=
”center”>
    <tr valign=”top”>
     <td>
          <p><img src=”telecentro.gif” width=
”640" height=”44”></p>
          <!mensagem>
           <p><b>Formul&aacute;rio de Exclus&atilde;o: <br>
           </b></p>
           <form method=”post” action=”exclusao.php”>
            <p>Nome Completo:
              <input type=”text” name=”nome” size=
”25" maxlength=”50">
           </p>
           <p>
             <input type=”submit” name=”Submit” value=
”Enviar”>
           </p>
           </form>
            <center> <b> <a href=”index.html”>Home</a> </b>
</center>
     </td>
   </tr>
```

```
    <tr>
      <td> </td>
    </tr>
</table>
</body>
</html>
```

Após digitar o código, salve-o e teste-o.

Quando o formulário for submetido, dará um erro, alertando que o programa exclusao.php, para o qual você está encaminhando os dados, não existe. Então, você precisa criá-lo.

5.2) PROGRAMA EXCLUSAO.PHP

O programa exclusao.php vai receber o nome do formulário, confirma através de consulta ao banco de dados se o aluno está cadastrado, e exclui o registro.

```
<html>
<head>
<title>Exclusao.php</title>
</head>
<body bgcolor="#FFFFFF">
<?php

// Tira os espaços em branco das variáveis recebidas
pelo formulário
$nome = trim($nome);

// Consiste Nome
if (empty($nome)) {
$html = file("exclusao.html");
$html = implode(" ",$html);
$erro = "<center><font color=\"#FF0000\"> Preencha o
```

```
campo<b>Nome</b></font></center>";
$html = str_replace("<!mensagem>",$erro,$html);
echo ($html);
}
else {
echo ("<p><center><img src=\"telecentro.gif\" width=\"640\"
height=\"44\"></center></p>");

// Cria uma conexão com o servidor MySQL
$conec = mysql_connect ("localhost","root","telecentros");

// Declaração do SQL
$declar = "SELECT nome from alunos where nome =
'$nome'";

// Roda a query, verifica se o aluno está cadastrado
$query = mysql_db_query ('telecentros', $declar, $conec)
or die ("Erro no
acesso ao banco");
$achou = mysql_num_rows($query);
//echo ($achou);
// Se encontrou exclui, senão mostra mensagem
if ($achou > 0) {



 echo ("<BR><BR>");
        echo ("<center> Aluno: $nome </center>");
       echo ("<BR>");

        // Exclui registro na tabela de alunos

         $declar2 = "DELETE from alunos where nome
 = '$nome'";
         if (mysql_db_query ('telecentros', $declar2, $conec)) {
```

```
            echo (“<BR><BR>”);
            echo (“<center> <b> <font size = 4>
Exclusão Efetuada </font> </b>
                  </center>”);
            echo (“<BR><BR>”);
echo (“<center> <b> <a href=\”exclusao.html\”>Voltar</a>
</b>
</center>”);
}
      else {
            echo (“<BR><BR>”);
            echo (“<center> <b> <font size = 4>
Erro - Exclus&atilde;o n&atilde;o Efetuada
</font>
            </b> </center>”);
            echo (“<BR><BR>”);
           echo (“<center> <b> <a href=\”exclusao.html\”>
Voltar</a> </b>
</center>”);
}
}
else {
          echo (“<BR><BR>”);
           echo (“<center> <b> Aluno n&atilde;o cadastrado
</b>
</center>”);
          echo (“<BR><BR>”);
echo (“<center> <b> <a href=\”exclusao.html\”>Voltar
</a> </b>
</center>”);
    }
mysql_close ($conec);
}
?>
</body>
</html>
```

5.3) TESTANDO O MÓDULO DE EXCLUSÃO

Roteiro:

1. Abra o Mozilla e digite o seguinte endereço do site:
http://localhost/<endereço>/index.html

2.No menu da página principal, clique na opção Exclusão.

3.Deixe o campo nome do aluno em branco e clique em Enviar. Deverá aparecer uma mensagem de erro.

4.Preencha o formulário com o nome completo do aluno e clique em Enviar. Deverá aparecer a mensagem “Exclusão efetuada”.

5.Preencha o formulário com o nome do aluno que você acabou de excluir e clique em Enviar. Deverá mostrar a mensagem “Aluno não cadastrado”.

6 - MÓDULO DE ALTERAÇÃO

Você irá criar a página com o formulário de alteração.

6.1) FORMULÁRIO ALTERACAO.HTML

Após digitar o código, salve-o e teste-o.

Quando o formulário for submetido, dará um erro, alertando que o programa “alteracao.php”, para o qual você está encaminhando os dados, não existe. Então, você precisa criá-lo.

6.2) PROGRAMA ALTERACAO.PHP

O programa alteracao.php vai receber o dado do formulário, recuperar as informações do banco de dados e mostrá-las num formulário para que elas sejam alteradas. Para montar o formulário, você criará uma função em PHP.

