Módulo 5 – Construção de páginas web dinâmicas

Php

- Hypertext Preprocessor;
- Executado do lado o servidor;
  - A linguagem é interpretada ao nível do servidor.
- Para a visualização num browser de páginas programadas em php, devemos fazer o upload dos ficheiros para um servidor que interprete esta linguagem.
- Existem alguns sites gratuitos onde é possível registar um espaço e alojar páginas em php.
- Pode ser simulado localmente
  - O computador deve simular um servidor web, com interpretado php localmente (Apache)
  - Existem alguns pacotes: WampServer, Appserver
  - http://localhost/ficheiro.php ou http://127.0.0.1/ficheiro.php

Declaração

<?php
 (escrever o código aqui)
?>

- Ou

<script language="php">
 (escrever o código aqui)
 </script>

- Ou

<?
 (escrever o código aqui)
 </script>

- Onde
  - Nas tags head ou body do html

Operadores básicos

Operadores aritméticos	
+	soma
-	subtração
*	Multiplicação
/	Divisão
%	Resto da divisão inteira

Operadores de incremento / Decremento
++
--

Operadores aritméticos de atribuição	
+=	
-=	
*=	
/=	
%=	

Operadores de indicação de tipo	
GetType()	Indica o tipo (String number, ...)

Operadores de comparação e lógicos

Operadores de comparação
== (igual a)
!= (Diferente de)
<
<=
>
>=

Operadores lógicos
! (Not – negação)
&& (and –e)
|| (or - ou)

Notas:
Os **comentário** são feitos com // ou /* (código a comentar) */

Para “imprimir” utilizar o **echo**
Exemplo echo “Olá”;

Variáveis

- Declaração: Não precisam ser declaradas
- Não têm tipo definido
- A declaração implica a precedência de um $
- Devem começar por uma letra
- O php é case-sensitive (Teste != teste != tEste)
- Exemplos:

```
<?php
   //Declaração de variáveis em php
   $x=10;
   echo GetType($x);
   echo $x;
   $x = “joana”;
   echo GetType($x);
  echo $x;  ?>
```

Estruturas de controlo

```
If(condição){
  instruções;
}
Elseif(condição){
  instruções;
}
Else{
  instruções;
}
```

Exemplo

```
<?php
$x =2;
  if($x == 0)
    echo "Numero " . $x;
 elseif($x == 1)
   echo "Numero " . $x;
 else
   echo "Numero errado";
?>
```

Estruturas de controlo

```
switch(expressão){
case valor1: instruções; break;
case valor2: instruções; break;
case valorN: instruções; break;
Default: instruções
}
```

```
<?php
                switch($x)
               {
   case 25: echo "idade = " . $x; break;
   case 29: echo "idade = " . $x; break;
   case 30: echo "idade = " . $x; break;
   case 34: echo "idade = " . $x; break;
   default: echo ”<br>Idade não definida";
               }
?>
```

Ciclos

```
for(inicialização; condição; incremento){
  instruções;
}
while(condição)
{
  instruções;
}
do{
  instruções
}while(condição);
```

```
<?php
 for($i=1; $i <= 16; $i++)
   echo "Numero " . $i . "<br>";
?>
```

```
<?php

 $i = 1;

 while($i <= 16){
   echo $i . "<br>";
   $i++;
 }
?>
```

```
<?php

 $i = 1;

 do{
   echo $i . "<br>";
   $i++;
 }while($i <= 16);
?>
```

Exercício pratico

1.Copie o código da página 6 (variáveis) para o editor. Grave com o nome printVar.php. Verifique os resultados no browser.
2.Copie o código do exemplo anterior (ciclo for) para o editor. Grave o ficheiro com o nome for.php. Altere-o para que imprima todos os números de 23 a 35. Teste o código escrito no browser.

Funções, classes e objectos

- Funções: Definidas pela palavra function, podendo ou não receber parametros, retornar ou não valores.

```
function nome_da_funcao($parametro1, $parametro2, ...,$parametron)
{
  intruções;
  return [$valor ou expressão];
}
```

```
function alerta()
{
  echo "Dados incorrectos";
}
```

```
function soma($v1, $v2)
{
  $sv = $v1 + $v2;

  return $sv;
}
```

Variáveis (locais e globais)

- Variáveis globais – sempre que é necessário que a variável exista durante todo o código.
- Variáveis locais – em apenas certos momentos do código.
- Por defeito as variáveis são locais.
- Declaração de uma variável como global: GLOBAL $var;

```
<?php
$a = 3;


function ver()
{
  $b = $a + 2;
  echo $b;
}


echo " chamar funcao ";
Echo ver();


echo "<br>";
echo  "Imprimir o valor de a " . $a;
echo "<br>";
echo "Imprimir o valor de b " . $b ;
```

```
<?php
$a = 3;


function ver()
{
  GLOBAL $a;
  $b = $a + 2;
  echo $b;
}


ver();


echo "<br>";
echo $a;
echo "<br>";
echo $b;
```

Exercício pratico

- Copie os dois códigos da página anterior. Grave com os nome Var1.php e Var2.php.
- Verifique os resultados no browser.

