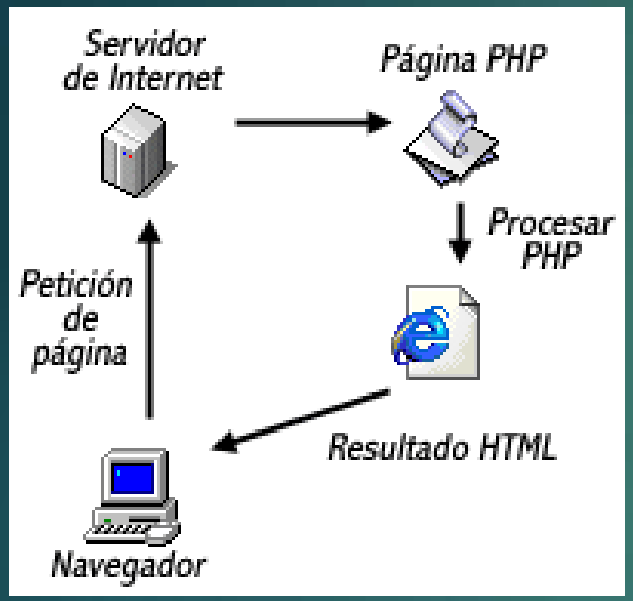
Manual PHP Básico

CAMILO CARTAGENA

Conceptos básicos



EL LENGUAJE PHP ES UN LENGUAJE DE PROGRAMACIÓN DE ESTILO CLÁSICO, CON ESTO QUIERO DECIR QUE ES UN LENGUAJE DE PROGRAMACIÓN CON VARIABLES, SENTENCIAS CONDICIONALES, BUCLES, FUNCIONES.... NO ES UN LENGUAJE DE MARCAS COMO PODRÍA SER HTML, XML O WML. ESTÁ MAS CERCANO A JAVASCRIPT O A C, PARA AQUELLOS QUE CONOCEN ESTOS LENGUAJES.

PERO A DIFERENCIA DE JAVA O JAVASCRIPT QUE SE EJECUTAN EN EL NAVEGADOR, PHP SE EJECUTA EN EL SERVIDOR, POR ESO NOS PERMITE ACCEDER A LOS RECURSOS QUE TENGA EL SERVIDOR COMO POR EJEMPLO PODRÍA SER UNA BASE DE DATOS. EL PROGRAMA PHP ES EJECUTADO EN EL SERVIDOR Y EL RESULTADO ENVIADO AL NAVEGADOR. EL RESULTADO ES NORMALMENTE UNA PÁGINA HTML PERO IGUALMENTE PODRÍA SER UNA PAGINA WML.

AL SER PHP UN LENGUAJE QUE SE EJECUTA EN EL SERVIDOR NO ES NECESARIO QUE SU NAVEGADOR LO SOPORTE, ES INDEPENDIENTE DEL NAVEGADOR, PERO SIN EMBARGO PARA QUE SUS PÁGINAS PHP FUNCIONEN, EL SERVIDOR DONDE ESTÁN ALOJADAS DEBE SOPORTAR PHP.

Conceptos básicos - Nuestro primer PHP

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
//Parte de HTML normal.
<BR><BR>
<?php
  echo "Parte de PHP<br>";
  for($i=0;$i<10;$i++)
  {
    echo "Linea ".$i."<br>";
  }
?>
</body>
</html>
```

La ventaja que tiene PHP sobre otros lenguajes de programación que se **ejecutan en el servidor** (como podrían ser los script CGI Perl), es que nos permite intercalar las sentencias PHP en las paginas HTML, es un concepto algo complicado de entender si no se ha visto nunca como funciona unas paginas PHP o ASP.

Vamos a ver un ejemplo sencillo para comprenderlo mejor. En azul está el código HTML y en rojo el código PHP. Seguiremos este criterio durante todo el manual.

Conceptos básicos - Variables

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $a = 1;
  $b = 3.34;
  $c = "Hola Mundo";
  echo $a,"<br>",$b,"<br>",$c;
?>
</body>
</html>
```

Una variable es un contenedor de información, en el que podemos meter números enteros, números decimales, carácteres... el contenido de las variables se puede leer y se puede cambiar durante la ejecución de una página PHP.

