

PHP

Aula 5

Tópicos

- 1 Conversão de variáveis
 - 1.1 Casting
- 2 Operadores de Comparação (relacionais)
- 3 Formas reduzidas
- 4 Operadores Lógicos
- 5 Mais um pouco sobre os métodos GET e POST dos formulários
 - 5.1 Criar um formulário
 - 5.2 O método GET
 - 5.3 O método POST
 - 5.4 Tratando as informações recebidas
 - 5.5 Algumas funções especiais para formatação de dados
- 6 Estruturas de controle
 - 6.1 Switch
 - 6.2 Do .. while
- 7 Exercícios
 - 7.1 Exercício 1
 - 7.2 Exercício 2
 - 7.3 Exercício 3
 - 7.4 Exercício 4
 - 7.5 Exercício 5

1 Conversão de variáveis

Se tivermos uma string contendo somente números, o PHP somará normalmente esse valor com outra variável do tipo numérico. Se houver textos e números em uma string, o PHP utilizará somente a parte em que estão os números para efetuar operações aritméticas.

Exemplo:

```
<?php
$nomequalquer = "5";
$numero=3;
$texto="20 anos";
$frase="tenho 40 anos";
echo $nomequalquer+$numero."<br>";
echo $numero+$texto."<br>";
echo $numero+$frase."<br>";
?>
```

Na tela:

8
23
3

1.1 Casting

O PHP não requer (ou suporta) a definição de tipo explícita na declaração de variáveis: o tipo de uma variável é determinado pelo contexto em que a variável é utilizada. Isto significa que, se você atribuir um valor string para a variável \$var, \$var se torna uma string. Se você então atribuir um valor inteiro para \$var,

ela se torna um inteiro. Porém, algumas vezes precisamos fazer uma conversão manualmente para realizar certos tipos de cálculos. A essa operação chamamos de casting.

A conversão de tipos no PHP funciona como no C: o nome de um tipo desejado é escrito entre parênteses antes da variável em que se deseja a moldagem.

As moldagens permitidas são:

- (int), (integer) - molde para inteiro
- (bool), (boolean) - converte para booleano
- (float), (double), (real) - converte para número de ponto flutuante
- (string) - converte para string
- (binary) - converte para string binária (PHP 6)
- (array) - converte para array
- (object) - converte para objeto
- (unset) - converte para NULL (PHP 5)

Exemplo 1 (casting parcial): <pre><?php \$var1 = 1.5; \$var2 = 2.5; \$soma = (int) \$var1+\$var2; echo \$soma; ?></pre>	Exemplo 2 (casting de toda a expressão): <pre><?php \$var1 = 1.5; \$var2 = 2.5; \$soma = (int) (\$var1+\$var2); echo \$soma; ?></pre>
Na tela: 3.5	Na tela: 4

2 Operadores de Comparação (relacionais)

São os mesmos operadores que utilizamos na linguagem C, com algumas pequenas modificações, a saber:

valor1 > valor2	Maior
valor1 >= valor2	Maior ou igual
valor1 < valor2	Menor
valor1 <= valor2	Menor ou igual
valor1 == valor2	Igual
valor1 != valor2	Diferente
valor1 <> valor2	Diferente

Exemplo:

```
<?php
$var = 3;
if($var <> 2)
    echo $var." é diferente de 2";
else
    echo $var." é igual de 2";
?>
```

Na tela:

3 é diferente de 2

3 Formas reduzidas

Utilizamos as mesmas formas reduzidas que utilizamos na linguagem C, com algumas pequenas modificações, a saber:

Forma normal	Forma reduzida
\$x = \$x + 1;	\$x++;
\$x = \$x - 1;	\$x--;
\$x = \$x + 3;	\$x+=3;
\$x = \$x - 3;	\$x-=3;
\$x = \$x * 3;	\$x*=3;
\$x = \$x / 2;	\$x/=2;
\$x = \$x + 1; \$y = \$x;	\$y = ++\$x;
\$y = \$x; \$x = \$x + 1;	\$y = \$x++;
\$x = \$x % 3;	\$x%=3;
\$y = \$y . \$x;	\$y.= \$x;