```
<html>
<head>
<title>Alteracao.php</title>
</head>
<body bgcolor=”#FFFFFF”>
<?php
include (“funcoes.php”);

// Tira os espaços em branco das variáveis recebidas
pelo formulário
$nome = trim($nome);

// Consiste Nome
if (empty($nome)) {
$html = file(“alteracao.html”);
$html = implode(“ “,$html);
$erro = “<center><font color=\”#FF0000\”> Preencha o
campo<b>Nome</b></font></center>”;
$html = str_replace(“<!mensagem>”,$erro,$html);
echo ($html);
}
else {
echo (“<p><center><img src=\”telecentro.gif\” width=
\”640\”height=\”44\”></center></p>”);
```

```
// Cria uma conexão com o servidor MySQL
// Parâmetros: host, username, senha
$conec = mysql_connect (“localhost”,”root”,”telecentros”);

// Declaração do SQL
$declar = “SELECT cargo, telefone, email from alunos
where nome = ‘$nome’”;

// Roda a query e verifica se encontrou registro
$query = mysql_db_query (‘telecentros’, $declar, $conec)
or die (“Erro no
acesso ao banco”);
$achou = mysql_num_rows($query);
// echo ($achou);

// Se encontrou, guarda as variáveis
if ($achou > 0) {

$row = mysql_fetch_row ($query);
      $cargo = $row[0];
      $telefone = $row[1];
      $email = $row[2];
        monta_pagina($nome,$cargo,$telefone,$email,);
}
else {
     echo (“<BR><BR>”);
      echo (“<center> <b> Aluno n&atilde;o cadastrado </b>
</center>”);
     echo (“<BR>”);
      echo (“<center> <b> <a href=\”alteracao.html\”>Voltar
</a> </b>
</center>”);
}
}
?>
```

```
</body>
</html>
```

Observe que, no código que você acabou de digitar, ele chama a função **monta_pagina**, passando como parâmetros as variáveis nome, cargo, telefone e email.

As **funções** são úteis porque podem ser reutilizadas em vários programas, além disso, o tamanho do código do programa chamador diminui consideravelmente.

Você pode criar um único programa, exemplo: **funcoes.php** que conterá todas as funções. Um detalhe importante que você não pode esquecer é que você irá precisar incluir esse programa de funções no seu programa chamador. No código visto anteriormente você tem o comando **include** ("funcoes.php") logo no início do código php.

6.3) PROGRAMA FUNCOES.PHP

O programa **"funcoes.php"** pode armazenar todas as funções que serão utilizadas no site. Neste curso você usará apenas a função **monta_pagina**. Essa função serve para montar o formulário já preenchido, com as informações que foram passadas como parâmetros no programa anterior.

Observação: Com algumas pequenas alterações, essa função também poderia ser usada para recriar o formulário do módulo de inclusão.