En PHP todas las variables comienzan con el símbolo del dólar $ y no es necesario definir una variable antes de usarla. Tampoco tienen tipos, es decir que una misma variable puede contener un número y luego puede contener carácteres.

Conceptos básicos - Aritméticos

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $a = 8;
  $b = 3;
  echo $a + $b,"<br>";
  echo $a - $b,"<br>";
  echo $a * $b,"<br>";
  echo $a / $b,"<br>";
  $a++;
  echo $a,"<br>";
  $b--;
  echo $b,"<br>";
?>
</body>
</html>
```

Los operadores de PHP son muy parecidos a los de C y JavaScript, si usted conoce estos lenguajes le resultaran familiares y fáciles de reconocer.

Estos son los operadores que se pueden aplicar a las variables y constantes numérico.

Conceptos básicos - Comparación

LOS OPERADORES DE COMPARACIÓN SON USADOS PARA COMPARAR VALORES Y ASÍ PODER TOMAR DECISIONES.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $a = 8;
  $b = 3;
  $c = 3;
  echo $a == $b,"<br>";
  echo $a != $b,"<br>";
  echo $a < $b,"<br>";
  echo $a > $b,"<br>";
  echo $a >= $c,"<br>";
  echo $b <= $c,"<br>";
?>
</body>
</html>
```

Conceptos básicos - Lógicos

LOS OPERADORES LÓGICOS SON USADOS PARA EVALUAR VARIAS COMPARACIONES, COMBINANDO LOS POSIBLES VALORES DE ESTAS.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $a = 8;
  $b = 3;
  $c = 3;
  echo ($a == $b) && ($c > $b),"<br>";
  echo ($a == $b) || ($b == $c),"<br>";
  echo !($b <= $c),"<br>";
?>
</body>
</html>
```

Conceptos básicos - Condicionales

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $a = 8;
  $b = 3;
  if ($a < $b)
  {
    echo "a es menor que b";
  }
  else
  {
    echo "a no es menor que b";
  }
?>
</body>
</html>
```

LAS SENTENCIAS CONDICIONALES NOS PERMITEN EJECUTAR O NO UNAS CIERTAS INSTRUCCIONES DEPENDIENDO DEL RESULTADO DE EVALUAR UNA CONDICIÓN. LAS MÁS FRECUENTES SON LA INSTRUCCIÓN IF Y LA INSTRUCCIÓN SWITCH.

SENTENCIA IF ... ELSE

Conceptos básicos - Bucles

LOS BUCLES NOS PERMITEN **ITERAR CONJUNTOS DE INSTRUCCIONES**, ES DECIR REPETIR LA EJECUCIÓN DE UN CONJUNTO DE INSTRUCCIONES MIENTRAS SE CUMPLA UNA CONDICIÓN.

SENTENCIA WHILE

```
<!-- Manual de PHP de WebEstilo.com -->
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
Inicio<BR>
<?php
  $i=0;
  while ($i<10)
  {
    echo "El valor de i es ", $i,"<br>";
    $i++;
  }
?>
Final<BR>
</body>
</html>
```

Conceptos básicos - Salida

HASTA AHORA HEMOS USADO LA INSTRUCCIÓN ECHO PARA REALIZAR SALIDA A PANTALLA, ESTA INSTRUCCIÓN ES BASTANTE LIMITADA YA QUE NO NOS PERMITE FORMATEAR LA SALIDA. EN ESTA PÁGINA VEREMOS LA INSTRUCCIÓN PRINTF QUE NOS DA MUCHA MÁS POTENCIA.