Exemplo:

```
<?php
    $x = 5;
    $y = 3;
    $y.= $x;
    echo "y = ".$y."<br>";
    $x%=3;
    echo "x = ".$x."<br>";
?>
```

Na tela:

y = 35
x = 2

4 Operadores Lógicos

São os mesmos operadores que utilizamos na linguagem C, com algumas pequenas modificações, a saber:

Se todos os testes forem verdadeiros	(x && y)
	(x AND y)
Se pelo menos um dos testes for verdadeiro	(x y)
	(x OR y)
Se só um dos testes for verdadeiro	(x XOR y)
Se o teste for falso (operador NOT)	(!x)

Usando XOR

Teste1	Teste2	Resultado
V	V	F
V	F	V
F	V	V
F	F	F

Usando NOT

Teste	Resultado
V	F
F	V

Exemplo 1:

```
<?php
$x = 5;
$y = 3;
if($x>1 AND $y>1)
    echo "ambos os testes são verdadeiros<br>";
else
    echo "um dos testes é falso<br>";
?>
```

ambos os testes são verdadeiros

Exemplo 2:

```
<?php
$x = 5;
$y = 3;
if($x>1 XOR $y>1)
    echo "apenas um dos testes é verdadeiro<br>";
else
    echo "ou os dois testes são verdadeiros ou os dois são falsos<br>";
?>
```

ou os dois testes são verdadeiros ou os dois são falsos

Exemplo 3:

```
<?php
$x = true;
if(!$x)
    echo "teste retorna true<br>";
else
    echo "teste retorna false<br>";
?>
```

teste retorna false

5 Mais um pouco sobre os métodos GET e POST dos formulários

É comum utilizar os formulários junto com o PHP. Para isso, precisamos pensar e seguir um roteiro que poderia ser assim:

- Criar um formulário
- Enviar as informações para um programa PHP
 - Usar o Método GET; ou
 - Usar o Método POST
- Tratar as informações recebidas

5.1 Criar um formulário

Os formulários são utilizados para aumentar a interatividade, fazendo comunicação entre o usuário e o site, e são criados por meio do HTML, composto de no mínimo um campo entrada de dados e um botão para enviar as informações contidas nele.

Por exemplo, se implementarmos o código abaixo, as informações digitadas serão perdidas, pois o navegador não sabe o que fazer com elas.

```
<body>
  <form>
    <p>Digite seu nome: <input type="text" name="usuario" size="20"></p>
    <p><input type="submit" value="Enviar!" name="enviar"></p>
  </form>
</body>
```

Para o tornar útil podemos usar a opção **action**, informando ao navegador para onde enviar a informações para serem processadas.

```
<body>
  <form action="recebe_dados.php" method="GET">
    <p>Digite seu nome: <input type="text" name="usuario" size="20"></p>
    <p><input type="submit" value="Enviar!" name="enviar"></p>
  </form>
</body>
```

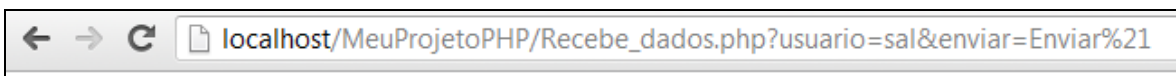
Utilizamos a opção da action da tag form do HTML, que também pode receber dois métodos de passagem de parâmetros para o arquivo **recebe_dados.php**, conhecidos como:

- GET
`<form action="recebe_dados.php" method="GET">`
- POST
`<form action="recebe_dados.php" method="POST">`

5.2 O método GET

Método GET é o padrão de envio de dados, se não for especificado o método na tag action, GET é assumido pelo PHP. Assim, os dados são enviados juntamente com o nome da página (URL) para o envio de dados.

