

Reconhecida pela Portaria nº 821/MEC – D.O. U de 01.06.94

COORDENAÇÃO UVA / EVOLUTIVO

**DISCIPLINA:** CRIAÇÃO DE SITES NA WEB II

**PROFESSOR (A):** JOÃO CLÁUDIO

**AULA 3 (29/01/2008)**

**EXERCÍCIO 1**

Veremos como se escrever  a data por extenso em PHP com o seguinte formato:

***Quarta-feira, 11 de Julho de 2008***

Para a criação do script foi utilizada a função getdate() do PHP, que monta um array com as informações da data. Para que a data fique no formato acima indicado usaremos o comando switch para que seja possível converter o formato da data gerado pelo PHP. Vamos iniciar o nosso script.  Abra o Bloco de Notas e salve com o nome de *dt\_extenso.php*, em seguida digite os código abaixo:

1. <?php
2. $hoje = [getdate](http://www.php.net/getdate)();
4. // Nessa parte do código foi criada a variável $hoje, que receberá os valores da data.
6. switch ($hoje['wday'])
7. {
8. case 0:
9. [echo](http://www.php.net/echo) "Domingo, ";
10. break;
11. case 1:
12. [echo](http://www.php.net/echo) "Segunda-Feira, ";
13. break;
14. case 2:
15. [echo](http://www.php.net/echo) "Terça-Feira, ";
16. break;
17. case 3:
18. [echo](http://www.php.net/echo) "Quarta-Feira, ";
19. break;
20. case 4:
21. [echo](http://www.php.net/echo) "Quinta-Feira, ";
22. break;
23. case 5:
24. [echo](http://www.php.net/echo) "Sexta-Feira, ";
25. break;
26. case 6:
27. [echo](http://www.php.net/echo) "Sábado, ";
28. break;
29. }
31. // Acima foi utilizada a instrução switch para que o dia da semana possa ser apresentado por
32. // extenso, já que o PHP retorna em números. Perceba que dentro de cada instrução case tem uma
33. // instrução echo que escreve o dia da semana na tela.
35. [echo](http://www.php.net/echo) $hoje['mday'];
37. // A instrução echo $hoje[‘mday’]; escreve na tela o data em número,
38. // conforme retorna o PHP, não precisando de conversão.
40. switch ($hoje['mon'])
41. {
42. case 1:
43. [echo](http://www.php.net/echo) " de Janeiro de ";
44. break;
45. case 2:
46. [echo](http://www.php.net/echo) " de Fevereiro de ";
47. break;
48. case 3:
49. [echo](http://www.php.net/echo) " de Março de ";
50. break;
51. case 4:
52. [echo](http://www.php.net/echo) " de Abril de ";
53. break;
54. case 5:
55. [echo](http://www.php.net/echo) " de Maio de ";
56. break;
57. case 6:
58. [echo](http://www.php.net/echo) "de Junho de ";
59. break;
60. case 7:
61. [echo](http://www.php.net/echo) " de Julho de ";
62. break;
63. case 8:
64. [echo](http://www.php.net/echo) " de Agosto de ";
65. break;
66. case 9:
67. [echo](http://www.php.net/echo) " de Setembro de ";
68. break;
69. case 10:
70. [echo](http://www.php.net/echo) " de Outubro de ";
71. break;
72. case 11:
73. [echo](http://www.php.net/echo) "de Novembro de ";
74. break;
75. case 12:
76. [echo](http://www.php.net/echo) " de Dezembro de ";
77. break;
78. }
80. // A parte do código acima tem a mesma função que o primeiro switch utilizado,
81. // só que agora ele é usado para apresentar o mês.
83. [echo](http://www.php.net/echo) $hoje['year'];
84. ?>

**EXERCÍCIO 2**

**Conceito**  
Orientação a objetos é um termo que descreve uma série de técnicas para estruturar soluções para problemas computacionais. No nosso caso específico, vamos falar de programação OO, que é um paradigma de programação no qual um programa é estruturado em objetos, e que enfatiza os aspectos abstração, encapsulamento, polimorfismo e herança.  
  
**Objeto**  
Objetos são a unidade fundamental de qualquer sistema orientado a objetos. Tudo é um objeto -- tipos, valores, classes, funções, métodos, pode ser uma pessoa, um lugar, um carro, um avião, e é claro, instâncias: todos possuem atributos e métodos associados. Características que definem um objeto são os atributos e comportamentos chamados de  métodos.  
  
**Atributos seriam os aspectos de um objeto:**  
Objeto Carro:  
· Marca  
· Modelo  
· Cor  
· Ano  
  
**As ações de um objeto seriam os métodos:**  
Objeto Carro:  
  
· Correr  
· Freiar  
· Bater  
· Parar  
  
**Construtores**  
 O Construtor é referenciado no PHP como **\_\_construct()**, é uma função definida na classe e que é executada sempre que o objeto é criado, ou seja, sempre que a classe é instanciada.  
  
**Destrutores**  
São chamadas no momento em que o objeto está sendo destruído.Podem servir para fechar uma conexão com bando de dados, no PHP5 usamos a nomenclatura padrão **\_\_destruct()** para implementarmos destrutores.  
  
**Visibilidade de Atributos e Métodos**Os métodos e atributos de uma classe no PHP podem ser definidos como: **private**, **public**, **protected  
  
Public**O atributo ou método definido como **public** torna-o acessível em qualquer lugar da classe, de suas subclasses ou em qualquer parte dos scritps que fazem parte da classe.  
  