SENTENCIA PRINTF

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $var="texto";
  $num=3;
  printf("Puede fácimente intercalar <b>%s</b> con números
<b>%d</b> <br>",$var,$num);

  printf("<TABLE BORDER=1 CELLPADDING=20>");
  for ($i=0;$i<10;$i++)
  {
    printf("<tr><td>%10.d</td></tr>",$i);
  }
  printf("</table>");
?>
</body>
</html>
```

Conceptos básicos - Salida

HASTA AHORA HEMOS USADO LA INSTRUCCIÓN ECHO PARA REALIZAR SALIDA A PANTALLA, ESTA INSTRUCCIÓN ES BASTANTE LIMITADA YA QUE NO NOS PERMITE FORMATEAR LA SALIDA. EN ESTA PÁGINA VEREMOS LA INSTRUCCIÓN PRINTF QUE NOS DA MUCHA MÁS POTENCIA.

SENTENCIA PRINTF

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  $var="texto";
  $num=3;
  printf("Puede fácimente intercalar <b>%s</b> con números <b>%d</b>
<br>",$var,$num);

  printf("<TABLE BORDER=1 CELLPADDING=20>");
  for ($i=0;$i<10;$i++)
  {
    printf("<tr><td>%10.d</td></tr>",$i);
  }
  printf("</table>");
?>
</body>
</html>
```

Manual PHP Avanzado

CAMILO CARTAGENA

Conceptos Avanzados - Manejo de cadenas

DADO EL USO DEL LENGUAJE PHP EL TRATAMIENTO DE CADENAS ES MUY IMPORTANTE, EXISTEN BASTANTES FUNCIONES PARA EL MANEJO DE CADENAS, A CONTINUACIÓN EXPLICAREMOS LAS MÁS USADAS.

1. STRLEN(CADENA). NOS DEVUELVE EL NÚMERO DE CARÁCTERES DE UNA CADENA.
2. SPLIT(SEPARADOR,CADENA). DIVIDE UNA CADENA EN VARIAS USANDO UN CARÁCTER SEPARADOR.
3. SPRINTF(CADENA DE FORMATO, VAR1, VAR2...). FORMATEA UNA CADENA DE TEXTO AL IGUAL QUE PRINTF PERO EL RESULTADO ES DEVUELTO COMO UNA CADENA.
4. SUBSTR(CADENA, INICIO, LONGITUD). DEVUELVE UNA SUBCADENA DE OTRA, EMPEZANDO POR INICIO Y DE LONGITUD LONGITUD.
5. CHOP(CADENA). ELIMINA LOS SALTOS DE LÍNEA Y LOS ESPACIOS FINALES DE UNA CADENA.
6. STRPOS(CADENA1, CADENA2). BUSCA LA CADENA2 DENTRO DE CADENA1 INDICÁNDONOS LA POSICIÓN EN LA QUE SE ENCUENTRA.
7. STR_REPLACE(CADENA1, CADENA2, TEXTO). REEMPLAZA LA CADENA1 POR LA CADENA2 EN EL TEXTO.

Conceptos Avanzados - Manejo de cadenas

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php
  echo strlen("12345"),"<br>";

  $palabras=split(" ","Esto es una prueba");
  for($i=0;$palabras[$i];$i++)
    echo $palabras[$i],"<br>";

  $resultado=sprintf("8x5 = %d <br>",8*5);
  echo $resultado,"<br>";

  echo substr("Devuelve una subcadena de otra",9,3),"<br><br>";

  if (chop("Cadena \n\n ") == "Cadena")
    echo "Iguales<br><br>";

  echo strpos("Busca la palabra dentro de la frase", "palabra"),"<br><br>";

  echo str_replace("verde","rojo","Un pez de color verde, como verde es la hierba."),"<br>";

?>
</body>
</html>
```

Conceptos Avanzados - Funciones

EL USO DE FUNCIONES NOS DA LA CAPACIDAD DE AGRUPAR VARIAS INSTRUCCIONES BAJO UN SOLO NOMBRE Y PODER LLAMARLAS A ESTAS VARIAS VECES DESDE DIFERENTES SITIOS, AHORRÁNDONOS LA NECESIDAD DE ESCRIBIRLAS DE NUEVO.

OPCIONALMENTE PODEMOS PASARLE PARÁMETROS A LAS FUNCIONES QUE SE TRATARAN COMO VARIABLE LOCALES Y ASÍ MISMO PODEMOS DEVOLVER UN RESULTADO CON LA INSTRUCCIÓN RETURN VALOR; ESTO PRODUCE LA TERMINACIÓN DE LA FUNCIÓN RETORNANDO UN VALOR.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<?php

  function media_aritmetica($a, $b)
  {
    $media=($a+$b)/2;
    return $media;
  }

  echo media_aritmetica(4,6),"<br>";
  echo
media_aritmetica(3242,524543),"<br>";

?>
</body>
</html>
```

