PHP Avançado

Índice

9.1 - Trabalhando com Arrays em PHP

Trabalhando com Arrays

Um array é uma variável, mas diferente das demais ele armazena uma coleção de valores e não somente um. E ainda por cima podem conter outras variáveis e de tipos diferentes.

Detalhe importante: Quando em uma função precisarmos rtornar mais de um valor array é a saída, basta retornar todos os valors em forma de array.

Além disso é semelhante ao que estudamos na matemática: linhas e colunas. Matriz 3x4 (3 linhas e 4 colunas).

Um array no PHP é um mapa ordenado, que relaciona valores com chaves (em linhas e colunas).

Especificando um array()

array([chave =>] valor, ...);

A chave pode ser uma string ou um inteiro. O valor pode ser qualquer coisa.

Algumas das funções

Essas funções abaixo permitem a interação e manipulação de arrays de várias formas. Arrays são essenciais para armazenar, gerenciar, operar sobre um conjunto de variáveis.

Arrays (matrizes) simples e multidimensionais são suportados e podem ser criados pelo usuário ou por outras funções. Existem diversas funções específicas para bancos de dados, que preenchem arrays com os dados retornados em consultas, e vários outros tipos de funções também retornam arrays.

array_fill -- Preenche um array com valores

array array_fill (int start_index, int num, mixed value)

```
<?php
$a = array_fill(5, 6, 'banana');
print_r($a);
?>
```

array_merge -- Funde dois ou mais arrays

array array_merge (array array1, array array2 [, array ...])

```
<?php
$array1 = array();
$array2 = array(1 => "data");
$result = array_merge($array1, $array2);
?>
```

Não esqueça que as chaves numéricas serão reordenadas!

```
Array
(
   [0] => data
)
```

Se você quer preservar os arrays e apenas concatená-los, o operador +:

```
<?php
$array1 = array();
$array2 = array(1 => "data");
$result = $array1 + $array2;
?>
```

As chaves numéricas serão preservadas e as associações originais permanecem.

array_pad -- Expande um array para um certo comprimento utilizando um determinado valor

array array_pad (array input, int pad_size, mixed pad_value)

Exemplo 1. Exemplo de array_pad()

```
<?php
$input = array(12, 10, 9);

$result = array_pad($input, 5, 0);
// $result é array(12, 10, 9, 0, 0)

$result = array_pad($input, -7, -1);
// $result é array(-1, -1, -1, -1, 12, 10, 9)

$result = array_pad($input, 2, "noop");
// Não será expandido.
?>
```

array_pop -- Retira um elemento do final do array

mixed array_pop (array array)

```
<?php
$cesta = array("laranja", "banana", "melancia", "morango");
$fruta = array_pop($cesta);
print_r($cesta);
?>
```

array_push -- Adiciona um ou mais elementos no final de um array

int array_push (array array, mixed var [, mixed ...])

```
<?php
$cesta = array("laranja", "morango");
array_push($cesta, "melancia", "batata");
print_r($cesta);
?>
```

array_reverse -- Retorna um array com os elementos na ordem inversa

array array_reverse (array array [, bool preserve_keys])

```
<?php
$input = array("php", 4.0, array ("verde", "vermelho"));
```

```
$result = array_reverse($input);
$result_keyed = array_reverse($input, TRUE);
print_r($result_keyed);
?>
```

array_search

-- Procura por um valor em um array e retorna sua chave correspondente caso seja encontrado. Caso contrário retorna FALSE.

mixed array_search (mixed procurar_este, array procurar_neste [, bool strict])

```
<?php
$a=array("a","b",0,"c","d");
echo "a: ".array_search("a",$a)."<br>";
echo "b: ".array_search("b",$a)."<br>";
echo "c: ".array_search("c",$a)."<br>";
echo "d: ".array_search("d",$a)."<br>";
echo "0: ".array_search("0",$a)."<br>";
echo "x: ".array_search("x",$a)."<br>";
echo "1: ".array_search("1",$a);
?>

<?php
if (array_search($needle, $array)!== FALSE) {
   //code goes here (
}
?>

<?php

function array_replace($search, $replace, &$array) {
  foreach($array as $key => $value) {
   if($value == $search) {
     $array[$key] = $replace;
   }
  }
}
?>

<?
$Projects[0] = array(123, "Text 1");
$Projects[1] = array(456, "Text 2");
$Projects[2] = array(789, "Text 3");

$search_value = "ext 3";

foreach ($Projects as $key => $row){
     foreach($row as $cell){
       if (strpos($cell, $search_value) !== FALSE){
                echo  "<p>Project ".$key;
       }
     }
 }
?>
```

array_shift -- Retira o primeiro elemento de um array

mixed array_shift (array array)

```
<?php
$cesta = array("laranja", "banana", "melancia", "morango");
$fruta = array_shift($cesta);
print_r($cesta);
?>
```

array_sum -- Calcula a soma dos elementos de um array

mixed array_sum (array arr)

```
<?php
$a = array(2, 4, 6, 8);
echo "soma(a) = ".array_sum($a)."<br>";

$b = array("a" => 1.2, "b" => 2.3, "c" => 3.4);
echo "soma(b) = ".array_sum($b)."<br>";
?>
```

array_unique -- Remove os valores duplicados de um array

array array_unique (array array)

```
<?php
$input = array("a" => "verde", "vermelho", "b" => "verde", "azul", "vermelho");
$result = array_unique($input);
print_r($result);
?>
```

Exemplo 2. array_unique() e tipos

```
<?php
$input = array(4, "4", "3", 4, 3, "3");
$result = array_unique($input);
var_dump($result);
?>
<pre>


<h2>array_values -- Retorna todos os valores de um array</h2>

array array_values ( array input )

//Retorna os valores, as chaves não
<pre>
<?php
$array = array("tamanho" => "G", "cor" => "dourado");
print_r(array_values ($array));
?>
```

array -- Cria um array

array array ([mixed ...])

Exemplo 1. Exemplo de array()

```
<?php
$frutas = array (
   "frutas" => array("a"=>"laranja", "b"=>"banana", "c"=>"maçã"),
   "numeros" => array(1, 2, 3, 4, 5, 6),
   "buracos" => array("primeiro", 5 => "segundo", "terceiro")
)
?>
```

Exemplo 2. Indexação automática com array()

```
<?php
$array = array(1, 1, 1, 1,  1, 8 => 1,  4 => 1, 19, 3 => 13);
print_r($array);
?>
```

arsort

-- Ordena um array em ordem descrescente dos valores mantendo a associação entre índices e valores

void arsort (array array [, int sort_flags])

```
<?php
$frutas = array("d" => "limao", "a" => "laranja", "b" => "banana", "c" =>
"melancia");
arsort($frutas);
reset($frutas);
while (list($chave, $valor) = each($frutas)) {
   echo "$chave = $valor\n";
}
?>
```

asort

-- Ordena um array em ordem crescente dos valores mantendo a associação entre índices e valores

void asort (array array [, int sort_flags])

```
<?php
$frutas = array("d" => "limao", "a" => "laranja", "b" => "banana", "c" =>
"melancia");
asort($frutas);
reset($frutas);
```

```
while (list($chave, $valor) = each($frutas)) {
   echo "$chave = $valor\n";
}
?>
```

count -- Conta o número de elementos de uma variável

int count (mixed var [, int mode])

```
<?php
$a[0] = 1;
$a[1] = 3;
$a[2] = 5;
$a[3] = 6;
$result = count($a);
// $result == 4
print $result."<br>";

$b[0] = 7;
$b[5] = 9;
$b[10] = 11;
$result = count($b);
// $result == 3;
print $result;
?>
```

Exemplo 2. Uso recursivo da função count() (PHP >= 4.2.0)

```
<?php
$food = array( 'fruits'  => array('orange', 'banana', 'apple'),
'veggie'  => array('carrot', 'collard','pea'));
// recursive count
echo count($food,COUNT_RECURSIVE);  // mostra 8
// normal count
echo count($food);                  // mostra2 2
?>

<?php
$food = array( 'fruits'  => array('orange', 'banana', 'apple'),
'veggie'  => array('carrot', 'collard','pea'));
// recursive count
echo count($food,COUNT_RECURSIVE)."<br>";  // mostra 8
// normal count
echo count($food);                  // mostra2 2
?>
```

current -- Retorna o elemento corrente em um array

mixed current (array array)

```
<?php
```

```
$transport = array('foot', 'bike', 'car', 'plane');
$mode = current($transport); // $mode = 'foot';
echo "Atual $mode<br>";
$mode = next($transport);    // $mode = 'bike';
echo "Atual $mode<br>";
$mode = current($transport); // $mode = 'bike';
echo "Atual $mode<br>";
$mode = prev($transport);    // $mode = 'foot';
echo "Atual $mode<br>";
$mode = end($transport);    // $mode = 'plane';
echo "Atual $mode<br>";
$mode = current($transport); // $mode = 'plane';
echo "Atual $mode<br>";
?>
```

each -- Retorna o par chave/valor corrente de um array e avança o seu cursor

array each (array array)

```
<?php
$foo = array("bob", "fred", "jussi", "jouni", "egon", "marliese");
$bar = each($foo);
print_r($bar);
?>

<?php
  $foo = array("Robert" => "Bob", "Seppo" => "Sepi");
  $bar = each($foo);
  print_r($bar);
?>
```

Percorrendo um array com each()

```
<?php
$fruit = array('a' => 'apple', 'b' => 'banana', 'c' => 'cranberry');
reset($fruit);
while (list($key, $val) = each($fruit)) {
     echo "$key => $val\n";
}
/* Saída:

  a => apple
  b => banana
  c => cranberry

  */
?>
```

end -- Faz o ponteiro interno de um array apontar para o seu último elemento

mixed end (array array)

```
<?php
  $frutas = array('melancia', 'banana', 'morango');
  print end($frutas); // morango
?>
```

key -- Retorna uma chave da posição atual de um array associativo

mixed key (array array)

```
<?php
  $array = array(
     'fruit1' => 'apple',
     'fruit2' => 'orange',
     'fruit3' => 'grape',
     'fruit4' => 'apple',
     'fruit5' => 'apple');

  // este ciclo  exibirá todas as chaves do array associativo
  // ausiliado pela função next()
  while ($fruit_name = current($array)) {
         echo key($array).'<br>';
             next($array);
  }
  ?>

<?php
  $array = array(
     'fruit1' => 'apple',
     'fruit2' => 'orange',
     'fruit3' => 'grape',
     'fruit4' => 'apple',
     'fruit5' => 'apple');

  // este ciclo  exibirá toda a chave do array associativo
  // onde o valor é igual a "apple"
  while ($fruit_name = current($array)) {
     if ($fruit_name == 'apple') {
         echo key($array).'<br>';
     }
     next($array);
  }
  ?>
```

next -- Avança o ponteiro interno de um array

mixed next (array array)

```
<?php
  $transport = array('foot', 'bike', 'car', 'plane');
  $mode = current($transport); // $mode = 'foot';
  print"$mode<br>";
  $mode = next($transport);    // $mode = 'bike';
  print"$mode<br>";
  $mode = next($transport);    // $mode = 'car';
  print"$mode<br>";
  $mode = prev($transport);    // $mode = 'bike';
  print"$mode<br>";
  $mode = end($transport);    // $mode = 'plane';
  print"$mode<br>";
?>
```

prev -- Retrocede o ponteiro interno de um array

mixed prev (array array)

```
<?php
  $transport = array('foot', 'bike', 'car', 'plane');
  $mode = current($transport); // $mode = 'foot';
print"$mode<br>";
  $mode = next($transport);    // $mode = 'bike';
print"$mode<br>";
  $mode = next($transport);    // $mode = 'car';
print"$mode<br>";
  $mode = prev($transport);    // $mode = 'bike';
print"$mode<br>";
  $mode = end($transport);    // $mode = 'plane';
print"$mode<br>";
?>
<pre>


<h2>reset --  Faz o ponteiro interno de um array apontar para o seu primeiro
elemento</h2>

mixed reset ( array array )
<pre>
<?php
  $array = array('primero passo', 'segundo passo', 'terceiro passo', 'quarto
passo');

  // por definição, o ponteiro está sobre o primeiro elemento
  echo current($array)."<br>\n"; // "Primeiro passo"

  // pula dois passos
  next($array);
  next($array);
  echo current($array)."<br>\n"; // "passo três"
```

```
  // reinicia o ponteiro, começa novamente o primeiro passo
  reset($array);
  echo "Depois de resetado...: " . current($array)."<br>\n"; // "primeiro passo"
?>
```

sizeof -- Apelido de count()

sort -- Ordena um array pelo seu valor

void sort (array array [, int sort_flags])

```
<?php

$frutas = array("limao", "laranja", "banana", "melancia");
sort($frutas);
reset($frutas);
while (list($chave, $valor) = each($frutas)) {
   echo "frutas[".$chave."] = ".$valor."<br>";
}
?>
```

Os seguintes também são funcionalmente idênticos:

```
<?php
$arr = array("one", "two", "three");
reset($arr);
while (list($key, $value) = each ($arr)) {
   echo "Chave: $key; Valor: $value<br />\n";
}

foreach ($arr as $key => $value) {
   echo "Chave: $key; Valor: $value<br />\n";
}
?>
```

Mais alguns exemplos para demonstrar os usos:

```
<?php
/* exemplo foreach 1: somente valores */

$a = array(1, 2, 3, 17);

foreach ($a as $v) {
   echo "Valor atual de \$a: $v.\n";
}

/* exemplo foreach 2: valores (com as chaves impressas para ilustração) */

$a = array(1, 2, 3, 17);
```

```
$i = 0; /* para exemplo somente */

foreach ($a as $v) {
   echo "\$a[$i] => $v.\n";
   $i++;
}

/* exemplo foreach 3: chaves e valores */

$a = array (
   "um" => 1,
   "dois" => 2,
   "três" => 3,
   "dezessete" => 17
);

foreach ($a as $k => $v) {
   echo "\$a[$k] => $v.\n";
}

/* exemplo foreach 4: arrays multidimensionais */

$a[0][0] = "a";
$a[0][1] = "b";
$a[1][0] = "y";
$a[1][1] = "z";

foreach ($a as $v1) {
   foreach ($v1 as $v2) {
       echo "$v2\n";
   }
}

/* exemplo foreach 5: arrays dinâmicos */

foreach (array(1, 2, 3, 4, 5) as $v) {
   echo "$v\n";
}
?>
```

Exemplo de array multidimensional

```
$produto[1][codigo] = "1";
$produto[1][nome] = "João Pereira Brito";
$produto[1][email] = "joao@joao.org";
$produto[1][rua] = "Vasco da Gama";
$produto[1][numero] = "1345";

$produto[2][codigo] = "2";
$produto[2][nome] = "Antônio queiroz";
```

Exemplo de Array

```
$i=0;
while($i < $numregs){
        $codigo=pg_result($consulta,$i,codigo);
        $nome=pg_result($consulta,$i,nome);
        $venc=pg_result($consulta,$i,vencimento);
        $apartamento=pg_result($consulta,$i,apartamento);
        $pessoas=pg_result($consulta,$i,pessoas);
        $cota_agua=pg_result($consulta,$i,cota_agua);
        $cota_condominio=pg_result($consulta,$i,cota_condominio);
        $cota_reserva=pg_result($consulta,$i,cota_reserva);

        $total = $cota_agua + $cota_condominio + $cota_reserva;
        $total = number_format($total,2, ',','.');
...
$i++;
}
```

Também podemos ter um array formado por outros arrays (neste caso, cada sub array é uma linha do principal)

```
$arrayvarios = array(
        array(1, 3, 5, 7),
        array(2, 4, 6, 8),
        array(1, 1, 1, 1)
);
```

Neste caso temos um array 2x4 (2 linhas por 4 colunas, que iniciam sempre com índice zero).

Então se queremos retornar o valor 8, que está na linha 2 e coluna 4, devemos retornar o índice 1,3 (linha2=índice 1, coluna4=índice3).

print $arrayvarios[1][3];

Agora veremos com detalhes os pares: chave => valor:

```
$alunos = array(
        "0732355" => "Ribamar FS",
        "0823456" => "Antônio Brito",
        "0654345" => "Roberto Queiroz"
);
```

O que isto retornaria?

```
print $alunos["0732355"];
print $alunos[0];
```

Experimente!!

Atribuindo valores às chaves de arrays

Também podemos fazer diretamente assim:

print $alunos["0732355"] = "João Brito";

Lembrando que, a chave, é exclusiva. Podemos ter

```
$alunos["0732355"] = "João Brito";
```

```
$alunos["0932355"] = "João Brito";

Mas não podemos ter:
$alunos["0732355"] = "João Brito";
$alunos["0732355"] = "Ribamar FS";
```

Anexo agora um excelente tutorial sobre Arrays do Celso Goya publicado em:
http://www.xoopstotal.com.br/modules/wfsection/article.php?articleid=51

Trabalhando com arrays

Visão geral Para facilitar o entendimento, vamos definir array como um conjunto de valores, que podem ser identificados em grupo ou então separadamente. Estes conjuntos podem ser muito úteis enquanto programamos, pois em alguns casos podem substituir uma tabela em banco de dados ou então utilizando métodos mais avançados podemos carregá-los dinâmicamente e utilizar quase como um banco de dados em memória.

A linguagem PHP oferece uma incrível gama de recursos para se trabalhar com arrays. Com destaque para as funções auxiliares que permitem fazer desde uma simples contagem de elementos até a conversão automática de um array em string.

Neste artigo desenvolveremos como exemplo uma função para gerar combo boxes com os estados do Brasil. Muitas vezes criamos uma tabela no banco de dados para armazenar a lista de estados do Brasil sendo que neste caso existe um número finito de registros e menor que 100, então as operações de banco de dados não são tão ágeis quanto o uso de um array.

Criando o primeiro array

Para utilizar um array, antes de mais nada é preciso criar uma variável do tipo array.

```
<?php
$estados = array();
?>
```

O próximo passo é montar nossa lista de estados.

```
<?php
$estados = array();
$estados[0] = "Acre";
$estados[1] = "Alagoas";
$estados[2] = "Amapá";
$estados[3] = "Amazonas";
?>
```

Os colchetes servem para identificar qual elemento do nosso conjunto estamos nos referindo e o número entre colchetes é o código identificador do elemento.

Podemos fazer o seguinte teste:

```
<?php
$estados = array();
$estados[0] = "Acre";
$estados[1] = "Alagoas";
```

```
$estados[2] = "Amapá";
$estados[3] = "Amazonas";
echo($estados[0]);
?>
```

Neste caso será exibida a palavra Acre, pois indicamos o ítem [0] da variável $estados, que é um array.

Você deve estar se perguntando "O que há de tão fantástico em um array?". Agora vamos mostrar alguns recursos.

Criando o array de estados

Nosso array não será de grande valia se não permitir que as siglas dos estados sejam armazenadas também, pois desta forma podemos guardar no banco de dados apenas os dois caractéres correspondentes à sigla do estado, ou seja, utilizaremos apenas dois bytes no banco de dados.

Então vamos criar um array com duas colunas, sendo a primeira a sigla do estado e a segunda seu nome por extenso.

```
<?php
$estados = array();
$estado[0][0] = "AC";
$estado[0][1] = "Acre";
$estado[1][0] = "AL";
$estado[1][1] = "Alagoas";
$estado[2][0] = "AP";
$estado[2][1] = "Amapá";
$estado[3][0] = "AM";
$estado[3][1] = "Amazonas";
$estado[4][0] = "BA";
$estado[4][1] = "Bahia";
$estado[5][0] = "CE";
$estado[5][1] = "Ceará";
$estado[6][0] = "DF";
$estado[6][1] = "Distrito Federal";
$estado[7][0] = "ES";
$estado[7][1] = "Espírito Santo";
$estado[8][0] = "GO";
$estado[8][1] = "Goiás";
$estado[9][0] = "MA";
$estado[9][1] = "Maranhão";
$estado[10][0] = "MG";
$estado[10][1] = "Minas Gerais";
$estado[11][0] = "MT";
$estado[11][1] = "Mato Grosso";
$estado[12][0] = "MS";
$estado[12][1] = "Mato Grosso do Sul";
$estado[13][0] = "PA";
$estado[13][1] = "Pará";
$estado[14][0] = "PR";
$estado[14][1] = "Paraná";
$estado[15][0] = "PE";
$estado[15][1] = "Pernambuco";
```

```
$estado[16][0] = "PI";
$estado[16][1] = "Piauí";
$estado[17][0] = "RJ";
$estado[17][1] = "Rio de Janeiro";
$estado[18][0] = "RN";
$estado[18][1] = "Rio Grande do Norte";
$estado[19][0] = "RS";
$estado[19][1] = "Rio Grande do Sul";
$estado[20][0] = "RO";
$estado[20][1] = "Rondônia";
$estado[21][0] = "RR";
$estado[21][1] = "Roraima";
$estado[22][0] = "SC";
$estado[22][1] = "Santa Catarina";
$estado[23][0] = "SP";
$estado[23][1] = "São Paulo";
$estado[24][0] = "SE";
$estado[24][1] = "Sergipe";
$estado[25][0] = "TO";
$estado[25][1] = "Tocantins";
?>
```

A diferença neste exemplo é que utilizamos dois identificadores de elemento, ou seja, agora para cada elemento do array possuimos mais outros dois dependentes. Da mesma forma que criamos dois elementos o 0 e 1 para cada item de estado poderiamos criar n novos sub-elementos, por exemplo:

```
<?php
$estado = array();

$estado[0][0] = "SP";
$estado[0][1] = "São Paulo";
$estado[0][2] = "Sudeste";
?>
```

Vamos considerar à partir de agora que um array possui linhas e colunas, onde as linhas são equivalentes ao primeiro conjunto de colchetes e as colunas são equivalentes ao segundo conjunto de colchetes.

A função de exibição do combo box de estados

Agora vamos exibir todos os elementos de nosso array em uma função:

```
<?php
/*Nossa função reebe 3 parâmetros
$pNome :: Corresponde ao nome do SELECT
$pSelected :: Corresponde ao elemento que deverá possuir o status de selecionado
automaticamente
$extra :: Caso precise adicionar um style, ou então opção de multiple
*/
function renderCombo($pNome = "", $pSelected = "SP", $extra = ""){
        echo("<SELECT NAME='".$pNome."' ".$extra.">");

/*
Para exibir todos os itens do nosso combo utilizamos o comando for ,
lembre-se que como usamos números para identificar nosso array,
então podemos substituí-lo automaticamente com o for
```

```
*/
        //Realiza o loop em todos os elementos do array
        for( $i = 0; $i < 26;$i++ ){
                //Imprime a TAG OPTION usando a primeira coluna do array
                echo("<OPTION VALUE='".$estado[$i][0]."'");
                //Efetua a comparação para verificar se este é o ítem
selecionado
                if( $estado[$i][0] == $pSelected ){
                        //Caso a comparação seja verdadeira seleciona o ítem
                        echo(" SELECTED");
                }
                //Imprime o nome por extenso do estado, equilavente a segunda
coluna do array
                echo(">".$estado[$i][1]."</option>\n");
        }
        //Finaliza a tag SELECT
        echo("</SELECT>\n");
}
?>
```

Eureka! Esta feita uma função para exibir um combo de estados.

Identificadores alternativos

Na linguagem PHP podemos utilizar palavras para identificar um elemento de um array, este recurso é muito bom, pois facilita muito a depuração e o entendimento de programas que utilizam arrays.

Vamos utilizar nosso array de estados como exemplo:

```
<?php
$estado[0]["sigla"] = "SP";
$estado[0]["nome"] = "São Paulo";
$estado[0]["regiao"] = "Sudeste";

echo($estado[0]["sigla"]);
?>
```

Desta forma podemos deixar o código de nossos programas mais fáceis de se compreender. Repare que utilizamos uma string simples para identificar um elemento do array, sendo assim, podemos utilizar variáveis para identificar um item do array, por exemplo:

```
<?php
$estado[0]["sigla"] = "SP";
$estado[0]["nome"] = "São Paulo";
$estado[0]["regiao"] = "Sudeste";

$variavel = "sigla";

echo($estado[0][$variavel]);
?>
```

É importante lembrar que mesmo existindo uma string para identificar um elemento do array ainda podemos utilizar números se quisermos, por exemplo:

```
<?php
$estado[0]["sigla"] = "SP";
$estado[0]["nome"] = "São Paulo";
$estado[0]["regiao"] = "Sudeste";

echo($estado[0][0]);
?>
```

Nos três casos o resultado é o mesmo, diferindo apenas no método como chamamos o array.

O que você viu neste artigo é o básico sobre arrays, caso você se interessar pelo assunto e queira dar uma pesquisada rápida na web, vai encontrar outras formas de declarar arrays bem como usos diferenciados. O XOOPS utiliza muito este recurso. É só dar uma olhada em algum arquivo xoops_version.php, que você vai encontrar um exemplo prático do uso de arrays.

Final do tutorial do Celso Goya.

Convertendo objetos para um array

http://www.revistaphp.com.br/print.php?id=147

```
$array = array();

if(is_array($array)){
    echo 'Variável $array é um array';
} else {
    echo 'Variable is not an array';
}
```

Usando Array em Laços

For

```
<?php

 $animals = array( 'dingo', 'wombat', 'Steve Irwin', 'playpus', 'emu' );

 /*** Recebe o tamanho do array ***/
 $size = sizeof( $animals );

 for( $i=0; $i<$size; $i++ ){
      echo $animals[$i].'<br />';
 }

?>
```

While

```
<?php

 $animals = array( 'dingo', 'wombat', 'Steve Irwin', 'playpus', 'emu' );

 $size = sizeof( $animals );

 /*** set um contador ***/
 $i = 0;

 while( $i < $size )
 {
      echo $animals[$i].'<br />';
      $i++;
 }

?>
```

foreach
O foreach varre todos os elementos de um array

```
<?php

 $animals = array( 'dingo', 'wombat', 'Steve Irwin', 'playpus', 'emu' );

 foreach( $animals as $animal )
 {
      echo $animal.'<br />';
 }

?>
```

Adicionar um Elemento para um Array
Usar a função array_push().

```
<?php

$animals = array(
      'ernie'=>'dingo',
      'wally'=>'wombat',
      'rat bag'=>'Steve Irwin',
      'playto'=>'platypus',
      'marty'=>'emu'
      );

 /*** Adicionar um novo elemento para o array ***/
 array_push( $animals, 'wallaby', 'stingray' );
```

```
 foreach( $animals as $key=>$animal ){
       echo $key.' - '.$animal.'<br />';
 }

?>
```

Excluir elemento de array
Usar a função unset()

```
<?php

  /*** create an array ***/
  $animals = array(
       'ernie'=>'dingo',
       'wally'=>'wombat',
       'rat bag'=>'Steve Irwin',
       'playto'=>'platypus',
       'marty'=>'emu'
       );

 /*** delete an array element ***/
 unset( $animals['rat bag'] );

 /*** loop over the array ***/
 foreach( $animals as $key=>$animal )
 {
       echo $key.' - '.$animal.'<br />';
 }

?>
```

9.2 - Trabalhando com Formulários em PHP

Manipulando dados de formulários com PHP – Parte 1

Uma das dúvidas mais freqüentes entre programadores PHP iniciantes é como manipular os dados de formulário enviados para os scripts PHP, principalmente dados de "checkbox" e upload de arquivos. Nessa primeira parte desse artigo, estarei mostrando como receber e manipular dados de campos comuns de formulários.

Na próxima semana estarei mostrando como manipular o upload de arquivos através dos formulários.

Para facilitar, esta primeira parte está divida nos seguintes tópicos:

1 - Introdução

Um formulário HTML é apenas um "rosto bonito" para onde os usuários poderão inserir informações que serão interpretados de alguma maneira por algum script do lado do servidor. E no nosso caso, esse script é um script PHP.

Primeiro: antes para poder enviar as informações, seu formulário deve conter um botão "submit", isso se consegue através do comando:

```
<input type=submit value="Texto do Botão">
```

Segundo: todos os campos que serão tratados no script PHP devem conter o parâmetro "NAME", caso contrário, os dados não serão passados para o script PHP. Ex: <input type=text name=nome_do_campo>

Como as informações do formulário são passadas para esse script PHP e como as informações do formulário enviado são tratadas, dependem de você.

Existem 2 métodos como as informações podem ser passadas: GET e POST. O recomendável sempre, para todos os formulários é usar o método POST, onde os dados enviados não são visíveis nas URLs, ocultando possíveis importantes informações e permitindo o envio de longas informações. O GET é totalmente o contrário disso.

Como as informações chegam para o script PHP?

Assuma o seguinte formulário:

```
<form action="script.php" method="post">
Campo 1: <input type=text name=campo1><br>
Campo 2: <input type=text name=campo2><br>
<input type=submit value="OK">
</form>
```

Esse formulário usa o método POST para envio das informações, então em "script.php":

```
<?php
echo "O valor de CAMPO 1 é: " . $_POST["campo1"];
echo "<br>O valor de CAMPO 2 é: " . $_POST["campo2"];
?>
```

Se o formulário tivesse sido enviado usando o método GET, você simplesmente usaria $_GET no lugar de $_POST.

Observações:

Em vez de usar $_GET ou $_POST você pode escrever a variável com o mesmo nome do campo do formulário (no exemplo, $campo1 e $campo2). Mas, esse uso não é recomendado, pois se a diretiva "register_globals" na configuração do seu PHP estiver desativada, as variáveis com nome dos campos dos formulários, terão um valor vazio.

Uma solução para isso é usar a função **import_request_variables** no começo dos seus scripts que interpretam formulários. Essa função aceita 3 letras como argumentos: P, G e C, referentes a $_POST, $_GET e $_COOKIE respectivamente. Exemplo de uso:

```
<?php
import_request_variables("gP");
?>
```

O que acontece?

Exemplo: Você possui formulário com os campos "nome", "endereço" e "idade". Assuma que a diretiva "register_globals" do seu PHP esteja desligada, mas, você já havia programado o script usando as variáveis no escopo global, no lugar de $_POST.

Adicionando aquela função no começo do script, as variáveis do seu formulário postado: $_POST["nome"], $_POST["endereco"] e $_POST["idade"] serão extraídas cada para uma variável diferente: $nome, $endereco e $idade.

