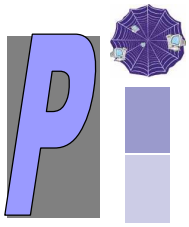


Tema 5. PHP

Contenido

1. Introducción
2. Fundamentos de PHP
   - Tipos básicos y variables
   - Operadores
   - Estructuras de Control
   - Arrays
   - Fechas
   - Funciones
   - Clases
3. Variables predefinidas en PHP
   - Parámetros enviados en la petición
   - Cookies
   - Sesiones
4. Conexión con bases de datos

Universidad de Huelva

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA, SISTEMAS INFORMÁTICOS Y AUTOMÁTICA

Introducción

- Características de PHP
  - PHP - (acrónimo de "PHP: Hypertext Preprocessor") es un lenguaje de script del lado del servidor.
  - Código incluido dentro de un documento xHTML (XML) que añade dinamismo a una página web estática.
  - Open Source (código libre)
  - Funcionamiento
    - El navegador del cliente solicita el documento PHP.
    - Llega la solicitud al servidor que localiza el documento, lanza el intérprete de PHP y ejecuta el código.
    - Como resultado de la ejecución se genera un documento xHTML y el servidor transfiere este documento al cliente que lo muestra en el navegador.

Introducción

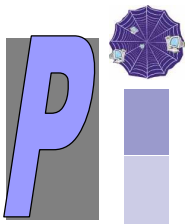
- Características de PHP
  - Inclusión en el código xHTML
    - **<?php … ?>**
    - **<? … ?>**
    - **<script languaje="php"> … </script>**
  - Ejemplos

```
<?php // Ejemplo1
   phpinfo();
?>
```

La función ***echo*** devuelve la cadena pasada por parámetros.
Esta función es muy limitada, es posible usar la función ***printf***, igual que C.

printf('cadena', vars)

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>

<body>
<h1>Mi Primer Ejemplo</h1>
<?php // Ejemplo2
  echo '<p>Hola Mundo</p>';
?>
</body>
</html>
```

Tema 5. PHP

Contenido

Universidad de Huelva
DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA, SISTEMAS INFORMÁTICOS Y AUTOMÁTICA

Fundamentos de PHP

- Tipos de Datos

  Enteros (int, long, numeric) Números enteros positivos y negativos
  Real (float, double, real, numeric) Números decimales o de coma flotante
  Boolean (bool) True o False
  Cadena (string) Cadenas de texto
  Array (array) Tipo especial de colección de valores
  Objetos (object) Tipo especial de dato

- Variables
  - No es necesario declararlas y no tiene un tipo determinado
  - Comienzan con el signo dólar ($)

```
<?php // Ejemplo3
  $a = 1;
  $b = 3.34;
  $c = "Hola Mundo";
  echo $a . '<br>' . $b . '<br>'  . $c;
?>
```

  - Constantes: define(*nombre*, *valor*);

NOTA: Funciones típicas
is_xxxx**(***variable***)** : True / False
Donde xxxx tipo de dato.
P.ej. is_double

gettype(*variable***)** : Devuelve tipo
settype(*variable***,'***tipoVariable***')**: Cambia el tipo de la variable.
(También se usa casting)

Fundamentos de PHP

- Operadores

Aritméticos

Operador	Nombre	Ejemplo
+	Suma	5 + 6
-	Resta	7 - 9
*	Multiplicación	6 * 3
/	División	4 / 8
%	Módulo	7 % 2
++	Suma 1	$a++
--	Resta 1	$a--

Lógicos

Operador	Nombre	Ejemplo
&&	Y	(7>2) && (2<4)
and	Y	(7>2) and (2<4)
||	O	(7>2) || (2<4)
or	O	(7>2) or (2<4)
!	No	! (7>2)

Comparación

Operador	Nombre	Ejemplo
==	Igual	$a == $b
!=	Distinto	$a != $b
<	Menor que	$a < $b
>	Mayor que	$a > $b
<=	Menor o igual	$a <= $b
>=	Mayor o igual	$a >= $b

Asignación: Igual (=)

Concatenación: Punto (.)

Fundamentos de PHP

- Estructuras de Control
  - If-else

