PHP

Aula 4

Tópicos

1 Sintaxe Básica

1.1 Identificando o código em PHP

O PHP tem uma sintaxe muito simples e enxuta. O código PHP fica embutido no próprio HTML. O interpretador identifica quando um código é PHP pelas seguintes tags:

Tag padrão	Short tag	Script tag	ASP tag
<?php comandos ?>	<script language="php"> comandos </script>	<? comandos ?>	<% comandos %>

A tag padrão é a recomendada, pois funcionará em qualquer ambiente/servidor. Já as outras tags precisam ser configuradas no PHP.INI.

O uso das short tags (que já foram o padrão do PHP) é desencorajado pois pode conflitar com arquivos XML.

1.2 Separador de instruções

Para cada fim de linha de código tem que haver um ponto e vírgula, indicando ao sistema fim de instrução. Exemplo.

```
<?php
        echo 'com ponto e vírgula' ;
?>
```

echo é uma função que imprime uma ou mais strings.

```
echo (string arg1, string argn....);
```

Como no C ou Perl, o PHP requer que as instruções sejam terminadas com um ponto-e-vírgula ao final de cada comando. A tag de fechamento de um bloco de código PHP automaticamente implica em um ponto-e-vírgula; você não precisa ter um ponto-e-vírgula terminando a última linha de um bloco PHP. A tag de fechamento irá incluir uma nova linha logo após, se estiver presente.

1.3 Comentários

O PHP suporta comentários no estilo 'C', 'C++' e shell do Unix shell (estilo Perl).

Tipos de comentários:

- // ou # é o comentário de estilo "uma linha", ou seja, que comentam apenas até o final da linha ou do bloco PHP de código corrente, o que chegar primeiro.
- /* ... */ são os comentários no estilo 'C' termina ao primeiro */ encontrado.

Exemplo

```
<?php
  echo "teste de comentários<br>";# exemplo de comentário de fim de linha
  echo "teste de comentários<br>";//exemplo de comentário de fim de linha
  echo "teste de comentários<br>";/*exemplo de comentário
   de mais de uma linha*/
?>
```

Na tela

1.4 Variáveis

Toda variável em PHP é iniciada pelo caracter $, seguido pelo nome (identificador) da variável.

Observações sobre o nome da variável:

- deve iniciar por uma letra ou o caracter "_"
- não podem ter como primeiro caracter de seu nome um número
- não podem caracteres acentuados ou de pontuação
- não podem ter espaço
- o PHP é case sensitive, ou seja, as variáveis $php e $PHP são diferentes
- o PHP já possui alguma variáveis pré−definidas cujos nomes são formados por letras maiúsculas. Então evite usar nome s de variáveis com letras maiúsculas.

Formato básico

$nomeDaVariável = valor;

Exemplo

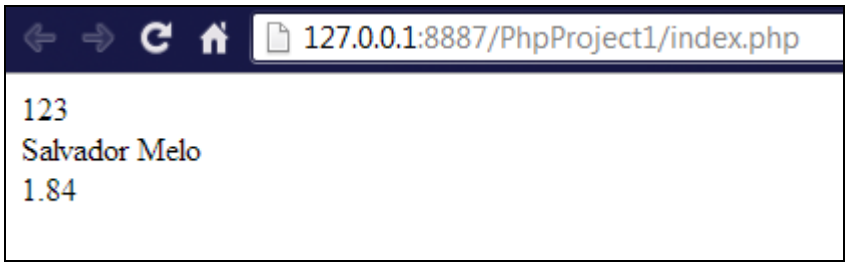
```
<?php
  $var=123;
  echo $var;
?>
```

Podemos implicitamente declarar o tipo de variável, porém estas variáveis podem ser facilmente convertidas para outro tipo, de forma totalmente transparente. Veja:

```
<?php
  $var=123;
  echo "$var<br>";
  $var="Salvador Melo";
  echo "$var<br>";
```

```
    $var=1.84;
    echo "$var<br>";
?>
```

Na tela:



Observações:

- Você pode usar algumas marcações do HTML em conjunto com as instruções do PHP, como o
 para quebra de linha;
- Usando aspas simples, o valor da variável será exatamente o texto contido entre as referidas aspas;
- Usando aspas duplas, qualquer caracter de escape será expandido antes de ser atribuído.
- Não é necessário variáveis inicializadas no PHP, contudo é uma ótima prática.