2 - Campos Hidden

Os campos hidden são usados para passar informações que não podem ser alteradas pelo usuário que estará inserindo informações no formulário. Por exemplo: você tem um site com sistema de login e o usuário quer alterar as informações de login dele. O script que irá manipular esse formulário, precisa saber o ID do usuário para poder alterar as informações no banco de dados,

então esse ID é um campo hidden.

Códigos Exemplos:

hidden.html

```
<form action="hidden.php" method="post">
<input type=hidden name=escondido value="valor do escondido">
<input type=hidden name=id value="111">
<input type=submit>
</form>
```

hidden.php

```
<?php
echo "Campo Hidden: " . $_POST["escondido"];
echo "<br>Oi, seu ID é: " . $_POST["id"];
?>
```

3 - Campos Text e Textarea

Os campos text e textarea são os tipos mais simples, onde há somente um possível valor por campo. Dispensam maiores explicações.

Códigos Exemplos:

texts.html

```
<form action="texts.php" method="post">
Nome: <input type=text name=nome><br>
Email: <input type=text name=email><br><br>
Mensagem: <textarea name=mensagem cols=8 rows=3></textarea><br>
<input type=submit>
</form>
```

texts.php

```
<?php
echo "Olá " . $_POST["nome"] . " (email: " . $_POST["email"] . ")<br><br>";
echo "Sua mensagem: " . $_POST["mensagem"];
?>
```

4 - Campos Radio

Campos Radio permitem um relacionamento de um para muitos entre identificador e valor, ou seja, eles têm múltiplos possíveis valores, mas somente um pode ser pré-exibido ou selecionado. Por exemplo: você tem um sistema de "quiz". Cada pergunta possui 5 possíveis respostas. Cada resposta é um radio, onde os 5 radios dessa pergunta possuem o mesmo identificador, mas cada com valores diferentes.

Códigos Exemplos:

radio.html

```
<form action="radio.php" method="post">
<B>Qual seu sistema operacional?</B><br>
<input type=radio name=sistema value="Windows 98"> Win 98
<input type=radio name=sistema value="Windows XP"> Win XP
<input type=radio name=sistema value="Linux"> Linux
<input type=radio name=sistema value="Mac"> Mac
<br><br>
<B>Qual a marca de seu monitor?</B><br>
<input type=radio name=monitor value="Samsung"> Samsung
<input type=radio name=monitor value="LG"> LG
<input type=radio name=monitor value="Desconhecido"> Desconhecido
<br><br>
<input type=submit>
</form>
```

radio.php

```
<?php
echo "Seu sistema operacional é: " . $_POST["sistema"];
echo "<br>Seu monitor é: " . $_POST["monitor"];
?>
```

5 - Campos Checkbox

O tipo Checkbox tem somente um possível valor por entrada: on value (marcado) ou no value (desmarcado). No script você deve fazer a verificação para saber se o campo foi marcado ou não.

Se é possível também utilizar grupos de checkbox com o mesmo nome. Para você deve adicionar "[]" no final do nome, para o PHP interpretar como array, veja o código exemplo.

Códigos Exemplos:

checkbox.html

```
<form action="checkbox.php" method="post">
<B>Escolha os numeros de sua preferência:</B><br>
<input type=checkbox name="numeros[]" value=10> 10<br>
<input type=checkbox name="numeros[]" value=100> 100<br>
<input type=checkbox name="numeros[]" value=1000> 1000<br>
<input type=checkbox name="numeros[]" value=10000> 10000<br>
<input type=checkbox name="numeros[]" value=90> 90<br>
<input type=checkbox name="numeros[]" value=50> 50<br>
<input type=checkbox name="numeros[]" value=30> 30<br>
<input type=checkbox name="numeros[]" value=15> 15<br><BR>
<input type=checkbox name="news" value=1> <B>Receber
Newsletter?</B><br><BR>
<input type=submit>
</form>
```

checkbox.php

```
<?php
// Verifica se usuário escolheu algum número
if(isset($_POST["numeros"]))
{
	echo "Os números de sua preferência são:<BR>";

	// Faz loop pelo array dos numeros
	foreach($_POST["numeros"] as $numero)
	{
		echo "- " . $numero . "<BR>";
	}
}
else
{
	echo "Você não escolheu número preferido!<br>";
}

// Verifica se usuário quer receber newsletter
if(isset($_POST["news"]))
{
	echo "Você deseja receber as novidades por email!";
}
else
{
	echo "Você não quer receber novidades por email...";
}
?>
```

6 - Campos Select

Os campos select permitem tratar uma variedade de opções, onde o usuário pode selecionar apenas uma opção ou múltiplas opções. Quando você permite múltiplas seleções, deve adicionar "[]" no final do nome, para o PHP interpretar como array.

Nos exemplos, mostro o funcionamento e tratamento de ambas.

Códigos Exemplos:

select.html

```
<form action="select.php" method="post">
<B>Qual seu processador?</B><br>
<select name=processador>
<option value=Pentium>Pentium</option>
<option value=AMD>AMD</option>
<option value=Celeron>Celeron</option>
</select><BR><BR>
<B>Livros que deseja comprar?</B><br>
Obs: segure "CTRL" para selecionar mais de um.<BR>
<select name="livros[]" multiple>
<option value="Biblia do PHP 4">Biblia do PHP 4</option>
```

```
<option value="PHP Professional">PHP Professional</option>
<option value="Iniciando em PHP">Iniciando em PHP</option>
<option value="Novidades do PHP 5">Novidades do PHP 5</option>
<option value="Biblia do MySQL">Biblia do MySQL</option>
</select><BR><BR>
<input type=submit>
</form>
```

select.php

```
<?php
echo "Seu processador é: " . $_POST["processador"] . "<BR>";

// Verifica se usuário escolheu algum livro
if(isset($_POST["livros"]))
{
	echo "O(s) livro(s) que você deseja comprar:<br>";
	// Faz loop para os livros
	foreach($_POST["livros"] as $livro)
	{
		echo "- " . $livro . "<br>";
	}
}
else
{
	echo "Você não escolheu nenhum livro!";
}
?>
```

Clique aqui para baixar os códigos desse artigo

Quaisquer dúvidas que tiver, não hesite em contatar-me!

Até a próxima semana, onde estarei mostrando como manipular o upload de arquivos de formulários e algumas boas técnicas para com formulários => PHP.

Alfred Reinold Baudisch
Desenvolvedor de Sistemas Web
alfred@auriumsoft.com.br
www.estacaonet.com

Excluir Marcados

```
<?php
/*
Banco - excluir_varios
Tabela
CREATE TABLE `produtos` (
  `id` int(11) NOT NULL,
```

```
  `produto` char(45) default NULL,
  `categoria` char(45) default NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
*/

// Form com arquivo chamando a si mesmo
$conexao = mysql_connect('localhost','root','ribafs');
mysql_select_db('excluir_varios',$conexao);
$consultar = "SELECT * FROM produtos ORDER BY id";
$resultado = mysql_query($consultar, $conexao);

if(mysql_num_rows($resultado) != 0){
     echo "<form name='frmExcluir' method='post' action='teste.php'>";
     echo "<table border=1><tr><th> </th><th>Produto:</th><th>Categoria:</th></tr>";

     while($linha = mysql_fetch_row($resultado)){
          echo "<td><input type='checkbox' name='id[]' value='$linha[0]'></td>
                <td>$linha[1]</td>
                <td>$linha[2]</td></tr>";
     }
     echo "<tr><td colspan='3'><input type='submit' name='excluir' value='Excluir!'></td></tr>";
     echo "</table></form>";
}else{
     echo "Nenhum registro foi encontrado!";
}
if(isset($_POST['id'])){
     $opcoes = $_POST['id'];
     $opcoes_text = implode(", ", $opcoes);
     $strexcluir = "DELETE FROM produtos WHERE id in (" . $opcoes_text . ")";
     mysql_query($strexcluir, $conexao) or die("Ocorreu algum erro");
}else{
     echo "É necessário escolher quem será excluído<br>";
     echo "<a href='javascript: history.back();'>Voltar</a>";
}
?>
```

Básico de GET e POST

Usar GET em situações seguras e POST em situações que precisem de segurança.

GET suporta apenas 1024 caracteres na string.
POST é sem limite.

Exemplo de uso de formulário com PHP chamando outro script

```
<body onLoad="document.form1.nome.focus()">
<form name="form1" method="POST" action="inserir.php">
Nome.<input type="text" name="nome"><br>
E-mail<input type="text" name="email"><br>
<input type="hidden" name="oculto" value="OK">
Senha<input type="password" name="senha"><br>
SenhaCript<input type="password" name="senhacript"><br>
<input type="submit" value="Enviar"><br>
</form>

<?php
//$nome=$_POST["nome"];
//$email=$_POST["email"];
if (isset($nome)){
    echo "O nome digitado foi " . $nome . "<br>";
    echo "O e-mail digitado foi " . $email . "<br>";
    echo "O campo oculto contém " . $oculto . "<br>";
    echo "A senha digitada foi " . $senha . "<br>";
    echo "A senha md5 digitada foi " . md5($senhacript) . "<br>";
}
?>
```

Exemplo de uso de formulário com PHP chamando o mesmo script

```
<body onLoad="document.form1.nome.focus()">
<form name="form1" method="POST" action="<?php $_PHPSELF ?>">
Nome.<input type="text" name="nome"><br>
E-mail<input type="text" name="email"><br>
<input type="submit" name="enviar" value="Enviar"><br>
</form>

<?php
if(isset($_POST['enviar'])){
$nome = $_POST["nome"];
$email = $_POST["email"];

    if(isset($nome)){
        echo "O nome digitado foi " . $nome . "<br>";
        echo "O e-mail digitado foi " . $email . "<br>";
    }
}
?>
```

import_request_variables("gP");
import_request_variables -- Import GET/POST/Cookie variables into the global scope

Muito indicado para quem tem o register_globals desativado e não quer digitar
$_POST, $_GET ou $_COOKIE.

Sempre deve vir na primeira linha do script ou antes do uso da variável.

import_request_variables("gP", "rvar_");

echo $rvar_foo;

Alerta: evite usar o recurso acima em produção, use apenas em caso de teste e ainda assim deve ser evitado e em teste devemos reproduzir o que iremos usar em produção.

As informações digitadas em um formulário podem ser capturadas por um script PHP.
Veja como:

form.html
```
<form action="recebe.php" method="POST">
Nome <input type="text" name="nome"><br>
Idade <input type="text" name="idade"><br>
<input type="submit" value="Enviar">
</form>
```

recebe.php

```
<?php
echo $_POST["nome"];
echo "<br>";
echo $_POST["idade"];
?>
```

Recebendo dados via URL

Se temos a seguinte situação:
Temos um arquivo c:\www\teste.php, com o seguinte código:

```
<?php
   echo $_GET['codigo'];
?>
```

Queremos passar uma variável código com valor 5, fazemos:
http://127.0.0.1/teste.php?codigo=5

Para receber mais de uma variável ou campo usar:
http://127.0.0.1/teste.php?codigo=5&nome="Antônio"&email="ribafs@yahoo.com"

echo "Código = ".$_GET['codigo']." Nome = ".$_GET['nome']." E-mail = ".$_GET['email'];

Também podemos passar variáveis pela URL, assim teste.php?codigo=$cod

9.3 - Trabalhando com Arquivos em PHP

Quando abrimos um arquivo para leitura (r+) o ponteiro situa-se no início do arquivo.
Quando abrimos um arquivo para anexar (a+) o ponteiro situa-se no final do arquivo.

Sempre usar fclose() quando finalizar de trabalhar com o arquivo para economizar memória.

O PHP suporta duas funções básicas para escrever em arquivos:
file_put_contents() - que escreve ou anexa uma string para um arquivo.
fwrite() - função que incrementalmente escreve dados para um arquivo texto.

Exemplos:

file_put_contents($string, $arquivo);

A função file($arquivo) lê arquivos texto.

Ler todo um arquivo em uma string:
```
$file = file_get_contents("/windows/system32/drivers/etc/services");
print("Size of the file: ".strlen($file)."\n");
```

Abrir um arquivo para leitura
```
$file = fopen("/windows/system32/drivers/etc/hosts", "r");
print("Tipo de manipulador de arquivo: " . gettype($file) . "\n");
print("Primeira linha do manipulador do arquivo: " . fgets($file));
fclose($file);
```

Ler um único caractere de um arquivo:
```
<?php
$file = fopen("/windows/system32/drivers/etc/services", "r");
$count = 0;
while ( ($char=fgetc($file)) !== false ) {
  if ($char=="/") $count++;
}
fclose($file);
print("Number of /: $count\n");
?>
```

Conteúdo do meuarquivo.txt

```
1|Meu nome|Meu site|Olá pessoal
2|Meu nome1|Meu site|Olá pessoal
3|Meu nome2|Meu site|Olá pessoal
4|Meu nome3|Meu site|Olá pessoal
5|Meu nome4|Meu site|Olá pessoal
```

Primeiro, vamos colocar todo o arquivo num array. Use o seguinte código:
```
$meuArray = file("nomedoarquivo.txt");
```

Vamos excluir a linha que tenha o primeiro valor do array igual a 3.

Para isto, temos que ler cada linha do array e fazer um "explode" delas, separando os "|" como se fosse em colunas.

Depois, faremos um [cfIF[/cf] perguntando se a coluna desejada é igual ao valor que queremos: 3

```
for($n=0; $n < count($meuArray); $n++) {
      $cadaLinha = explode("|", $meuArray[$n]);
     if($cadaLinha[0] == 3) {
          echo "Meu nome: $cadalinha[1] - Site: $cadalinha[2]";
     }
}
```

Excluindo linhas do array e salvando nos arquivos:

```
$meuArray = file("nomedoarquivo.txt"); // coloco todo o arquivo num array
$arrayModificado = array(); // crio um array vazio.

for($n=0; $n < count($meuArray); $n++) {
     $cadaLinha = explode("|", $meuArray[$n]);

     if($cadaLinha[0] <> 3) {
          $arrayModificado[] = $meuArray[$n];
     }
}
```

Depois é só salvar o array no arquivo, incluindo cada linha do array já modificado:

```
$bufferArquivo = fopen('nomedoarquivo.txt',"w");

for($n=0; $n < count($arrayModificado); $n++) {
    fwrite($bufferArquivo, $arrayModificado[$n]);
}

fclose($bufferArquivo);
```

Autor: Emannuel em http://www.codigofonte.net/dicas/php/130_trabalhando-com-arquivos

Veja também:
http://onlamp.com/pub/a/php/2002/10/03/php_foundations.html
http://onjava.com/pub/a/php/2002/12/12/php_foundations.html
http://onlamp.com/pub/a/php/2003/01/09/php_foundations.html

Abrir arquivo

fopen(filename, mode, [use_include_path]);

filename: pode ser simplesmente um nome, ou um caminho completo.

Exemplos: "arquivo.txt", "./arquivo.dat", "/data/data.txt".

mode: especifica o modo de abertura, ou seja, se o arquivo deve ser aberto para leitura, escrita, etc.
Modos de abertura:

```
- r: abre o arquivo no modo somente leitura e posiciona o ponteiro no início do
arquivo;
        o arquivo já deve existir;
- r+: abre o arquivo para leitura/escrita, posiciona o ponteiro no início do
arquivo;
- w: abre o arquivo no modo somente escrita; se o arquivo já existir, será
sobrescrito;
        senão, será criado um novo;
- w+: abre o arquivo para escrita/leitura; se o arquivo já existir, será
sobrescrito;
        senão, será criado um novo;
- a: abre o arquivo para anexar dados, posiciona o ponteiro no final do arquivo;
        se o arquivo não existir, será criado um novo;
- a+: abre o arquivo para anexo/leitura, posiciona o ponteiro no final do
arquivo;
        se o arquivo não existir, será criado um novo;

<?php
        $fp = fopen("./arquivo.dat", "r"); // $fp conterá o handle do arquivo
que abrimos
?>
<pre>


<h2>Fechar Arquivo</h2>

<pre>
fclose(handle_arquivo);

<?php
fclose($fp);
?>
```

Gravar em Arquivo

```
<?php
$fp = fopen("./dados.txt", "w");
fwrite($fp, "Olá mundo do PHP!"); // grava a string no arquivo. Se o arquivo não
existir ele será criado
fclose($fp);
?>
```

Ler Arquivo

```
<?php

$fp = fopen("./dados.txt", "r");
$texto = fread($fp, 20); // lê 20 bytes do arquivo e armazena em $texto
fclose($fp);
echo $texto;
?>
```

fgets()

Esta função é usada na leitura de strings de um arquivo. fgets() lê "length - 1" bytes do arquivo. Por default lê 1024 bytes para o comprimento da linha. Se for encontrado o final da linha e o número de bytes especificados não tiver sido atingido, fgets() terminará a leitura no final da linha (ou no final do arquivo, se for o caso). Eis a sua sintaxe:

fgets(handle, length);

- handle: handle do arquivo de onde os dados serão lidos;

- length: tamanho em bytes do buffer de leitura;

Exemplo:

```
<?php
$fp = fopen("./dados.txt", "r");
$texto = fgets($fp, 3);
fclose($fp);
echo $texto;
?>
```

fgetc()

Esta função permite ler caractere por caractere de um arquivo. Seguem a sintaxe e um exemplo de utilização:

fgetc(handle);

- handle: manipulador do arquivo de onde os dados serão lidos;

Ler Todo um Arquivo

```
<?php
$fp = fopen("./dados.txt", "r");
while (!feof($fp)){
    $char .= fgetc($fp);
}
fclose($fp);
echo $char."<br><br>";
?>
```

file()

Esta função lê um arquivo completo, e armazena cada linha do arquivo como um elemento de um array. Depois de ler todo o conteúdo do arquivo, file() o fecha automaticamente, não sendo necessária uma chamada a fclose(); Vejamos a sintaxe:

Ler Arquivo Via URL

$fh = fopen("http://127.0.0.1/", "r");

file(filename);

- filename: nome ou caminho completo de um arquivo.

Exemplo:

```
<?php
// file() lê todo o arquivo
$file_lines = file("./dados.txt");
echo "Primeira linha: " . $file_lines[0]."<br>";
echo "Segunda linha: " . $file_lines[1]."<br>";
echo "Terceira linha: " . $file_lines[2];
?>
```

Além dessas funções para leitura e escrita, existe ainda uma função bastante útil, que testa se o final do arquivo foi atingido. É a função feof(), que tem a seguinte sintaxe:

feof(handle)

- handle: handle do arquivo;

Exemplo:

```
<?php
$fp = fopen("./dados.txt", "r");
while(!feof($fp)) {
    $char .= fgetc($fp);
}
fclose($fp);
echo $char;
?>
```

Contando o Número de Linhas de um arquivo

```
<?php
// Contar o número de linhas de um arquivo, iniciando com 1
$fp = "./dados.txt";
$line_count = count (file ($fp));
echo $line_count;
?>
```

Contar palavras de um arquivo mostrando duplicadas

```
/*
// Contar palavras repetidas em um arquivo
$fn = "./dados.txt";
$f_contents = preg_split ("/[\s+/", implode ("", file ($fn)));
foreach ($f_content as $palavra) {
        $ar[$palavra]++;
}
print "A seguinte palavra tem duplicatas<br>";
foreach ($ar as $palavra => $conta_palavra) {
        if (conta_palavra > 1) {
                print "Palavra: $palavra<br>Número de ocorrências:
$conta_palavra<br><br>";
        }
}
*/
```

Ler de forma inversa um arquivo, linha a linha

```
<?php
$fn = "./dados.txt";
$f_contents = array_reverse (file ($fn));
foreach ($f_contents as $linha_inversa) {
        print $linha_inversa;
}
?>
```

Ler aleatoriamente linha de arquivo

```
<?php
$fn = "./pensamentos.txt";
$f_contents = file ($fn);
srand ((double)microtime()*1000000);
$linha_aleatoria = $f_contents[ rand (0, (count ($f_contents) - 1)) ];
print $linha_aleatoria;
?>
```

Ler linha específica de arquivo

```
<?php
$fn = "./dados.txt";
$nr_linha = 38;
$f_contents = file ($fn);
$sua_linha = $f_contents [$nr_linha];
print $sua_linha;
?>
```

Operações com Diretórios

Mostrando conteúdo de diretório

```
<?php
$dn = opendir ("/home/1www/");
while ($file = readdir ($dn)) {
        print "$file<br>";
}
closedir($dn);
?>
```

Excluindo arquivos do SO

```
<?php
$fn = "./dados0.txt";
// Excluindo arquivo
$ret = unlink ($fn);
if ($ret){
         die ("Arquivo excluído!");
}else{
        die ("Erro ao excluir arquivo");
}
?>
```

Copiando arquivos

```
<?php
$fn = "./dados.txt";

if (copy ($fn, "dados0.txt")){
         die ("Arquivo '$fn' copiado para dados0.txt ");
}else{
        die ("Erro ao copiar arquivo");
}
?>
```

Processando todos os arquivos de um diretório

```
<?php
$dh = dir ("/home/1www/");
while ($entrada = $dh->read()) {
        print $entrada . "<br>";
}
$dh->close();
?>
```

Teste se Arquivo pode ser lido

```
<?php
if (is_readable('http://127.0.0.1/index.html')) {
   header('Location: http://127.0.0.1/index.html');
}else{
        echo "Este arquivo não pode ser lido!";
}

?>


<?php
// TESTAR SE ARQUIVO PERMITE LEITURA

print '<br>';
$filename = 'teste2.php';
if (is_readable($filename)) {
   echo 'O arquivo permite leitura';
} else {
   echo 'O arquivo não permite leitura';
}
print '<br>';
```

//Outro

<?php if (is_readable('http://127.0.0.1/index.html')) {

```
  header('Location: http://127.0.0.1/index.html');
```

}else{ echo "Este arquivo não pode ser lido!"; }

?>

Testar se Arquivo Permite Escrita

```
if (is_writable($filename)) {
   echo 'O arquivo permite escrita';
} else {
   echo 'O arquivo não permite escrita';
}

?>
```

Testar se Arquivo Existe

```
<?php
print '<br>';
// TESTAR SE ARQUIVO EXISTE
$filename = 'teste2.php';

if (file_exists($filename)) {
```

```
   echo "O arquivo $filename existe";
} else {
   echo "O arquivo $filename não existe";
}
?>
```

Testar se é Arquivo ou Diretório

```
<?php
print '<br>';
// TESTAR SE ARQUIVO É UM ARQUIVO COMUN OU SE É DIRETÓRIO
$filename = 'teste2.php';

$filename2 = 'c:\windows';

if (is_file($filename)) {
   echo "O arquivo $filename é comun";
}else{
  echo "O arquivo $filename não é um arquivo comun";
}
print '<br>';

if (is_file($filename2)){
   echo "O arquivo $filename2 é comun";
}else{
  echo "O arquivo $filename2 não é um arquivo comun";
}

print '<br>';

if (is_dir($filename2)){
   echo "$filename2 é um diretório";
}else{
  echo "$filename2 não é um diretório";
}
```

Outras Funções

```
is_link($diretorio)
readlink($dir_link) // retorna o path completo do link
bool symlink ( string $destino, string $linkorigem ) // Cria um link simbólico
```

Espaço Total no Disco

<?php // ESPAÇO TOTAL NO DISCO $diretorio="c:/"; print disk_total_space($diretorio); print "
";

Espaço Livre no Disco

// ESPAÇO LIVRE NO DISCO // $df contém o número de bytes disponível em "/" $df = disk_free_space("c:/"); print $df ?>

Tamanho de Diretório, número de arquivos e sub-diretórios

<?php

// CALCULANDO TAMANHO OCUPADO POR UM DIRETÓRIO, NR DE ARQUIVOS E SUBDIRETÓRIOS

// http://www.go4expert.com/forums/showthread.php?t=290

function getDirectorySize($path) {

```
$totalsize = 0;
$totalcount = 0;
$dircount = 0;
if ($handle = opendir ($path))
{
  while (false !== ($file = readdir($handle)))
  {
    $nextpath = $path . '/' . $file;
    if ($file != '.' && $file != '..' && !is_link ($nextpath))
    {
      if (is_dir ($nextpath))
      {
        $dircount++;
        $result = getDirectorySize($nextpath);
        $totalsize += $result['size'];
        $totalcount += $result['count'];
        $dircount += $result['dircount'];
      }
      elseif (is_file ($nextpath))
      {
        $totalsize += filesize ($nextpath);
        $totalcount++;
      }
    }
  }
}
closedir ($handle);
$total['size'] = $totalsize;
$total['count'] = $totalcount;
$total['dircount'] = $dircount;
return $total;
```

}

function sizeFormat($size) {

```
  if($size<1024)
  {
```

```
        return $size." bytes";
    }
    else if($size<(1024*1024))
    {
        $size=round($size/1024,1);
        return $size." KB";
    }
    else if($size<(1024*1024*1024))
    {
        $size=round($size/(1024*1024),1);
        return $size." MB";
    }
    else
    {
        $size=round($size/(1024*1024*1024),1);
        return $size." GB";
    }

}
```

// Usando

$path="D:/_xampplite/htdocs/desweb/7AplicativosExemplo/extras"; $ar=getDirectorySize($path);

echo "

Detalhes : $path

";

echo "Tamanho total : ".sizeFormat($ar['size'])."
"; echo "No. de arquivos : ".$ar['count']."
"; echo "No. de diretórios : ".$ar['dircount']."
"; ?>

Tamanho de diretório

<? // CALCULANDO TAMANHO (BYTES) OCUPADO POR UM DIRETÓRIO // http://www.weberdev.com/get_example-4171.html

function dir_size($dir) {

```
    if( !$dir or !is_dir( $dir ) )
    {
        return 0;
    }

    $ret = 0;
    $sub = opendir( $dir );
    while( $file = readdir( $sub ) )
    {
        if( is_dir( $dir . '/' . $file ) && $file !== ".." && $file !== "." )
        {
            $ret += dir_size( $dir . '/' . $file );
            unset( $file );
```

```
        }
        elseif( !is_dir( $dir . '/' . $file ) )
        {
            $stats = stat( $dir . '/' . $file );
            $ret += $stats['size'];
            unset( $file );
        }
    }
    closedir( $sub );
    unset( $sub );
    return $ret;

}
echo dir_size("D:/_xampplite/htdocs/desweb/1LinguagemPHP/php/tutoriais/trabalhando_com"); ?>


<?php
// MAIS UMA ÓTIMA FUNÇÃO PARA LER O TAMANHO DE UM DIRETÓRIO
/*
* PHP Freaks Code Library
* http://www.phpfreaks.com/quickcode.php
*
* Title: Directory Size
* Version: 1.0
* Author: Nathan Taylor aka(Lakario)
* Date: Saturday, 12/20/2003 - 12:34 PM
*
*
*
* NOTICE: This code is available from PHPFreaks.com code Library.
*         This code is not Copyrighted by PHP Freaks.
*
*         PHP Freaks does not claim authorship of this code.
*
*         This code was submitted to our website by a user.
*
*         The user may or may not claim authorship of this code.
*
*         If there are any questions about the origin of this code,
*         please contact the person who submitted it, not PHPFreaks.com!
*
*         USE THIS CODE AT YOUR OWN RISK! NO GUARANTEES ARE GIVEN!
*
* SHAMELESS PLUG: Need WebHosting? Checkout WebHost Freaks:
*                 http://www.webhostfreaks.com
*                 WebHosting by PHP Freaks / The Web Freaks!
```

- /
  // * Description / Example: // * // * This code will allow an individual to quickly obtain the size and number of files inside a directory recursively. // * // * It also includes a convenient byte value converter to kilobyte, megabyte, gigabyte, or trilobyte accordingly.

?>

Trechos do Tutorial - The right way to read files with PHP da IBM

USANDO fscanf

fscanf

Coming back to string processing, fscanf again follows the traditional C file library functions. If you're unfamiliar with it, fscanf reads field data into variables from a file.

list ($field1, $field2, $field3) = fscanf($fh, "%s %s %s");

FUNÇÃO fpassthru

No matter how you've been reading your file, you can dump the rest of your data to your standard output channel using fpassthru.

fpassthru($fh);

my_file = file_get_contents("myfilename");

echo $my_file;

Although it isn't best practice, you can write this command even more concisely as:

echo file_get_contents("myfilename");

This article is primarily about dealing with local files, but it's worth noting that you can grab, echo, and parse other Web pages with these functions, as well.

echo file_get_contents("http://127.0.0.1/");

This command is effectively the same as:

$fh = fopen("http://127.0.0.1/", "r");

fpassthru($fh);

You must be looking at this and thinking, "That's still way too much effort." The PHP developers agree with you. So you can shorten the above command to:

readfile("http://127.0.0.1/");

The readfile function dumps the entire contents of a file or Web page to the default output buffer. By default, this command prints an error message if it fails. To avoid this behavior (if you want to), try:

@readfile("http://127.0.0.1/");

Of course, if you actually want to parse your files, the single string that file_get_contents returns might be a bit overwhelming. Your first inclination might be to break it up a little bit with the split()

function.

$array = split("\n", file_get_contents("myfile"));

But why go through all that trouble when there's a perfectly good function to do it for you? PHP's file() function does this in one step: It returns an array of strings broken up by lines.

$array = file("myfile");

It should be noted that there is a slight difference between the above two examples. While the split command drops the newlines, the newlines are still attached to the strings in the array when using the file command (as with the fgets command).

PHP's power goes far beyond this, though. You can parse entire PHP-style .ini files in a single command using parse_ini_file. The parse_ini_file command accepts files similar to Listing 4.

Listing 4. A sample .ini file

```
; Comment
[personal information]
name = "King Arthur"
quest = To seek the holy grail
favorite color = Blue

[more stuff]
Samuel Clemens = Mark Twain
Caryn Johnson = Whoopi Goldberg
```

The following commands would dump this file into an array, then print that array:

```
$file_array = parse_ini_file("holy_grail.ini");
print_r $file_array;
```

The following output is the result:

```
Listing 5. Output
Array
(
    [name] => King Arthur
    [quest] => To seek the Holy Grail
    [favorite color] => Blue
    [Samuel Clemens] => Mark Twain
    [Caryn Johnson] => Whoopi Goldberg
)
```

Of course, you might notice that this command merged the sections. This is the default behavior, but you can fix it easily by passing a second argument to parse_ini_file: process_sections, which is a Boolean variable. Set process_sections to True.