```
<?php
function
monta_pagina($nome,$cargo,$telefone,$email)
{
echo “<table width=\”640\” border=\”0\” cellspacing=\”0\”
align=\”center\”>”;
echo “<tr>”;
echo “<td>”;
echo “<p><b>Formul&aacute;rio de Altera&ccedil;&atilde;
o: <br></b></p>”;
echo “<form method=\”post\” action=\”alteracao2.php\”>”;
echo “<p>Nome: $nome </p>”;
echo “<p>Cargo: <input type=\”text\” name=\”cargo
\” value=\”$cargo\”
      size=\”40\” maxlength=\”40\”> </p>”;
echo “<p>Telefone: <input type=\”text\” name=\”telefone\”
value=\”$telefone\”  maxlength=\”10\” size=\”10\”> </p>”;
echo “<p>E-mail: <input type=\”text\” name=\”email
\” value=\”$email\”
size=\”25\” maxlength=\”25\”> </p>”;
echo “<p> <input type=\”submit\” name=\”Submit
\” value=\”Enviar\”>
      </p>”;
echo “<p> <input type=\”hidden\” name=\”nome
\” value=\”$nome\”></p>”;
echo “</form>”;
echo “</td>”;
echo “</tr>”;
echo “<tr>”;
echo “</tr>”;
echo “</table>”;
echo (“<center> <b> <a href=\”alteracao.html\”>Voltar</a>
</b>
</center>”);
return; }
?>
```

Observe que o formulário criado pela função **monta_pagina** chama o programa **alteracao2.php**. Isso porque, para você completar o módulo de alteração precisará de mais um programa que pegue as informações que foram alteradas e as inclua no banco de dados.

Observação: Como o campo nome não é passado para o programa alteracao2.php, por não se tratar de uma variável do formulário, temos que passá-lo como um campo escondido input type="hidden".

6.4) PROGRAMA ALTERACAO2.PHP

O programa "alteracao2.php" vai pegar as informações alteradas e fazer um update (atualizar) no banco de dados.

```
<html>
<head>
<title>Alteracao2.php</title>
<meta http-equiv="Content-Type" content="text/
html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF">
<p><center><img src="telecentro.gif" width=
"640" height="44"></center></p>
<?php
include ("funcoes.php");
// Tira os espaços em branco das variáveis recebidas
pelo formulário
$nome = trim($nome);
$cargo = trim($cargo);
$telefone = trim($telefone);
$email = trim($email);

if (empty($nome) || empty($cargo) || empty($telefone))
```

```
{
echo (“<font color=\”#FF0000\”>
<b>Campo(s) obrigat&oacute;rio(s) n&atilde;o preenchido(s)
</b></font>”);
monta_pagina($nome, $cargo, $telefone, $email);
}
else
{
// Cria uma conexão com o servidor MySQL
$conec = mysql_connect (“localhost”,”root”,”telecentros”)
or die
// consulta SQL pra atualizar a tabela do banco de dados
$declar=”UPDATE alunos SET nome=
’$nome’, cargo=’$cargo’, email=’$email’, telefone=
’$telefone’ WHERE nome=’$nome’”;
// Roda a query e trata o resultado
if (mysql_db_query (“telecentros”, $declar, $conec)) {
     echo (“<BR><BR>”);
    echo “<center> <b> <font size = 4> Altera&ção Efetuada!
</font> </b>
     </center>”;
     echo (“<br>”);


    echo (“<center> <b> <a href=\”alteracao.html\”>Voltar
</a> </b>
</center>”);
}
else {
     echo (“<BR><BR>”);
     echo (“<center> <b> <font size = 4> Erro - Alteração
não Efetuada
</font>
     </b> </center>”);
     echo (“<BR>”);
```

```
        echo ("<center> <b> <a href=\"alteracao.html\">Voltar<
/a> </b>
</center>");
}

// Fecha a conexão com o servidor MySQL (Opcional)
mysql_close ($conec);
}
?>
</body>
</html>
```

6.5) TESTANDO O MÓDULO DE ALTERAÇÃO

Roteiro:

1. Abra o Mozilla e digite o seguinte endereço do site:
   http://localhost/<endereço>/index.html

2.No menu da página principal, clique na opção **Alteração**.

3.Deixe o campo nome do aluno em branco e clique em **Enviar**. Deverá aparecer uma mensagem de erro.

4.Preencha o formulário com um nome de aluno que não existe e clique em **Enviar**. Deverá mostrar a mensagem "Aluno não cadastrado".

5.Preencha o formulário com um nome de aluno válido e clique em Enviar. Será mostrado um formulário com os dados desse aluno.

6.Altere alguns campos e clique em **Enviar**. Deverá mostrar a mensagem "Alteração efetuada".

7.Entre no módulo de consulta e confira se os dados foram realmente alterados.

7 - CRIAÇÃO DA PÁGINA SAIR.HTML

A página sair.html é utilizada no nosso site apenas para agradecer ao usuário por utilizar o sistema. Utilize o Quanta Plus

para criar uma página simples que agradece ao usuário pelo uso do sistema, e salve-a com o nome de sair.html.

Dicas:

COMO OBTER DATA E HORA DO SISTEMA.

No exemplo a seguir você obterá data e hora usando a função **date**, joga o conteúdo em variáveis e mostra essas variáveis na tela.

PARÂMETROS UTILIZADOS NA FUNÇÃO DATE:

j: dia
m: mês
Y: ano
H: hora
i: minutos
s: segundos