```
<?php
if (expr)
    sentencia
?>
```

```
<?php
if ($a > $b) {
    print "a es mayor que b";
} else {
    print "a NO es mayor que b";
}
?>
```

  - Switch

```
switch ($i) {
    case 0:
        print "i equals 0";
        break;
    case 1:
        print "i equals 1";
        break;
    case 2:
        print "i equals 2";
        break;
}
?>
```

  - While

```
while (expr) sentencia
```

  - Do-while

```
<?php
$i = 0;
do {
    print $i;
} while ($i>0);
?>
```

  - For

```
for (expr1; expr2; expr3) sentencia
```

  - Foreach

```
foreach(expresion_array as $value) sentencia
foreach(expresion_array as $key => $value) sentencia
```

```
<?php
$a = array("uno", "dos", "tres");
foreach ($a as $v) { echo "<p>Valor: $v</p>"; }
foreach( $a as $k => $v ) {   echo "Indice: $k; Valor: $v <br>"; }
?>
```

  - Continue
  - Break

Fundamentos de PHP

- Estructuras de Control
  - Ejemplo

```
<?php

$a = array ('Uno', 'Dos', 'Tres', 'next', 'Cuatro', 'Cinco', 'stop', 'Seis');

foreach ($a as $val) {

   if ($val == 'stop') {
      break;
   }
   if ($val == 'next') {
      continue;
   }

   echo "$val<br>\n";
}

?>
```

Resultado

Uno
Dos
Tres
Cuatro [se salta el valor *next*]
Cinco [El bucle finaliza con el valor *stop*]

Fundamentos de PHP

- Arrays
  - Lista de variables estructurada no necesariamente del mismo tipo.
    Indexados (comienzan en 0) ó Asociativos
  - Declaración
    - De forma implicita

```
$a[ ] = “primero”;
$a[ ] = “segundo”;
```

```
clave
  ↓
$a[“Uno”] = “primero”;
$a[“Dos”] = “segundo”;
```

    - Mediante el constructor *array*

```
$a = array (‘primero’, segundo’);
```

```
$a = array(‘Uno'=>‘primero',‘Dos'=>‘segundo');
```

  - Utilización

```
$a[0]  …  $a[1]

for($i=0; $i<max; $i++)  echo $a[$i]
```

```
$a[‘Uno’]  …  $a[‘Dos’]