$file_array = parse_ini_file("holy_grail.ini", true);

print_r $file_array;

And you'll get the following output:

Listing 6. Output

```
Array
(
    [personal information] => Array
        (
            [name] => King Arthur
            [quest] => To seek the Holy Grail
            [favorite color] => Blue
        )

    [more stuff] => Array
        (
            [Samuel Clemens] => Mark Twain
            [Caryn Johnson] => Whoopi Goldberg
        )

)
```

PHP placed the data into an easily parsable multidimensional array.

This is just the tip of the iceberg when it comes to PHP file processing. More complex functions like tidy_parse_file and xml_parse can help you handle HTML and XML documents, respectively. See Resources for details on how these particular functions work. These are well worth looking at if you'll be dealing with those types of files, but instead of considering every possible file type you might run into in detail in this article, here are a few good general rules for dealing with the functions I've described thus far.

Good practice

Never assume that everything in your program will work as planned. For example, what if the file you're looking for has moved? What if the permissions have been altered and you're unable to read the contents? You can check for these things in advance by using file_exists and is_readable.

Listing 7. Use file_exists and is_readable

```
$filename = "myfile";
if (file_exists($filename) && is_readable ($filename)) {
        $fh = fopen($filename, "r");
        # Processing
        fclose($fh);
}

In practice, however, such code is probably overkill. Processing the return
value of fopen is simpler and more accurate.

if ($fh = fopen($filename, "r")) {
        # Processing
        fclose($fh);
```

```
}
```

Final do trecho do tut IBM----------

Lê e imprime todo o conteúdo de um arquivo CSV

```
<?php
// Lê e imprime todo o conteúdo de um arquivo CSV
$row = 1;
$handle = fopen ("test.csv","r");
while ($data = fgetcsv ($handle, 1000, ",")) {
   $num = count ($data);
   print "<p> $num campos na linha $row: <br>";
   $row++;
   for ($c=0; $c < $num; $c++) {
       print $data[$c] . "<br>";
   }
}
fclose ($handle);
?>
<pre>

Outra:
<pre>
<?php

define('CSV_BOTH', 1);
define('CSV_ASSOC', 2);
define('CSV_NUM', 3);

function parse_csv($filename, $result_type = CSV_BOTH) {
   if(!file_exists($filename)) {
       die("file (" . $filename . ") does not exist\n");
   }

   $lines = file($filename);

   $title_line = trim(array_shift($lines));
   $titles = split(",", $title_line);

   $records = array();
   foreach($lines as $line_num => $line) {
       $subject = trim($line);
       $fields = array();
       for($field_num = 0; $field_num < count($titles); $field_num++) {
           if($subject{0} == '"') {
               preg_match('/^"(([^"]|\\")*)",?(.*)$/', $subject, $matches);

               $value = $matches[1];
               $subject = $matches[3];

               if($result_type == CSV_BOTH || $result_type == CSV_ASSOC) {
                   $fields[$titles[$field_num]] = $value;
               }
```

```
                if($result_type == CSV_BOTH || $result_type == CSV_NUM) {
                    $fields[$field_num] = $value;
                }
            } else {
                preg_match('/^([^,]*),?(.*)$/', $subject, $matches);

                $value = $matches[1];
                $subject = $matches[2];

                if($result_type == CSV_BOTH || $result_type == CSV_ASSOC) {
                    $fields[$titles[$field_num]] = $value;
                }

                if($result_type == CSV_BOTH || $result_type == CSV_NUM) {
                    $fields[$field_num] = $value;
                }
            }
        }

        $records[] = $fields;
    }

    return $records;
}

?>
<pre>

This version is conditional - it only adds quotes if needed:
<pre>
<?
function csv_escape($str) {
    $str = str_replace(array('"', ',', "\n", "\r"), array('""', ',', "\n", "\r"),
$str, &$count);
    if($count) {
        return '"' . $str . '"';
    } else {
        return $str;
    }
}
?>



<?php
$caminho = "/home/httpd/html/index.php";
$arquivo = basename ($caminho);        // $arquivo = "index.php"
$arquivo = basename ($caminho,".php"); // $arquivo = "index"
?>
```

rmdir -- Remove um diretório

This functions deletes or empties the directory. Without using recursive functions!

```
<?php
```

```
/**
 * Removes the directory and all its contents.
 *
 * @param string the directory name to remove
 * @param boolean whether to just empty the given directory, without deleting
the given directory.
 * @return boolean True/False whether the directory was deleted.
 */
function deleteDirectory($dirname,$only_empty=false) {
   if (!is_dir($dirname))
       return false;
   $dscan = array(realpath($dirname));
   $darr = array();
   while (!empty($dscan)) {
       $dcur = array_pop($dscan);
       $darr[] = $dcur;
       if ($d=opendir($dcur)) {
           while ($f=readdir($d)) {
               if ($f=='.' || $f=='..')
                   continue;
               $f=$dcur.'/'.$f;
               if (is_dir($f))
                   $dscan[] = $f;
               else
                   unlink($f);
           }
           closedir($d);
       }
   }
   $i_until = ($only_empty)? 1 : 0;
   for ($i=count($darr)-1; $i>=$i_until; $i--) {
       echo "\nDeleting '".$darr[$i]."' ... ";
       if (rmdir($darr[$i]))
           echo "ok";
       else
           echo "FAIL";
   }
   return (($only_empty)? (count(scandir)<=2) : (!is_dir($dirname)));
}

?>

//Outra

<?php
/* Function to remove directories, even if they contain files or
subdirectories.  Returns array of removed/deleted items, or false if nothing
was removed/deleted.

by Justin Frim.  2007-01-18

Feel free to use this in your own code.
*/

function rmdirtree($dirname) {
   if (is_dir($dirname)) {    //Operate on dirs only
```

```
       $result=array();
       if (substr($dirname,-1)!='/') {$dirname.='/';}    //Append slash if
necessary
       $handle = opendir($dirname);
       while (false !== ($file = readdir($handle))) {
           if ($file!='.' && $file!= '..') {    //Ignore . and ..
               $path = $dirname.$file;
               if (is_dir($path)) {    //Recurse if subdir, Delete if file
                   $result=array_merge($result,rmdirtree($path));
               }else{
                   unlink($path);
                   $result[].=$path;
               }
           }
       }
       closedir($handle);
       rmdir($dirname);    //Remove dir
       $result[].=$dirname;
       return $result;    //Return array of deleted items
   }else{
       return false;    //Return false if attempting to operate on a file
   }
}
?>
```

rename -- Renomear um arquivo ($antigo, $novo)

```
<?php
  rename("/tmp/tmp_file.txt", "/home/user/login/docs/my_file.txt");
?>
```

unlink -- Apaga um arquivo

mkdir -- Criar um diretório

mkdir ("/path/to/my/dir", 0700);

is_file -- Diz se o arquivo é um arquivo comum (não é diretório)

file -- Le um arquivo inteiro para um array

```
<?php
// Le um arquivo em um array. Nesse exemplo você pode obter via HTTP para obter
// o código fonte HTML de uma URL.
$lines = file ('http://www.exemplo.com/');

// Roda através do array, mostrando o fonte HTML com numeração de linhas.
foreach ($lines as $line_num => $line) {
   echo "Linha #<b>{$line_num}</b> : " . htmlspecialchars($line) . "<br>\n";
}
// Outro exemplo, onde obtemos a página web inteira como uma string. Veja também
file_get_contents().
$html = implode ('', file ('http://www.exemplo.com/'));
?>
```

file_exists -- Checa se um arquivo ou diretório existe

```
$filename = '/caminho/para/qualquer.txt';

if (file_exists($filename)) {
   print "O arquivo $filename existe";
} else {
   print "O arquivo $filename não existe";
}
```

disk_free_space -- Retorna o espaço disponivel no diretório

Receber Conteúdo de URL

<?php // Trazer conteúdo de arquivo ou de página para string

// Define a context for HTTP. $aContext = array(

```
  'http' => array(
      'proxy' => 'tcp://10.0.0.1:3128', // This needs to be the server and the
port of the NTLM Authentication Proxy Server.
      'request_fulluri' => True,
      ),
  );
```

$cxContext = stream_context_create($aContext);

// Now all file stream functions can use this context. $sFile =
file_get_contents("http://www.google.com", False, $cxContext); echo $sFile; ?>

Recursively find files by filename pattern

http://snippets.dzone.com/posts/show/4147

Scans a directory, and all subdirectories for files, matching a regular expression. Each match is sent to the callback provided as third argument. A simple example:

function my_handler($filename) {

```
 echo $filename . "\n";
```

} find_files('c:/', '/php$/', 'my_handler');

And the actual snippet

function find_files($path, $pattern, $callback) {

```
 $path = rtrim(str_replace("\\", "/", $path), '/') . '/';
```

```
 $matches = Array();
 $entries = Array();
 $dir = dir($path);
 while (false !== ($entry = $dir->read())) {
   $entries[] = $entry;
 }
 $dir->close();
 foreach ($entries as $entry) {
   $fullname = $path . $entry;
   if ($entry != '.' && $entry != '..' && is_dir($fullname)) {
     find_files($fullname, $pattern, $callback);
   } else if (is_file($fullname) && preg_match($pattern, $entry)) {
     call_user_func($callback, $fullname);
   }
 }

}


<?php

$file='file.inc';
$php_errormsg = 'Erro na operação do arquivo!';

// Determinaod onde o penteiro está
/*
  $fr = fopen($file, 'r');
  fseek($fr, rand(1, 1024));
  $offset = ftell($fr);
  echo "We randomly landed on the $offset byte of the file.<BR>";
*/
?>

<?php
// Escreve no início do arquivo
$file_name= $file;
if(file_exists($file_name))
{
    //open file for writng and place pointer at the end
    $handle = fopen($file_name, 'a+');

    if(!$handle)
    {
        die("couldn't open file <i>$file_name</i>");
    }

    //place pointer at the beginning of the file.
    //rewind($handle);

    $data = fgets($handle, 4096);
```

```
    // move de volta para o inicio do arquivo
    // o mesmo que rewind($fp);
    fseek($handle, 0);

    //write to file
    //fwrite($handle, "Student ID: 12345");
    echo "success writing to start of file";
} else {
    echo "file <i>$file_name</i> doesn't exists";
}

fclose($handle);

// Posição atual
$fp = fopen($file, 'a+');
$data = fgets($fp, 4096);
echo '<br>Onde estamos '.ftell($fp).'<br>';
fclose($fp);


// Retornar a última linha do arquivo

function readlastline($file)
{
      $fp = @fopen($file, "r");
      $pos = -1;
      $t = " ";
      while ($t != "\n") {
          fseek($fp, $pos, SEEK_END);
          $t = fgetc($fp);
          $pos = $pos - 1;
      }
      $t = fgets($fp);
      fclose($fp);
      return $t;
}
print readlastline($file);

?>

<?php
/*
$file_name= $file;
if(file_exists($file_name))
{
    //open file for writng and place pointer at the end
    $handle = fopen($file_name, 'a+');
```

```
    if(!$handle)
    {
        die("couldn't open file <i>$file_name</i>");
    }

    fwrite($handle, "Student GPA: 2.9");
    echo "success writing to file";
} else {
    echo "file <i>$file_name</i> doesn't exists";
}

fclose($handle);
*/
?>


<?php

/*
// move de volta para o inicio do arquivo
// o mesmo que rewind($fp);
fseek($fp, 0);
// Escrevendo com quebra de linha
$file_name = "myfile.txt";
if(file_exists($file_name))
{
    //open file for writng and place pointer at the end
    $handle = fopen($file_name, 'w');

    if(!$handle)
    {
        die("couldn't open file <i>$file_name</i>");
    }
    $str.= "Student Name: Mark Fendisen\r\n";
    $str.= "Student ID: 12345\r\n";
    $str.= "Student GPA: 2.9\r\n";

    fwrite($handle, $str);
    echo "success writing to file";
} else {
    echo "file <i>$file_name</i> doesn't exists";
}

fclose($handle);
*/
?>
```

Referência
http://phpbrasil.com/articles/print.php/id/310

9.4 - Trabalhando com Diretórios no PHP

```
$Dir = “C:\\PHP”;
$DirOpen = opendir($Dir);
while ($CurFile = readdir($DirOpen)) {
        echo $CurFile . “<br />”;
}
closedir($DirOpen);

$Dir = “C:\\PHP”;
$DirEntries = scandir($Dir);
foreach ($DirEntries as $Entry) {
        echo $Entry . “<br />”;
}

mkdir($diretorio);

is_writable($dir);

file_exists($dir);

is_dir($dir);

copy($arq_origem, $arq_destino);

unlink($arquivo);

rmdir($dir);



<?php
// Listar conteúdo de diretório
function listar_dir($arquivo){
    $dn = opendir ($arquivo);
    while ($file = readdir ($dn)) {
      return "$file<br>";
    }
    closedir($dn);
}

// Espaço total em disco em KB
function total_space($dir){
    return round(disk_total_space($dir)/(1024*1024),2) . ' KB';
}
//print total_space('/home/ribafs');

// Espaço livre em disco em KB
```

```
function free_space($dir){
    return round(disk_free_space($dir)/(1024*1024),2) . ' KB';
}
//print free_space('/home/ribafs');

// Receber tamanho de diretório recursivamente
function getdirsize($dir) {
    // Initialize the size value
    $size = 0;
    // Get the files and directories
    // inside the given directory
    $files = scandir($dir);
    // Loop through the files
    foreach ($files as $file) {
        // Filter-out .. and .
        if (strlen(str_replace('.', '', $file))>0) {
            if (is_dir($dir.'/'.$file)) {
                // If the item is a directory
                // run this function again and
                // get the size of the files inside
                $size += getdirsize($dir.'/'.$file);
            } else {
                // If it's a file, calculate the size
                // and add it to the size variable
                $size += filesize($dir.'/'.$file);
            }
        }
    }
    // Finally, return the calculated size
    return $size;
}
//print getdirsize('/home/ribafs/uteis');

function dir_size($directory, $type, $round) {
  $dir = scandir($directory);
  $size = 0;
  foreach($dir as $value) {
    if(is_dir($directory.'/'.$value)) {
      $init = scandir($directory.'/'.$value);
        foreach($init as $subfile) {
          $filesize = filesize($directory.'/'.$value.'/'.$subfile);
          $size = $size + $filesize;
        }
      if(is_file($directory.'/'.$value)) {
        $filesize = filesize($directory.'/'.$value);
        $size = $size = $filesize;
      }
    }
```

```
  }
  if($type==1) {
    // returns KB
    $size = $size / 1024;
    if(isset($round))
      $size = round($size, $round);
    echo $size.'KB';
  }
  elseif($type==2) {
    $size = $size / 1048576;
    if(isset($round))
      $size = round($size, $round);
    echo $size.'MB';
  }
  elseif($type==3) {
    $size = $size / 1073741824;
    if(isset($round))
      $size = round($size, $round);
    echo $size.'GB';
  }
  else {
    if(isset($round))
      $size = round($size, $round);
    echo $size.'Bytes';
  }
}
/*
 * #1 with how you would like the data to be shown (1 = KB, 2 = MB, 3 = GB, 0 or nothing is Bytes), and #2 is optional, if you want it to make the number perttier enter how many numbers AFTER the . to have, so if you put in 2, and the number before it rounded it was 3.4325214 it would be something like, 3.44 or so after it rounds it.
 */
dir_size("/home/ribafs", 1, 2);


function dir_size2( $dir )
{
   if( !$dir or !is_dir( $dir ) )
   {
      return 0;
   }

   $ret = 0;
   $sub = opendir( $dir );
   while( $file = readdir( $sub ) )
   {
      if( is_dir( $dir . '/' . $file ) && $file !== ".." && $file !== "." )
      {
```

```
            $ret += dir_size( $dir . '/' . $file );
            unset( $file );
        }
        elseif( !is_dir( $dir . '/' . $file ) )
        {
            $stats = stat( $dir . '/' . $file );
            $ret += $stats['size'];
            unset( $file );
        }
    }
    closedir( $sub );
    unset( $sub );
    return $ret;
}

//print dir_size2('/home/ribafs/htdocs');

/**
 * Removes the directory and all its contents.
 *
 * @param string the directory name to remove
 * @param boolean whether to just empty the given directory, without deleting the given directory.
 * @return boolean True/False whether the directory was deleted.
 */
function delete_dir($dirname,$only_empty=false) {
  if (!is_dir($dirname))
     return false;
  $dscan = array(realpath($dirname));
  $darr = array();
  while (!empty($dscan)) {
     $dcur = array_pop($dscan);
     $darr[] = $dcur;
     if ($d=opendir($dcur)) {
        while ($f=readdir($d)) {
           if ($f=='.' || $f=='..')
              continue;
           $f=$dcur.'/'.$f;
           if (is_dir($f))
              $dscan[] = $f;
           else
              unlink($f);
        }
        closedir($d);
     }
  }
  $i_until = ($only_empty)? 1 : 0;
  for ($i=count($darr)-1; $i>=$i_until; $i--) {
     echo "\nDeleting '".$darr[$i]."' ... ";
```

```
    if (rmdir($darr[$i]))
       echo "ok";
    else
       echo "FAIL";
  }
  return (($only_empty)? (count(scandir)<=2) : (!is_dir($dirname)));
}

/* Function to remove directories, even if they contain files or
subdirectories.  Returns array of removed/deleted items, or false if nothing
was removed/deleted.

by Justin Frim.  2007-01-18

Feel free to use this in your own code.
*/

function rmdirtree($dirname) {
  if (is_dir($dirname)) {    //Operate on dirs only
    $result=array();
    if (substr($dirname,-1)!='/') {$dirname.='/';}    //Append slash if necessary
    $handle = opendir($dirname);
    while (false !== ($file = readdir($handle))) {
      if ($file!='.' && $file!= '..') {    //Ignore . and ..
        $path = $dirname.$file;
        if (is_dir($path)) {    //Recurse if subdir, Delete if file
          $result=array_merge($result,rmdirtree($path));
        }else{
          unlink($path);
          $result[].=$path;
        }
      }
    }
    closedir($handle);
    rmdir($dirname);    //Remove dir
    $result[].=$dirname;
    return $result;    //Return array of deleted items
  }else{
    return false;    //Return false if attempting to operate on a file
  }
}

?>
```

9.5 - Trabalhando com Path no PHP

Exemplos simples de uso de funções para path do PHP

```
<?php
   $path=dirname(realpath($_SERVER['SCRIPT_FILENAME']));
   $path=substr($path,0,strlen($path) - 5);
echo "<br>Path deste script sem 5 finais caracteres - " . $path;

echo "<br><br>Diretório atual - ".dirname(__FILE__);
echo "<br>Caminho completo do script atual - ".__FILE__;

echo "<br>URL do script atual - " . "http://" . $_SERVER['HTTP_HOST'] .
$HTTP_SERVER_VARS["SCRIPT_NAME"];
?>

<?php
$path = "/etc/passwd";
$file = dirname($path); // $file is set to "/etc"
?>

<?php
$path = "/home/httpd/html/index.php";
$file = basename($path);        // $file is set to "index.php"
$file = basename($path, ".php"); // $file is set to "index"
?>

<?php
echo dirname($_SERVER["REQUEST_URI"]);
?>
```

Recebendo o Path Absoluto do Script Atual

dirname(__FILE__)

Recebendo o path relativo do webserver do script atual

```
function GetRelativePath($path){
   $npath = str_replace('\\', '/', $path);
   return str_replace(GetVar('DOCUMENT_ROOT'), '', $npath);
}
```

GetRelativePath(dirname(__FILE__));

<?php

if (DIRECTORY_SEPARATOR=='/')

```
 $absolute_path = dirname(__FILE__).'/';
```

else

```
 $absolute_path = str_replace('\\\\', '/', dirname(__FILE__)).'/';
```

?>

Resultará em um path absoluto no estilo UNIX que funciona também em PHP5 sob Windows.

Em algumas instalações (< 4.4.1) $_SERVER['REQUEST_URI'] não está configurado, usado o código para corrigir:

```
<?php

if (!isset($_SERVER['REQUEST_URI'])) {
   $_SERVER['REQUEST_URI'] = substr($_SERVER['PHP_SELF'],1);
   if (isset($_SERVER['QUERY_STRING'])) $_SERVER['REQUEST_URI'].='?'.
$_SERVER['QUERY_STRING'];
}

?>

$my_uri = "http://" . $_SERVER['HTTP_HOST'] . $HTTP_SERVER_VARS["SCRIPT_NAME"];

// então

<?php echo ("$my_uri");?>

<?php
include("{$_SERVER['DOCUMENT_ROOT']}/includes/my_include.php");
?>
```

Você pode usar isso para receber o diretório pai:

dirname(dirname(__FILE__))

...include a file relative to file path:

include(dirname(__FILE__) . '/path/relative/file_to_include.php');

Isso colocará ambos os paths "www" e "file" de forma fácil para transportar o array.

```
<?php
// build the www path:
$me = $_SERVER['PHP_SELF'];
$Apathweb = explode("/", $me);
$myFileName = array_pop($Apathweb);
$pathweb = implode("/", $Apathweb);
$myURL = "http://".$_SERVER['HTTP_HOST'].$pathweb."/".$myFileName;
$PAGE_BASE['www'] = $myURL;
```

```
// build the file path:
strstr( PHP_OS, "WIN") ? $strPathSeparator = "\\" : $strPathSeparator = "/";
$pathfile = getcwd ();
$PAGE_BASE['physical'] = $pathfile.$strPathSeparator.$myFileName;

// this is so you can verify the results:
$www = $PAGE_BASE['www'];
$physical = $PAGE_BASE['physical'];

echo "$physical<p>";
echo "$www<p>";
?>
```

retornará algo como:

Windows:

F:\dev\Inetpub\wwwroot\somedirectory\index.php

http://devserver/somedirectory/index.php

Unix:

/home/somepathto/gieson.com/webroot/index.php

http://www.gieson.com/index.php

Path absoluto do script em execução

$path=dirname(realpath($_SERVER['SCRIPT_FILENAME']));

```
print 'Diretório atual=> '.dirname($_SERVER["SCRIPT_FILENAME"]);
print '<br>Diretório atual=> '.dirname(__FILE__);
print '<br>Diretório anterior=> '.dirname(dirname(__FILE__));
print '<br>Diretório anterior 2 níveis=> '.dirname(dirname(dirname(__FILE__)));
print '<br>Diretório raiz geral => '.dirname($_SERVER['PHP_SELF']);
print '<br>Diretório atual isolado => '.basename(dirname(__FILE__));
print '<br>Arquivo atual isolado => '.$_SERVER['REQUEST_URI'];
print '<br>Arquivo atual path completo => '.__FILE__;
print '<br>Arquivo raiz geral => '.dirname($_SERVER["SCRIPT_NAME"]);
```

```
 $site_path = realpath(dirname(__FILE__));
 define ('__SITE_PATH', $site_path);
```

```
function fixPath ($path)
(
      return dirname($path . '/.');
)
```

```
fixPath('/one');               // '/one'
fixPath('/one/');              // '/one'
fixPath('/one//two');          // '/one/two'
fixPath('/one////two////');    // '/one/two'
```

Checar permissão de escrita:

```
error_reporting(E_ALL);
$dir_name = '/var/www/virtual/phpintra/htdocs/php/';
do {
   $b_is_writable = is_writable($dir_name);
   echo sprintf("Dir[%s]Writable[%s]\n", $dir_name, $b_is_writable? 'YES':'NO');
}while (($dir_name = dirname($dir_name)) !='/');

//Restringindo acesso direto
if (!defined("BASE_PATH")) define('BASE_PATH',
dirname($_SERVER['SCRIPT_NAME'])=='/' ? './' : str_repeat("../",
substr_count(dirname($_SERVER["SCRIPT_NAME"]), "/")));


function myRealPath($path) {
   // Check if path begins with "/". => use === with strpos
   if (strpos($path,"/") === 0) {
      $path = $_SERVER['DOCUMENT_ROOT'].$path;
   }
   else {
      // Strip slashes and convert to arrays.
      $currentDir = preg_split("/\//",dirname($_SERVER['PATH_TRANSLATED']));
      $newDir = preg_split('/\//',$path);
      // Drop one directory from the array for each ".."; add one otherwise.
      foreach ($newDir as $dir) {
         if ($dir == "..")
            array_pop($currentDir);
         elseif ($dir != ".") //test for "." which represents current dir (do nothing in that case)
            array_push($currentDir,$dir);
      }
      // Return the slashes.
      $path = implode($currentDir,"/");
   }
   return $path;
}
//print myRealPath("./");

$p = getcwd(); // recebe o diretório atual
//echo $p;

$dir_path = str_replace( $_SERVER['DOCUMENT_ROOT'], "", dirname(realpath(__FILE__)) ) .
```

```
DIRECTORY_SEPARATOR;
//print $dir_path;

/*
echo realpath(dirname(__FILE__)); //Similar ao getcwd()

$inc = $_SERVER["DOCUMENT_ROOT"]."/inc";
include $inc."/config.php";
include $about."/setup.php";
include $inc."/header.php";
include $inc."/topnav.php";
include $inc."/left.php";
include $inc."/right.php";


echo __FILE__AbsolutePath(__FILE__)."\n";

function __FILE__AbsolutePath($Filename)
 {
      switch (PHP_OS)
       {
            case "WINNT": $needle = "\\"; break;
            case "Linux": $needle = "/";  break;
            default:      $needle = "/";  break; // TODO : Mac check
       }
      $AbsPath = substr($Filename, 0, strrpos($Filename, $needle));
      return $AbsPath;
 }
*/
function __FILE__AbsolutePath()
 {
      $d = debug_backtrace();
      $Filename = $d[0]['file'];

      switch (PHP_OS)
       {
            case "WINNT": $needle = "\\"; break;
            case "Linux": $needle = "/";  break;
            default:      $needle = "/";  break; // TODO : Mac check
       }
      $AbsPath = substr($Filename, 0, strrpos($Filename, $needle));
      return $AbsPath;
 }
//print __FILE__AbsolutePath() ;
```

9.6 - Trabalhando com Includes em PHP

INCLUDE - inclue e avalia o conteúdo do arquivo.
REQUIRE - também inclue e avalia o conteúdo do arquivo incluído.

A diferença entre ambos é que o include ao encontrar um erro, lança um warning apenas, enquanto que o require lança um Fatal Error, que pára o processamento.

Uso do require: para códigos que requerem maior segurança.

INCLUDE_ONCE e REQUIRE_ONCE - são semelhantes ao include e require, sendo que estes incluem um arquivo somente uma vez.

Mostrando uso do require_once:

echo.php
```
<?php
echo "Hello";
?>
```

teste.php
```
<?php
require('echo.php');
require_once('echo.php');
?>
```

Executar teste.php
saída: "Hello".

Agora teste2.php:
```
<?php
require('echo.php');
require('echo.php');
?>
```

Executar teste2.php
saída: "HelloHello".

Agora teste3.php:
```
<?php
require_once('echo.php');
require('echo.php');
?>
```

Executar teste3.php
saída: "HelloHello".

Ou seja, ao encontrar require_once, ele verifica se o arquivo já foi incluído, e somente o incluirá

novamente se ele ainda não tiver sido incluído.

```
<?php
// Isto está errado e não funcionará como desejado
if ($condition)
  include $arquivo;
else
  include $outro;


// E este está correto
if ($condition) {
  include $arquivo;
} else {
  include $outro;
}

?>

<?php
$path="/full/path/to/script/";
if (getdomain($path) == 'yourdomain'){
    include($path.'somefile.php');
}
?>

"variables.php"
<?php
$includer = basename($_SERVER['SCRIPT_NAME']);

switch ($includer) {
  case 'a.php':
  $this_variable = 'included by script a.php';
  break;

  case 'b.php':
  $this_variable = 'included by script b.php';
  break;

  default:
  $this_variable = 'included by unkown script';
}
echo $this_variable;
?>
Test with 3 different files "a.php", "b.php", "c.php", all with the same content:
<?php
include 'variables.php';
?>
```

9.7 - Trabalhando com Funções no PHP

O PHP conta com uma grande quantidade de boas bibliotecas de funções internar sobre vários assuntos, como strings, arrays, arquivos, diretórios, data e hora, etc.

Além das funções embutidas também podemos criar nossas próprias funções quando não encontrarmos uma função que atenda as nossas necessidades.

Funções são blocos de código, conjuntos de instruções, que ficam quietos, sem vida, até que sejam chamadas em algum lugar do código. Funções reduzem o trabalho de digitação, o tamanho dos scripts em geral, os erros, reaproveitando código, facilitando e organizando o trabalho de desenvolvimento.