**Protected**  
O atributo ou método definido como **protected** são visíveis pela classe que os criou e por suas subclasses, e não são acessíveis fora desse contexto.  
  
**Private**  
O atributo ou método definido como private são visíveis apenas na classe que os criou, ou seja, subclasses ou script que contêm a classe não podem acessar esses atributos ou métodos.  
  
Vamos a um exemplo básico para melhor entendimento  
  
**Arquivo carro.class.php**

1. <?php
2. class CARRO
3. {
4. // Aqui declaramos o atributo privado \_marca
5. private $\_marca;
6. // Aqui declaramos o atributo privado \_modelo
7. private $\_modelo;
8. // Aqui declaramos o atributo privado \_cor
9. private $\_cor;
11. // Função pública, lembrando que ela é executada no momento de criação do objeto
12. public function \_\_construct() {
13. // Mensagem é exibida no momento da criação do objeto
14. [echo](http://www.php.net/echo) 'Passei por aqui, logo criei um objeto<br>';
16. // O $this é utilizada para referenciar ao próprio objeto, ou seja,
17. // $this->\_marca estaria acessando ao atributo $\_marca.
19. //chama-se a função setAtributos atribuindo valores "NULL" a Marca do Carro
20. $this->setAtributos(null,null,null);
21. }

24. // Recebe como parametros uma marca, um modelo e uma cor
25. public function setAtributos($marca, $modelo, $cor) {
26. // Atributo \_marca recebe a marca passada por parâmetro
27. $this->\_marca = $marca;
28. // Atributo \_modelo recebe um modelo passado por parâmetro
29. $this->\_modelo = $modelo;
30. // Atributo \_cor recebe uma cor passada por parâmetro
31. $this->\_cor = $cor;
32. }
34. // Função que retorna apenas a Marca do carro
35. public function getMarca() {
36. return $this->\_marca;
37. }
39. // Função que retorna apenas o Modelo do carro
40. public function getModelo() {
41. return $this->\_modelo;
42. }
44. // Função que retorna apenas a Cor do carro
45. public function getCor() {
46. return $this->\_cor;
47. }
48. }
49. ?>

**Arquivo index.php**

1. <?php
2. // Fazemos nosso include da nossa classe
3. include 'carro.class.php';
5. //criação de uma nova instância da classe CARRO ($um\_carro)
6. $um\_carro = new CARRO;
7. //criação de uma nova instância da classe CARRO ($dois\_carros)
8. $dois\_carros = new CARRO;
9. //criação de uma nova instância da classe CARRO ($tres\_carros)
10. $tres\_carros = new CARRO;
12. // Chamamos a função setAtributos, que seta os atributos do objeto $um\_carro
13. $um\_carro->setAtributos('Munstang','GT','Vermelho');
14. // Chamamos a função setAtributos, que seta os atributos do objeto $dois\_carros
15. $dois\_carros->setAtributos('Audi','A3','Preto');
16. // Chamamos a função setAtributos, que seta os atributos do objeto $tres\_carros
17. $tres\_carros->setAtributos('Peugeot','307','Prata');
19. // Eximos uma linha para separação
20. [echo](http://www.php.net/echo) '<hr>';
21. [echo](http://www.php.net/echo) '<p>';
22. // Exibimos a Marca do carro e quebramos uma linha com o <br />
23. [echo](http://www.php.net/echo) '<b>Carro:</b> '. $um\_carro->getMarca() . '<br />';
24. // Exibimos o Modelo do carro e quebramos uma linha com o <br />
25. [echo](http://www.php.net/echo) '<b>Modelo:</b> '. $um\_carro->getModelo() . '<br />';
26. // Exibimos a Cor do carro e quebramos uma linha com o <br />
27. [echo](http://www.php.net/echo) '<b>Cor:</b> '. $um\_carro->getCor() . '<br />';
28. [echo](http://www.php.net/echo) '</p>';
29. // Eximos uma linha para separação
30. [echo](http://www.php.net/echo) '<hr>';
32. [echo](http://www.php.net/echo) '<p>';
33. // Exibimos a Marca do carro e quebramos uma linha com o <br />
34. [echo](http://www.php.net/echo) '<b>Carro:</b> '. $dois\_carros->getMarca() . '<br />';
35. // Exibimos o Modelo do carro e quebramos uma linha com o <br />
36. [echo](http://www.php.net/echo) '<b>Modelo:</b> '. $dois\_carros->getModelo() . '<br />';
37. // Exibimos a Cor do carro e quebramos uma linha com o <br />
38. [echo](http://www.php.net/echo) '<b>Cor:</b> '. $dois\_carros->getCor() . '<br />';
39. [echo](http://www.php.net/echo) '</p>';
40. // Eximos uma linha para separação
41. [echo](http://www.php.net/echo) '<hr>';
43. [echo](http://www.php.net/echo) '<p>';
44. // Exibimos a Marca do carro e quebramos uma linha com o <br />
45. [echo](http://www.php.net/echo) '<b>Carro:</b> '. $tres\_carros->getMarca() . '<br />';
46. // Exibimos o Modelo do carro e quebramos uma linha com o <br />
47. [echo](http://www.php.net/echo) '<b>Modelo:</b> '. $tres\_carros->getModelo() . '<br />';
48. // Exibimos a Cor do carro e quebramos uma linha com o <br />
49. [echo](http://www.php.net/echo) '<b>Cor:</b> '. $tres\_carros->getCor() . '<br />';
50. [echo](http://www.php.net/echo) '</p>';
51. // Eximos uma linha para separação
52. [echo](http://www.php.net/echo) '<hr>';
53. ?>