foreach( $a as $dato) echo $dato

while ($dato = each($a)) {
      echo $dato[0];   // clave. Uno, Dos
      echo $dato[1];   // valor: primero, …
```

La función **each**(*array*) devuelve una matriz con dos posiciones, en 0 almacena la clave o índice y en 1 el valor.

Fundamentos de PHP

- Arrays: Principales funciones [más información en http://www.php.net/manual/es/ref.array.php]
  - Recorrido
    reset(*array*) : Se posiciona en el primer elemento
    end(*array*) : Se posiciona en el último elemento
    next(*array*) : Se posiciona en el siguiente elemento
    prev(*array*) : Se posiciona en el elemento anterior
    current(*array*) : Devuelve el contenido de la posición actual o false
    cont (*array*) : Devuelve el número de elementos. Similar es sizeof (*array*)
  - Otras funciones
    array_push(matriz,variable1,variableN): Añade elementos al final del array
    array_unshift(matriz,variable1,variableN): Añade elementos al principio del array

    array_shift(matriz) : Elimina el primer elemento de la matriz
    array_pop(matriz) : Elimina el último elemento de la matriz

    sort(matriz) : Ordenación ascendente (matriz indexada)
    rsort(matriz) : Ordenación descendente (matriz indexada)
    asort(matriz) : Ordenación ascendente por valor (matriz asociativa)
    arsort(matriz): Ordenación descendente por valor (matriz asociativa)
    ksort(matriz) : Ordenación ascendente por clave (matriz asociativa)
    krsort(matriz) : Ordenación descendente por clave (matriz asociativa)

Fundamentos de PHP

- Cadenas

Una cadena es una secuencia de caracteres comprendidos entre:

- Comillas simples ' ‘ ‘esto es una cadena: $i’
- Comillas dobles " " “esto es otra cadena: $i”

- Principales funciones

echo ***cadena*** : imprime la cadena

printf(cadena_formato, valores) : imprime la cadena sustituyendo los valores

strlen(cadena) : Devuelve el número de caracteres de la cadena.

substr(cadena, inicio, longitud). Devuelve una subcadena de otra, empezando por inicio y de longitud longitud.

chop(cadena). Elimina los saltos de línea y los espacios finales de una cadena.

strpos(cadena1, cadena2). Busca cadena2 en cadena1 devolviendo en índice.

str_replace(cadena1, cadena2, texto). Cambia cadena1 por cadena2 en el texto.

ltrim(cadena), rtrim(cadena) y trim(cadena) : Elimina los espacios en blanco

strtolower(cadena) : Convierte la cadena a minúsculas

strtoupper(cadena); Convierte la cadena a mayúsculas

Fundamentos de PHP

- Fechas: PHP permite el manejo de fechas y horas.

getdate() : Devuelve un array asociativo con la información de la fecha y hora.

Índices: hours, minutes, seconds, mday, wday, weekday, mon, month, year, yday

Esta función acepta un parámetro que indica los segundos desde 01/01/1970

time() : Devuelve el número de segundos transcurridos desde 01/01/1970

mktime(hr, mt, sg, m, d, a) : Devuelve los segundos trascurridos desde el 01/01/1970

date(formato_fecha) : Devuelve la fecha actual en una cadena formateada, según:

d	Día del mes, 2 dígitos con ceros iniciales	*01* a *31*
D	Una representación textual de un día, tres letras	*Mon* a *Sun*
j	Día del mes sin ceros iniciales	*1* a *31*
l	Una representación textual completa del día de la semana	*Sunday* a *Saturday*
z	El día del año (comenzando en 0)	*0* a *365*
F	Una representación textual completa de un mes	*January* a *December*
m	Representación numérica de un mes, con ceros iniciales	*01* a *12*
M	Una representación textual corta de un mes, tres letras	*Jan* a *Dec*
n	Representación numérica de un mes, sin ceros iniciales	*1* a *12*
t	Número de días en el mes dado	*28* a *31*
Y	Una representación numérica de un año, 4 dígitos	Ejemplos: *1999* o *2003*
g	formato de 12-horas de una hora sin ceros iniciales	*1* a *12*
G	formato de 24-horas de una hora sin ceros iniciales	*0* a *23*
h	formato de 12-horas de una hora con ceros iniciales	*01* a *12*
H	formato de 24-horas de una hora con ceros iniciales	*00* a *23*
i	Minutos con ceros iniciales	*00* a *59*
s	Segundos, con ceros iniciales	*00* a *59*

Fundamentos de PHP

- Funciones
  - Declaración
    function *nombre*(*parámetros*)**{**
    Sentencias
    }
  - Llamada
    nombre(*Val_parámetros*)**;**
  - Valor de retorno (return)
  - Parámetros
    - por valor
    - por referencia: &
    - por defecto
  - Funciones:
    func_num_args() : Numero de parámetros
    func_get_args(i) : Parámetro i
    func_get_args () : Array de parámetros

Ejemplos

```
function sumar($v1,$v2){
    $ suma=$v1+$v2
    echo $v1."+".$v2."=".$suma;
}

sumar(6,4);

function incrementa($v1){
    return $v1+1;
}
A = incrementa(2);


function cambiar(&$v1){
    $v1=5;
}


function opera( $v1, $v2=2) {
    return $v1*$v2;
}
```


Fundamentos de PHP

- Código desde un fichero

require('fichero') : incluye y evalúa el archivo. Si se produce un problema genera un error de sistema E_ERROR

include("fichero") : incluye y evalúa el archivo. Si se produce un problema genera un aviso del sistema E_WARNING

```
// funciones.php
function unaFuncion($a) { …

function otraFuncion( ) { …

etc…
```

```
…
<html>
<head> … </head>
<body>
…
<? php
require ('funciones.php');

unaFuncion(5);
…
otraFuncion( );
…
?>
…
</body>
</html>
```

Fundamentos de PHP

- Clases
  - Declaración