Observe:

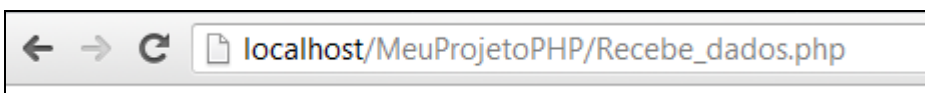


- Desvantagens
 - Limite de caracteres é de 2.000;
 - Os dados enviados são visíveis na barra de endereço do navegador. O método POST resolve isso.
- Vantagem
 - Pode ser utilizado para passagem de parâmetros por link.

5.3 O método POST

Diferente do GET, o POST envia os dados por meio do corpo da mensagem encaminhada ao servidor.

Observe:



- Vantagens
 - Não é visível a cadeia de variáveis
 - Não há limites no tamanho dos dados, sendo mais usado para formulários com grande quantidade de informações
 - Permite enviar outros tipos de dados, não aceitos pelo GET, como imagens ou outros arquivos (usar valor file na opção type da tag input)
- Desvantagens
 - Não é possível a passagem de parâmetros por link

5.4 Tratando as informações recebidas

Existem duas maneiras de acessar os dados recebidos:

- Tratar como variáveis adicionando \$ ao nome dos campos especificados no formulário. O campo nome ficaria \$nome e o campo idade \$idade.
 - É necessário que a opção register_globals seja ativada (funcionalidade obsoleta)
- Usar os arrays superglobais predefinidos pelo php: \$_GET e \$_POST
 - Os nome dos campos são usado como chaves associativas \$_GET["nome"] ou \$_POST["idade"]

Por questões de segurança os desenvolvedores do PHP recomendam o uso do arrays \$_GET e \$_POST. Caso queira usar a primeira opção, cuide para não criar outras variáveis com o mesmo nome, pois o valores serão sobrescritos.

Exemplo (dois arquivos do mesmo projeto PHP):

index.php

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>
  <form action="Recebe_dados.php" method="POST">
    <p>Digite seu nome: <input type="text" name="usuario" size="20"></p>
    <p><input type="submit" value="Enviar!" name="enviar"></p>
  </form>
</body>
</html>
```

Recebe_dados.php

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>
  <?php
    $usuario = $_POST["usuario"];
    $erro = FALSE;
    if(strlen($usuario)<5)
      {echo "O nome deve ter ao menos 5 caracteres. <br>"; $erro=TRUE;}
    if(!$erro)
      {echo "Todos os dados foram digitados corretamente! <br>";}
  ?>
</body>
</html>
```

Na tela:

Digite seu nome:

O nome deve ter ao menos 5 caracteres.

5.5 Algumas funções especiais para formatação de dados

- `empty($nomeVariável)`
 - Verifica se uma variável esta vazia.
 - Essa função somente verifica variáveis, qualquer outra coisa então irá resultar em um erro.
 - Retorna TRUE quando uma variável for vazia e FALSE quando uma variável não for vazia.
 - Exemplo:

```
if(empty($var))
    echo "variável com conteúdo";
else
    echo "variável vazia";
```
- `isset($nomeVariável)`
 - Verifica se uma variável existe.
 - Essa função somente verifica variáveis, qualquer outra coisa então irá resultar em um erro.
 - Retorna TRUE quando uma variável existir e FALSE quando uma variável não existir.
 - Para uma variável “não existir” ela não pode ter sido usada/definida em nenhum momento [anterior] do script.
 - Exemplo:

```
<?php
if (isset($var))
    echo "Essa variável existe.";
else
    echo "Essa variável NÃO existe.";
?>
```
- `strstr()`
 - Encontra a primeira ocorrência de uma string.
 - Esta função diferencia maiúsculas de minúsculas. Para pesquisas que não diferenciem, use `stristr()`.
 - Exemplo:

```
<?php
$email = 'name@example.com';
$domain = strstr($email, '@');
echo $domain."<br>"; // prints @example.com

$user = strstr($email, '@', true); // A partir do PHP 5.3.0
echo $user; // prints name

if(!strstr($email, '@'))
    echo "endereço de email inválido";
?>
```
- `strlen`
 - Retorna o número de caracteres de uma string.
 - Exemplo:

```
<?php
    $str = 'abcdef';
    echo strlen($str). "<br>"; // 6

    $str = ' ab cd ';
    echo strlen($str); // 7
?>
```

6 Estruturas de controle

O uso do if, for e while são idênticos àquele que vimos na linguagem C. Além dessas estruturas, temos ainda o switch e o do...while, que veremos a seguir.