Exemplos de funções definidas pelo usuário

```
function quadrado($numero) {
        print $numero*$numero;
}
```

Executando:

quadrado(6); // Saída: 36

Variáveis globais

```
$var1 = 5;
function testeGlobal1(){
        $var1 = 1;
        print "<br>Valor de \$var1: $var1";
}
echo testeGlobal1();
```

Variável externa acessível na função

```
$var2 = 10;
function testeGlobal2(){
        global $var2;
        print "<br>Valor de \$var2 $var2";
}
echo testeGlobal2();

$var5 = 15;
function testeGlobal5(){
        $var5 = 5;
    print "<br><br>A variável global vale $GLOBALS[var5], ";
    print "Já a variável local vale $var5<br><br>";
}

testeGlobal5();

function cliente($codigo, $idade = 18){
        print "Código = $codigo, Idade = $idade";
}

cliente(1);  //Exibirá: Código = 1, Idade = 18

function cubo($num){
        return ($num*$num*$num);
}

$var1 = 2 * cubo(5);echo "<br>".$var1;
```

Acessando variáveis Externas

Normalmente de dentro de uma função não temos acesso às variáveis que estão fora dela. Mas veja como contornar isso:

Variável externa inacessível

```
$var1 = 5;
function testeGlobal1(){
        $var1 = 1;
        print ”Valor de \$var1: $var1”;
}
echo testeGlobal1();
Variável externa acessível na função

$var2 = 10;
function testeGlobal2(){
        global $var2;
        print ”Valor de \$var2 $var2”;
}
echo testeGlobal2();
```

Outra alternativa é usar o array $GLOBALS[], que contém todas as variáveis globais. Veja um exemplo:

```
$var5 = 15;
function testeGlobal5(){
        $var5 = 5;
        print "<br><br>A variável global vale $GLOBALS[var5], ";
        print "Já a variável local vale $var5<br><br>";
}

testeGlobal5();
```

Variáveis estáticas

```
function contador(){
        static $x = 0;
        return $x++;
}
echo "<br>";
echo contador();echo contador();echo contador();
//A saída será: 012

function contador2(){
        $x = 0;
        return $x++;
}
echo "<br>";
echo contador2();echo contador2();echo contador2();
//A saída será: 000.

  function staticfunction() {
   static $count = 0;
```

```
   $count++;
        if ($count==1){
       echo "A Função foi executada $count vez<br>";
        }else{
           echo "A Função foi executada $count vezes<br>";
        }
  }

  for($i = 0; $i < 5; $i++) {
   staticfunction();
  }

function Teste1()
{
    static $a = 0;
    echo $a;
    $a++;
}

for($x=0;$x<=10;$x++){
        echo Teste1()." ";
}
```

Recursiva

```
function Teste()
{
    static $count = 0;

    $count++;
    echo $count." ";
    if ($count < 10) {
        Teste ();
    }
    $count--;
}

Teste();
```

Declarando variáveis static

```
function foo(){
    static $int = 0;          // correro
   // static $int = 1+2;        // errado (é uma expressão)
   // static $int = sqrt(121);  // errado  (é uma expressão também)

    $int++;
    echo $int;
}
```

```
function aumentoSalario($sal, $perc=5){
        $salario = $sal * $perc/100;
        echo $salario;
}
echo "<br>Aumento: " . aumentoSalario(1956);

function redirecionar($url){
        header("Location: $url");
}
echo "<br>";
redirecionar("http://ribafs.phpnet.us/");
echo "<br>";
```

Retornar mais de um valor de uma função

usa-se arrays e list()

array() retorna e list() exibe

//Exemplo:

```
function recebe_hoje(){
        $data_abreviada=date("d/m/Y");
        $data_extensa=date("l, d F \d\e Y");

        return array($data_abreviada, $data_extensa);
}

list($data_abreviada, $data_extensa)=recebe_hoje();
        print $data_extensa;
echo "<br>";
        print $data_abreviada;


<h2>Declaração dinâmica de função</h2>

<pre>
if ($f == 1){
        function f1(){
                echo "funcao1";
        }
}else{
        function f2(){
                echo "funcao2";
        }
}

<h2>Retornando o número de argumentos de uma função</h2>

<pre>
function ret_args_funcao() {
    $numargs = func_num_args();
    echo "Número de argumentos: $numargs<br>\n";
    if ($numargs >= 2) {
    echo "Segundo argumento vale : " . func_get_arg (1) . "<br>\n";
    }
    $arg_list = func_get_args();
```

```
    for ($i = 0; $i < $numargs; $i++) {
    echo "Argumento $i vale: " . $arg_list[$i] . "<br>\n";
    }
}

ret_args_funcao (1, 2, 3);
```

Passando Argumentos através de Funções

O default é 'por valor', que passa uma cópia do valor da variável.

Também podemos passar 'por referência', onde se passa o endereço da própria variável. Quando atribuímos uma variável a outra passando como referência, não fazemos cópia alguma, mas sim passamos o endereço da variável original, portanto qualquer alteração nesta referência refletirá na variável original.

ByRef é mais eficiente em termos de memória ao lidar com variáveis e arrays grandes e também permite alterar o valor da variável original, o que não acontece com o ByVal, mas a vantagem de desempnho somente é percebida em grandes arrays em grandes loops.

Para maior segurança setar:

allow_call_time_pass_reference no php.ini

Impede a passagem de valores por referência nas chamadas, mas permite somente na definição das funções.

$var1 = & $var2;

Agora ambas apontam para o mesmo endereço e valor.

Reter valor de variáveis entre chamadas (static)

(Guarda o valor da última chamada até o final da execução do script, tantas vezes quantas a função for chamada).

Exemplo:

```
$valor = 4;
$ref = &$valor;

$ref = 3;

$valor = 4;
$ref = &$valor; // Aqui tornamos ambas as variáveis com o mesmo endereço
                // O que alterarmos em $ref alteramos em $valor

$ref = 3;       // Com isso também alteramos $valor para 3, veja abaixo.
echo $valor . "<br>";

$valor=0;          // Com isso também alteramos $ref para 0, veja abaixo.

echo $ref;
```

Por Valor

```
function val_subtracao($num1, $num2){
        if($num1 < $num2){
                die("Números negativos");
        }else{
                $return_result=0;
                while($num1 > $num2){
                        $num1 = $num1 - 1;
                        $return_result = $return_result + 1;
                }
        }
        return($return_result);
}

$primeiro_op=493;
$segundo_op=355;
$resultado1 = val_subtracao($primeiro_op, $segundo_op);
print ("Resultado1 é $resultado1<br>");
$resultado2 = val_subtracao($primeiro_op, $segundo_op);
print("Resultado2 é $resultado2<br>");
```

Por Referência

```
function subtracao_ref(&$num1, &$num2){
        if($num1 < $num2){
                die("Números negativos");
        }else{
                $return_result=0;
                while($num1 > $num2){
                        $num1 = $num1 - 1;
                        $return_result = $return_result + 1;
                }
        }
        return($return_result);
}

$primeiro_op=493;
$segundo_op=355;
$resultado1 = subtracao_ref($primeiro_op, $segundo_op);
print ("<br><br>Resultado1 é $resultado1<br>");
$resultado2 = subtracao_ref($primeiro_op, $segundo_op);
print("Resultado2 é $resultado2<br>");

echo "Agora, se se nós executarmos exatamente a mesma chamada da subtração como
fizemos a primeira vez, receberemos a saída:
resultado1 é 138 e resultado2 é 0";


/*
Sugestão de chamada de função:

if (nome_funcao($argumento){
        ....
        ....
```

```
}else{
        ....
}

*/
?>

<?php
// Retorna o tipo e o valor de variável
function ss_as_string (&$thing, $column = 0) {
    if (is_object($thing)) {
        return ss_object_as_string($thing, $column);
    }
    elseif (is_array($thing)) {
        return ss_array_as_string($thing, $column);
    }
    elseif (is_double($thing)) {
        return "Double(".$thing.")";
    }
    elseif (is_long($thing)) {
        return "Long(".$thing.")";
    }
    elseif (is_string($thing)) {
        return "String(".$thing.")";
    }
    else {
        return "Unknown(".$thing.")";
    }
}

// Retorna o tipo e o valor de array
function ss_array_as_string (&$array, $column = 0) {
    $str = "Array(<BR>\n";
    while(list($var, $val) = each($array)){
        for ($i = 0; $i < $column+1; $i++){
            $str .= "    ";
        }
        $str .= $var.' ==> ';
        $str .= ss_as_string($val, $column+1)."<BR>\n";
    }
    for ($i = 0; $i < $column; $i++){
        $str .= "    ";
    }
    return $str.')';
}

// Retorna o tipo e o valor de objeto
function ss_object_as_string (&$object, $column = 0) {
    if (empty($object->classname)) {
        return "$object";
    }
    else {
        $str = $object->classname."(<BR>\n";
        while (list(,$var) = each($object->persistent_slots)) {
            for ($i = 0; $i < $column; $i++){
                $str .= "    ";
            }
```

```
            global $$var;
            $str .= $var.' ==> ';
            $str .= ss_as_string($$var, column+1)."<BR>\n";
        }
        for ($i = 0; $i < $column; $i++){
            $str .= "    ";
        }
        return $str.')';
    }
}

$var="Riba";
echo ss_as_string($var);
//echo ss_as_string($GLOBALS);
```

Otimização do tempo de execução

```
function ss_timing_start ($name = 'default') {
    global $ss_timing_start_times;
    $ss_timing_start_times[$name] = explode(' ', microtime());
}

function ss_timing_stop ($name = 'default') {
    global $ss_timing_stop_times;
    $ss_timing_stop_times[$name] = explode(' ', microtime());
}

function ss_timing_current ($name = 'default') {
    global $ss_timing_start_times, $ss_timing_stop_times;
    if (!isset($ss_timing_start_times[$name])) {
        return 0;
    }
    if (!isset($ss_timing_stop_times[$name])) {
        $stop_time = explode(' ', microtime());
    }
    else {
        $stop_time = $ss_timing_stop_times[$name];
    }
    // do the big numbers first so the small ones aren't lost
    $current = $stop_time[1] - $ss_timing_start_times[$name][1];
    $current += $stop_time[0] - $ss_timing_start_times[$name][0];
    return $current;
}

?>
```

Função com parâmetro default

Parâmetro default é aquele cujo valor já faz parte da função, sendo opcional.

```
function cliente($codigo, $idade = 18){
        print "Código = $codigo, Idade = $idade";
}
```

```
cliente(1);  //Exibirá: Código = 1, Idade = 18
```

Trabalhando com Funções OB

- Artigo do Adriano Oliveira Gonçalves na Revista do PHP

9.8 - Trabalhando com Session no PHP

Sessões são usadas para guardar dados enquanto a janela do browser estiver aberta. São geralmente usadas para manter dados como nome do usuário, tipo do usuário (se é o administrador ou se é um visitante não cadastrado, por exemplo), entre outros dados importantes. Dica: NUNCA COLOQUE A SENHA NA SESSÃO. VOCÊ SÓ PRECISA DELA PARA AUTENTICAÇÃO!
Para iniciar uma sessão no PHP usamos a função:

```
session_start();
```

Se quisermos destruir a sessão (no caso de o usuário ter feito logoff), usamos a função:

```
session_destroy();
```

Agora precisamos apenas setar os valores que ficarão na sessão. No PHP os valores são armazenados em um vetor associativo chamado $_SESSION. As associações são feitas com pares chave e valor. Por exemplo, vamos setar o nome do usuário e a permissão dele, e colocar esses dados na seção. Vamos buscar esses dados a partir de um formulário fictício, via POST.

```
< ?php
    session_start(); //iniciamos a sessão
    $_SESSION['usuario'] = $_POST['usuario']; //colocamos na sessão o valor do
campo usuário vindo do formulário
    $_SESSION['permissao'] = $_POST['admin']; //da mesma forma setamos suas
permissões
    header("location:pagina_principal.php"); //Redirecionamos para a pagina
principal
? >
```

Com isso feito, podemos acessar as variáveis da sessão de qualquer lugar da nossa aplicação através do vetor $_SESSION.

Fonte: http://maozinhadaweb.blogspot.com/2007/05/tutorial-de-php-parte-3-sesses-e.html

Sessions em PHP

Session ou sessão é o método ou via pela qual podemos gravar informação de variáveis, funcionando assim como uma variável global. A grande vantagem das sessions é podermos navegar por várias páginas sem que esta informação se perca evitando assim passar o valor das variáveis indivídualmente de página em página.

Ao contrário dos cookies que ficam gravadas no computador do utilizador, as sessions são gravadas no servidor e automaticamente eliminadas quando o utilizador fecha o browser.

Exemplificando:
Vamos criar a página pagina1.php

```
<?php
//Iniciar sessão
session_start();
//Gravar variáveis em sessão
$_SESSION[‘id’]=1;
$_SESSION[‘nome’]=’Paulo’;
//Imprimir o valor das variáveis guardadas
print(“$id”);
print(“$nome”);
?>
```

Podemos verificar que são imprimidos os valores das variáveis atribuídas à session: 1 e Paulo, ou seja, esses valores ficam gravados numa session e podem ser utilizados em qualquer página onde essa sessão esteja aberta.

Modificar e remover sessions

Quando utilizamos sessions podemos ter a necessidade de modificar ou remover variáveis de sessão.
Para alterar o valor de uma variável basta fazer nova atribuição:

```
$_SESSION[‘nome’]=’Pedro’;
```

A variável $nome passaria a ter o valor de Pedro a partir desta linha.

Para remover todas as variáveis da session basta colocar o comando session_unset() ou apenas unset() para remover variáveis indivídualmente.
Para eliminar ou destruir completamente toda a sessão utilizamos o comando session_destroy().

Como referido anteriormente, a sessão acaba sempre que o utilizador fechar o browser, mas a duração das sessões também podem ser alteradas para um espaço de tempo pre-definido. Para isso deverá ser alterado o php.ini na linha session.cookie_lifetime = 0 colocando o número de segundos desejados no lugar do 0 (zero).

Gravar sessions num directório

As sessões também podem ser gravadas num directório no servidor. Para que esta operação seja realizada devera ser dada a instrução através da função session_save_path().
Nesta função deverá ser dado o caminho para a pasta que guardará as sessões:
session_save_path(“sessions”). Neste caso a pasta sessions que guardará as sessões fica localizada na raís.

Para que esta função funcione deverá ser colocada sempre antes do comando session_start().

Vamos experimentar:

```
 <?php
//Pasta onde vai gravar  sessão
```

```
session_save_path(“sessions”);
//Iniciar sessão
session_start();
//Gravar variáveis em sessão
$_SESSION[‘id’]=1;
$_SESSION[‘nome’]=’Paulo’;
//Imprimir o valor das variáveis guardadas
print(“$id”);
print(“$nome”);
?>
```

Pode confirmar que é gravado um ficheiro na pasta sessions com um nome do tipo “sess_t4k884rd8uunp8ojt83picc9j1”. Se abrir este ficheiro com o Notepad pode verificar que se encontram lá gravadas as variáveis dadas à sessão.

Convém salientar que, por razões óbvias, a pasta onde são gravadas as sessions deverá estar protegida.

Fonte: http://www.techcaffe.net/site/tutoriais.php?recID=4

Exemplo prático de Session em PHP

index.php

```
<?php
session_start();
?>
<h1 align="center">Usando SESSION em PHP</h1>
Podemos preservar valores de variáveis enquanto durar uma sessão do browser através do uso de SESSION.<br>
Para isso devemos startar a sessão em cada página que desejamos usar esta variável com<br><br>
sesssion_start();<br><br>
Lembrando que esta função deve vir antes de qualquer comando que mande algo para a tela, caso<br>
o session esteja configurado para usar cookie.<br>
Na primeira página deve ter um formulário com algum campo que devemos usar no session.<br>
Experimente gravar a URL de uma das páginas internas e acessar diretamente (http://localhost/session3)
<br>Primeiro feche todas as seções do browser e depois abra o browser com essa URL.<br>
<br>
Veja que SESSION é muito bom para preservar o valor de variáveis entre páginas de um site numa seção.<br>
```

Portanto seu uso é muito útil quando pretendemos autenticas os visitantes de todas as as páginas de um site.

Como também para outros usos em que pretendemos reaproveitar o valor de variáveis (algo como global).

Acompanhe este exemplo para ver detalhes.


```
<form method=post action=session2.php>
Login<input type=text size=8 name=login><br>
<input type=submit value=Enviar>
</form></center>
```

session2.php

```
<?php
session_start();
$_SESSION['login']=$_POST['login'];
if (isset($_SESSION['login'])){
echo "Entre. Session2. <a href=session3.php>Session3</a>";
} else {
echo "Acesso não autenticado!";
}
?>
```

session3.php

```
<?php
session_start();
if (isset($_SESSION['login'])){
  echo "Entre. Session3. <a href=session4.php>Session4</a><br>";
  echo "<a href=destruir.php>Drestruir Sessão</a>";
} else {
  echo "Acesso não autenticado!";
}
?>
```

session4.php

```
<?php
session_start();
if (isset($_SESSION['login'])){
echo "Entre. Session4. <a href=session5.php>Session5</a>";
} else {
echo "Acesso não autenticado!";
}
?>
```

session5.php

```
<?php
session_start();
if (isset($_SESSION['login'])){
echo "Entre. Session5. <a href=index.php>Index</a><br><br>";
echo "Informações: <br>ID da Sessão: <b>" . session_id() . "</b><br>Variável mantida pela SuperGlobal \$_SESSION: <b>" . $_SESSION['login'];
} else {
echo "Acesso não autenticado!";
}
?>
```

Dicas sobre Session:

```
print ini_get('max_execution_time');

ini_set("session.gc_maxlifetime","3600");
session_save_path("/my/own/path/without/url/sessions");
```

Bom tutorial em: http://www.phpro.org/tutorials/Introduction-To-PHP-Sessions.html

9.9 - Trabalhando com Cookies no PHP

Setando o Cookie

Para gravar um cookie no navegador do visitante usa-se a função setcookie:

setcookie*(nome, valor, data_expiração)*:

Exemplo:

```
//Calculate 60 dias no futuro
//segundos * minutos * horas * dias + tempo atual
$emDoisMeses = 60 * 60 * 24 * 60 + time();
setcookie('UltimaVisita', date("G:i - d/m/Y"), $emDoisMeses);
```

Recebendo o Cookie

```
if(isset($_COOKIE['UltimaVisita']))
   $visita = $_COOKIE['UltimaVisita'];
else
   print "You've got some stale cookies!";

echo "Sua última visita foi - ". $visita;
```

Fonte: http://www.tizag.com/phpT/phpcookies.php

9.10 - Tratamento de Erros no PHP

Os 25 erros de programação mais perigosos segundo a SANS
By coredump

Saiu no site da SANS a lista criada com o consenso entre varios profissionais e empresas do ramo de segurança e desenvolvimento descrevendo os 25 erros de programação mais perigosos para o desenvolvimento seguro. Eu vou traduzir os nomes e informação básicos mas o melhor é ler o artigo na íntegra, em inglês.

Os erros estão separados em três categorias: Interação insegura entre componentes, Gerenciamento arriscado de recursos, Defensas porosas.

Categoria: Interação insegura entre componentes

1. **Validação Imprópria de Entradas**: Entradas que recebem dados e os aceitam mesmo sem certificar que eles são do tipo/formato esperado.
2. **Codificação ou Escape Impróprios de Saída**: Saídas que não são codificadas ou escapadas corretamente são a maior fonte de ataques de injeção de código.
3. **Falha ao Preservar a Estrutura da Busca SQL (conhecido como Injeção de SQL)**: Se os atacantes podem influenciar as procuras SQL do seu programa, então eles podem controlar o seu banco de dados.
4. **Falha ao Preservar a Estrutura do Código da Página (conhecido como "Cross-site Scripting")**: Assim como o anterior, se os atacantes podem injetar código ou scripts em sua página, eles podem controlar a página.
5. **Falha ao Preservar a Estrutura de Comandos do Sistema Operacional**: Se você permitir que entradas ilegais sejam passadas para aplicativos do sistema operacional, o atacante pode controlar o servidor.
6. **Transmissão de Dados Sensíveis em Texto Puro**: Senhas, dados de cartão e qualquer informação considerada sensível deve ser criptografada.
7. **Falsificação de Requisição Entre Sites**: Um atacante pode criar uma requisição que é enviada a outro site forjando a origem e fazendo o mesmo partir de um usuário inocente, aproveitando credenciais de autenticação e acessos.
8. **Condição de Corrida**: Atacantes vão sempre procurar por condições de corrida no software para conferir se alguma informação importante não é obtida no processo.
9. **Vazamento de Informações em Mensagens de Erro**: Atacantes vão procurar por mensagens de erro que descrevam mais que o necessário, como nomes de campos SQL, objetos e bibliotecas sendo utilizadas.

Categoria: Gerenciamento arriscado de recursos:

1. **Falha ao Limitar Operações aos Limites de um Buffer de Memória**: O conhecido buffer overflow.
2. **Controle Externo de Dados Sensíveis**: Informações críticas que são mantidas fora de um banco de dados por questões de performance não deviam ser facilmente acessíveis por atacantes.
3. **Controle Externo de de Caminho ou Nome de Arquivo**: Quando você usa dados externos para montar um nome de arquivo ou caminho de gravação, você está se arriscando a ser atacado.
4. **Caminho de Procura Inseguro**: Se o caminho de procura de recursos estiver em algum lugar sob controle de um atacante, bibliotecas ou código pode ser inserido a revelia.

5. **Falha ao Controlar a Geração de Código**: Caso o atacante consiga influenciar a geração de código dinâmico (se geração de código dinâmico for utilizada no programa) ele poderá controlar todo seu código.
6. **Download de Código sem Verificação de Integridade**: Se você executa código obtido por download, você confia na fonte. Atacantes podem aproveitar esta confiança.
7. **Desligamento ou Liberação Impróprias de Recursos**: Arquivos, conexões e classes precisam ser corretamente encerradas.
8. **Inicialização Imprópria**: Dados, bibliotecas e sistemas inicializados incorretamente podem abrir margens para problemas.
9. **Cálculos Incorretos**: Quando o atacante tem algum controle sobre as entradas usadas em operações matemáticas, isso pode gerar vulnerabilidades.

Categoria: Defensas porosas:

1. **Controle de Acesso Impróprio**: Se você não garante que seus usuários estão fazendo apenas o que deviam, os atacantes irão se aproveitar de sua autenticação.
2. **Uso de um Algoritmo Criptográfico Quebrado ou Vulnerável**: Utilização de algoritmos fracos ou comprometidos levam a falhas de criptografia e vulnerabilidades.
3. **Senha no Código**: deixar um usuário e uma senha no próprio código traz inúmeros problemas.
4. **Permissão de Acesso Insegura para Recurso Crítico**: Configurações, arquivos de dados e bancos de dados devem ter suas permissões de acesso protegidas.
5. **Uso de Valores Insuficientemente Aleatórios**: Se você usa tipos de segurança que dependem de aleatoriedade, usar um gerador aleatório insuficiente só vai causar problemas.
6. **Execução com Privilégios Desnecessários**: Se seu programa precisa de privilégios elevados para executar suas funções, ele deve abrir mão destes direitos assim que ele termina de executar as ações que precisavam dos privilégios.
7. **Aplicação de Segurança do Lado do Servidor pelo Cliente**: Atacantes podem usar engenharia reversa em um cliente de software e escrever seus próprios clientes removendo testes e aplicações de segurança.

Algumas coisas foram realmente chatas de traduzir, sinta-se livre para sugerir correções.

intel

PS: Lembre-se sempre “os 25 mais” não quer dizer “os 25 únicos”. *Grain of Salt* faz bem.

Fonte: http://core.eti.br/2009/01/22/os-25-erros-de-programacao-mais-perigosos-segundo-a-sans/

Exemplo simples de uso de Exception

```
<?php
/*
http://www.sourcecode4you.com/article/articleview/956dd1bb-a5a2-42ad-ae31-ad836eec24f3/try-catch-block-in-php.aspx

Try Catch Block In Php

Try catch block handle runtime exception which occur in function.
*/

function dividebyZero()
{
    try
    {
        $k= 10/0;
    }
    catch(Exception $ex)
    {
        echo $ex;
    }
}

dividebyZero();

function throwException()
{
    try
    {
        throw new Exception("My Custom Exception");
    }
    catch(Exception $ex)
    {
        echo $ex;
    }
}
throwException();
?>
```

Introdução a manipulação de erros em PHP

Autor: Lorran Luiz <lorranluiz at click21.com.br>
Data: 22/01/2009

Introdução

Na história humana muitos conquistaram grandes coisas. Tiveram problemas é claro, mas em suas decisões não erraram ou erraram pouco (pois é, com os erros que aprendemos). Os erros enfim tiveram um papel relativamente importante para o sucesso dessas pessoas. Foi com eles (os erros) que as soluções apareceram, pois não existe solução se não há um erro para ser consertado.

É claro que erros nos decepcionam, mas eles só refletem a realidade - de que ainda não estamos completamente qualificados a fazer tal ato com êxito completo. Os erros mostram que não somos perfeitos.

A fato de não sermos perfeitos esbarra no que fazemos. Enfim, os programadores erram, os usuários podem errar ou até mesmo alguma parte do programa que está a ser executado pode falhar de alguma maneira.

São nesses erros que envolvem o aplicativo e/ou os atores envolvidos no sistema que grandes desastres começam a se formar. Verdadeiras bolas de neve vão crescendo a medida que um programador digita seu código sem preocupar-se com a menor possibilidade de algo dar errado.

Depois que um sistema está completamente construído, sem base alguma para o tratamento correto de erros, o "problema quase que irresolvível" estará montado! A cada passo na vida desse sistema a possibilidade da ocorrência de erros "não identificáveis" aumentará, e a cada erro, mais e mais do seu tempo se consumirá para se chegar a resolução do mesmo. O sistema enfim se torna insustentável e todas aquelas horas de trabalho foram desperdiçadas!

Situações como estas poderiam ser evitadas se medidas simples como a construção de um código "supervisionado" com cláusulas e blocos que garantem a manipulação correta e segura de erros fossem implementadas logo nas primeiras linhas.

Situações de possíveis erros

Veremos a partir de então o básico quando o assunto é manipulação de erros. Com tal conhecimento o programador se capacitará a avançar no estudo de tratamento de erros e posteriormente aprender técnicas para tal.

Algumas situações

Suspeite de qualquer situação em que seu código fará algo decisivo para o funcionamento do seu sistema. Coisas como conectar a um banco de dados, abrir algum arquivo que contenha parâmetros importantes, carregar uma classe (ou uma biblioteca) essencial na aplicação ou armazenar alguma informação numa tabela de banco de dados, entre muitas outras situações requerem uma execução sem falhas e por isso deve haver uma preocupação maior com os erros que acontecerem.

Podemos dizer que em certas partes de seu código necessitaremos de "vigilantes" atentos a possíveis situações anormais. Veremos então quem são os vigilantes da programação no PHP.

Os "vigilantes"

Podemos rodear determinadas áreas de seu script com "vigilantes", que esperam que algo dê errado para passar o controle da situação aos seus "superiores".

Usamos a palavra-chave *try* para representar esses vigias virtuais no seu código fonte.

O try é seguido de um bloco de códigos no qual ele estará responsável por capturar eventuais erros. Os erros são denominados exceções, ou seja, situações que saem do esperado.

Sintaxe:

```
try {
//Código suspeito
}
```

Exceções

Em orientação a objeto, no PHP, exceções representam um objeto. Esse objeto necessita de um "molde" para ser construído, esse molde é denominado *Classe* e a classe usada para "modelar" uma exceção é a classe *Exception*.

Abaixo vemos a classe Exception:

```
<?php
class Exception {

  protected $message = 'Unknown exception'; // Mensagem da exceção
  protected $code = 0;                      // Código da exceção definido pelo usuário
  protected $file;                          // Arquivo gerador da exceção
  protected $line;                          // Linha geradora da exceção

  function __construct(string $message=NULL, int code=0);

  final function getMessage();              // Mensagem da exceção
  final function getCode();                 // Código da exceção
  final function getFile();                 // Arquivo gerador
  final function getTrace();                // um array com o backtrace()
  final function getTraceAsString();        // String formatada do trace

  /* Sobrecarregável */
  function _toString();                     // String formatada para ser mostrada

}
```

?>

A classe Exception não precisa ser inserida em seu código, ela é uma classe nativa do PHP introduzida no Zend 2.

Os "superiores"

Os "superiores" são aqueles que tem competência para fazer algo, devem resolver um problema ou passar essa responsabilidade para quem resolva. No PHP usamos a palavra-chave *catch* para representar esse superior.

O "catch" é responsável pela identificação do erro, ou seja, ele "sabe" qual o tipo de exceção ocorreu, de acordo com a classe, derivada de Exception, que "modelou" o objeto de exceção passado para ele.

O "catch" necessita de um parâmetro, o nome da classe Exception ou derivada seguida da variável que conterá o objeto da exceção e um bloco de código.

Sintaxe:

```
catch (Exception $variavel) {
//Código que manipulará tal exceção
}
```

A palavra-chave catch deve ser posicionada logo após o bloco de código rodeado pela cláusula try. Ficando então assim:

Sintaxe:

```
try { //Código suspeito } catch (Exception $variavel) { //Código que manipulará tal exceção }
```

Preparando seu código para as exceções

Já vimos como cercar uma parte do código para manipular os erros que nela ocorrerem. Veremos a partir de então como "delatar" os erros e alertar o sistema quanto a eles.

Construindo uma exceção

O método construtor da classe Exception requer 2 parâmetros, sendo 1 opcional:

```
public Exception::__construct ([ string $message=NULL [, int $code ]] )
```

O primeiro parâmetro é a mensagem do erro e o segundo o código do erro (opcional).

Disparando o alarme

Em PHP usamos a palavra-chave *throw* para alertar o sistema da ocorrência de um erro, uma

exceção. Throw é seguido de um código que solicita a construção de um novo objeto de exceção.

Para instanciar uma classe no PHP usamos a palavra-chave *new* seguida do nome da classe. Então a sintaxe ficará assim:

throw new Exception("Mensagem", 1);

Como usar "throw"

Podemos usar a condicional *if* para decidirmos quando disparar uma exceção.

Veja o exemplo abaixo:

if (!@mysql_connect ("localhost", "usuário", "senha")) throw new Exception("Não foi possível conectar ao banco de dados");

Obs.: Podemos usar o operador de supressão (@) de erros para evitar que sejam exibidas as mensagens de erro padrão do PHP.

Estendendo e especificando exceções

Podemos tornar nossas exceções mais específicas e por consequência saber com mais precisão que tipo de erro ocorreu, manipulando-o assim da melhor maneira possível.

Usamos a palavra-chave *extends* para estender uma classe, ou seja, criar uma classe derivada (ou classe filha). Quando uma classe estende outra, ela herda da última atributos e métodos dependendo do nível de acesso de cada elemento que constitui o corpo da classe. A classe filha pode sobrecarregar métodos (redefiní-los).

Exception é uma classe que pode ser herdada, sendo assim podemos definir classes que derivem e aumentem sua funcionalidade, passando de uma simples exceção genérica para exceções mais precisas.

Exceções derivadas

Definamos algumas classes derivadas de Exception:

```
class DBException extends Exception {
   protected $variavel; //Uma variavel qualquer

   public function __construct($exceptionMessage, $errorCode = 0) {
      parent::__construct($exceptionMessage, $errorCode);
   }

   //Aqui vemos alguns métodos característicos da classe derivada de Exception
   public function getVariavel() {
      return $this->variavel_qualquer;
   }
```

```
    public function setVariavel($valor = NULL) {
        $this->variavel = $valor;
    }
}

class FopenException extends Exception {
    protected $arquivo; //Variável contendo o nome de um arquivo

    public function __construct($nomeDoArquivo, $exceptionMessage, $errorCode = 0) {
        parent::__construct($exceptionMessage, $errorCode);
        $this->arquivo = $nomeDoArquivo;
    }

    //Métodos característicos dessa classe
    public function getNomeDoArquivo() {
        return $this->arquivo;
    }

    public function exibirNomeDoArquivo() {
        echo "Nome do arquivo: {$this->arquivo}";
    }

}
```

Um código básico com tratamento de erros

Veremos e entenderemos como funcionaria então um código básico completo que trataria de uma maneira certa os possíveis erros que ocorressem durante sua execução.

