PHP - Módulo 2: O primeiro script

Por Valdir Dias

Agora que você já conheceu um pouco mais do PHP, vamos conhecer mais detalhes da linguagem. Neste módulo você aprenderá como tratar string e números no PHP e no final estará apto a criar um script de envio de informações de um formulário por e-mail.

Para aprender uma nova linguagem, costumamos verificar os pontos que listo abaixo:

1) Comentários
Todo programa deve possuir comentários, visando o entendimento do código em consultas posteriores.
No PHP, existem três tipos de marcadores de comentário, que são:
// e # para comentário de uma linha. Por exemplo:

```
// atribui o nome à variável
$nome = "Valdir Leite"

$email = "valdirleite@ig.com.br" # atribui o E-mail à variável
```

e para comentários que ocupem mais de uma linha, usamos os marcadores /* */.

```
/*
Nas linhas abaixo, atribuiremos os valores
Do nome e do e-mail às respectivas variáveis
*/
$nome = "Valdir Leite"
$email = "valdirleite@ig.com.br"
```

2) Variáveis
Para começar, vamos ver como o PHP trata suas variáveis (ou constantes), que podem ser variáveis escalares ou não-escalares. As variáveis escalares são aquelas que podem ser retrabalhadas, ou "divididas em pedaços menores", enquanto as não escalares são as arrays (matrizes) e os objetos.

A identificação de uma variável, independente do seu tipo é pelo sinal $ colocado como primeiro caractere, como abaixo:

```
$nome = "Valdir Leite";
$matricula = 9843825
$conex = ocilogon("scott", "tiger", "ORA8");
```

A primeira variável é do tipo string; a segunda, inteiro (ambas escalares); e a terceira é uma variável que possui a referência de um objeto de conexão com um banco de dados Oracle.

Vale lembrar que, como a linguagem C, as variáveis $nome e $Nome são consideradas diferentes, pois o PHP as trata como sensíveis ao caso.

Na versão 4 do PHP, as variáveis podem receber valor por referência. Isto significa que ao para atribuir o valor a uma variável não usamos um valor, mas um "ponteiro" para o valor em questão. Na verdade, este "ponteiro" é uma outra variável:

```
$nome = "Valdir";
$identificacao = &$nome;
```

Deste modo, a variável $identificacao recebe o valor de $nome e, se uma das duas for atualizada, a outra também será, mantendo o mesmo valor em ambas. Mas não vamos aprofundar nisto, pois é ciosa de versão 4.0, que será tratada posteriormente.

3) Operações Matemáticas
As operações no PHP também seguem o padrão das outras linguagens (+, -, *, /, sin(), cos()). Além destas, o PHP tem um completo conjunto de operações matemáticas, que podem ser consultadas nesta página: (http://br.php.net/manual/ref.math.php3)

Um exemplo para calcular o valor líquido de um preço, depois de aplicar 10% de desconto sobre o preço bruto:

```
$valorbruto = 10;
$desconto = 10 * $valorbruto / 100;
$valorliquido = $valorbruto - $desconto;
```

4) Operações com strings
Operações com strings são uma das características mais desenvolvidas do PHP. Entre as mais importantes estão:
strlen(), que permite saber quantos caracteres possui a string:

```
echo "A palavra 'internet' possui " . strlen("internet") . " caracteres ";
```

substr(), que devolve uma substring da string informada:

```
echo substr("abcde", 2 , 2); // Esta linha irá exibir os caracteres "cd";
```

strpos (), para saber se determinado caractere (ou substring) está contida em uma string:

```
if strpos ($email, "@") {
  echo("Seu e-mail parece estar correto!\n");
} else {
  echo("O e-mail está inválido\n");
}
```

No exemplo acima, verificamos se o caractere "@" está contida em uma variável $email. Se estiver, exibe a primeira mensagem. Do contrário, exibe a segunda.