Constantes

- Define(constante, valor_da_constante)
- Exemplos

Define(“A”, 3)

Define(“Nome”, “Joana”)

```
<?php
Define("A", 3);
Define("Nome", "Joana");


function ver($a)
{
$b = $a + 2;
  echo $b;
}


ver(10);


echo "<br>";
echo A;
echo "<br>";
echo Nome;


?>
```

Arrays (I)

- Para criar um array

  $nome_array = array(conteudo_pos0, conteudo_pos1, ..., conteudo_posn)

  O primeiro elemento do array encontra-se na posição zero.

- Exemplo

```
$a = array(10, 20, 30); // array com 3 posições preenchidas
echo $a; // irá aparecer a palavra array no browser
```

- Para aceder ao conteúdo de uma posição de um array

```
$nome_array[posicao];
Exemplo: echo $a[0];
```

- Um array poderá misturar no seu conteúdo strings, inteiros, floats, etc.

  Exemplo $a = array(10, “joana”, 3.5)

Arrays (II)

- Para acrescentar um elemento a um vector

  $a[] = 600;

  o valor 600 ficou na posição 3.

  Em alternativa pode-se fazer: $a[3] = 600;

Exemplo – Arrays e ciclo for

```
for($i = 0; $i < 3; $i++)
{
   echo $a[$i];
}
```

Ciclo Foreach e arrays

- Por vezes não se sabe o tamanho de um array, por exemplo quando o array é preenchido com informação de uma base de dados.
- Nestes casos pode-se utilizar o ciclo foreach.
  - Percorre todas as posições ocupadas de um array, imprimindo por cada iteração o seu conteúdo.
  - O ciclo para quando não existirem mais informações preenchidas com informação no array.
- Ciclo foreach

  foreach($nome_array_existente as $nome_escolhido)
- Exemplo:

  foreach($a as $cont)

  {echo $cont;}

Arrays associativos

- Nos arrays associativos, em vez de se utilizar identificadores numericos para o acesso às posições, utilizam nomes.
- Este tipo de array associam uma chave a um valor

  ($chave=>$valor)

- Para criar um array associativo

$nome_array_associativo = array("nome_chave_pos0"=>valor_pos0, ..., "nome_chave_posn"=>valor_posn)

- Exemplo 2

```
$a = array("Joana" => 24, "Rute" => 33, "Ricardo" => 28);
echo "O Ricardo tem " . $a["Ricardo"] . " anos";

foreach($a as $nome=>$idade)
  echo "Nome" . $nome . "idade " . $idade . "<br>";
```

Arrays associativos

- Acrescentar um valor ao array associativo:
  $a[“Hugo”] = “23”
- Alguns erros com os arrays associativos

$a = array(“Joana” => 24, “Rute” => 33, “Ricardo” => 28);

1. foreach($a as $nome)
   echo “Nome: ” . $nome . “
”;

Neste caso só imprime todas as idades do array

2. for($i=0; $i<2; $i++)
   echo $a[$i];

Um array associativo não pode ser acedido através de posições numéricas.

Exercício

- Copiar para o editor (javascript free editor, por exemplo) o código da exemplo 2, alterando os dados do array(nome, idade)
- Acrescentar um valor ao array associativo
- Grave o ficheiro com o nome arrayasso.php
- Verifica os resultados no browser.

Funções

- A linguagem php dispõe de várias funções que podem ser consultadas, por exemplo em www.php.net

- Alguns exemplos de funções
  - header[“Location: página destino.php”]
  - É utilizada quando pretendemos que determinada acção nos encaminhe para uma página diferente da atual.
  - Exemplo:
  - if($login==“aluno” && $pasw == 1234)

```
{
  header[“Location:main.php”];
}
else{
  header[“Location:erro.php”];
}
```

Funções

- isset(variavel)
  - Verifica se uma variável está ou não definida.
  - Retorna 0 (false) se a variável não está definida
- Exemplo

```
$a=2;
 isset($a);
 isset($b);
```

Form Action e method

- <form action="nome_ficheiro.php" method=....>
  - O action num formulário serve para indicar que existe um ficheiro que irá tratar dos dados após a submissão.
  - Method – Mostra a forma como os dados serão capturados
    - Existem dois métodos de captura de dados: GET e POST
    - Exemplo:

<form action="registo.php" method="POST">

<form action="captar.php" method="GET">

Get e Post

- Get
  - Não existe confidencialidade – os dados passados por este método são mostrados na barra de endereços. Dados confidenciais como passwords não devem ser passados na barra de endereços.
  - Limite de caracteres – Os dados a passar têm um limite de 100 caracteres.