O código

O código que se seguirá é meramente explicativo e por isso talvez não possua utilidade numa situação real.

- Criaremos um código onde será criado um manipulador de arquivos;
- O conteúdo do arquivo será lido;
- Efetuaremos uma conexão com o servidor de banco de dados;
- Esse conteúdo será inserido numa célula de uma tabela de um banco de dados;
- E se tudo ocorrer bem, será exibida a mensagem: Operação efetuada com sucesso;
- Caso ocorra um erro será exibida uma mensagem de erro que será reportada ao programador.

Veja abaixo:

```
//A CLASSE:

class ArmazenadorDeArquivoEmDB {
```

```
    protected $db_obj;
    protected $db_usuario;
    protected $db_senha;
    protected $db_dsn;
    protected $nomeDoArquivo;
    protected $manipuladorDoArquivo;
    protected $conteudoDoArquivo;
    protected $db_tabela;
    protected $db_dbName;

    public function __construct($usuario, $senha, $tabela, $db, $servidor) { //Construir objeto
        $this->db_usuario = $usuario;
        $this->db_senha = $senha;
        $this->db_tabela = $tabela;
        $this->db_dbName = $db;
        $this->db_dsn = $servidor;
    }

    public function conectar() {
        if(!($this->db_obj = @mysql_connect($this->db_dsn, $this->db_usuario, $this->db_senha))) {
            throw new DBException("Não foi possível conectar ao banco de dados", 281); //Lembrando
que o código de erro não é um parâmetro obrigatório
        }
    }

    public function armazenarArquivoNoDB($nomeDoArquivo) { //O parâmetro requer endereço
completo do arquivo
        $this->nomeDoArquivo = $nomeDoArquivo;
        try {
            if (empty($this->db_obj)) { //Se não existir uma conexão aberta com o DB
                $this->conectar();
            }
        } catch  ( DBException $e ) { //Caso haja algum erro durante a tentativa de conexão...
            throw $e;                    // ...o "alarme" irá disparar
        }

        //Iremos abrir o arquivo:
        if (!file_exists($nomeDoArquivo)) throw new FopenException($nomeDoArquivo, "O arquivo
especificado não existe.", 282);
        if (!($this->manipuladorDoArquivo = @fopen($nomeDoArquivo, "r"))) throw new
FopenException($nomeDoArquivo, "Não foi possível abrir um manipulador para este arquivo.",
283);
        if (!($this->conteudoDoArquivo = @fread($this->manipuladorDoArquivo,
intval(sprintf("%u", filesize($nomeDoArquivo)))))) throw new FopenException($nomeDoArquivo,
"Não foi possível abrir o conteúdo deste arquivo.", 284); //O 2º parâmetro de fread() possibilita a
leitura de arquivos maiores que 2GB

        //Armazenaremos agora o conteúdo do arquivo no DB
```

```
        $sql = 'INSERT INTO `' . $this->db_dbName . '`.`' . $this->db_tabela . '` (`nome`, `conteudo`) VALUES (\" . $this->nomeDoArquivo . '\', \". $this->conteudoDoArquivo . '\');';
        if(!($insert = @mysql_query($sql))) throw new DBException("Não foi possível inserir o arquivo no banco de dados.", 285);
        else echo '<html><head></head><body onload="alert(\'O arquivo foi armazenado com sucesso no banco de dados\')"></body></html>';
    }
}

//O OBJETO EM AÇÃO:

$armazenadorDeArquivoEmDB = new ArmazenadorDeArquivoEmDB("usuario", "senha", "tabela", "banco_de_dados", "sevidor"); //Criamos um objeto que armazena um arquivo num banco de dados

try { //Tentiva de conexão
    $armazenadorDeArquivoEmDB->armazenarArquivoNoDB("Ergue-te Marcos.txt"); //Aqui entra o nome do arquivo a ser armazenado no DB
} catch ( DBException $e ) {
    //Houve um erro relativo ao banco de dados
    echo nl2br("<b>{$e->getMessage()}</b>\n<br />Detalhes:\n{$e->__toString()}"); //Exibir string contendo informações sobre a exceção
    $gossipy->reportar($e); //A nível exemplificativo, temos este objeto fictício, imaginário, que envia por email informações sobre o erro ao programador
} catch ( FopenException $e ) {
    //Houve um erro referente ao arquivo, e podemos dar um tratamento específico a este tipo de exceção!
    echo nl2br("<b>{$e->getMessage()}</b>\n<br />Detalhes:\n{$e->__toString()}"); //Exibir string contendo informações sobre a exceção
    $gossipy->reportar($e); //A nível exemplificativo, temos este objeto fictício, imaginário, que envia por email informações sobre o erro ao programador
}
```

Poderíamos ter explorado mais desta fantástica linguagem que é o PHP e ter feito um tratamento de erros mais proveitoso no código acima, mas acho que para fins meramente explicativos e didáticos não é necessário tanta complexidade.

Vocês podem ver que poderíamos ter criado no código acima subclasses de Exception bem mais específicas, a idéia é a mesma de quando criamos as classes "FopenException" e "DBException". Classes mais específicas para a manipulação de exceções relativas ao banco de dados poderiam ser definidas apenas estendendo a classe "DBException", veja:

```
SQLException extends DBException { }
```

E cada vez podemos ir tornando as exceções mais específicas (e a manipulação mais precisa).

Resumindo

Faremos então uma compilação de tudo o que aprendemos:

- Neste artigo revisamos conceitos básicos do poderoso recurso que é a orientação a objetos do PHP 5;
- Vimos algumas situações de risco que podem surgir durante a execução de seu código;
- Vimos também que podemos aprender com as situações frustradoras se tratarmos melhor as informações relativas a aquele problema;
- Vimos que o PHP 5 conta com excelentes ferramentas para tratarmos da maneira certa as exceções que ocorrerem;
- Estudamos as principais palavras-chave quando o assunto é tratamento de erros em PHP 5: try, catch e throw;
- Estudamos superficialmente a classe Exception;
- Aprendemos a disparar o "sinal de alerta" aproveitando a condicional "if" e usando-a em conjunto com throw.

Aprendemos o funcionamento básico do "sistema" de manipulação dos erros no PHP 5:

Quando um erro (exceção) ocorre, este é "percebido" (com por exemplo a condicional "if"), instancia-se então a classe Exception (com "throw new Exception('msg', 01)") ou uma derivada, o controle é passado para o bloco "catch" correspondente, que por vez pode usar as informações que recebeu a respeito do erro na sua manipulação, como melhor convir.

Depois que todo o código do bloco catch que manipulou o erro é executado, o controle do código volta ao escopo mais geral (exceto se uma função exit() ou similar for executada dentro do bloco catch, o que terminaria a execução do código neste instante).

Encerrando

Espero sinceramente ter sido claro, e mesmo sem aprofundar muito neste estudo, ter passado o máximo de conhecimento a respeito dos fundamentos do tratamento de erro com exceções em PHP.

Continuemos a estudar e talvez um dia melhorar nosso país com o conhecimento!

Um abraço para todos da comunidade VOL!

L. Luiz

---

http://www.vivaolinux.com.br/artigo/Introducao-a-manipulacao-de-erros-em-PHP

9.11 - Trabalhando com Validação de Dados no PHP

Ótimo artigo em 5 partes na Revista do PHP de autoria do Er Abbott

- Validação de Formulários - 5º e última parte: Validando no servidor

- Validação de Formulários - Parte 04 (A fonteira cliente/servidor)

- Validação de Formulários - Parte 3: O baile de máscaras

- Validação de Formulários - Parte 2: Os Campos Especiais

- Validação de formulários - Parte 1 (O Planejamento)

Turbinando a Validação de Formulários

Validação de e-mails

Check valid e-mail

```
function esEmailValido($email)
{
    if (ereg("^[_a-zA-Z0-9-]+(\.[_a-zA-Z0-9-]+)*@([_a-zA-Z0-9-]+\.)*[a-zA-Z0-9-]
{2,200}\.[a-zA-Z]{2,6}$", $email ) )
        {
       return true;
    }
        else
        {
       return false;
    }
}
```

http://snippets.dzone.com/posts/show/4346

Tipos de Variáveis

```
<?php

function ss_array_as_string (&$array, $column = 0) {
    $str = "Array(<BR>\n";
    while(list($var, $val) = each($array)){
        for ($i = 0; $i < $column+1; $i++){
            $str .= "    ";
        }
        $str .= $var.' ==> ';
```

```
        $str .= ss_as_string($val, $column+1)."<BR>\n";
    }
    for ($i = 0; $i < $column; $i++){
        $str .= "    ";
    }
    return $str.')';
}

function ss_object_as_string (&$object, $column = 0) {
    if (empty($object->classname)) {
        return "$object";
    }
    else {
        $str = $object->classname."(<BR>\n";
        while (list(,$var) = each($object->persistent_slots)) {
            for ($i = 0; $i < $column; $i++){
                $str .= "    ";
            }
            global $$var;
            $str .= $var.' ==> ';
            $str .= ss_as_string($$var, column+1)."<BR>\n";
        }
        for ($i = 0; $i < $column; $i++){
            $str .= "    ";
        }
        return $str.')';
    }
}

function ss_as_string (&$thing, $column = 0) {
    if (is_object($thing)) {
        return ss_object_as_string($thing, $column);
    }
    elseif (is_array($thing)) {
        return ss_array_as_string($thing, $column);
    }
    elseif (is_double($thing)) {
        return "Double(".$thing.")";
    }
    elseif (is_long($thing)) {
        return "Long(".$thing.")";
    }
    elseif (is_string($thing)) {
        return "String(".$thing.")";
    }
    else {
        return "Unknown(".$thing.")";
    }
}

$my_variable=3;
//echo ss_as_string($my_variable);
echo ss_as_string($GLOBALS);
?>
```

Validação Usando Filtros do PHP

`Suportados pelo PHP com versão 5.2 ou superior.`

Constantes pré-definidas

As contantes abaixo são definidas por esta extensão e somente estarão disponíveis quando a extensão foi compilada com o PHP ou carregada dinamicamente durante a execução.

INPUT_POST (integer)
Variáveis POST.
INPUT_GET (integer)
Variáveis GET.
INPUT_COOKIE (integer)
Variáveis COOKIE.
INPUT_ENV (integer)
Variáveis ENV.
INPUT_SERVER (integer)
Variáveis SERVER.
INPUT_SESSION (integer)
Variáveis SESSION. (ainda não implementada)
INPUT_REQUEST (integer)
Variáveis REQUEST. (ainda não implementada)

FILTER_FLAG_NONE (integer)
Sem flags.
FILTER_REQUIRE_SCALAR (integer)
Flag usada para requerir escalar como entrada
FILTER_REQUIRE_ARRAY (integer)
Requer um array como entrada.
FILTER_FORCE_ARRAY (integer)
Sempre retorna um array.
FILTER_NULL_ON_FAILURE (integer)
Usa NULL ao invés de FALSE em falha.
FILTER_VALIDATE_INT (integer)
ID do filtro "int".
FILTER_VALIDATE_BOOLEAN (integer)
ID do filtro "boolean".
FILTER_VALIDATE_FLOAT (integer)
ID do filtro "float".
FILTER_VALIDATE_REGEXP (integer)
ID do filtro "validate_regexp".
FILTER_VALIDATE_URL (integer)
ID do filtro "validate_url".
FILTER_VALIDATE_EMAIL (integer)
ID do filtro "validate_email".
FILTER_VALIDATE_IP (integer)
ID do filtro "validate_ip".

FILTER_DEFAULT (integer)
ID do filtro padrão ("string").
FILTER_UNSAFE_RAW (integer)
ID do filtro "unsafe_raw".
FILTER_SANITIZE_STRING (integer)
ID do filtro "string".
FILTER_SANITIZE_STRIPPED (integer)
ID do filtro "stripped".
FILTER_SANITIZE_ENCODED (integer)
ID do filtro "encoded".
FILTER_SANITIZE_SPECIAL_CHARS (integer)
ID do filtro "special_chars".
FILTER_SANITIZE_EMAIL (integer)
ID do filtro "email".
FILTER_SANITIZE_URL (integer)
ID do filtro "url".
FILTER_SANITIZE_NUMBER_INT (integer)
ID do filtro "number_int".
FILTER_SANITIZE_NUMBER_FLOAT (integer)
ID of "number_float" filter.
FILTER_SANITIZE_MAGIC_QUOTES (integer)
ID do filtro "magic_quotes".
FILTER_CALLBACK (integer)
ID do filtro "callback".
FILTER_FLAG_ALLOW_OCTAL (integer)
Permite notação octal (*0[0-7]+*) no filtro "int".
FILTER_FLAG_ALLOW_HEX (integer)
Permite notação hexadecimal (*0x[0-9a-fA-F]+*) no filtro "int".
FILTER_FLAG_STRIP_LOW (integer)
Remove caracteres com valor ASCII menor que 32.
FILTER_FLAG_STRIP_HIGH (integer)
Remove caracteres com valor ASCII maior que 127.
FILTER_FLAG_ENCODE_LOW (integer)
Codifica caracteres com valor ASCII menor que 32.
FILTER_FLAG_ENCODE_HIGH (integer)
Codifica caracteres com valor ASCII maior que 127.
FILTER_FLAG_ENCODE_AMP (integer)
Codifica *&*.
FILTER_FLAG_NO_ENCODE_QUOTES (integer)
Não codifica *'* e *"*.
FILTER_FLAG_EMPTY_STRING_NULL (integer)
(Nenhum uso no momento.)
FILTER_FLAG_ALLOW_FRACTION (integer)
Permite parte fracional no filtro "number_float".
FILTER_FLAG_ALLOW_THOUSAND (integer)
Permite separador de milhar (*,*) no filtro "number_float".
FILTER_FLAG_ALLOW_SCIENTIFIC (integer)
Permite notação científica (*e*, *E*) no filtro "number_float".

FILTER_FLAG_SCHEME_REQUIRED (integer)
Requer scheme no filtro "validate_url".
FILTER_FLAG_HOST_REQUIRED (integer)
Requer host no filtro "validate_url".
FILTER_FLAG_PATH_REQUIRED (integer)
Requer path no filtro "validate_url".
FILTER_FLAG_QUERY_REQUIRED (integer)
Requer query no filtro "validate_url".
FILTER_FLAG_IPV4 (integer)
Permite somente endereço IPv4 no filtro "validate_ip".
FILTER_FLAG_IPV6 (integer)
Permite somente endereço IPv6 no filtro "validate_ip".
FILTER_FLAG_NO_RES_RANGE (integer)
Não permite endereços reservados no filtro "validate_ip".
FILTER_FLAG_NO_PRIV_RANGE (integer)
Não permite endereços privados no filtro "validate_ip".

Funções de Filtragem

filter_has_var — Verifica se a variável é de um especificado tipo existente

- filter_id — Retorna o ID de um dado nome de filtro
- filter_input_array — Obtem variáveis externas e opcionalmente as filtra
- filter_input — Obtem a específica variável externa pelo nome e opcionalmente a filtra
- filter_list — Retorna a lista de todos filtros suportados
- filter_var_array — Obtêm múltiplas variáveis e opcionalmente as filtra
- filter_var — Filtra a variável com um especificado filtro

Detectando o tipo de variável:

```
$_GET['test'] = 1;
echo filter_has_var(INPUT_GET, 'test') ? 'Sim' : 'Não';

Retornando uma lista com os tipos de filtros suportados
```

```
<?php
print_r(filter_list());
?>
```

Filter_input:

```
$search_html = filter_input(INPUT_GET, 'search', FILTER_SANITIZE_SPECIAL_CHARS);
$search_url = filter_input(INPUT_GET, 'search', FILTER_SANITIZE_ENCODED);
echo "You have searched for $search_html.\n";
echo "<a href='?search=$search_url'>Search again.</a>";
```

filter_var

var_dump(filter_var('bob@example.com', FILTER_VALIDATE_EMAIL));
var_dump(filter_var('example.com', FILTER_VALIDATE_URL, FILTER_FLAG_SCHEME_REQUIRED));

```
<?php
error_reporting(E_ALL | E_STRICT);
$data = array(
    'product_id'    => 'libgd<script>',
    'component'     => '10',
    'versions'      => '2.0.33',
    'testscalar'    => array('2', '23', '10', '12'),
    'testarray'     => '2',
);

$args = array(
    'product_id'   => FILTER_SANITIZE_ENCODED,
    'component'    => array('filter'    => FILTER_VALIDATE_INT,
                            'flags'     => FILTER_FORCE_ARRAY,
                            'options'   => array('min_range' => 1,
'max_range' => 10)
                           ),
    'versions'     => FILTER_SANITIZE_ENCODED,
    'doesnotexist' => FILTER_VALIDATE_INT,
    'testscalar'   => array(
                            'filter' => FILTER_VALIDATE_INT,
                            'flags'  => FILTER_REQUIRE_SCALAR,
                           ),
    'testarray'    => array(
                            'filter' => FILTER_VALIDATE_INT,
                            'flags'  => FILTER_FORCE_ARRAY,
                           )

);

$myinputs = filter_var_array($data, $args);

var_dump($myinputs);
echo "\n";
?>
```

```
Just a little filter to validate IP v4 & v6
This little script display result in function of the query
<?php
$ipv6="2a01:e35:aaa4:6860:a5e7:5ba9:965e:cc93";
$ipv4="82.237.3.3";
$fake = "3342423423";
$ipv4priv = "255.255.255.255";
```

```
$ipv6priv = "::1";
echo "<pre>";
echo $ipv4;
echo "<br />";
var_dump(filter_var($ipv4,FILTER_VALIDATE_IP));
echo "<br />";
echo $ipv6;
echo "<br />";
var_dump(filter_var($ipv6,FILTER_VALIDATE_IP));
echo "<br />";
echo $fake;
echo "<br />";
var_dump(filter_var($fake,FILTER_VALIDATE_IP));
echo "<br />";
echo $ipv4priv;
echo "<br/>";
echo "FILTER_VALIDATE_IP, FILTER_FLAG_NO_RES_RANGE";
echo "<br />";
var_dump(filter_var($ipv4priv,FILTER_VALIDATE_IP,
FILTER_FLAG_NO_RES_RANGE));
echo "<br />";
echo $ipv4priv;
echo "<br/>";
echo "FILTER_FLAG_NO_PRIV_RANGE";
echo "<br />";
var_dump(filter_var($ipv4priv,FILTER_FLAG_NO_PRIV_RANGE));
echo "<br />";
echo $ipv6priv;
echo "<br/>";
echo "FILTER_FLAG_NO_PRIV_RANGE";
echo "<br />";
var_dump(filter_var($ipv6priv,FILTER_FLAG_NO_PRIV_RANGE));
echo "<br />";
echo $ipv6priv;
echo "<br />";
var_dump(filter_var($ipv6priv,FILTER_VALIDATE_IP));

echo "</pre>";
```

9.12 – Trabalhando com XML em PHP

XML no Manual do PHP - http://www.php.net/manual/pt_BR/book.xml.php
Introdução ao XML - http://www.criarweb.com/manuais/24

Lendo extrutura completa de arquivo XML:

data.xml

```
<?xml version='1.0'?>
<!DOCTYPE chapter SYSTEM "/just/a/test.dtd" [
<!ENTITY plainEntity "FOO entity">
<!ENTITY systemEntity SYSTEM "xmltest2.xml">
]>
<chapter>
 <TITLE>Titulo</TITLE>
 <para>
  <informaltable>
   <tgroup cols="3">
    <tbody>
     <row><entry>a1</entry><entry morerows="1">b1</entry><entry>c1</entry></row>
     <row><entry>a2</entry><entry>c2</entry></row>
     <row><entry>a3</entry><entry>b3</entry><entry>c3</entry></row>
    </tbody>
   </tgroup>
  </informaltable>
 </para>
 &systemEntity;
 <section id="about">
  <title>Sobre este documento</title>
  <para>
   <!-- this is a comment -->
   <?php echo 'Hi!  This is PHP version '.phpversion(); ?>
  </para>
 </section>
</chapter>
```

ler_xml.php

```
<?php
//Exemplo de Entity externa

$file = "data.xml";

function trustedFile($file) {
   // only trust local files owned by ourselves
   if (!eregi("^([a-z]+)://", $file)
       && fileowner($file) == getmyuid()) {
           return true;
   }
```

```
  return false;
}

function startElement($parser, $name, $attribs) {
  echo "&lt;<font color=\"#0000cc\">$name</font>";
  if (sizeof($attribs)) {
     while (list($k, $v) = each($attribs)) {
        echo " <font color=\"#009900\">$k</font>=\"<font
              color=\"#990000\">$v</font>\"";
     }
  }
  echo "&gt;";
}

function endElement($parser, $name) {
  echo "&lt;/<font color=\"#0000cc\">$name</font>&gt;";
}

function characterData($parser, $data) {
  echo "<b>$data</b>";
}

function PIHandler($parser, $target, $data) {
  switch (strtolower($target)) {
     case "php":
        global $parser_file;
        // If the parsed document is "trusted", we say it is safe
        // to execute PHP code inside it.  If not, display the code
        // instead.
        if (trustedFile($parser_file[$parser])) {
           eval($data);
        } else {
           printf("Untrusted PHP code: <i>%s</i>",
                 htmlspecialchars($data));
        }
        break;
  }
}

function defaultHandler($parser, $data) {
  if (substr($data, 0, 1) == "&" && substr($data, -1, 1) == ";") {
     printf('<font color="#aa00aa">%s</font>',
           htmlspecialchars($data));
  } else {
     printf('<font size="-1">%s</font>',
           htmlspecialchars($data));
  }
}
```

```
function externalEntityRefHandler($parser, $openEntityNames, $base, $systemId,
                          $publicId) {
  if ($systemId) {
     if (!list($parser, $fp) = new_xml_parser($systemId)) {
        printf("Could not open entity %s at %s\n", $openEntityNames,
              $systemId);
        return false;
     }
     while ($data = fread($fp, 4096)) {
        if (!xml_parse($parser, $data, feof($fp))) {
           printf("XML error: %s at line %d while parsing entity %s\n",
                 xml_error_string(xml_get_error_code($parser)),
                 xml_get_current_line_number($parser), $openEntityNames);
           xml_parser_free($parser);
           return false;
        }
     }
     xml_parser_free($parser);
     return true;
  }
  return false;
}

function new_xml_parser($file) {
  global $parser_file;

  $xml_parser = xml_parser_create();
  xml_parser_set_option($xml_parser, XML_OPTION_CASE_FOLDING, 1);
  xml_set_element_handler($xml_parser, "startElement", "endElement");
  xml_set_character_data_handler($xml_parser, "characterData");
  xml_set_processing_instruction_handler($xml_parser, "PIHandler");
  xml_set_default_handler($xml_parser, "defaultHandler");
  xml_set_external_entity_ref_handler($xml_parser, "externalEntityRefHandler");

  if (!($fp = @fopen($file, "r"))) {
     return false;
  }
  if (!is_array($parser_file)) {
     settype($parser_file, "array");
  }
  $parser_file[$xml_parser] = $file;
  return array($xml_parser, $fp);
}

if (!(list($xml_parser, $fp) = new_xml_parser($file))) {
  die("could not open XML input");
}
```

```
echo "<pre>";
while ($data = fread($fp, 4096)) {
   if (!xml_parse($xml_parser, $data, feof($fp))) {
      die(sprintf("XML error: %s at line %d\n",
                xml_error_string(xml_get_error_code($xml_parser)),
                xml_get_current_line_number($xml_parser)));
   }
}
echo "</pre>";
echo "parse complete\n";
xml_parser_free($xml_parser);
?>
```

Ler XML usando simplexml:
xml_lido.php

```
<?php
$xmlstr = <<<XML
<?xml version='1.0' encoding='ISO-8859-1' ?>
<filmes>
 <filme>
  <titulo>PHP: Iniciando o Parser</titulo>
  <personagens>
   <personagem>
   <nome>João de Brito</nome>
   <actor>Brito</actor>
   </personagem>
   <personagem>
   <nome>Manoel Cunha</nome>
   <actor>Manoel</actor>
   </personagem>
  </personagens>
  <comentario>
   O XML é uma linguagem. Ela é como uma linguagem de programação. Ou uma
   linguagem de script? Tudo será revelado após ler bem toda a
   documentação.
  </comentario>
  <votos type="thumbs">7</votos>
  <votos type="stars">5</votos>
 </filme>
</filmes>
XML;
?>
```

xml_ler.php

```
<?php
include 'xml_lido.php';
$xml = simplexml_load_string($xmlstr);
```

```
echo $xml->filme[0]->comentario; // "So this language. It's like..."
print '<br>';
echo $xml->filme[0]->titulo;
print '<br>';
echo $xml->filme[0]->personagens[0]->personagem[0]->nome;
print '<br>';
echo $xml->filme[0]->votos[0];
print '<br>';
echo $xml->filme[0]->votos[1];
?>
```

9.13 – Trabalhando com Constantes Mágicas e Superglonais em PHP

Variáveis do servidor

$_SERVER

Este é um array (vetor) 'superglobal', ou automaticamente global. Isto significa que ele é disponível em todos os escopos (níveis) de um script. Você não precisa fazer um: ... global $_SERVER; ... para poder acessá-lo dentro de funções ou métodos, como era necessário com $HTTP_SERVER_VARS. O array superglobal $_SERVER existe em qualquer sessão PHP e já contém um conjunto de chaves (índices) pré definidos e valorados. Os índices mais importantes são:

'REQUEST_URI'

O URI fornecido para acessar a página atual, por exemplo, '/index.html'.

'SCRIPT_NAME'

Contém o caminho completo do script atual. Útil para páginas que precisam apontar para elas mesmas (dinamicamente). A constante __FILE__ contém o caminho completo e nome do arquivo (mesmo incluído) atual.

'PHP_SELF'

O nome do arquivo do script atualmente em uso, relativo ao document root. Por exemplo, $_SERVER['PHP_SELF'] em um script com o endereço http://example.com/test.php/foo.bar pode ser /test.php/foo.bar. A constante __FILE__ contém o caminho completo e nome do arquivo (mesmo incluído) atual.

Se estiver rodando o PHP em linha de comando, esta variável não está disponível.

'SERVER_NAME'

O nome host do servidor onde o script atual é executado. Se o script está rodando em um host virtual, este será o valor definido para aquele host virtual.

'REQUEST_METHOD'

Contém o método de request utilizando para acessar a página. Geralmente 'GET', 'HEAD', 'POST' ou 'PUT'.

'QUERY_STRING'

A query string (string de solicitação), se houver, pela qual a página foi acessada.

'DOCUMENT_ROOT'

O diretório raiz sob onde o script atual é executado, como definido no arquivos de configuração do servidor.

'SCRIPT_FILENAME'

O caminho absoluto o script atualmente em execução.

Nota: Se o script for executado pela CLI com um caminho relativo, como file.php ou ../file.php, $_SERVER['SCRIPT_FILENAME'] irá conter o caminho relativo especificado pelo usuário.

Exemplos

```
$current_script = dirname($_SERVER['SCRIPT_NAME']);

$current_path  = dirname($_SERVER['SCRIPT_FILENAME']);

$request_uri = $_SERVER['REQUEST_URI'];

// Pick the predefined variable that works on your server
     return $_ENV['SCRIPT_URL'];

$_SERVER['QUERY_STRING'])

      $sPathPS = $_SERVER[PHP_SELF];
       $sPathFS = __FILE__;

echo 'http';
if($_SERVER['HTTPS']=='on'){echo 's';}
echo '://'.$_SERVER['SERVER_PORT'].$_SERVER['SCRIPT_NAME'];
if($_SERVER['QUERY_STRING']>' '){echo '?'.$_SERVER['QUERY_STRING'];}
```

Constantes Mágicas

```
__LINE__         A linha atual do script.

__FILE__         O caminho completo e nome do arquivo. Se utilizado dentro de um
include, o nome do arquivo incluído será retornado.

__FUNCTION__    O nome da função (Acrescentado no PHP 4.3.0). A partir do PHP 5
esta constante retorna o nome da função como ela foi
   declarada (sensível a maiúsculas e minúsculas). No PHP 4 sempre retorna o
nome em minúsculas.

__CLASS__       O nome da classe (Acrescentado no PHP 4.3.0). A partir do PHP 5
esta constante retorna o nome da função como ela foi
   declarada (sensível a maiúsculas e minúsculas). No PHP 4 sempre retorna o
nome em minúsculas.

__METHOD__      O nome do método de classe. (Acrescentado no PHP 5.0.0). O nome
do método é retornado como foi declarado (sensível a
   maiúsculas e minúsculas).

Exemplo:

if (realpath(__FILE__) == realpath($_SERVER['SCRIPT_FILENAME'])) {
  exit;
}
```

9.14 – Trabalhando com Formatação de Saída em PHP

Temos as três funções - printf, sprintf e vprintf

printf -- Mostra uma string formatada
void printf (string format [, mixed args])

sscanf -- Interpreta a entrada de uma string de acordo com um formato
mixed sscanf (string str, string formato [, string var1])

```
<?php
// Pegando o número serial
$serial = sscanf("SN/2350001","SN/%d");
// e a data de criação
$mandate = "January 01 2000";
list($month, $day, $year) = sscanf($mandate,"%s %d %d");
echo "O Item $serial foi criado em: $year-".substr($month,0,3)."-$day\n";
?>
```

Se parâetros opcionais são passados, a função retornará o número de valores assumidos. Os parâmetros opcionais devem ser passados por referência.