Outras funções relacionadas à operações com strings podem ser encontradas em:
http://br.php.net/manual/ref.strings.html

5) Controlando o fluxo e LOOPS
As funções usadas para controlar o fluxo do programa e execução de "loops" são:

if ... else ... else if, que segue o padrão da linguagem C:

```
if ($sexo == "m") {
    echo "Você é do sexo Masculino\n";
} elseif ($sexo == "f") {
    echo "Você é do sexo Feminino\n";
} else {
    echo "Por favor, informe corretamente seu sexo\n";
}
```

switch, uma maneira de controlar o fluxo onde a variável de controle do fluxo pode ter várias opções de valores. Este tipo de controle poderia ser feito com uma seqüência de "ifs" e "elseifs", mas o uso do switch torna o código mais legível e faz com que seja executado mais rapidamente, pois a verificação da variável "$sexo" só é feita uma vez e depois comparada com as opções de cada "case". Se não estiver em nenhuma delas, é executado o bloco sob o "default". Já com o "elseif", a comparação é feita novamente a cada sentença. Neste exemplo, a diferença não é tão grande, mas quando o tipo de verificação vai ficando mais complexo a velocidade começa a ser sentida. Na maioria dos casos, vale a pena optar pelo switch.

```
switch ($sexo) {
    case "m":
      echo "Você é do sexo Masculino\n";
      break;
    case "f"
      echo "Você é do sexo Feminino\n";
      break;
    case default:
echo "Por favor, informe corretamente seu sexo\n";
break;
}
```

Sempre inclua o comando break no final do case. Caso contrário, a execução continuará até encontrar o final do switch (ou a instrução break), fazendo com que as instruções de mais de um case sejam executadas.

while, que permite repetir o código enquanto uma condição for verdadeira:

```
while ($contador > 0) {
    $contador = $contador - 2;
}
```

for, para execução de um loop determinada quantidade de vezes:

```
for ($i==0; $i<100; $i++) {
    echo "$i\n";
}
```

Tratando formulários
Com as informações que você já possui, podemos passar para exemplos práticos, úteis no dia-a-dia de um webmaster.

Vamos fazer, passo-a-passo, um script para receber os dados de um formulário, consistir as informações e enviar o resultado por e-mail. Este formulário possui campos para digitação do nome, e-mail e telefone. Todos os campos são obrigatórios e a consistência do campo e-mail deve ser feita apenas verificando a existência do caractere @, para facilitar as coisas. Já o campo telefone deve ter, sete oi oito caracteres. Tendo este cenário, mãos a obra!

Vamos ao programa:

```
<?php

$erro = "";

if ($nome == "") {
    $erro .= "Digite seu Nome\n"; }

if ((strlen($telefone) > 8) or (strlen($telefone) < 7)) {
    $erro .= "O número do telefone deve ter sete ou oito caracteres\n";
}

if strpos ($email, "@") = 0 {
    $erro .= "O e-mail digitado não é válido\n"
}
```

Esta primeira parte faz a consistência dos dados e altera o valor da variável $erro, caso alguma das condições não seja satisfeita. Para prosseguir, devemos verificar a ocorrência de erros e então enviar o e-mail se erros não tiverem ocorrido ou enviar uma tela de resposta informando qual o erro aconteceu. Como o valor de $erro antes da verificação dos campos é "", basta testar se a variável ainda tem este valor para saber se aconteceu ou não um erro. Vamos continuar:

```
echo("<html><title>Envie o formulário
abaixo</title><body><center>\n"); # Cabeçalho de resposta.

If ($erro == "") { // Não houve nenhum erro no preenchimento do formulário
mail("valdirleite@ig.com.br", "Dados do Formulário"," Nome: $nome\n E-mail: $email\n Telefone: $telefone\n","From: $email\nDate:
$date\n";
echo("Obrigado por enviar este formulário!\n");
} else
echo("Não foi possível enviar o formulário!<br>Verifique as mensagens abaixo<br><br><b> $erro \n");
}
echo("</center></body></html>\n");
```

Pronto!

As novidades neste script são:
· O comando mail() que é a função nativa do PHP para envio de e-mail. Sua sintaxe é a seguinte: mail(Destinatário, Assunto, Mensagem, Informações_Adicionais);
· O echo() que é o equivalente do print no PERL e do response.write do ASP;

Veja no próximo módulo
No próximo tutorial veremos outras funções de tratamento de strings e como trabalhar com banco de dados. Até lá.