6.1 Switch

O comando switch é parecido com o if, pois ambos avaliam o valor de uma expressão (teste), para escolher qual bloco de instruções deve ser executado. Porém, o if usa apenas uma expressão e o switch pode usar várias.

Estrutura básica:

```
switch(opção)
{
    case valor1: comandos;break;
    case valor2: comandos;break;
    case valor3: comandos;break;
    ...
    default: comandos;break;
}
```

O break termina a execução do comando atual, e pode ser utilizado com um for, while e switch. Quando o break é encontrado numa dessas estruturas, o fluxo de execução passa para o próximo comando depois da referida estrutura.

Exemplo:

```
<?php
    $var=2;
    switch($var){
        case 1: echo "armazenado o valor 1";break;
        case 2: echo "armazenado o valor 2";break;
        case 3: echo "armazenado o valor 3";break;
        default: echo "não foi armazenado nem 1, nem 2 e nem 3";break;
    }
?>
```

6.2 Do .. while

A diferença entre o while e o do...while é que o teste é avaliado no final do laço e não no começo. Como consequência, as instruções que estiverem dentro do do...while serão executadas pelo menos uma vez.

Estrutura básica:

```
do{  
    instruções
```

```
}while(teste);
```

Exemplo:

```
<?php  
$num=1;  
do{  
    echo "valor atual da variável num = $num<br>";  
    $num++;  
}while($num <=4);  
?>
```

Na tela:

```
valor atual da variável num = 1  
valor atual da variável num = 2  
valor atual da variável num = 3  
valor atual da variável num = 4
```

7 Exercícios**7.1 Exercício 1**

Implemente todos os exemplos, analise o código e anote as suas dúvidas.

7.2 Exercício 2

Faça o teste de mesa, informe o último valor armazenado nas variáveis declaradas no código abaixo.

```
<?php  
$num = 5;  
$resultado = 8 + 3 * 2 + ++$num;  
echo "num = $num<br>";  
echo "resultado = $resultado";  
?>
```

```
num = 6  
resultado = 20
```

7.3 Exercício 3

Reproduza a página ilustrada abaixo, e faça um comentário, usando o switch, para escolha que o usuário fizer. Formulário num arquivo e aplicação em php, com o switch, em outro arquivo.

```
O que você achou do switch?  
  
☒ Muito bom ☐ Bom ☐ Regular ☐ Um Lixo  
  

```

7.4 Exercício 4

Crie uma página que simula o valor das parcelas de um empréstimo imobiliário pelo Tabela SAC (Sistema de Amortização Constante), que é o atual sistema utilizado pela Caixa Econômica Federal. O usuário deve entrar com os seguintes dados:

- Valor do empréstimo (exemplo: R\$ 100.000,00)
- Prazo para pagamento (exemplo: 10 meses)
- A taxa de juros ao mês cobrado pelo banco (exemplo: 25% ao mês)

Alguns cálculos necessários:

- $\text{amortização} = \text{valor do empréstimo} / \text{prazo para pagamento}$
- $\text{saldo devedor (inicial)} = \text{valor do empréstimo}$
- $\text{juros do mês} = \text{taxa de juros do banco} * \text{saldo devedor atual} / 100,0$
- $\text{valor da prestação do mês} = \text{valor da amortização} + \text{juros do mês}$
- $\text{saldo devedor atual} = \text{saldo devedor anterior} - \text{valor da amortização}$

No fim do período, o saldo devedor deverá ser zero.