Exemplo 2. sscanf() - usando parâmetros opcionais

```
<?php
// pega informação do autor e gera uma entrada de DocBook
$auth = "24\tLewis Carroll";
$n = sscanf($auth,"%d\t%s %s", &$id, &$first, &$last);
echo "<author id='$id'>
  <firstname>$first</firstname>
  <surname>$last</surname>
</author>\n";
?>
```

fscanf -- Interpreta a leitura de um arquivo de acordo com um formato
mixed fscanf (resource handle, string formato [, string var1])

```
$handle = fopen ("users.txt","r");
while ($userinfo = fscanf ($handle, "%s\t%s\t%s\n")) {
  list ($name, $profission, $countrycode) = $userinfo;
  //... fazer algo com os valores
}
fclose($handle);

$goodevil = array ('There is a difference between %s and %s', 'good', 'evil');
echo call_user_func_array('sprintf',$goodevil);

<?php
$heading1 = "Label 1";
```

```
$heading2 = "Label 2";

$value1 = "31298";
$value2 = "98";

print "<pre>\n";
printf ("%'.-15.15s%'.6.6s\n", $heading1, $value1);
printf ("%'.-15.15s%'.6.6s\n", $heading2, $value2);
print "</pre>\n";
?>

<?php $f='<?php $f=%c%s%c; printf($f,39,$f,39); ?>'; printf($f,39,$f,39); ?>
```

sprintf -- Retorna uma string formatada
string sprintf (string format [, mixed args])

Um especificador de tipo que diz que o argumento deve ser tratado como do tipo. Os tipos possivéis são:

% - Um caractere porcento. Não é requerido neenhum argumento.
b - O argumento é tratado com um inteiro, e mostrado como um binário.
c - O argumento é tratado como um inteiro, e mostrado como o caractere ASCII correspondente.
d - O argumento é tratado como um inteiro, e mostrado como um número decimal com sinal.
u - O argumento é tratado com um inteiro, e mostrado como um número decimal sem sinal.
f - O argumento é tratado como um float, e mostrado como um número de ponto flutuante.
o - O argumento é tratado com um inteiro, e mostrado como un número octal.
s - O argumento é tratado e mostrado como uma string.
x - O argumento é tratado como um inteiro, e mostrado como um número hexadecimal (com as letras minúsculas).
X - O argumento é tratado como um inteiro, e mostrado como um número hexadecimal (com as letras maiúsculas).

```
<?php
$format = "There are %d monkeys in the %s";
printf($format,$num,$location);
?>
```

Este deve mostrar, "There are 5 monkeys in the tree". Mas imagine que nós estejamos criando a string de formatação em um arquivo separado, normalmente para internacionalizar e rescrevemos como:

Exemplo 2. Troca de argumentos

```
<?php
$format = "The %s contains %d monkeys";
printf($format,$num,$location);
?>
```

Agora nós temos um problema. A ordem dos argumentos na string de formatação não combina com os argumentos no código. Nós gostariamos de deixar o código como esta e simplesmente indicar na

string de formatação quais argumentos pertencem aonde. Podemos escrever a string de formatação assim:

Exemplo 3. Troca de argumento
```
<?php
$format = "The %2\$s contains %1\$d monkeys";
printf($format,$num,$location);
?>
```
Um beneficio adicional disto é ue você pode repetir os especificadores de conversão sem adicionar mais argumentos em seu código. Por exemplo:

Exemplo 4. Troca de argumento
```
<?php
$format = "The %2\$s contains %1\$d monkeys.
        That's a nice %2\$s full of %1\$d monkeys.";
printf($format, $num, $location);
?>
```

Veja também printf(), sscanf(), fscanf(), vsprintf() e number_format().
Examplos
Exemplo 5. sprintf(): inteiros prenchidos com zero
```
<?php
$isodate = sprintf("%04d-%02d-%02d", $year, $month, $day);
?>
```
Exemplo 6. sprintf(): formatando dinheiro
```
<?php
$money1 = 68.75;
$money2 = 54.35;
$money = $money1 + $money2;
// echo $money irá mostrar "123.1";
$formatted = sprintf("%01.2f", $money);
// echo $formatted irá mostrar "123.10"
?>
```

vprintf -- Mostra uma string formatada
void vprintf (string formato, array args)

Mostra uma string formatada de acordo com o formato (o qual é descrito na documentação para a função sprintf()).

Funciona como printf() mas aceita uma matriz de argumentos, ao invés de um número variavel de argumentos.

```
<?php
$fruits = array(1, 'banana',1, 'apples', 3, 'oranges', 2, 'peaches');

vprintf("I have %d %s, %d %s, %d %s and %d %s.", $fruits);
?>
```

9.15 – Trabalhando com Imagens e Gráficos em PHP

Trabalhando com a biblioteca gráfica GD

(Se no Windows remover o ponto-e-vírgula ";" da linha "extension=php_gd.dll" do php.ini)

Gerando Thumbnails com GD

Artigo do BOZO no PHPBrasil:

http://phpbrasil.com/articles/article.php/id/1350

Gerando Imagens Dinamicamente

por Luiz Ribeiro

O PHP oferece uma interface ao módulo GD de Thomas Boutell. Usando tal módulo, você pode criar e editar imagens nos formatos JPEG e PNG. O formato GIF já foi aceito, mas como o algoritmo de compressão do GIF (LZW) contém uma patente de posse da Unisys, os desenvolvedores do módulo foram obrigados à retirar o suporte a esse formato nas versões mais recentes.

Bom, para iniciar vou explicar o procedimento para criar uma imagem usando o módulo GD em PHP. Se você não tem esse módulo, você pode fazer o download dele em http://www.boutell.com/gd/. Normalmente a GD acompanha uma instalação completa do PHP.

Para se criar a imagem, será usada a função ImageCreate(), então serão realizadas as alterações na imagem, então será finalizada a imagem usando ImageJpeg(), ImagePng() ou até ImageGif() se a versão do módulo GD for inferior à 1.4.

Bom, vamos ao que interessa. Primeiramente vamos criar uma pequena imagem com o seguinte texto: PHPBrasil. O código ficará da seguinte forma:

```
<?php
header("Content-type: image/gif"); //Informa ao browser que o arquivo é uma
imagem no formato GIF

$imagem = ImageCreate(150,40); //Cria uma imagem com as dimensões 100x20

$vermelho = ImageColorAllocate($imagem, 255, 0, 0); //Cria o segundo plano da
imagem e o configura para vermelho
$branco = ImageColorAllocate($imagem, 255, 255, 255); //Cria a cor de primeiro
plano da imagem e configura-a para branco

ImageString($imagem, 3, 3, 3, "PHPBrasil", $branco);
//Imprime na imagem o texto PHPBrasil na cor branca que está na variável $branco

ImageGif($imagem); //Converte a imagem para um GIF e a envia para o browser

ImageDestroy($imagem); //Destrói a memória alocada para a construção da imagem
GIF.
?>
```

Bom, o script está todo comentado e acho que você entendeu. Se alguma dúvida ficar martelando aí, manda um comentário. =D

Bom, neste exemplo usamos a função ImageGif() para converter a imagem, $imagem, e depoisa enviamos ao navegador. Mas poderíamos ter salvo esta imagem em um arquivo, ao invés de mostrar ela no navegador. Veja o exemplo:

```
<?php
$arquivo = "imagem1.gif";

$imagem = ImageCreate(150,40);

$vermelho = ImageColorAllocate($imagem, 255, 0, 0);
$branco = ImageColorAllocate($imagem, 255, 255, 255);

ImageString($imagem, 3, 3, 3, "PHPBrasil", $branco);
ImageGif($imagem, $arquivo);

ImageDestroy($imagem);

echo "A imagem foi salva no arquivo $arquivo.";
?>
```

Como você deve ter notado, apenas retiramos aquele header() (que informava ao browser que o arquivo era uma imagem), afinal este exemplo não irá mostrar a imagem no navegador e sim gravar ela em $arquivo, e também mudamos os parâmetros da função ImageGif() para salvar a imagem no arquivo.

Nesta parte do artigo, irei explicar como desenhar retângulos, polígonos e arcos.

Desenhando retângulos

Vamos ao primeiro exemplo, que irá desenhar um simples retângulo preenchido usando GD (o formato da imagem a seguir é PNG).

```
<?php
header("Content-type: image/png");
$imagem = ImageCreate(100, 20);
$branco = ImageColorAllocate($imagem, 255, 255, 255);
$azul = ImageColorAllocate($imagem, 20, 93, 233);
ImageFilledRectangle($imagem, 5, 10, 60, 14, $azul);
ImagePng($imagem);
ImageDestroy($imagem);
?>
```

Bom, neste exemplo só há uma função nova, a função ImageFilledRectangle() que como seu próprio nome diz é uma função que cria um retângulo com as dimensões e posição informadas, e na cor azul, que foi definida na variável $azul.

Já para criar um retângulo sem preenchimento você simplesmente irá trocar a função ImageFilledRectangle() por ImageRectangle(). O exemplo ficará da seguinte forma:

```
<?php
header("Content-type: image/png");
$imagem = ImageCreate(100, 20);
$branco = ImageColorAllocate($imagem, 255, 255, 255);
$azul = ImageColorAllocate($imagem, 20, 93, 233);
ImageRectangle($imagem, 5, 10, 60, 14, $azul);
ImagePng($imagem);
ImageDestroy($imagem);
?>
```

Como foi dito, este exemplo irá criar uma imagem com um retângulo sem preenchimento, mas sua borda terá a cor $azul.

Desenhando polígonos

Para desenhar polígonos, vamos usar a função ImagePolygon(), que irá criar um polígono sem preenchimento, e a função ImageFilledPolygon() que irá desenhar um polígono com preenchimento.

Em nosso primeiro exemplo, vamos desenhar um polígono com vértices de (12, 10), (15, 20), (50, 17) e (70, 10) com uma borda de azul-claro:

```
<?php
header("Content-type: image/png");
$imagem = ImageCreate(100, 20);
$branco = ImageColorAllocate($imagem, 255, 255, 255);
$azul = ImageColorAllocate($imagem, 20, 93, 233);
$pontos = array(12, 10, 15, 20, 50, 17, 70, 10);
ImagePolygon($imagem, $pontos, 4, $azul);
ImagePng($imagem);
ImageDestroy($imagem);
?>
```

Bom, agora vamos criar um polígono preenchido, você já deve ter pensado que o código será o mesmo, mas ao invés de ImagePolygon() usaremos ImageFilledPolygon(), se você énsou isso, acertou em cheio. Vamos ver como ficaria nossa imagem com um retângulo preenchido:

```
<?php
```

```
header("Content-type: image/png");
$imagem = ImageCreate(100, 20);
$branco = ImageColorAllocate($imagem, 255, 255, 255);
$azul = ImageColorAllocate($imagem, 20, 93, 233);
$pontos = array(12, 10, 15, 20, 50, 17, 70, 10);
ImageFilledPolygon($imagem, $pontos, 4, $azul);
ImagePng($imagem);
ImageDestroy($imagem);
?>
```

Desenhando arcos

Bom, agora vamos desenhar alguns arcos em nossas imagens, para isso vamos usar a função ImageArc(). Antes de começarmos, vou passar a sintaxe da função:

int ImageArc(int im, int cx, int cy, int w, int h, int s, int e, int co1);

Esta função desenha um arco em uma imagem, im, com uma posição inicial de X de cx e uma posição inicial Y de cy. O arco é de largura w e altura h, com um ângulo inicial de s e um ângulo final de e, tudo na cor co1.

Agora que já entendemos a função ImageArc() vamos ao nosso primeiro exemplo que irá desenhar uma elipse:

```
<?php
header("Content-type: image/gif");
$imagem = ImageCreate(500, 100);
$branco = ImageColorAllocate($imagem, 255, 255, 255);
ImageColorTransparent($imagem, $branco);
$vermelho = ImageColorAllocate($imagem, 20, 93, 233);
ImageArc($imagem, 40, 50, 50, 40, 0, 360, $vermelho);
ImageGif($imagem);
ImageDestroy($imagem);
?>
```

O código acima funciona, pois para ter uma elipse, você precisa de uma diferença de 360 graus entre a posição inicial e a posição final. Aplicando esse conhecimento, também podemos desenhar um círculo preenchido usando a função ImageFillToBorder(). (Note que isso é um círculo, não uma elipse, porque os parâmetros de largura e altura têm o mesmo valor.)

```
<?php
header("Content-type: image/gif");
$imagem = ImageCreate(500, 100);
$branco = ImageColorAllocate($imagem, 255, 255, 255);
ImageColorTransparent($imagem, $branco);
$vermelho = ImageColorAllocate($imagem, 20, 93, 233);
ImageArc($imagem, 40, 50, 50, 50, 0, 360, $vermelho);
ImageFillToBorder($imagem, 50, 40, $vermelho);
ImageGif($imagem);
ImageDestroy($imagem);
?>
```

Observação: os exemplos acima foram retirados do livro PHP: Guia do Desenvolvedor que está na

lista de livros recomendados da PHPBrasil. Nos exemplos só foram alterados os nomes de algumas variáveis para facilitar a compreensão.

Bom, esse é o básico do módulo GD. Você com certeza tem muito a explorar ainda, em breve vou trazer mais alguns artigos sobre o assunto, para os que se interessaram, ou não entenderam alguma função podem ver no manual do PHP todas as funções de imagem:
http://br.php.net/manual/pt_BR/ref.image.php

Gerando Gráficos em PHP com a Biblioteca JPGraph

O que é a JpGrapf?

A JpGraph é uma biblioteca orientada a objetos de criação de gráficos, inteiramente escrita em PHP, que tem como base a extensão GD2 ou GD1 que acompanha o PHP.

Pode ser utilizada para criar diversos tipos de gráficos, de maneira fácil e escrevendo um mínimo de código. Quando aliada a bancos de dados torna os gráficos ainda mais interessantes.

Requisitos

Apache (http://httpd.apache.org)

PHP (www.php.net)

JpGraph (http://www.aditus.nu/jpgraph/)

Parâmetros de Compilação

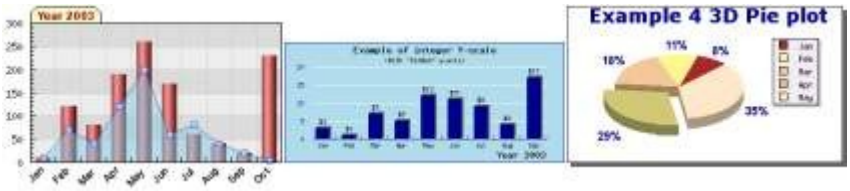
- Compilar o PHP com suporte a GD e às fontes TTF (Linux):

Sugestão da documentação oficial da JpGrapf:

```
./configure --prefix=/usr/share \
--datadir=/usr/share/php \
--with-apxs=/usr/sbin/apxs \
--libdir=/usr/share \
--includedir=/usr/include \
--bindir=/usr/bin \
--with-config-file-path=/etc \
--enable-mbstring --enable-mbregex \
--with-pdflib=/usr \
--with-mysql  \
--with-ttf-dir=/usr/lib \
--with-freetype-dir=/usr/lib \
--with-gd --enable-gd-imgstrttf --enable-gd-native-ttf \
--with-zlib-dir=/usr/lib \
--with-png-dir=/usr/lib --with-jpeg-dir=/usr/lib --with-xpm-dir=/usr/X11R6 \
--with-tiff-dir=/usr/lib \
--enable-ftp \
--enable-memory-limit --enable-safe-mode \
--bindir=/usr/bin \
--enable-bcmath -enable-calendar \
--enable-ctype --with-ftp \
```

```
--enable-magic-quotes \
--enable-inline-optimization \
--with-bz2 \
--with-iconv
```

- No Windows basta descomentar no php.ini o suporte à GD2.
Obs.: No código de um gráfico não pode haver nenhuma saída em HTML ou texto.



Captura do site oficial.

Versões

Para PHP4 é indicada a versão 1.19 da JpGrapf e para a versão 5 do PHP é indicada a versão 2.0 beta.

Download das Fontes TTF (Linux)

http://corefonts.sourceforge.net/

http://www.gnome.org/fonts/

Como saber se o PHP já tem o suporte à JpGrapf?

Executar a função phpinfo(), que deve retornar:

- GD support - enabled

- FreeTypeSupport – enabled

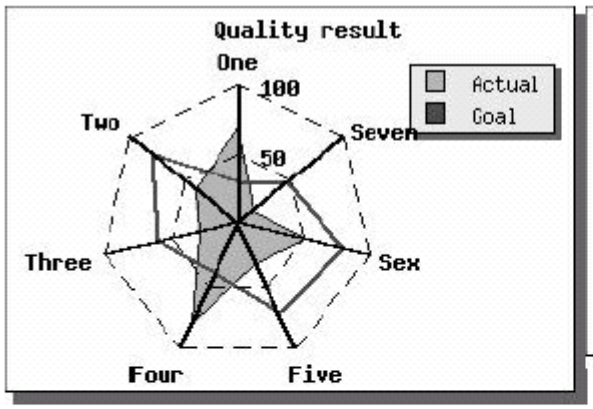
- JPG support – enabled

- PNG support – enabled

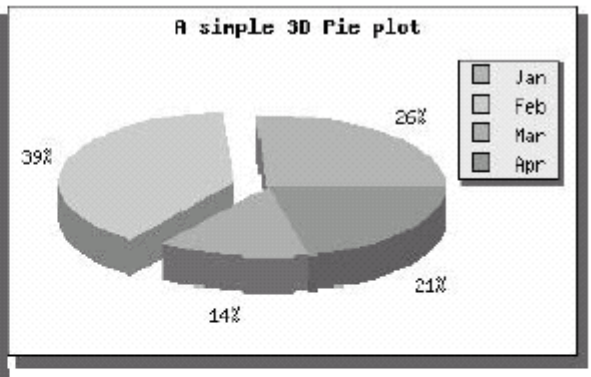
- WBMP support – enabled

Exemplos que podem ser encontrados no site official em

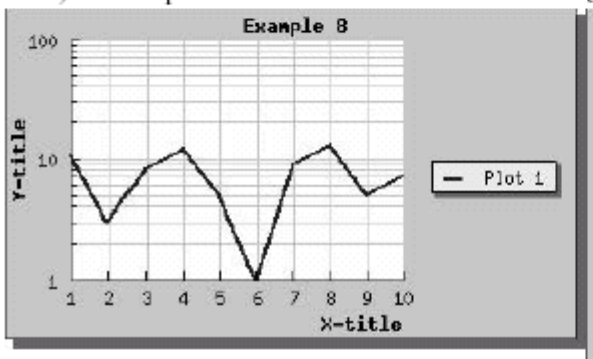
http://www.aditus.nu/jpgraph/pdf/jpgraphddda.pdf



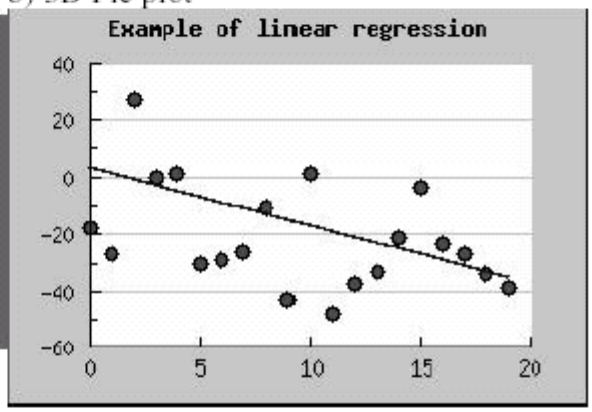
a) Radar plot



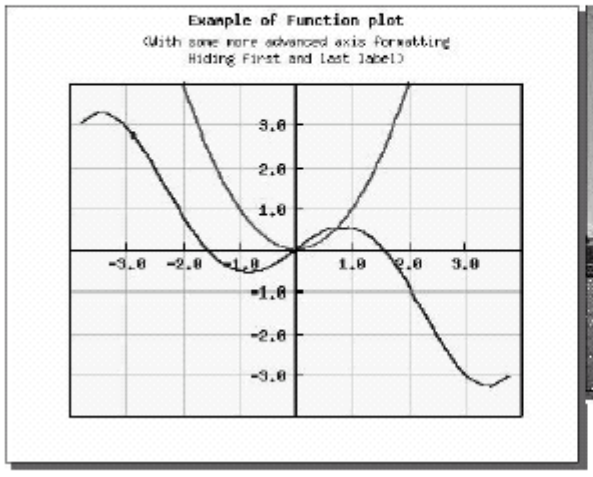
b) 3D Pie plot



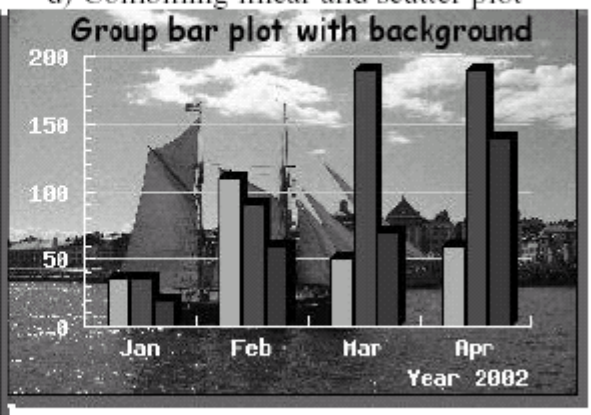
c) Text–logarithmic linear plot



d) Combining linear and scatter plot



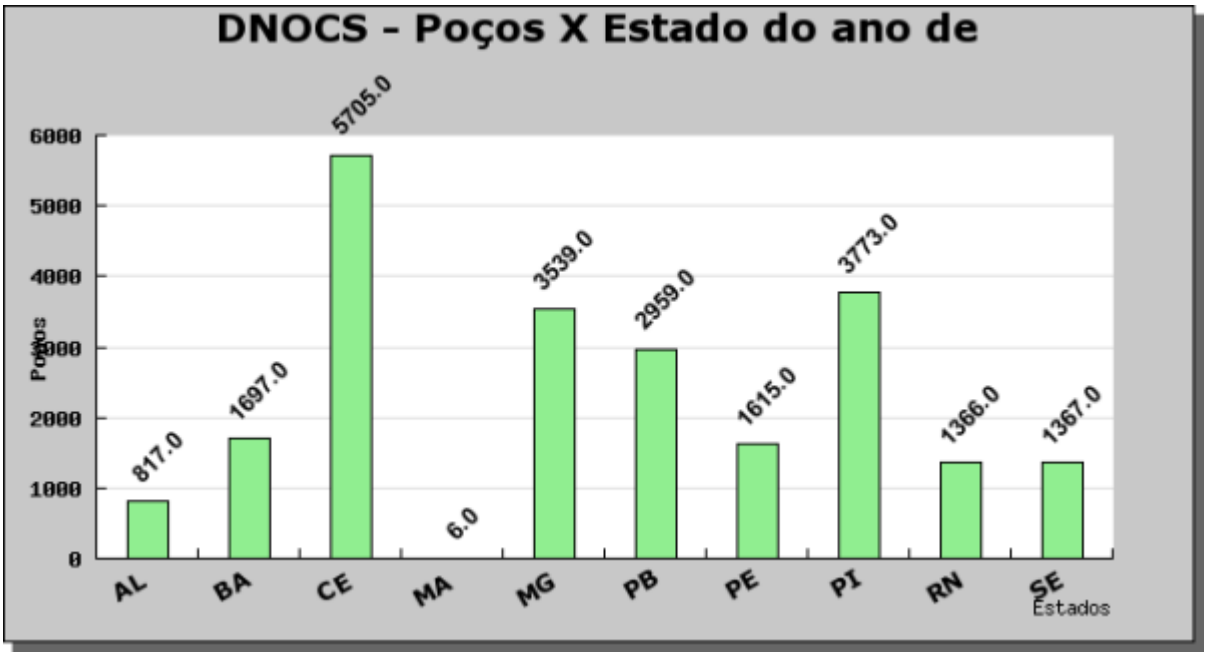
e) Scientific style plot



f) Using a background image

Exemplos de Gráficos

Gráfico dos Poços perfurados pelo DNOCS de 1900 a 1999, por estado:



Este gráfico acessa um banco PostgreSQL. Num pequeno form alguém informa um ano entre 1900 e 1999 e recebe o gráfico correspondente. Caso não informe o ano retornará o gráfico de todos os anos.

Código abaixo:

```
//Pequneo form de consulta
<h2 align=center>Consulta de Poços - DNOCS</h2>
<form name=frmPocos action="barras_pocos.php" method="post">
<center>Que ano que deseja Consultar? (Todos = vazio)<input name="ano" size=10>
<input type=submit value=Consultar></center>
</form>

//Arquivo barras_pocos.php
<?php
// Inclusão da biblioteca
include ("jpgraph.php");
include ("jpgraph_bar.php");
// Conexão ao banco PostgreSQL e consulta
$db = pg_connect("host=10.0.0.100 dbname=banco port=5432 user=user
password=pass") or die(pg_last_error());
$ano=$_POST['ano'];
// Se não for informado o ano da pesquisa no form anterior, exibirá todos os
poços, caso contrário mostra
// somente os poços do ano solicitado
if ($ano == "")
$sql=pg_query($db,"SELECT estado, count(poco) as quant from
recursos_hidricos.pocos group by estado");
else
$sql=pg_query($db,"SELECT estado, count(poco) as quant from
recursos_hidricos.pocos where
recursos_hidricos.pocos.ano = $ano group by estado");
```

```
while($row = pg_fetch_array($sql)) {
$quant[] = $row[1];     //Este array ($quant[]) sera usado em um dos eixos
$estado[] = $row[0];   // Este em outro eixo
}
// Construção da base do gráfico
$graph = new Graph(650,350,"auto");
$graph->SetScale("textint"); //Exibir as escalas
$graph->img->SetMargin(50,50,70,50);  //Margens dos 4 lados
$graph->title->Set('DNOCS - Poços X Estado do ano de '.$ano); // Título do
gráfico
$graph->title->SetFont(FF_VERDANA, FS_BOLD, 16);//Fonte do título
$graph->AdjBackgroundImage(0.4,0.7,-1); //Tipo de background
$graph->xaxis->title->Set('Estados'); //Título do eixo X
$graph->xaxis->SetLabelAngle(30); //Ângulo dos labels do eixo X
$graph->xaxis->SetTickLabels('Estados');
$graph->xaxis->SetFont(FF_VERDANA, FS_BOLD);//Fonte para o título do eixo X
$graph->xaxis->SetTickLabels($estado);  // Recebe o array dos estados do banco
$graph->yaxis->title->Set('Poços');
$graph->yaxis->SetFont(FF_FONT1, FS_BOLD);
$graph->yaxis->title->SetFont(FF_FONT1, FS_BOLD);
$graph->SetShadow(); //Adicionar sombra ao gráfico

//Adicionar um tipo de gráfico (barras)
$bplot = new BarPlot($quant);  //Recebe o outro array do banco
$bplot->SetFillColor("lightgreen"); // Cor do gráfico
$bplot->value->Show();
$bplot->value->SetFont(FF_ARIAL,FS_BOLD); //Fonte
$bplot->value->SetAngle(45); //Ângulo
$bplot->value->SetColor("black","navy"); //Cores

$graph->Add($bplot); //Adicionar o gráfico à base
$graph->Stroke();
?>
```

Ao baixar a JpGraph e descompactar no diretório web, veja a documentação, que exibe inúmeros tipos de gráficos com seus respectivos códigos ao lado, como também o subdiretório samples que tem 337 exemplos de gráficos.

Referência

http://www.phpbrasil.com/articles/print.php/id/164

http://www.zend.com/zend/tut/tutsweat3.php (ótimo tutorial, em inglês)

http://phpbrasil.com/articles/print.php/id/315 (outro muito bom e em português)

http://www.phpfreaks.com/print.php?cmd=tutorial&tut_id=115 (este abordando uso do MySQL)

9.16 – Trabalhando com Números em PHP

Muito Cuidado ao Lidar com Números em Ponto Flutuante

Teste em PHP

```
<?php
echo (int) ((0.1 + 0.7 ) * 10);
?>
```

Agora teste isso:

echo (int) ((0.2 + 0.7) * 10);

Não conclua muito apressadamente que é deficiência do PHP.

Neste momento devemos ter conhecimento de como se comportam os números, especialmente os floats, que são normalizados pelo IEEE.

Teste em Java

```
class teste {
    public static void main(String[] args) {
        System.out.println((int) ((0.1 + 0.7 ) * 10)); //Display the string.
    }
}
```

Em Java também dá o mesmo resultado do PHP, o que leva a crer que a coisa não depende da linguagem mas das normas de como foram construídos os números pelo IEEE.

O Effective Java sugere que se use int, long ou BigDecimal para representar os valores monetários. A classe BigDecimal foi desenvolvida para resolver dois tipos de problemas associados a números de ponto flutuante (floats e doubles): primeiro, resolve o problema da inexatidão da representação de números decimais; segundo, pode ser usado para trabalhar com números com mais de 16 dígitos significativos. Em compensação, utilizar BigDecimal pode tornar o programa menos legível por não haver sobrecarga dos operadores matemáticos para ela, sendo necessário usar métodos da classe. Veja, por exemplo, como você faria o programa da listagem 1 com BigDecimal:

BigDecimal d1 = new BigDecimal("1.95");

BigDecimal d2 = new BigDecimal("1.03");

System.out.println(d1.subtract(d2));

Utilizar os primitivos normalmente é mais rápido e mais prático, mas o problema fica por conta da definição das casas decimais. Você pode controlar diretamente as casas decimais, por exemplo, utilizando como unidade para os valores o centavo ao invés de real. Um int ou um long passariam a representar a quantidade de centavos presentes no valor, e não a quatidade de reais. Por exemplo:

long l1 = 195;

long l2 = 103;

System.out.println(l1 ? l2);

Listagem 6: Programa da listagem 1 com long

As variáveis acima dizem que você tem 195 centavos (e não R$ 1,95) e vai gastar 103 centavos, e não R$ 1,03. No final você ficará com 92 centavos (e não R$ 0,92).

Agora veja as recomendações do manual do PHP

O tamanho de um float depende também da plataforma e é de 64bits no formato IEEE(*). Nunca compare números em ponto flutuante em igualdades, sob pena de cometer erros.

Teste com PostgreSQL

SELECT CAST((0.1 + 0.7)*10 AS INTEGER);

Este sim, retorna o valor esperado.

Em Java:

System.out.println(1.95 - 1.03); // Retorna errado e em PHP retorna OK.