```
class nombre_clase  {
    // atributos
    var nombre_atributo;

    // métodos (acceso a atributos this-> )
    function nomb_metodo( parámetros) {
            cuerpo
    }
}
```

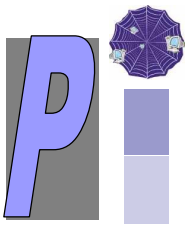
  - Instanciación de objetos

```
$obj = new nombre_clase();
```

  - Invocación de métodos (->)

```
$obj -> nomb_metodo (val_parametros)
```

```
class num {

  var $numero;

  function num() {  // CONSTRUCTOR
    $this->numero=10;
  }

  function mas($i=0) {
    $this->numero=$this->numero+$i;
  }

  function valor() {
    return $this->numero;
  }
}


$n = new num();
$n->mas(5);
echo "El valor es " . $n->valor
```

Tema 5. PHP

Contenido

Universidad de Huelva
DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA, SISTEMAS INFORMÁTICOS Y AUTOMÁTICA

Variables predefinidas en PHP

- Principales variables globales [ver phpinfo()]

  A partir de la versión 4.1

  - $_SERVER: Array asociativo con información sobre el servidor web
  - $_ENV: Array asociativo con información sobre el entorno de ejecución
  - $_GET: Array asociativo con la información enviada por el usuario, si la petición ha sido tipo GET.
  - $_POST: Array asociativo con la información enviada por el usuario, si la petición ha sido tipo POST.
  - $_SESSION: Array asociativo con las variables de la sesión.
  - $_COOKIE: Array asociativo con las cookies enviadas en la petición.
  - $_FILES: Array asociativo con la información de cada archivo enviado
  - $_REQUEST: Array asociativo con la concatenación de $_GET, $_POST, $_COOKIE Y $_FILES.

Tabla equivalencia entre versiones de PHP

A partir de 4.1.0	Versiones anteriores
$_SERVER	$HTTP_SERVER_VARS
$_ENV	$HTTP_ENV_VARS
$_GET	$HTTP_GET_VARS
$_POST	$HTTP_POST_VARS
$_SESSION	$HTTP_SESSION_VARS
$_COOKIE	$HTTP_COOKIE_VARS

Variables predefinidas en PHP

- Parámetros enviados en una petición

$_GET, $_POST ó $_REQUEST

- Peticiones GET

[Directamente en el navegador

http://*servidor:puerto/ruta/fichero.php*?var1=val1&var2=val2 o desde un enlace]

```
<%php
…
echo $_GET[var1];   // mostraría val1
echo $_GET[var2];   // mostraría val2
…
%>
```

Variables predefinidas en PHP

- Parámetros enviados en una petición
  $_GET, $_POST ó $_REQUEST

  - Peticiones POST

```
<form action="ruta/fichero.php" method="POST" >
...
   Var1: <input type="text" name="var1"  … ><br />
   Var2: <input type="text" name="var2"  … > <br />
   <input type="submit" value="Enviar"  … >
...
</form>
```


```
<%php     // fichero.php
...
echo $_POST[var1];   // mostraría val1
echo $_POST[var2];   //  mostraría val2
...
%>
```


Variables predefinidas en PHP

- Cookies

  Array $_COOKIE y función setcookie();

  - Asignación de Cookies (forma parte de la cabecera de la petición)

    setcookie(nombre, valor, caducidad, ruta, dominio, segura);

    Ejemplo: setcookie("contador", $cont, time()+60)

  - Acceso a las cookies

    $_COOKIE['nomb_cookie']

    Ejemplo: $_COOKIE['contador']

Variables predefinidas en PHP

- Cookies
  - Ejemplo:

```
<?php  // ATENCIÓN: Debe ser lo primero incluido en el fichero
$cont = $_COOKIE['contador'];
$cont++;
setcookie('contador', $cont, time()+60);
?>
<html>
<head> <title>Ejemplo Cookies</title> </head>
<body>
        <h1>Ejemplo con una Cookie contador</h1>
        <h2>Número de accesos:
        <?php echo $cont; ?>      </h2>
</body>
</html>
```


Variables predefinidas en PHP

- Sesiones

  Principales funciones

  session_start() : Inicia una nueva sesión o verifica la actual. Esta función debe estar incluida en todos los scripts que usen sesiones.

  session_register('variable') : establece la variable indicada por parámetros como una variable de sesión que será mantenida durante la sesión. La función inversa es session_unregister('variable');

  session_unset() : Elimina de la sesión todas las variables.

  session_destroy() : finaliza una sesión

  session_name([nombre]) devuelve el nombre de la sesión actual. Si se especifica un nombre como parámetro lo asigna como nombre de la sesión

  Array $_SESSION

  Las variables establecidas como de la sesión son accedidas mediante: $_SESSION

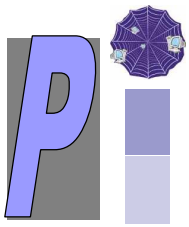
  Ej. session_register('contador') $_SESSION['contador']

Variables predefinidas en PHP

- Sesiones

  Ejemplo

```
<?php
        session_start();
        if (!isset($_SESSION['count'])) {
                $_SESSION['count'] = 0;
        } else {
                $_SESSION['count']++;
}
?>
<html> <head> … </head>

<?php if ($_SESSION['count'] ==0) { ?>
        <p>Es la primera vez que accedes</p>     ← Esto es código HTML
<?php }  else { ?>
        <p>ya has accedido más veces</p>
<?php }  ?>

</html>
```

Tema 5. PHP

Contenido

Universidad de Huelva

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA, SISTEMAS INFORMÁTICOS Y AUTOMÁTICA

Conexión Base datos

- MySQL
  - Conectando con la base de datos

int mysql_connect (string host, string usuario, string password)

Conecta con el servidor de base de datos indicado, devolviendo el enlace de la conexión o falso en caso de error.

int mysql_select_db (string bd, [int enlace])

Conecta con la base de datos bd, el parámetro enlace es opcional si no se indica se utiliza el obtenido en la última llamada a mysql_connect. Devuelve cierto si tiene éxito y falso en caso de error.

Ejemplo:

```
$enlace = mysql_mysql_connect("mi servidor","usario","password") or die("Fallo en la conexión" . mysql_error());
mysql_select_db("mi base de datos") or die("Fallo al seleccionar la BD" . msyql.error());
```


Conexión Base datos

- MySQL
  - Estableciendo la sentencia a ejecutar sobre la base de datos

Para poder ejecutar alguna sentencia sobre la base de datos es necesario tener permisos para ello.

Las sentencias básicas son:

- SELECT campos FROM basedatos WHERE condiciones
- INSERT INTO basedatos VALUES (valoresCampos)
- DELETE FROM basedatos WHERE condiciones

Ejemplo.

```
$sql1 = " SELECT * FROM clientes WHERE dni='11111111' " ;
$sql2 = "INSERT INTO cliente VALUES ('11111111','PEPE', 30)" ;
$sql3 = "DELETE FROM cliente WHERE dni='11111111' " ;
```

Conexión Base datos

- MySQL
  - Ejecución de la sentencia

resource **mysql_query** (string cta, [int enl])
Ejecuta la consulta *cta* sobre la base de datos cuyo enlace de conexión es *enl,* si se omite se utiliza el obtenido en la última llamada a mysql_connect.
Devuelve un resource con el resultado o falso si ha fallado.

Ejemplos:

```
$resultado = mysql_query( $sql1);
// Tratamiento del resource para mostrar los datos consultados

$resultado = mysql_query( $sql2); // siendo $sql2 un INSERT
if ($resultado) echo "Inserción realizada correctamente" ;
else  die("La inserción no ha podido realizarse por: ". mysql_error());