As imagens que devem aparecer na tela deverão ser semelhantes ou melhores que esta:

Simulação de Financiamento pela Tabela SAC

Valor Financiado:

Número de Meses:

Taxa de Juros Mensal:

Figura 1: imagem antes de clicar no botão simular.

Simulação de Financiamento pela Tabela SAC

Valor Financiado:

Número de Meses:

Taxa de Juros Mensal:

Mês	Amortização	Parcela	Saldo
1	833,33	2.133,33	99.166,67
2	833,33	2.122,50	98.333,33
3	833,33	2.111,67	97.500,00
4	833,33	2.100,83	96.666,67
5	833,33	2.090,00	95.833,33
6	833,33	2.079,17	95.000,00
7	833,33	2.068,33	94.166,67
8	833,33	2.057,50	93.333,33
9	833,33	2.046,67	92.500,00
10	833,33	2.035,83	91.666,67
11	833,33	2.025,00	90.833,33
12	833,33	2.014,17	90.000,00
13	833,33	2.003,33	89.166,67

Figura 2: imagem depois que clicar no botão simular.

Dicas:

- Use um formulário para entrada de dados;
- Chame o próprio arquivo do formulário para trabalhar com o php;
- Coloque os comandos do php logo abaixo do form;
- Com a tag `<input>` use o `type="number"` para definir o número de casas decimais;
 - Exemplo:


```
echo number_format($numero, 2, ',', '.');
```

onde:

1º – O número a ser formatado

2º – A precisão decimal (quantidade de casas decimais que serão exibidas)

3º – Separador de dezenas (opcional)

4º – Separador de milhar (opcional)

- Use a função `isset()` para verificar se o botão enviar foi acionado;
- Use este método pouco utilizado de concatenamento da função `echo`, que também é mais rápido em termos de performance do que o tradicional concatenamento com pontos.

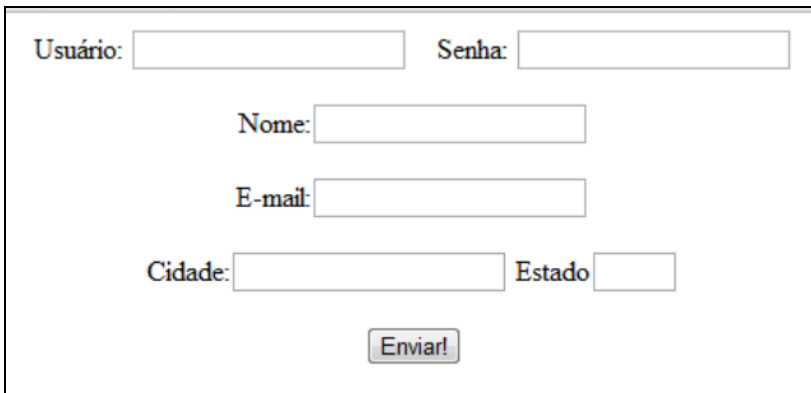
- Exemplo:

```
echo '<table class="resultado">'  
'<tr>'  
'<td>' . $cidade . '</td>'  
'<td>' . $estado . '</td>'  
'<td>' . $pais . '</td>'  
'</tr>'  
'</table>';
```

7.5 Exercício 5

Reproduza a página ilustrada abaixo, e faça as seguintes verificações:

- O nome do usuário deve ter pelo menos 5 caracteres
- A senha deve ter pelo menos 5 caracteres
- A senha não pode estar em branco
- O email não pode estar em branco, precisa ter o caracter @ e não pode ter menos que 8 caracteres
- A cidade não pode estar em branco
- O estado não pode nem menos nem mais que 2 caracteres
- Uma mensagem dizendo que todos os dados foram cadastrados corretamente, conforme os critérios acima.



Formulário de cadastro com os seguintes campos:

- Usuário:
- Senha:
- Nome:
- E-mail:
- Cidade:
- Estado:
- Botão: Enviar!