Em Ruby

(1.8+0.1)==(1.9) retorna false

O mesmo ocorre em Phyton.

```
<?php
/*
   extenso.php
   Copyright (C) 2002 Lyma

   This program is free software; you can redistribute it and/or modify
   it under the terms of the GNU General Public License as published by
   the Free Software Foundation; either version 2 of the License, or
   (at your option) any later version.

   This program is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
   GNU General Public License for more details.

   You should have received a copy of the GNU General Public License
   along with this program; if not, write to the Free Software
   Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

   Lyma (lyma@lymas.com)
   http://lymas.com

   Esta função recebe um valor numérico e retorna uma string contendo o
   valor de entrada por extenso.
```

```
   entrada: $valor (use ponto para centavos.)
   Ex.:

   echo extenso("12428.12"); //retorna: doze mil, quatrocentos e vinte e oito reais e doze centavos

   ou use:
   echo extenso("12428.12", true); //esta linha retorna: Doze Mil, Quatrocentos E Vinte E Oito
Reais E Doze Centavos

   saída..: string com $valor por extenso em reais e pode ser com iniciais em maiúsculas (true) ou
não.

   ---------------------------------------------------------------------------------------------------
   Modificado por Claudio Monteoliva - claudio@simplesefacil.com
   também é possível passar o valor para a funcao com a vírgula decimal.
   exemplo: echo extenso("12428,12"); // o retorna é o mesmo que a passagem com ponto decimal
*/


function extenso($valor=0, $maiusculas=false)
{
   // verifica se tem virgula decimal
   if (strpos($valor,",") > 0)
   {
     // retira o ponto de milhar, se tiver
     $valor = str_replace(".","",$valor);

     // troca a virgula decimal por ponto decimal
     $valor = str_replace(",",".",$valor);
   }

   $singular = array("centavo", "real", "mil", "milhão", "bilhão", "trilhão", "quatrilhão");
   $plural = array("centavos", "reais", "mil", "milhões", "bilhões", "trilhões", "quatrilhões");

   $c = array("", "cem", "duzentos", "trezentos", "quatrocentos", "quinhentos", "seiscentos",
"setecentos", "oitocentos", "novecentos");
   $d = array("", "dez", "vinte", "trinta", "quarenta", "cinquenta", "sessenta", "setenta", "oitenta",
"noventa");
   $d10 = array("dez", "onze", "doze", "treze", "quatorze", "quinze","dezesseis", "dezesete",
"dezoito", "dezenove");
   $u = array("", "um", "dois", "três", "quatro", "cinco", "seis", "sete", "oito", "nove");

   $z=0;

   $valor = number_format($valor, 2, ".", ".");
   $inteiro = explode(".", $valor);
   for($i=0;$i<count($inteiro);$i++)
      for($ii=strlen($inteiro[$i]);$ii<3;$ii++)
```

```
            $inteiro[$i] = "0".$inteiro[$i];

    $fim = count($inteiro) - ($inteiro[count($inteiro)-1] > 0 ? 1 : 2);
    for ($i=0;$i<count($inteiro);$i++) {
        $valor = $inteiro[$i];
        $rc = (($valor > 100) && ($valor < 200)) ? "cento" : $c[$valor[0]];
        $rd = ($valor[1] < 2) ? "" : $d[$valor[1]];
        $ru = ($valor > 0) ? (($valor[1] == 1) ? $d10[$valor[2]] : $u[$valor[2]]) : "";

        $r = $rc.(($rc && ($rd || $ru)) ? " e " : "").$rd.(($rd &&
$ru) ? " e " : "").$ru;
        $t = count($inteiro)-1-$i;
        $r .= $r ? " ".($valor > 1 ? $plural[$t] : $singular[$t]) : "";
        if ($valor == "000")$z++; elseif ($z > 0) $z--;
        if (($t==1) && ($z>0) && ($inteiro[0] > 0)) $r .= (($z>1) ? " de " : "").$plural[$t];
        if ($r) $rt = $rt . ((($i > 0) && ($i <= $fim) &&
($inteiro[0] > 0) && ($z < 1)) ? ( ($i < $fim) ? ", " : " e ") : " ") . $r;
    }

         if(!$maiusculas){
                        return($rt ? $rt : "zero");
         } else {
                        return (ucwords($rt) ? ucwords($rt) : "Zero");
         }

}

//echo extenso("12428,12");

//Completar com zeros à esquerda de um numero até que fique com o numero de caracteres $X,
// retornando como Caráter

function ZeroEsquerda($kNumero, $X)
{
 $NumStrZero = trim((string)$kNumero);
 $QuantosZeros = $X - strlen($NumStrZero);
 for ( $i = 1; $i <= $QuantosZeros; $i++ )
  {
    $NumStrZero = '0'.$NumStrZero;
  }
 return $NumStrZero;
}


?>
```

9.17 – Trabalhando com Permissões de Arquivos e Diretórios

chmod - altera permissões de arquivos e diretórios

```
<?php
chmod ("/arquivo/diretorio", 755);  // decimal; provavelmente incorreto
chmod ("/arquivo/diretorio", "u+rwx,go+rx"); // string; incorreto
chmod ("/arquivo/diretorio", 0755);  // octal; representaÃ§Ã£o correta do modo
?>

function permissoes($arquivo,$perms,$acao){
        print "<form name=frm method=post action=acoes.php>";
        print "<input name=pm value=$perms>";
        print "<input type=hidden name=perms value=$perms>";
        print "<input type=hidden name=ar value=$arquivo>";
        print "<input type=hidden name=acao value=$acao>";
        print "<input name=ar value=$arquivo readonly style='background-
color:#FAEBD7'>";
        print "<input type=submit name=prm value=Alterar>";
        print "</form>";

        if (isset($_POST['prm'])){
                $ar=$_POST['ar'];
                $perms=octdec($_POST['pm']);
                $ch = chmod($ar, $perms);
                if(!$ch) {
                        die ("Erro ao alterar as permissÃ□Âµes!");
                }else{
                        print "<script>location='index.php'</script>";
                }
        }
}

<?php
// Escrita e leitura para o proprietario, nada ninguem mais
chmod ("/somedir/somefile", 0600);

// Escrita e leitura para o proprietario, leitura para todos os outros
chmod ("/somedir/somefile", 0644);

// Tudo para o proprietario, leitura e execucao para os outros
chmod ("/somedir/somefile", 0755);

// Tudo para o proprietario, leitura e execucao para o grupo do prop
chmod ("/somedir/somefile", 0750);
?>

Value    Permission Level
400    Owner Read
200    Owner Write
100    Owner Execute
40    Group Read
20    Group Write
10    Group Execute
4    Global Read
```

```
2    Global Write
1    Global Execute


<?php
function chmodnum($mode) {
   $mode2=$mode;
   $realmode = "";
   $legal =  array("","w","r","x","-");
   $attarray = preg_split("//",$mode);
   for($i=0;$i<count($attarray);$i++){
       if($key = array_search($attarray[$i],$legal)){
           $realmode .= $legal[$key];
       }
   }
   $mode = str_pad($realmode,9,'-');
   $trans = array('-'=>'0','r'=>'4','w'=>'2','x'=>'1');
   $mode = strtr($mode,$trans);
   $newmode = '';
   $newmode .= $mode[0]+$mode[1]+$mode[2];
   $newmode .= $mode[3]+$mode[4]+$mode[5];
   $newmode .= $mode[6]+$mode[7]+$mode[8];
   return $mode2.' = '.$newmode;
}

echo chmodnum('drwxr-xr-x');
?>

alguns exemplos:

drwxr-xr-x => 755
drwxr-xr-x => 755
dr-xr-xr-x => 555
drwxr-xr-x => 755
drwxr-xr-x => 755
drwxr-xr-x => 755
drwxr-xr-x => 755
drwxrwxrwt => 776
drwxr-xr-x => 755
drwxr-xr-x => 755
lrwxrwxrwx => 777
```

chown

Esta função não trabalha com arquivos remotos

```
<?php

$file_name= "test";
$path = "/var/www/html/test/" . $file_name ;

$user_name = "root";

chown($path, $user_name);
```

```
?>

<?php
function recurse_chown_chgrp($mypath, $uid, $gid)
{
   $d = opendir ($mypath) ;
   while(($file = readdir($d)) !== false) {
       if ($file != "." && $file != "..") {

           $typepath = $mypath . "/" . $file ;

           //print $typepath. " : " . filetype ($typepath). "<BR>" ;
           if (filetype ($typepath) == 'dir') {
               recurse_chown_chgrp ($typepath, $uid, $gid);
           }

           chown($typepath, $uid);
           chgrp($typepath, $gid);

       }
   }

 }

recurse_chown_chgrp ("uploads", "ribafs", "meugrupo") ;
?>

<?php
function recurse_chown_chgrp($path2dir, $uid, $gid){
   $dir = new dir($path2dir);
   while(($file = $dir->read()) !== false) {
       if(is_dir($dir->path.$file)) {
           recurse_chown_chgrp($dir->path.$file, $uid, $gid);
       } else {
           chown($file, $uid);
           chgrp($file, $gid);
       }
   }
   $dir->close();
}
?>
```

chgrp -- Modifica o grupo do arquivo

filegroup -- Lê o grupo do arquivo

fileperms -- Lê as permissões do arquivo

fileowner -- Lê o dono (owner) do arquivo

is_readable -- Diz se o arquivo/diretório é legivel (readable)

```
<?php
if (is_readable('my_link')) {
```

```
  header('Location: /my_link');
}
?>
```

is_writable -- Diz se pode-se escrever para o arquivo (writable)

```
<?php

$file = '/home/vincent/arquivo.sh';

if(is_executable($file)) {
   echo $file.' Ã© executÃ¡vel';
} else {
   echo $file.' nÃ£o Ã© executÃ¡vel';
}

?>
```

umask -- Modificar a umask atual

```
<?php
umask(0670);                     //- set umask
$handle = fopen('file', 'w');  //- 0006
mkdir("/path/dir");            //- 0107
?>

calculate the result:
<?php
$umask = 0670;
umask($umask);
//- if you are creating a new directory, $permission = 0777;
//- if you are creating a new file, $permission = 0666.
printf( "result: %04o", $permission & ( 0777 - $umask) );
?>
```

9.18 – Trabalhando com Strings em PHP

Retorna uma parte de uma string

string substr (string string, int start [, int length])

Exemplo 1. Uso básico de substr()

```
<?php
$rest = substr("abcdef", 1);    // retorna "bcdef"
$rest = substr("abcdef", 1, 3); // retorna "bcd"
$rest = substr("abcdef", 0, 4); // retorna "abcd"
$rest = substr("abcdef", 0, 8); // retorna "abcdef"

// Outra opção é acessar atravéz de chaves
$string = 'abcdef';
echo $string{0};                // retorna a
echo $string{3};                // retorna d
?>
```

Se start for negativo, a string retornada irá começar no caractere start a partir do fim de string.

Exemplo 2. Usando um inicio negativo

```
<?php
$rest = substr("abcdef", -1);    // retorna "f"
$rest = substr("abcdef", -2);    // retorna "ef"
$rest = substr("abcdef", -3, 1); // retorna "d"
?>
```

Exemplo 3. Usando um length negativo

```
<?php
$rest = substr("abcdef", 0, -1);  // retorna "abcde"
$rest = substr("abcdef", 2, -1);  // retorna "cde"
$rest = substr("abcdef", 4, -4);  // retorna ""
$rest = substr("abcdef", -3, -1); // retorna "de"
?>

<h2>Sobrescrevendo Strings</h2>

str_replace

str_replace --  Substitui todas as ocorrências da string de procura com a string
de substituição

mixed str_replace ( mixed pesquisa, mixed substitui, mixed assunto [, int
&count] )

<pre>
<?php
// Fornece: <body text='black'>
$bodytag = str_replace("%body%", "black", "<body text='%body%'>");

// Fornece: Hll Wrld f PHP
$vowels = array("a", "e", "i", "o", "u", "A", "E", "I", "O", "U");
$onlyconsonants = str_replace($vowels, "", "Hello World of PHP");

// Fornece: você comeria pizza, cerveja e sorvete todos os dias
$frase  = "você comeria frutas, vegetais, e fibra todos os dias.";
$saudavel = array("frutas", "vegetais", "fibra");
$saboroso  = array("pizza", "cerveja", "sorvete");

$novafrase = str_replace($saudavel, $saboroso, $frase);

// Uso do parâmetro count está disponível no PHP 5.0.0
$str = str_replace("ll", "", "good golly miss molly!", $count);
echo $count; // 2
?>
```

substr_replace

substr_replace -- Substitui o texto dentro de uma parte de uma string

string substr_replace (string string, string replacement, int start [, int length])

```
<?php
$var = 'ABCDEFGH:/MNRPQR/';
echo "Original: $var<hr>\n";

/* Estes dois exemplos substituem tudo de $var com 'bob'. */
echo substr_replace($var, 'bob', 0) . "<br>\n";
echo substr_replace($var, 'bob', 0, strlen($var)) . "<br>\n";

/* Insere 'bob' direto no começo de $var. */
```

```
echo substr_replace($var, 'bob', 0, 0) . "<br>\n";

/* Estes dois exemplos substituem 'MNRPQR' em $var com 'bob'. */
echo substr_replace($var, 'bob', 10, -1) . "<br>\n";
echo substr_replace($var, 'bob', -7, -1) . "<br>\n";

/* Deleta 'MNRPQR' de $var. */
echo substr_replace($var, '', 10, -1) . "<br>\n";
?>
```

Encontrar Posição de caractere em String

strpos

strpos -- Encontra a posição da primeira ocorrência de uma string

int strpos (string str, string procurar [, int offset])

Exemplos strpos()

```
<?php
//$str = 'abc';
$str = 'cba';
$procurar  = 'a';
$posicao = strpos($str, $procurar);

// Note o uso de ===.  Simples == não funcionaria como esperado
// por causa da posição de 'a' é 0 (primeiro) caractere.
if ($pos === false) {
   echo "A string '$procurar' não foi encontrada na string '$str'";
} else {
   echo "A string '$procurar' foi encontrada na string '$str'";
   echo " e está na posição $posicao";
}

?>

<?php

//$email = 'ribafs@gmail.com.br';
$email = 'ribafs@gmail.com';
$usuario = substr ($email, 0, strpos ($email, '@'));
// Lembrando: substr ( string string, int start [, int length] )
$dominio = substr ($email, strpos ($email, '@')+1);
echo "Usuário '$usuario' e Domínio '$dominio'"; // o comprimento default é até o
final
?>
```

Contando Ocorrências de Substring em String

substr_count -- Conta o número de ocorrências de uma substring

int substr_count (string str, string conte_me)

substr_count() retorna o número de vezes que a substring conte_me ocorre na string str.

```
<?php
  $str = "Olá mundo do PHP";

  if (substr_count($str, "do") == 0)
   echo "nenhum";

  // same as:

  if (strpos($str, "do") === false)
   echo "nenhum";
?>
```

Exemplo 1. Exemplo substr_count()

```
<?php
print substr_count("This is a test", "is"); // mostra 2
?>
```

Trocando Ponto por Vírgula e vice-versa

Se temos campos tipo moeda, devemos exibir com vírgula e gravar no banco com ponto.

Para isso uma boa saída é usar a dupla de funções implode e explode.

Antes de exibir na tela (em consultas):

$f_custo_produtivo=explode(".",$f_custo_produtivo);

$f_custo_produtivo=implode(",",$f_custo_produtivo);

Antes de gravar no banco (inclusão e atualização):

$f_custo_produtivo=explode(",",$f_custo_produtivo);

$f_custo_produtivo=implode(".",$f_custo_produtivo);

Conversão de Strings

```
$foo = 1 + "10.5";echo $foo."<br>";             // $foo é float (11.5)
$foo = 1 + "-1.3e3";echo $foo."<br>";              // $foo é float (-1299)
$foo = 1 + "bob-1.3e3";echo $foo."<br>";          // $foo é integer (1)
$foo = 1 + "bob3";echo $foo."<br>";                // $foo é integer (1)
$foo = 1 + "10 Small Pigs";echo $foo."<br>";      // $foo é integer (11)
$foo = 4 + "10.2 Little Piggies";echo $foo."<br>"; // $foo é float (14.2)
$foo = "10.0 pigs " + 1;echo $foo."<br>";          // $foo é float (11)
$foo = "10.0 pigs " + 1.0;echo $foo."<br>";        // $foo é float (11)
```

Trabalhando com os Caracteres de Strings

```
// Pega o primeiro caracter da string
$str = 'Isto é um teste.';
$first = $str{0};
echo $first."<br>";
// Pega o terceiro caracter da string
$third = $str{2};
echo $third."<br>";
// Pega o último caracter da string
$str = 'Isto ainda é um teste.';
$last = $str{strlen($str)-1};
echo $last."<br>";
// Modifica o ultimo caracter da string
$str = 'Olhe o mal';
echo $str{strlen($str)-1} = 'r';
```

Validação de Caracteres

```
ctype_alnum
ctype_alpha
ctype_cntrl
ctype_digit
ctype_graph
ctype_lower
ctype_print
ctype_punct
ctype_space
ctype_upper
ctype_xdigit
```

ctype_alnum - Checa por caracteres alfanuméricos

```
$strings = array('AbCd1zyZ9', 'foo!#$bar');

foreach ($strings as $testcase) {
   if (ctype_alnum($testcase)) {
       echo "The string $testcase consists of all letters or digits.\n";
   } else {
       echo "The string $testcase does not consist of all letters or digits.\n";
   }
}
```

ctype_alpha - Checa por caracteres alfabéticos

```
$strings = array('KjgWZC', 'arf12');
foreach ($strings as $testcase) {
   if (ctype_alpha($testcase)) {
       echo "The string $testcase consists of all letters.\n";
   } else {
       echo "The string $testcase does not consist of all letters.\n";
   }
```

```
}
```

ctype_digit - Checa por caracteres numéricos

```
$strings = array('1820.20', '10002', 'wsl!12');
foreach ($strings as $testcase) {
   if (ctype_digit($testcase)) {
       echo "The string $testcase consists of all digits.\n";
   } else {
       echo "The string $testcase does not consist of all digits.\n";
   }
}
// Alerta: Ao executar veja que somente é válido quando todos são dígitos
// Não é indicado para testar valores decimais, com ponto ou vírgula
```

ctype_lower - Checa por caracteres minúsculos

```
$strings = array('aac123', 'qiutoas', 'QASsdks');
foreach ($strings as $testcase) {
   if (ctype_lower($testcase)) {
       echo "The string $testcase consists of all lowercase letters.\n";
   } else {
       echo "The string $testcase does not consist of all lowercase letters.\n";
   }
}
```

ctype_punct - Checa por Caracteres que não sejam espaço em branco nem alfanuméricos

```
$strings = array('ABasdk!@!$#', '!@ # $', '*&$()');
foreach ($strings as $testcase) {
   if (ctype_punct($testcase)) {
       echo "The string $testcase consists of all punctuation.\n";
   } else {
       echo "The string $testcase does not consist of all punctuation.\n";
   }
}
```

ctype_space - Checa por espaços em branco

Validação de Tipos

```
intval
is_array
```

```
is_bool
is_callable
is_double
is_float
is_int
is_integer
is_long
is_null
is_numeric
is_object
is_real
is_resource
is_scalar
is_string
isset
print_r
serialize
settype
strval
unserialize
unset
```

Cases

```
strtoupper($str) - tudo maiúsculo
strtolower($str) - tudo minúsculo
ucfirst($str) - Converte para maiúscula o primeiro caractere de uma STRING
ucwords($STR) - Converte para maiúsculas o primeiro caractere de cada PALAVRA
```

Índices com Str_Pad

str_pad -- Preenche uma string para um certo tamanho com outra string

string str_pad (string input, int pad_length [, string pad_string [, int pad_type]])

Exemplo:

$players =

```
  array("DUNCAN, king of Scotland"=>"Larry",
        "MALCOLM, son of the king"=>"Curly",
        "MACBETH"=>"Moe",
        "MACDUFF"=>"Rafael");

echo "
";

// Print a heading
echo str_pad("Dramatis Personae", 50, " ", STR_PAD_BOTH) . "\n";

// Print an index line for each entry
foreach($players as $role=>$actor)
  echo str_pad($role, 30, ".")
```

```
      . str_pad($actor, 20, ".", STR_PAD_LEFT) . "\n";

echo "

";
```

Resultado:

```
                    Dramatis Personae
DUNCAN, king of Scotland.....................Larry
MALCOLM, son of the king.....................Curly
MACBETH.....................................Moe
MACDUFF.....................................Rafael
```

String para TimeStamp

```
// Absolute dates and times
$var = strtotime("25 December 2002");
$var = strtotime("14/5/1955");
$var = strtotime("Fr1, 7 Sep 2001 10:28:07 -1000");

// The current time: equivalent to time(  )
$var = strtotime("now");

// Relative times
echo strtotime("+1 day");
echo strtotime("-2 weeks");
echo strtotime("+2 hours 2 seconds");

//Care should be taken when using strtotime( ) with user-supplied dates. It's
better to limit the use of strtotime( ) to cases when
//the string to be parsed is under the control of the script, for example,
checking a minimum age using a relative date:
// date of birth: timestamp for 16 August, 1983
$dob = mktime(0, 0, 0, 16, 8, 1982);

// Now check that the individual is over 18
if ((float)$dob < (float)strtotime("-18 years"))
  echo "Legal to drive in the state of Victoria";
```

9.19 – Trabalhando com URLs no PHP

Passando Parâmetros pela URL

Primeiro (âncora)

Link

```
arquivo.php
$par1=$_GET['parametro1'];
$par2=$_GET['parametro2'];
$par3=$_GET['parametro3'];
```

Segundo (action de form)

```
<form name=frm method=post action="arquivo2.php?
parametro1=valor1&parametro2=valor2">
...
```

arquivo2.php

```
$par1=$_POST['parametro1'];
$par2=$_POST['parametro1'];
```

Terceiro (URL)

http://localhost/teste.php?parametro1=valor1

teste.php

$par1=$_GET['parametro1'];

Quarto (location no javascript)

```
<?php
// Já vindo de outro script, chamado via POST
$a = $_POST['a'];

?>
<script>
if(confirm("Confirma?")){
        location="vai.php?a='<?=$a?>' ";
}else{
        location='volta.php';
}
</script>
```

Reconstruct URL string in PHP

```
// find out the domain:
$domain = $_SERVER['HTTP_HOST'];
// find out the path to the current file:
$path = $_SERVER['SCRIPT_NAME'];
// find out the QueryString:
$queryString = $_SERVER['QUERY_STRING'];
```

```
 // put it all together:
 $url = "http://" . $domain . $path . "?" . $queryString;
 echo "The current URL is: " . $url . "
";

 // An alternative way is to use REQUEST_URI instead of both
 // SCRIPT_NAME and QUERY_STRING, if you don't need them seperate:
 $url2 = "http://" . $domain . $_SERVER['REQUEST_URI'];
 echo "The alternative way: " . $url2;
```

Do site - http://snippets.dzone.com/posts/show/4054

9.20 – Trabalhando com Criptografia no PHP

Classe usando a função crypt do PHP para criptografar e descriptografar uma script.
Fonte: http://phpro.org

```
<?php

// make it of break it
error_reporting(E_ALL);

/**
* Class to provide 2 way encryption of data
*
* @author    Kevin Waterson
* @copyright    2009    PHPRO.ORG
*
*/
class proCrypt
{
    /**
    *
    * This is called when we wish to set a variable
    *
    * @access    public
    * @param    string    $name
    * @param    string    $value
    *
    */
    public function __set( $name, $value )
    {
        switch( $name)
        {
            case 'key':
            case 'ivs':
            case 'iv':
            $this->$name = $value;
            break;

            default:
            throw new Exception( "$name cannot be set" );
        }
    }

    /**
    *
    * Gettor - This is called when an non existant variable is cal
led
```

```
    *
    * @access    public
    * @param    string    $name
    *
    */
    public function __get( $name )
    {
        switch( $name )
        {
            case 'key':
            return 'keee';

            case 'ivs':
            return mcrypt_get_iv_size( MCRYPT_RIJNDAEL_128, MCRYPT
_MODE_ECB );

            case 'iv':
            return mcrypt_create_iv( $this->ivs );

            default:
            throw new Exception( "$name cannot be called" );
        }
    }

    /**
    *
    * Encrypt a string
    *
    * @access    public
    * @param    string    $text
    * @return    string    The encrypted string
    *
    */
    public function encrypt( $text )
    {
        // add end of text delimiter
        $data = mcrypt_encrypt( MCRYPT_RIJNDAEL_128, $this->key, $
text, MCRYPT_MODE_ECB, $this->iv );
        return base64_encode( $data );
    }

    /**
    *
    * Decrypt a string
    *
    * @access    public
    * @param    string    $text
    * @return    string    The decrypted string
```

```
    *
    */
    public function decrypt( $text )
    {
        $text = base64_decode( $text );
        return mcrypt_decrypt( MCRYPT_RIJNDAEL_128, $this->key, $t
ext, MCRYPT_MODE_ECB, $this->iv );
    }
} // end of class
```

Exemplo de Uso

```
<?php

// a new proCrypt instance
$crypt = new proCrypt;

// encrypt the string
$encoded = $crypt->encrypt( 'my message');
echo $encoded."\n";

// decrypt the string
echo $crypt->decrypt( $encoded ) . "\n";

?>
```

Encode e Decode

PHP/MySQL - Função ()encode e ()decode

Fala galera! Nesse artigo estarei dando continuação ao assunto criptografia em PHP. Como vocês puderam observar na semana passada, eu expliquei a função password(), que tem por finalidade fazer a criptografia de senhas.

Uma desvantagem do uso dessa função, é que não podemos descriptografar o dado depois. A seguir, vou apresentar um outro método de criptografia que é o encode (codifica o dado) e o decode (decodifica o dado).

Exemplo prático

1. Vamos criar uma tabela (mensagem.php):

CREATE TABLE tb_mensagem (
id int(3) NOT NULL auto_increment,
de varchar(80) NOT NULL DEFAULT '' ,
para varchar(80) NOT NULL DEFAULT '' ,
mensagem text NOT NULL DEFAULT '' ,

```
PRIMARY KEY (id)
);
```

2. "Popular" a nossa base de dados:

```
INSERT INTO tb_mensagem(de,para,mensagem)
VALUES('Júlio César Martini','iMasters',
encode('Mensagem sigilosa: Artigo sobre criptografia','teste'))
```

Resultado:

Id | De | Para | Mensagem
1 | Júlio César Martini | iMasters |$P²Ý5¾î_¯¹“ÁøJ®nzOAHG²³¼8¿ ê6êEòYÉ%O,{˜

Depois de ter efetuado este comando SQL, o resultado que você tem na sua tabela (tb_mensagem) é o apresentado acima.

Como você pode observar, o campo mensagem é o que recebeu a função encode(). Veja o monte de caracteres esquisitos que apareceram no lugar da mensagem original.

Agora vocês vão me perguntar, como eu faço para ele mostrar a mensagem certa? Para isso, vamos fazer uso da função decode().

3. Arquivo que conecta com a nossa base de dados MySQL:

```
<?
/* Conecta com um banco de dados MySQL conforme parâmetros enviados (servidor = localhost)
Banco de Dados: $dbname
Porta: $usuario
Senha: $senha*/

$dbname="nomedobd";
$usuario="";
$password="";

//1º passo - Conecta ao servidor MySQL
if(!($id = mysql_connect("localhost",$usuario,$password))) {
echo "<p align="center"><big><strong>Não foi possível estabelecer
uma conexão com o gerenciador MySQL. Favor Contactar o Administrador.
</strong></big></p>";
exit;
}

//2º passo - Seleciona o Banco de Dados
if(!($con=mysql_select_db($dbname,$id))) {
echo " <p align="center"><big><strong>Não foi possível estabelecer
uma conexão com o gerenciador MySQL. Favor Contactar o Administrador.
</strong></big></p>";
```

```
exit;
}
?>
```

4. Vamos criar uma página index.php que vai exibir a mensagem na tela:

```
<?
include "conecta.php"; //Arquivo que conecta com a nossa base de dados MySQL

$sql = mysql_query("SELECT de,para,decode(mensagem,'teste') FROM tb_mensagem") or
die("ERRO no SQL : ".mysql_error());
$array = mysql_fetch_array($sql);

echo $array['de']; echo "<br>";
echo $array['para']; echo "<br><br>";
echo $array['mensagem'];

?>
```

Na linha do SELECT, fazemos uso da função decode(), que vai decodificar aquela mensagem para que ela possa ser exibida corretamente na tela para o usuário.

Como vocês podem observar nessa linha: decode(mensagem,'teste'), passamos a chave para decodificar a mensagem. Nesse caso é teste.

Lembre-se que na hora de fazer o encode(), definimos teste como sendo a chave. Então, para decodificar, precisamos informar ela novamente.

Convido o leitor a tirar o decode() do SQL e ver o resultado.

A linha vai ficar assim:

```
<?
...
$sql = mysql_query("SELECT de,para,mensagem) FROM tb_mensagem") or die("ERRO no SQL :
".mysql_error());
...
?>
```

Não deixe de nos enviar críticas ou sugestões para o próximo assunto, afinal a coluna é de vocês.