- Post
  - Garante confidencialidade
  - Não existe limite de caracteres

Post e Get (II)

- Forma de aceder à informação captada
  - Se o método for o GET captamos os dados com $_GET[“....”]
  - Se o método for POST, captamos os dados com $_POST[“....”]

Name no formulário que enviou

- Exemplo

Ficheiro form.html

```
<form method="post" action="processar.php">
   Nome:*<input type='text' name='nome’>
   Idade: <input type='text' name='idade'> <br>

  <input type='submit' value='Enviar'>
  <input type='reset' value='Apagar'>
 </form>
```

Ficheiro processar.php

```
<?php
 $nome = $_POST["nome"];

 $idade = $_POST["idade"];

 echo "Nome enviado pelo formulário: " . $nome .
"<br>";
 echo "Idade enviado pelo formulário: " . $idade . "<br>";
?>
```

Exercício

- Mostrar no browser os dados enviados pelo formulário do ficheiro registo.html.
- Os dados são enviados pelo método GET.
- O ficheiro registo.html encontra-se no moodle da disciplina

Cookies e sessões

- Dados que são guardados com o objetivo de saber, por exemplo se o utilizador já acedeu ou não a uma página
- Exemplo: Página de nome entrar.php que contem formulário onde o utilizador escreve o username e a password que serão validados pelos sistema.

  Ao introduzir os dados correctos é redirecionado para a página main.php; se erra é redireccinado para a página erro.php
- Nada nos impede de aceder diretamente à página main.php (basta escrever o endereço da página main.php no browser) sem passar pela autenticação.
- Como a autenticação é necessária, a situação anterior não pode acontecer, devido a questões de segurança
- Como resolver a questão:
  - Através de Cookies ou de sessões (sessions).

Cookie vs session

Cookie	Session
Armazenados localmente	Armazenados no servidor
Tempo de vida limitado (expiram)	Não expiram
Baixa segurança	Maior segurança

cookies devem ser evitadas no caso de dados sigilosos pois ficam guardados
ficheiros temporários do computador.

Cookie

- São obrigatoriamente declaradas na tag <html>
- Criar cookies
  - setcookie(name, value,expire,path,domain,secure)
    - name – nome da cookie;
    - value – valor a guardar temporariamente;
    - expire – tempo em segundos de duração da cookie;
    - path – endereço da página que gerou a cookie – automático;
    - domain – domínio ao qual pertence a cookie – automático;
    - secure: 1 ou 0 para ligação segura (https) ou não segura, respectivamente.

Cookie - exemplo

- setcookie(“bolacha”, “torrada”, time()+60);
  - cookie de nome bolacha que expira em 60 segundos
- Para aceder ao valor de cookies criadas:
  - $_COOKIE[nome_da_cookie]
- Exemplo:
  - setcookie(“uti056”, “ghr443”, time()+60);
  - echo $_COOKIE[‘uti0563’];

Cookie – Exemplo 2

```
(....)
if(!issset($_COOKIE[‘uti0563’]))
  header[“Location: erro.php”];
```

•A criação da cookie pode ser feita, por exemplo após a autenticação com sucesso de um utilizador no site, criando-se uma cookie válida para cada utilizador.

Session

- São obrigatoriamente declaradas na tag <html>
- Criar / usar session
  - Session_start();
- Eliminar Session
  - Unset(); - destrói sessão especifica
  - Session_destroy(); - destroy sessão activa
- Para aceder ao valor de uma sesão
  - $_SESSION[‘nome_a_escolha’];

Exemplo - Session

(...)
Session_start(); //sempre necessário quando trabalhamos com sessões
$_SESSION[“login”] = $_POST[“login”]; // copia para a variável de sessão o login retirado de um formulário

Exemplo
(...)
Session_start();
if(!isset($_SESSION[“login”]))
  header[“location:erro.php”];
(...)

Strings e tratamentos de erros

- Alguns cuidados ao imprimir textos que contêm os seguintes caracteres: “, $ ou /