```
<html>
<head>
<title>Fun&ccedil;&atilde;o Data e Hora</title>
</head>
<body bgcolor="#FFFFFF">

<?php
$data = date("j/m/Y");
$hora = date("H:i:s");

echo ("Data: $data");
echo ("<br><br>");
echo ("Hora: $hora");
?>
</body>
</html>
```

Observação: Se você fosse gravar a data num banco de dados (aaaa/mm/dd), ao invés de mostrá-la, a sintaxe seria a seguinte:

$data = date("Y/m/j");

Guia rápido de funções preexistentes no PHP

FUNÇÕES RELACIONADAS AO HTML

• htmlspecialchars

string htmlspecialchars(string str);

Devolve a string fornecida, substituindo os seguintes caracteres:

& para '&'
" para '"'
< para '<'
> para '>'

• htmlentities

string htmlentities(string str);

Funciona de maneira semelhante ao comando anterior, mas de maneira mais completa, pois converte todos os caracteres da string que possuem uma representação especial em html, como por exemplo:

º para 'º'
ª para 'ª'
á para 'á'
ç para 'ç'

• nl2br

string nl2br(string str);

Devolve a string fornecida substituindo todas as quebras de linha (“\n”) por quebras de linhas em html (“
”).

Exemplo:

```
echo nl2br(“Mauricio\nVivas\n”);
```

Imprime:

```
Maurício<br>Vivas<br>
```

• get_meta_tags

array get_meta_tags(string ficheiro);

Abre um arquivo html e percorre o cabeçalho em busca de “meta” tags, Devolvendo num array todos os valores encontrados.

Exemplo:

No arquivo teste.html temos:

```
...
<head>
<meta name=”author” content=”jose”>
<meta name=”tags” content=”php3 documentation”>
...
</head><!— busca encerra aqui —>
...
```

a execução da função:

```
get_meta_tags(“teste.html”);
```

Devolve o array:

```
array(“author”=>”jose”,”tags”=>”php3 documentation”);
```

• strip_tags

string strip_tags(string str);

Devolve a string fornecida, retirando todas as tags html e/ou PHP encontradas.
Exemplo:
strip_tags('testando
');

Devolve a string "testando"

• urlencode

string urlencode(string str);

Devolve a string fornecida, convertida para o formato urlencode. Esta função é útil para passar variáveis para uma próxima página.

• urldecode

string urldecode(string str);

Funciona de maneira inversa a urlencode, desta vez decodificando a string fornecida do formato urlencode para texto normal.

FUNÇÕES RELACIONADAS A ARRAYS

• Implode e join

string implode(string separador, array partes);
string join(string separador, array partes);

As duas funções são idênticas. Devolvem uma string contendo todos os elementos do array fornecido separados pela string também fornecida.

Exemplo:

```
$partes = array(“a”, “casa número”, 13, “é azul”);
$inteiro = join(“ “,$partes);
```

$inteiro passa a conter a string:
“a casa número 13 é azul”

- split

array split(string padrao, string str, int [limite]);

Devolve um array contendo partes da string fornecida separadas pelo padrão fornecido, podendo limitar o número de elementos do array.

Exemplo:

```
$data = “11/14/1975”;
$data_array = split(“/”,$data);
```

O código acima faz com que a variável $data_array receba o valor: array(11,14,1975);

- explode

array explode(string padrao, string str);

Funciona de maneira bastante semelhante à função split, com a diferença que não é possível estabelecer um limite para o número de elementos do array.

COMPARAÇÕES ENTRE STRINGS

• similar_text

int similar_text(string str1, string str2, double [porcentagem]);

Compara as duas strings fornecidas e devolve o número de caracteres coincidentes. Opcionalmente pode ser fornecida uma variável, passada por referência (ver tópico sobre funções), que receberá o valor percentual de igualdade entre as strings. Esta função é case sensitive, ou seja, maiúsculas e minúsculas são tratadas como diferentes.

Exemplo:

$num = similar_text(“teste”, “testando”,&$porc);

As variáveis passam a ter os seguintes valores:
$num == 4; $porc == 61.538461538462

• strcasecmp

int strcasecmp(string str1, string str2);

Compara as duas strings e Devolve 0 (zero) se forem iguais, um valor maior que zero se str1 > str2, e um valor menor que zero se str1 < str2. Esta função é case insensitive, ou seja, maiúsculas e minúsculas são tratadas como iguais.

• strcmp

int strcasecmp(string str1, string str2);

Funciona de maneira semelhante à função strcasecmp, com a diferença que esta é case sensitive, ou seja, maiúsculas e minúsculas são tratadas como diferentes.

- strstr

string strstr(string str1, string str2);
string strchr(string str1, string str2);

As duas funções são idênticas. Procura a primeira ocorrência de str2 em str1. Se não encontrar, Devolve uma string vazia, e se encontrar Devolve todos os caracteres de str1 a partir desse ponto.

Exemplo:
strstr("Mauricio Vivas", "Viv"); // Devolve "Vivas"

- stristr

string strstr(string str1, string str2);

Funciona de maneira semelhante à função strstr, com a diferença que esta é case insensitive, ou seja, maiúsculas e minúsculas são tratadas como iguais.

- strpos

int strpos(string str1, string str2, int [offset]);

Devolve a posição da primeira ocorrência de str2 em str1, ou zero se não houver. O parâmetro opcional offset determina a partir de qual caracter de str1 será efetuada a busca. Mesmo utilizando o offset, o valor de retorno é referente ao início de str1.

- strrpos

int strrpos(string haystack, char needle);

Devolve a posição da última ocorrência de str2 em str1, ou zero se não houver.

FUNÇÕES PARA EDIÇÃO DE STRINGS

• chop

string chop(string str);

Retira espaços e linhas em branco do final da string fornecida.
Exemplo:
chop(“ Teste \n \n “); // Devolve “ Teste”

• ltrim

string ltrim(string str);

Retira espaços e linhas em branco do final da string fornecida.
Exemplo:
ltrim(“ Teste \n \n “); // Devolve “Teste \n \n”

• trim

string trim(string str);
Retira espaços e linhas em branco do início e do final da string fornecida.
Exemplo:
trim(“ Teste \n \n “); // Devolve “Teste”

• strrev

string strrev(string str);

Devolve a string fornecida invertida.
Exemplo:
trrev(“Teste”); // Devolve “etseT”

• strtolower

string strtolower(string str);

Devolve a string fornecida com todas as letras minúsculas.
Exemplo:
strtolower(“Teste”); // Devolve “teste”

• strtoupper

string strtoupper(string str);

Devolve a string fornecida com todas as letras maiúsculas.
Exemplo:
strtolower(“Teste”); // Devolve “TESTE”

• ucfirst

string ucfirst(string str);

Devolve a string fornecida com o primeiro caracter convertido para letra maiúscula.
Exemplo:
ucfirst(“teste de funcao”); // Devolve “Teste de funcao”

• ucwords

string ucwords(string str);

Devolve a string fornecida com todas as palavras iniciadas por letras maiúsculas.
Exemplo:
ucwords(“teste de funcao”); // Devolve “Teste De Funcao”

• str_replace

string str_replace(string str1, string str2, string str3);

Altera todas as ocorrências de str1 em str3 pela string str2.

FUNÇÕES DIVERSAS

• chr

string chr(int ascii);

Devolve o caracter correspondente ao código ASCII fornecido.

• ord

int ord(string string);

Devolve o código ASCII correspontente ao caracter fornecido.

• echo

echo(string arg1, string [argn]...);

Imprime os argumentos fornecidos.

• strlen

int strlen(string str);

Devolve o tamanho da string fornecida.

Todas as funções preexistentes no PHP podem ser vistas e consultadas no site www.php.net. Você pode utilizar esse site sempre que precisar. Nele existem guias em português e excelentes explicações sobre a linguagem.

Referência/fontes bibliográficas:

www.php.net

www.phpbuilder.com

www.phpwizard.net

Curso de Aplicaçoes WEB em PHP
Autor: Mauricio Vivas

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações

Anotações