Autor/fonte: Júlio César Martini
E-mail/Url: n/a
Site: http://www.htmlstaff.org

Encode e Decode usando base64

$str = 'Ribamar Ferreira de Sousa';
//echo base64_encode($str);

$str2 = 'UmliYW1hciBGZXJyZWlyYSBkZSBTb3VzYQ== ';
echo base64_decode($str2);

Encode-Decode in PHP

```php
<?php

/* Tutorial by AwesomePHP.com -> www.AwesomePHP.com */
/* Function: Encode or Decode anything based on a string */

$secretPass = 'kljhflk73#OO#*U$O(*YO';
$encodeThis = 'Please meet me at 05:44 time.';

/* Regular Encoding */
$encoded = Encode($encodeThis,$secretPass);
/* Another pass to decode */
$decoded = Encode($encoded,$secretPass);

echo 'Encoded String: '.$encoded;
echo '<br />Decoded String: '.$decoded;

/* Important: If passing this value via URL you might want to make it
explorer friendler */
$encoded = bin2hex(Encode($encodeThis,$secretPass));
/* Another pass to decode */
$decoded = Encode(hex2bin($encoded),$secretPass);

echo '<br /><br />Encoded String: '.$encoded;
echo '<br />Decoded String: '.$decoded;

function Encode($data,$pwd)
{
    $pwd_length = strlen($pwd);
    for ($i = 0; $i < 255; $i++) {
        $key[$i] = ord(substr($pwd, ($i % $pwd_length)+1, 1));
        $counter[$i] = $i;
    }
    for ($i = 0; $i < 255; $i++) {
        $x = ($x + $counter[$i] + $key[$i]) % 256;
        $temp_swap = $counter[$i];
        $counter[$i] = $counter[$x];
        $counter[$x] = $temp_swap;
    }
    for ($i = 0; $i < strlen($data); $i++) {
        $a = ($a + 1) % 256;
        $j = ($j + $counter[$a]) % 256;
        $temp = $counter[$a];
```

```
        $counter[$a] = $counter[$j];
        $counter[$j] = $temp;
        $k = $counter[(($counter[$a] + $counter[$j]) % 256)];
        $Zcipher = ord(substr($data, $i, 1)) ^ $k;
        $Zcrypt .= chr($Zcipher);
    }
    return $Zcrypt;
}

function hex2bin($hexdata) {
    for ($i=0;$i<strlen($hexdata);$i+=2) {
        $bindata.=chr(hexdec(substr($hexdata,$i,2)));
    }
    return $bindata;
}
?>
```

Site de origem: http://www.awesomephp.com/?Tutorials*3/Encode-Decode-in-PHP.html

```
<?php
Encriptar
$cod = base64_encode("senha");

print $cod;

Decriptar
echo "<br>".base64_decode($cod);
```

```
        /**
         * Decodes a text with many algorithms many times
         *
         * @param   string   $text    the text to decode
         * @param   string   $mode the sequence of decryption methods separated by commas [,]
         *                   G      : gEncryption method using the class gEncrypter
         *                   base64 : base64 decoding algorithm
         *                   uu     : decryption using the function uudecode()
         *                [default: G]
         * @param   string   $key     the key to use separated by commas [,]
         *                one for each "G" you put in the sequence
         *                [gEncrypter only] [default: ilarvet]
         * @return   string         the text finally decrypted
         */
        function decode($text, $mode = "G", $key = "ilarvet")
        {
           $exmode = explode(",", $mode);
           $exkey = explode(",", $key);

           $kcount = 0;
           $dec = $text;

           for ($i=0; $i<count($exmode); $i++)
```

```
    {
        $exmode[$i] = trim($exmode[$i]);

        switch (strtolower($exmode[$i]))
        {
            case "g":
                include_once($this->include_path . "gEncrypter.php");
                $e = new gEncrypter;
                $dec = $e->gED($dec, $exkey[$kcount]);
                $kcount ++;
                break;

            case "base64":
                $dec = base64_decode($dec);
                break;

            case "uu":
                $dec = convert_uudecode($dec);
                break;

            default:
                break;
        }
    }

    return $dec;
}

function encode($text, $mode = "G", $key = "ilarvet")
{
    $exmode = explode(",", $mode);
    $exkey = explode(",", $key);

    $kcount = 0;
    $enc = $text;

    for ($i=0; $i<count($exmode); $i++)
    {
        $exmode[$i] = trim($exmode[$i]);

        switch (strtolower($exmode[$i]))
        {
            case "g":
                include_once($this->include_path . "gEncrypter.php");
                $e = new gEncrypter;
                $enc = $e->gED($enc, $exkey[$kcount]);
                $kcount ++;
                break;
```

```
                case "base64":
                    $enc = base64_encode($enc);
                    break;

                case "uu":
                    $enc = convert_uuencode($enc);
                    break;

                default:
                    break;
            }
        }
    }

?>
```

9.21 – Trabalhando com e-mails em PHP

Enviar e-mail simples:
mail("cursos@ribafs.org", "Aqui fica o assunto", "Aqui o corpo do e-mail");

Envio de e-mail com tratamento de erro:
$to = cursos@ribafs.org';
$subject = 'Assunto';
$from = 'elias@ribafs.org';
$message = 'Como vai?';

if(mail($to, $subject, $message, "From: $from"))
  echo "E-mail enviado com sucesso!";
else
  echo "Falha no envio do e-mail – mensagem não enviada"; ?>

Mais um:
$subject = "Assunto";
	$content = "Text type email does not need br tags for line breaks;

A simple line break in your code will do the trick, as the Content-type is set to text in the header.";

	$headers = 'MIME-Version: 1.0' . "\n";
	$headers .= 'Content-type: text; charset=iso-8859-1' . "\n";

	$headers .= 'From: info@domain.com';
	mail($to,$subject,$content,$headers);

Mais um exemplo de envio de e-mail com PHP:
$headers = "From: Testando Envio de Email <$assunto>";

// pegando data
$date = date("d/m/Y h:i");

// teoricamente iremos enviar para esse email fictício, mas podemos trazer de um BD o email sem algum problema.
$seuemail = "joao@testando.com.br";

// assunto do email, como vai ficar em sua caixa de entrada, cuidado para não colocar nomes inseridas em blacklists de email.
$assunto = "Sistema de envio de Email";

// Corpo do texto, fiz um exemplo básico de cadastro trazendo em variavel.
$mensagem = "
		Nome: $variavel1
		Endereco: $variavel2
		Bairro: $variavel3

```
            Telefone: $variavel4
    Email: $variavel5

    Enviado em: $date";


// eis aqui o envio própriamente dito.... essa linha é onde se dispara o email.
mail($seuemail, $assunto, $mensagem, $headers);
```

Envio de E-mail no formato HTML

```
<?php
//define the receiver of the email
$to = 'youraddress@example.com';
//define the subject of the email
$subject = 'Test HTML email';
//create a boundary string. It must be unique
//so we use the MD5 algorithm to generate a random hash
$random_hash = md5(date('r', time()));
//define the headers we want passed. Note that they are separated with \r\n
$headers = "From: webmaster@example.com\r\nReply-To: webmaster@example.com";
//add boundary string and mime type specification
$headers .= "\r\nContent-Type: multipart/alternative; boundary=\"PHP-alt-".$random_hash."\"";
//define the body of the message.
ob_start(); //Turn on output buffering
?>
--PHP-alt-<?php echo $random_hash; ?>
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

Hello World!!!
This is simple text email message.

--PHP-alt-<?php echo $random_hash; ?>
Content-Type: text/html; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

<h2>Hello World!</h2>
<p>This is something with <b>HTML</b> formatting.</p>

--PHP-alt-<?php echo $random_hash; ?>--
<?php
//copy current buffer contents into $message variable and delete current output buffer
$message = ob_get_clean();
//send the email
$mail_sent = @mail( $to, $subject, $message, $headers );
//if the message is sent successfully print "Mail sent". Otherwise print "Mail failed"
echo $mail_sent ? "Mail sent" : "Mail failed";
```

```
?>
```

Envio de e-mail com anexo:

```
<?php
//define the receiver of the email
$to = 'youraddress@example.com';
//define the subject of the email
$subject = 'Test email with attachment';
//create a boundary string. It must be unique
//so we use the MD5 algorithm to generate a random hash
$random_hash = md5(date('r', time()));
//define the headers we want passed. Note that they are separated with \r\n
$headers = "From: webmaster@example.com\r\nReply-To: webmaster@example.com";
//add boundary string and mime type specification
$headers .= "\r\nContent-Type: multipart/mixed; boundary=\"PHP-mixed-".$random_hash."\"";
//read the atachment file contents into a string,
//encode it with MIME base64,
//and split it into smaller chunks
$attachment = chunk_split(base64_encode(file_get_contents('attachment.zip')));
//define the body of the message.
ob_start(); //Turn on output buffering
?>
--PHP-mixed-<?php echo $random_hash; ?>
Content-Type: multipart/alternative; boundary="PHP-alt-<?php echo $random_hash; ?>"

--PHP-alt-<?php echo $random_hash; ?>
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

Hello World!!!
This is simple text email message.

--PHP-alt-<?php echo $random_hash; ?>
Content-Type: text/html; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

<h2>Hello World!</h2>
<p>This is something with <b>HTML</b> formatting.</p>

--PHP-alt-<?php echo $random_hash; ?>--

--PHP-mixed-<?php echo $random_hash; ?>
Content-Type: application/zip; name="attachment.zip"
Content-Transfer-Encoding: base64
Content-Disposition: attachment

<?php echo $attachment; ?>
--PHP-mixed-<?php echo $random_hash; ?>--
```

```
<?php
//copy current buffer contents into $message variable and delete current output buffer
$message = ob_get_clean();
//send the email
$mail_sent = @mail( $to, $subject, $message, $headers );
//if the message is sent successfully print "Mail sent". Otherwise print "Mail failed"
echo $mail_sent ? "Mail sent" : "Mail failed";
?>
```

9.22 – Trabalhando com Data e Hora em PHP

```
<?php
// Funções diversas

// Definir o horário brasileiro:
//date_default_timezone_set('Brazil/East');


// Somar dias a data atual
function add_dias($dias)
{
    return date('d/m/Y',mktime(0,0,0,date('m'),date('d')+$dias,date('Y')));
}

   /*
      Função SomarDias()
      Usada para calcular a data de feriados móveis.
      Esta função adiciona 'n_dias' à data 'data', passado como argumento, a qual deve estar no formato dd/mm/yyyy ou yyyy-mm-dd.
      O argumento 'forma' serve para especificar o formato da data retornada. Ele pode conter os seguintes valores:

      "pt" => Retornará a data no formato DD/MM/YYYY
      "en" => Retornará a data no formato YYYY-MM-DD

      Se 'forma' não for especificada, adotar-se-á a forma brasileira (pt).
   */
   public function SomarDias ($data, $n_dias, $forma = "pt")
   {
      if (!is_int ($n_dias))
      {
         echo "<p>Função <strong>". __FUNCTION__ ."</strong>: o argumento \"n_dias\" deve ser um número inteiro.</p>";
         return false;
      }

      $forma = strtolower ($forma);
      if ($forma != "en" AND $forma != "pt")
         $forma = "pt";

      if (preg_match ("/^[0-9]{2}\/[0-9]{2}\/[0-9]{4}$/", $data))
         list ($dia, $mês, $ano) = explode ("/", $data);
      elseif (preg_match ("/^[0-9]{4}-[0-9]{2}-[0-9]{2}$/", $data))
         list ($ano, $mês, $dia) = explode ("-", $data);
      else
      {
         echo "<p>Função <strong>". __FUNCTION__ ."</strong>: Formato de data inválido (".
```

```
$data .").</p>";
            return false;
        }

        //transforma $n_dias em segundos
        //86400 = 60 * 60 * 24
        $segs_n_dias = $n_dias * 86400;

        // tranforma $data em timestamp
        $segs_data = strtotime ($ano . "-" . $mês . "-" . $dia);
        $segs_nova_data = $segs_data + $segs_n_dias;
        $nova_data = ($forma == "pt") ? date("d/m/Y", $segs_nova_data) : date("Y-m-d",
$segs_nova_data);

        return $nova_data;
    }



    /*
      Função SubtrairDias()
      Usada para calcular a data de feriados móveis.
      Esta função subtrai 'n_dias' da data 'data', passado como argumento, a qual deve estar no
formato dd/mm/yyyy ou yyyy-mm-dd.
      O argumento 'forma' serve para especificar o formato da data retornada. Ele pode conter os
seguintes valores:

      "pt" => Retornará a data no formato DD/MM/YYYY
      "en" => Retornará a data no formato YYYY-MM-DD

      Se 'forma' não for especificada, adotar-se-á a forma brasileira (pt).
    */
    public function SubtrairDias ($data, $n_dias, $forma = "pt")
    {
        if (!is_int ($n_dias))
        {
            echo "<p>Função <strong>". __FUNCTION__ ."</strong>: O argumento \"n_dias\" deve
ser um número inteiro.</p>";
            return false;
        }

        $forma = strtolower ($forma);
        if ($forma != "en" AND $forma != "pt")
            $forma = "pt";

        if (preg_match ("/^[0-9]{2}\/[0-9]{2}\/[0-9]{4}$/", $data))
            list ($dia, $mês, $ano) = explode ("/", $data);
        elseif (preg_match ("/^[0-9]{4}-[0-9]{2}-[0-9]{2}$/", $data))
```

```
            list ($ano, $mês, $dia) = explode ("-", $data);
        else
        {
            echo "<p>Função <strong>". __FUNCTION__ ."</strong>: Formato de data inválido (".
$data .").</p>";
            return false;
        }

        //transforma $n_dias em segundos
        //86400 = 60 * 60 * 24
        $segs_n_dias = $n_dias * 86400;

        // tranforma $data em timestamp
        $segs_data = strtotime ($ano . "-" . $mês . "-" . $dia);
        $segs_nova_data = $segs_data - $segs_n_dias;
        $nova_data = ($forma == "pt") ? date("d/m/Y", $segs_nova_data) : date("Y-m-d",
$segs_nova_data);

        return $nova_data;
    }

// Somar meses
function add_meses($meses)
{
    return date('d/m/Y',mktime(0,0,0,date('m')+$meses,date('d'),date('Y')));
}

// Somar anos
function add_anos($anos)
{
    return date('d/m/Y',mktime(0,0,0,date('m'),date('d'),date('Y')+$anos));
}

function somar_dias_uteis($str_data,$int_qtd_dias_somar = 7) {
    // Caso seja informado uma data do MySQL ou PostgreSQL do tipo DATETIME - aaaa-mm-dd
00:00:00
    // Transforma para DATE - aaaa-mm-dd

    $str_data = substr($str_data,0,10);
    // Se a data estiver no formato brasileiro: dd/mm/aaaa
    // Converte-a para o padrão americano: aaaa-mm-dd
    if ( preg_match("@/@",$str_data) == 1 ) {
        $str_data = implode("-", array_reverse(explode("/",$str_data)));
    }
    $array_data = explode('-', $str_data);
    $count_days = 0;
    $int_qtd_dias_uteis = 0;
    while ( $int_qtd_dias_uteis < $int_qtd_dias_somar ) {
```

```
        $count_days++;
            if ( ( $dias_da_semana = gmdate('w', strtotime('+'.$count_days.' day', mktime(0, 0, 0,
$array_data[1], $array_data[2], $array_data[0]))) ) != '0' && $dias_da_semana != '6' ) {
            $int_qtd_dias_uteis++;
        }
    }
    return gmdate('d/m/Y',strtotime('+'.$count_days.' day',strtotime($str_data)));
}

//Exemplo de uso:
//echo somar_dias_uteis('05/12/2006');
//echo somar_dias_uteis('2006-12-01',15);
// http://leandrovieira.com/archive/somando-dias-uteis-a-uma-data-especifica-com-php


function SomarData($data, $dias, $meses, $ano)
{
  //passe a data no formato dd/mm/yyyy
   $data = explode('/', $data);
   $newData = date('d/m/Y', mktime(0, 0, 0, $data[1] + $meses,
   $data[0] + $dias, $data[2] + $ano) );
   return $newData;
}

//Exemplo de como usar:
//echo SomarData('04/04/2007', 1, 2, 1);

//Este exemplo acima estamos adicionando 1 dia, 2 meses e 1 ano na data informada. O resultado
então seria '05/06/2008'
//http://www.codigofonte.com.br/codigo/php/data-hora/funcao-para-somar-datas-em-php


/*
Funções sub_data() e som_data()
Desenvolvidas por
InFog (Evaldo Junior Bento)
em Junho de 2007
junior_pd_bento@yahoo.com.br
Este script é disponibilizado utilizando
a licença GPL em sua versão mais atual.
Distribua, aprenda, ensine
mas mantenha os créditos do autor
Viva ao Software Livre e à livre informação
*/

/*
Função sub_data()
Esta função recebe a data no formato brasileiro dd/mm/AAAA
```

```
e o número de dias que serão subtraídos dela.
Certifique-se de checar se a data é válida antes de chamar a função
*/

function sub_data($data, $dias) {
    $data_e = explode("/",$data);
    $data2 = date("m/d/Y", mktime(0,0,0,$data_e[1],$data_e[0] - $dias,$data_e[2]));
    $data2_e = explode("/",$data2);
    $data_final = $data2_e[1] . "/". $data2_e[0] . "/" . $data2_e[2];
    return $data_final;
}

/*
Função soma_data()
Esta função recebe a data no formato brasileiro dd/mm/AAAA
e o número de dias que serão adicionados à dela.
Certifique-se de checar se a data é válida antes de chamar a função
*/

function soma_data($data, $dias) {
    $data_e = explode("/",$data);
    $data2 = date("m/d/Y", mktime(0,0,0,$data_e[1],$data_e[0] + $dias,$data_e[2]));
    $data2_e = explode("/",$data2);
    $data_final = $data2_e[1] . "/". $data2_e[0] . "/" . $data2_e[2];
    return $data_final;
}


//Autor/fonte: Evaldo Junior
//E-mail/Url: http://tuxmasters.blogspot.com/2007/06/somando-datas-no-php.html

function hoje(){
    return date("d/m/Y");
}

function agora(){
    return date("H:i:s");
}

function databr_atual(){
    return date("d/m/Y");
}

function hora_atual(){
    return date("H:i:s");
}
```

```
//Verifica se Data1 é menor que Data2, além de validá-las
//Entrada no formato Brasileiro
function ChecaVarData($data1,$data2)
{
 if (DataValida($data1) and DataValida($data1))
 {
  $data1=DataBrasilMySQL($data1);
  $data2=DataBrasilMySQL($data2);
  Return($data1<=$data2);
 }
 else
 {
  Return(false);
 }

}

function data_valida($data){
    $d = explode("/",$data);
    $d2 = checkdate ($d[1],$d[0],$d[2]);
    if (!$d2){
        ?><script>alert("A data "+<?=$d[0].$d[1].$d[2] ?>+" não é válida!")</script><?php
        exit;
    }
}

function idade($dia,$mes,$ano) {
    if($mes >= date(m)) {
        if($dia >=date(d)) {
            $idade = date(Y) - $ano + 1;
        }else{
            $idade = date(Y) - $ano;
        }
    }else{
        $idade = date(Y) - $ano;
    }
    return $idade;
}
//echo idade(03,08,1956);

/*
 *
 *
$arrDia=array ('Domingo','Segunda','Terça','Quarta','Quinta','Sexta','Sábado ');
$arrMes=array (1=>'Janeiro','Fevereiro','Março','Abril','Maio','Junho',
'Julho','Agosto','Setembro','Outubro','Novembro','Dezembro');
?>
<form>
```

```
Selecione a data completa<br>
<select name="semana">
<?php
for($i = 0; $i < 7; $i++){
echo"<option> $arrDia[$i]";
}
echo'</select><select name="dia">';
for ($i=1;$i<=31; $i++){
echo"<option>$i";
}
echo '</select> de <select name="mes">';
for ($i=1;$i<=12;$i++){
echo "<option value=\"$i\"> $arrMes[$i]";
}
echo '</select> de <select name="ano">'; //Faltava ;
$start_year = date('Y') - 10;
$end_year = $start_year + 20;
for($i = $start_year; $i <= $end_year; $i++){
echo "<option> $i";
}

 */

$arrDia=array
('Domingo','Segunda','Terça','Quarta','Quinta','Sexta','Sábado
');
$arrMes=array
(1=>'Janeiro','Fevereiro','Março','Abril','Maio','Junho',
'Julho','Agosto','Setembro','Outubro','Novembro','Dezembro');
?>
<form>
Selecione a data completa<br>
<select name="semana">
<?php
for($i = 0; $i < 7; $i++){
echo"<option> $arrDia[$i]";
}
echo'</select><select name="dia">';
for ($i=1;$i<=31; $i++){
echo"<option>$i";
}
echo '</select> de <select name="mes">';
for ($i=1;$i<=12;$i++){
echo "<option value=\"$i\"> $arrMes[$i]";
}
echo '</select> de <select name="ano">'; //Faltava ;
$start_year = date('Y') - 10;
$end_year = $start_year + 20;
```

```
for($i = $start_year; $i <= $end_year; $i++){
echo "<option> $i";
}


/* Código Original/ Original Code by Woodys
* http://www.zend.com/codex.php?id=176
*
* adaptação/tradução para o português e formato brasileiro das datas
*
* Alguns exclarecimentos
* - O que é TIMESTAMP do Unix?
* - É a contagem, em segundos, desde o dia 1 de janeiro de 1970 00:00:00 GMT
*     ,i.e., os segundos que se passaram até momento desde as ZERO horas do dia 1 de janeiro de
1970
*     exemplo:
*     timestamp = 1042752929  => passaram-se 1042752929 segundos desde 1/jan/1970 00horas
00min 00 seg
*
*  Tradução e Adaptação by Calvin
*/

?>

<?php

/**
* Calcula a quantidade de dias úteis entre duas datas (sem contar feriados)
* @author Marcos Regis
* @param String $datainicial
* @param String $datafinal=null
*/
function dias_uteis($datainicial,$datafinal=null){
  if (!isset($datainicial)) return false;
  if (!isset($datafinal)) $datafinal=time();

$segundos_datainicial = strtotime(preg_replace("#(\d{2})/(\d{2})/(\d{4})#","$3/$2/$1",
$datainicial));
$segundos_datafinal = strtotime(preg_replace("#(\d{2})/(\d{2})/(\d{4})#","$3/$2/$1",$datafinal));
$dias = abs(floor(floor(($segundos_datafinal-$segundos_datainicial)/3600)/24 ) );
$uteis=0;
for($i=1;$i<=$dias;$i++){
$diai = $segundos_datainicial+($i*3600*24);
$w = date('w',$diai);
if ($w==0){
//echo date('d/m/Y',$diai)." é Domingo<br />";
}elseif($w==6){
//echo date('d/m/Y',$diai)." é Sábado<br />";
```

```
}else{
//echo date('d/m/Y',$diai)." é dia útil<br />";
$uteis++;
}
}
return $uteis;
}
?>
```

ex. de uso

```
<?php
$data='28/02/2007';
echo "Existem ".dias_uteis($data)." dias úteis entre $data e hoje";
?>
```

```
<?php

//Impresso através do site http://www.htmlstaff.org

//Função para pegar todos os feriados do ano
```

//Uma função para pegar todos os feriados do ano (fixos e móveis). Coloquei dois estaduais, mas não será problema em editá-los conforme a cidade/estado.

//Função:

```
/**
* @param $ano ano em que se quer calcular os feriados
* @return array com os feriados do ano (fixo e moveis)
*/
function getFeriados($ano){
    $dia = 86400;
    $datas = array();
    $datas['pascoa'] = easter_date($ano);
    $datas['sexta_santa'] = $datas['pascoa'] - (2 * $dia);
    $datas['carnaval'] = $datas['pascoa'] - (47 * $dia);
    $datas['corpus_cristi'] = $datas['pascoa'] + (60 * $dia);
    $feriados = array (
            '01/01', // Confraternização universal
            '02/02', // Navegantes
            date('d/m',$datas['carnaval']),
            date('d/m',$datas['sexta_santa']),
            date('d/m',$datas['pascoa']),
            '21/04', //Tiradentes
            '01/05',
            date('d/m',$datas['corpus_cristi']),
            '20/09', // Revolução Farroupilha m/
            '12/10',
            '02/11',
```

```
                '15/11',
                '25/12',
                );
    return $feriados;
}
$feriados = getFeriados('2008');

for($x=0;$x<12;$x++){
    print $feriados[$x].'<br>';
}

//Autor/fonte: Maiquel Leonel
//E-mail/Url: http://www.vivaolinux.com.br/scripts/verScript.php?codigo=3430
?>
```

Feriados Brasileiros

Feriados nacionais

Os feriados nacionais são definidos pelas seguintes leis: nº 662 (de 1949)[1], nº 6.802 (de 1980)[2], nº 9.093 (de 1995)[3], e nº 10.607 (de 2002)[4]. Os feriados nacionais brasileiros são:

Data	Feriado	Motivação
1º de janeiro	Confraternização Universal	social
21 de abril	Tiradentes	cívica
1º de maio	Dia do Trabalho	social
7 de setembro	Independência do Brasil	cívica
12 de outubro	Nossa Senhora Aparecida	religiosa (católica)
2 de novembro	Finados	religiosa (católica)
15 de novembro	Proclamação da República	cívica
25 de dezembro	Natal	religiosa (cristã)

Feriados móveis

São feriados que dependem da Páscoa. Os judeus celebram a Páscoa segundo o que prescreve o livro do Êxodo, no capítulo 12, no dia 14 do mês de Nissan. É a celebração da libertação da escravidão do Egito para a liberdade da Terra Prometida por Deus a Abraão. O cristianismo celebra a Páscoa cristã, Ressurreição de Cristo, acompanhando de certa forma a data Páscoa judaica. Mas o calendário judeu é baseado na Lua, então a data da Páscoa cristã passou a ser móvel no calendário cristão, assim como as demais datas referentes à Páscoa, tanto na Igreja Católica como nas Igrejas Protestantes e Igrejas Ortodoxas. O primeiro Concilio geral da Igreja, o de Nicéia, no ano 325, determinou que a Páscoa cristã seria celebrada no domingo seguinte à primeira Lua cheia após o equinócio da primavera do hemisfério Norte (21 de março); podendo ocorrer entre 22 de março e 25 de abril.

Outros feriados que dependem da Páscoa:

Carnaval (terça-feira) - quarenta e sete dias antes da Páscoa.

Quaresma - inicia na quarta-feira de cinzas e termina no Domingo de Ramos (uma semana antes da Páscoa).

Sexta-feira Santa - a sexta-feira imediatamente anterior Sábado da Solene Vigília Pascal - o sábado de véspera

Pentecostes - o oitavo domingo após a Páscoa.

Corpus Christi - a quinta-feira imediatamente após o Pentecostes.

Feriados estaduais

A Lei nº 9.093, de 12 de setembro de 1995, incluiu entre os feriados civis, antes apenas os declarados em lei federal, a "data magna do Estado fixada em lei estadual".

Acre

Data	Feriado
23 de janeiro	Dia do evangélico
15 de junho	Aniversário do estado
6 de agosto	Início da Revolução Acreana
5 de setembro	Dia da Amazônia
17 de novembro	Assinatura do Tratado de Petrópolis

Alagoas

Data	Feriado
24 de junho	São João (decreto estadual n.4.087, de 18 de dezembro de 2008)
29 de junho	São Pedro (decreto estadual 4.087, de 18 de dezembro de 2008)
16 de setembro	Emancipação política

20 de novembro Dia da Consciência Negra

Amapá
Data Feriado
19 de março Dia de São José
5 de outubro Criação do estado
20 de novembro Dia da Consciência Negra

Amazonas
Data Feriado
5 de setembro Elevação do Amazonas à categoria de província
20 de novembro Dia da Consciência negra
8 de dezembro Dia de Nossa Senhora da Conceição

Bahia
Data Feriado
2 de julho Independência da Bahia
20 de novembro Dia da Consciência Negra

Ceará

O Ceará não tem data magna. Em 2007, o deputado estadual Lula Morais apresentou o Projeto de Lei nº 53, para definir o dia 19 de março como Data Magna do Estado do Ceará. O relator do projeto na Comissão de Constituição e Justiça da Assembleia Legislativa, deputado João Jaime, devolveu o projeto ao autor, com a sugestão de transformá-lo em projeto de indicação. Em junho de 2007, o projeto de lei foi retirado pelo autor e arquivado. Desde então, ninguém mais apresentou projeto que fixasse a data magna estadual cearense.

Distrito Federal
Data Feriado
21 de abril Fundação de Brasília
30 de novembro Dia do Evangélico

Espírito Santo
Data Feriado
23 de maio Colonização do solo espírito-santense
28 de outubro Dia do Servidor Público

Goiás
Data Feriado
28 de outubro Dia do Servidor Público

Maranhão
Data Feriado
28 de julho Adesão do Maranhão à independência do Brasil
8 de dezembro Dia de Nossa Senhora da Conceição

Mato Grosso
Data Feriado
20 de novembro Dia da Consciência Negra

Mato Grosso do Sul
Data Feriado
11 de outubro Criação do estado

Minas Gerais
Data Feriado
21 de abril Tiradentes

Pará
Data Feriado
15 de agosto Adesão do Grão-Pará à independência do Brasil
8 de dezembro Nossa Senhora da Conceição

Paraíba
Data Feriado
5 de agosto Emancipação política do estado

Paraná
Data Feriado
19 de dezembro Emancipação política (emancipação do Paraná de São Paulo)

Pernambuco
Data Feriado Legislação Observações
6 de março Revolução Pernambucana de 1817 Lei nº 13.386[1], de 24 de dezembro de 2007 ponto facultativo

Piauí
Data Feriado
13 de março Dia da Batalha do Jenipapo
19 de outubro Dia de Piauí

Rio de Janeiro
Data Feriado
23 de abril Dia de São Jorge
15 de outubro Dia do Comércio
20 de novembro Dia da Consciência Negra

Rio Grande do Norte
Data Feriado
29 de junho Dia de São Pedro
3 de outubro Mártires de Cunhaú e Uruaçu

Rio Grande do Sul
Data Feriado

20 de setembro Revolução Farroupilha

Rondônia
Data Feriado
4 de janeiro Criação do estado
18 de junho Dia do Evangélico

Roraima
Data Feriado
5 de outubro Criação do estado

Santa Catarina
Data Feriado
11 de agosto Criação da capitania, separando-se de São Paulo

São Paulo
Data Feriado
25 de janeiro Dia de São Paulo
9 de julho Revolução Constitucionalista de 1932
20 de novembro Dia da Consciência Negra
12 de outubro Nossa Senhora da Conceição Aparecida

Sergipe
Data Feriado
8 de julho Autonomia política de Sergipe

Tocantins
Data Feriado
5 de outubro Criação do estado

Feriados municipais

Os municípios podem declarar, em lei municipal, até quatro feriados religiosos, de acordo com a tradição local, entre eles a Sexta-Feira da Paixão. A Lei nº 9.335, de 10 de dezembro de 1996[7], acrescentou, ainda, como feriado civil, os dias do início e do término do ano do centenário de fundação do município, desde que fixado em lei municipal.

Fonte: http://pt.wikipedia.org/wiki/Feriados_no_Brasil

Referências

http://php.net
http://pt.wikibooks.org/wiki/Aplicativos_em_PHP