1.5 Quatro tipos básicos

- **integer**: Inteiros podem ser especificados em notação decimal (base 10), hexadecimal (base 16) ou octal (base 8), opcionalmente precedido de sinal (- ou +). Para usar a notação octal, você precisa preceder o número com um 0 (zero). Para utilizar a notação hexadecimal, preceda número com 0x.

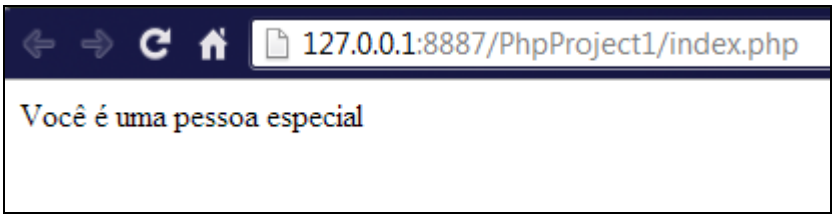
  Exemplo:
  ```
  <?php
        $a = 1234; // número decimal
        $a = -123; // um número negativo
        $a = 0123; // número octal (equivalente a 83 em decimal)
        $a = 0x1A; // número hexadecimal (equivalente a 26 em decimal)
  ?>
  ```

- **float** (número de ponto flutuante, ou também double)
  Exemplo:
  ```
  <?php
        $a = 1.234;
        $b = 1.2e3;
        $c = 7E-10;
  ?>
  ```
- **string:** uma string é uma série de caracteres. Antes do PHP 6, um caracter é o mesmo que um byte. Ou seja, há exatamente 256 caracteres diferentes possíveis. Isto implica que o PHP não tem suporte nativo ao Unicode. Por isso que em geral se acrescenta a seguinte linha de código html.
  ```
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  ```

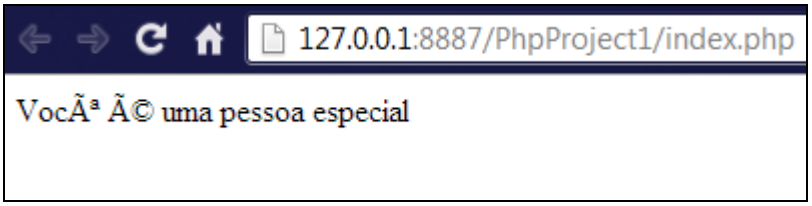
  Exemplo:
  ```
  <html>
  ```

```
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
   <title></title>
</head>
<body>
<?php
   echo "Você é uma pessoa especial";
?>
</body>
</html>
```

Na tela com o utf8:



Na tela sem o utf8:

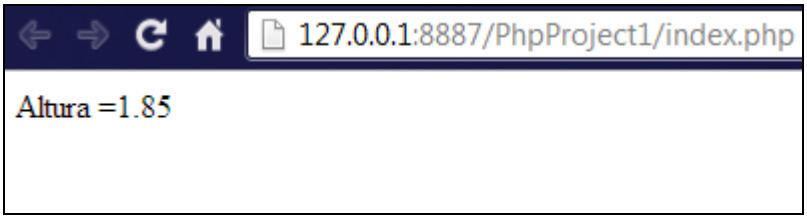


Strings podem ser concatenados utilizando o operador '.' (ponto). Note que o operador '+' (adição) não funciona para isso.

Exemplo:

```
<?php
  $valor=1.85;
  echo "Altura =".$valor;
?>
```

Na tela:

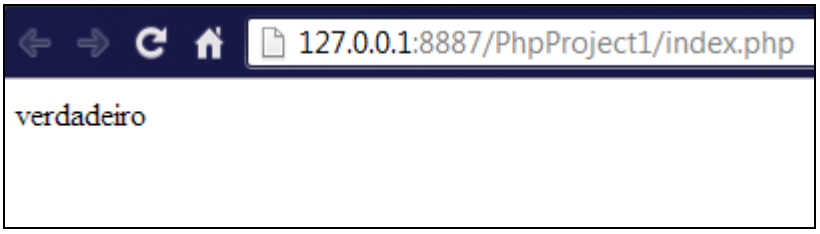


- **boolean**: um booleano expressa um valor verdade. Ele pode ser TRUE ou FALSE.
  Exemplo:

```
<?php
  $valor = true;
  if($valor == false)
     echo "falso";
```

```
    else
        echo "verdadeiro";
?>
```

Na tela:



1.5.1 Por valor ou por referência

Por padrão, as variáveis são sempre atribuídas por valor. Isto significa que quando você atribui uma expressão a uma variável, o valor da expressão original é copiado integralmente para a variável de destino. Isto significa também que, após atribuir o valor de uma variável a outra, a alteração de uma destas variáveis não afetará a outra. Para maiores informações sobre este tipo de atribuição, veja o capítulo em Expressões.

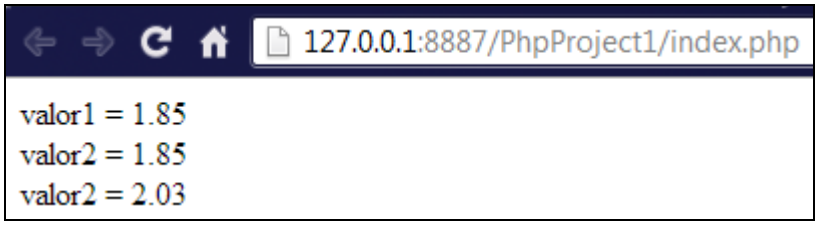
O PHP oferece um outro meio de atribuir valores a variáveis: atribuição por referência. Isto significa que a nova variável simplesmente referencia (em outras palavras, "torna-se um apelido para" ou "aponta para") a variável original. Alterações na nova variável afetam a original e vice versa.

Para atribuir por referência, simplesmente adicione um e-comercial (&) na frente do nome da variável que estiver sendo atribuída (variável de origem).

Exemplo:

```
<?php
  $valor1=1.85;
  echo "valor1 = $valor1<br>";
  $valor2=&$valor1;
  echo "valor2 = $valor2<br>";
  $valor1=2.03;
  echo "valor2 = $valor2<br>";
?>
```

Na tela:



1.5.2 Variáveis Pré-definidas

O PHP oferece um grande número de variáveis pré-definidas para qualquer script que ele execute.

1.6 Constantes

Uma constante é um identificador (nome) para um único valor. Como o nome sugere, esse valor não pode mudar durante a execução do script. As constantes são "Case Sensitive" (sensível ao tamanho

de letras) por padrão. Por convenção, o nomes de constantes são sempre em maiúsculas. O nome de uma constante tem as mesmas regras de qualquer identificador no PHP.

Você pode definir uma constante utilizando-se a:

- função define( )
- palavra-chave **const** fora da definição de uma classe a partir dor PHP 5.3.0.
- função constant( ) para ler o valor de uma constante, se você precisar obter seu valor dinamicamente.

Você pode obter o valor de uma constante simplesmente especificando seu nome. Diferentemente de variáveis, você não prefixa uma constante com um sinal de $.

Exemplo usando a função define( )

```
<?php
   define("CONSTANT", "Testando a constante");
   echo CONSTANT;
?>
```

Exemplo usando a palavra-chave const

```
<?php
   // Funciona a partir do PHP 5.3.0
   const CONSTANT = 'Testando a constante';
   echo CONSTANT;
?>
```

Observações

- Quando uma constante é definida, ela nunca mais poderá ser modificada ou anulada.
- A sintaxe da função define( ) é a seguinte:

  define(nome, valor);

  onde:

  nome = nome da constante.

  valor = O valor da constante; somente escalar e valores null são permitidos. Valores escalares são integer, float, string ou boolean.
- Ao contrário das funções definidas através de define( ), as constantes definidas usando a palavra-chave const devem ser declarados no escopo de topo (principal) pois são definidas no tempo de compilação. Isso significa que elas não podem ser definidas dentro de funções, laços ou ifs.
- A função constant( ) é útil se você precisa pegar o valor de uma constante, mas não sabe o seu nome. Ou seja, esta guardada em uma variável ou é retornada por uma função.

1.7 Operadores aritméticos

São os mesmos da linguagem C, vistos no semestre passado.

Exemplo	Nome	Resultado
-$a	Negação	Oposto de $a.
$a + $b	Adição	Soma de $a e $b.
$a - $b	Subtração	Diferença entre $a e $b.
$a * $b	Multiplicação	Produto de $a e $b.
$a / $b	Divisão	Quociente de $a por $b.
$a % $b	Módulo	Resto de $a dividido por $b.