$resultado = pq_query($sql3); // siendo $sql2 un DELETE
if ( | $resultado) die("El borrado no ha podido realizarse por: "mysql_error() ;
else  echo "Se han eliminado " . mysql_affected_rows()  . "registros";
```


Conexión Base datos

- MySQL
  - Recorrido de resource resultado en un SELECT

array **mysql_fetch_array** (int resultado, [int tipo_de_resultado])
Devuelve un array indexado (comenzando en 0) y/o asociativo (campos de la base de datos) del siguiente resultado del resource resultado o falso si ya no quedan más.

El array devuelto dependerá de valor del parámetro tipo_de_resultado:

- MYSQL_ASSOC Las columnas son devueltas en el array usando el nombre del campo como índice.
- MYSQL_BOTH Las columnas son devueltas en el array teniendo tanto un índice numérico como un índice correspondiente al nombre del campo. (VALOR POR DEFECTO)
- MYSQL_NUM Las columnas son devueltas en el array teniendo un índice numérico a los campos. Este índice comienza en 0, el primer campo del resultado.

MYSQL_BOTH es el parámetro por defecto si se omite tipo_de_resultado.

Conexión Base datos

- MySQL

array **array_fetch_row** (int resultado)

Devuelve un array indexado del siguiente resultado del resource resultado o falso si ya no quedan más.

object **array_fetch_object** (int resultado)

Devuelve un objeto, cuyos atributos son los campos de la base de datos, del siguiente elemento del resource resultado o falso si ya no quedan más.

Ejemplos. Suponer una base de datos con los campos dni, nombre y edad

```
$r1 = mysql_fetch_array ( $resultado );
echo "Bienvenido $r1['nombre']";

for ($i=1; i<mysql_num_rows( $resultado )-1; i++)  { $r2 = mysql_fetch_row($resultado); …

$obj = mysql_fetch_object($resultado);
echo $obj->nombre . "con" . obj->dni . "tiene" . $obj->edad . "años" ;
```


Conexión Base datos

- MySQL
  - Liberando recursos y cerrando la conexión

bool **mysql_free_result** (resource resultado)
Libera los recursos que utiliza un resource resultado de una consulta previa.

bool **mysql_close** (int enlace)
Cierra una conexión con el servidor de base de datos.

Ejemplo

```
mysql_free_result($resultado);

mysql_close($enlace);
```

Conexión Base datos

- MySQL

int **mysql_connect** (string host, string usuario, string password) : Conexión con una servidor de base de datos
int **mysql_select_db** (string bd): Conexión con una base de datos
bool **mysql_close** (int enlace) : Cerrar conexión con un servidor de base de datos

resource **mysql_query** (string query) : Ejecuta una sentencia sobre una base de datos.

array **mysql_fetch_array** (int resultado) : Array de una fila de la consulta
array **mysql_fetch_row** (int resultado) : Array de una fila de la consulta
object **mysql_fetch_object** (int resultado) : Objeto de una fila de la consulta

int **mysql_num_fields** (int resultado) : Número de campos obtenidos en la consulta
int **mysql_num_rows** (int resultado) : Número de filas de la consulta
int **mysql_affected_rows** ( ): Filas de la base de datos afectadas de la última sentencia INSERT, DELETE, UPDATE ejecutada

bool **mysql_free_result** (resource resultado) : Libera los recursos de una consulta.

string **mysql_error** (): Devuelve el texto del mensaje de error de la última operación MySQL

Conexión Base datos

- MySQL

```php
<?php
// Conexion, seleccion de base de datos
$enlace = mysql_connect('host_mysql', 'usuario_mysql', 'contrasenya_mysql')
    or die('No pudo conectarse : ' . mysql_error());
echo 'Conexi&oacute;n exitosa';
mysql_select_db('mi_base_de_datos') or die('No pudo seleccionarse la BD.');

// Realizar una consulta SQL
$consulta  = 'SELECT * FROM mi_tabla';
$resultado = mysql_query($consulta) or die('La consulta fall&oacute;: ' . mysql_error());

// Impresion de resultados en HTML
echo "<table>\n";
while ($linea = mysql_fetch_array($resultado, MYSQL_ASSOC)) {
    echo "\t<tr>\n";
    foreach ($linea as $valor_col) {
        echo "\t\t<td>$valor_col</td>\n";
    }
    echo "\t</tr>\n";
}
echo "</table>\n";

// Liberar conjunto de resultados
mysql_free_result($resultado);

// Cerrar la conexion
mysql_close($enlace);
?>
```