Conceptos Avanzados - Envío y recepción de dato

EL LENGUAJE PHP NOS PROPORCIONA UNA MANERA SENCILLA DE MANEJAR FORMULARIOS, PERMITIÉNDONOS DE ESTA MANERA PROCESAR LA INFORMACIÓN QUE EL USUARIO HA INTRODUCIDO.

AL DISEÑAR UN FORMULARIO DEBEMOS INDICAR LA PÁGINA PHP QUE PROCESARÁ EL FORMULARIO, ASÍ COMO EN MÉTODO POR EL QUE SE LE PASARÁ LA INFORMACIÓN A LA PÁGINA.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>
El nombre que ha introducido es: <?php echo
$_GET['nombre'] ?>
<br>
</body>
</html>
```

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>
Introduzca su nombre:
<FORM ACTION="procesa.phtml" METHOD="GET">
<INPUT TYPE="text" NAME="nombre"><BR>
<INPUT TYPE="submit" VALUE="Enviar">
</FORM>
</body>
</html>
```

Al pulsar el botón Enviar el contenido de cuadro de texto es enviado a la página que indicamos en el atributo ACTION de la etiqueta FORM.

En versiones anteriores a 4.2.0 PHP creaba una variable por cada elemento del FORM, esta variable creada tenía el mismo nombre que el cuadro de texto de la página anterior y el valor que habíamos introducido. Pero por razones de seguridad a partir de entonces para acceder a las variables del formulario hay que usar el array de parámetros $_POST[] o $_GET[] dependiendo del método usado para enviar los parámetros.

En este ejemplo se ha creado una entrada en el array $_GET[] con el índice 'nombre' y con el valor que haya introducido el navegante.

Conceptos Avanzados - Method GET y POST

En la página anterior hemos comentado que los datos de un formulario se envía mediante el método indicado en el atributo METHOD de la etiqueta FORM, los dos métodos posibles son GET y POST.

La diferencia entre estos dos métodos radica en la forma de enviar los datos a la página, mientras que el método GET envía los datos usando la URL, el método POST los envía por la entrada estándar STDIO.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de procesado de formularios</H1>
El nombre que ha introducido por GET es: <?php echo $_GET['nombre'],"
",$_GET['apellidos'] ?><br>
El nombre que ha introducido por POST es: <?php echo $_POST['nombre'],"
",$_POST['apellidos'] ?>
<br>
</body>
</html>
```

Conceptos Avanzados - Envió de emails

PHP NOS OFRECE LA POSIBILIDAD DE ENVIAR EMAILS DE UNA MANERA SENCILLA Y FÁCIL, PARA ELLO EL LENGUAJE NOS PROPORCIONA LA INSTRUCCIÓN MAIL( )

EN EL PARÁMETRO DESTINATARIO PONDREMOS LA DIRECCIÓN DE EMAIL A DONDE SE ENVIARÁ EL MENSAJE, EN EL PARÁMETRO TEMA EL TEMA O SUBJECT DEL MENSAJE Y EL PARÁMETRO TEXTO DEL MENSAJE EL CUERPO DEL MENSAJE EN FORMATO TEXTO PLANO.

EXISTE UNA SINTAXIS EXTENDIDA DE LA INSTRUCCIÓN MAIL( ) QUE NOS PERMITE AÑADIR INFORMACIÓN ADICIONAL A LA CABECERA DEL MENSAJE.

EN LA INFORMACIÓN DE CABECERA PODREMOS INCLUIR PARÁMETROS ADICIONALES AL MENSAJE COMO REPLY-TO:, FROM:, CONTENT-TYPE:... QUE NOS PERMITEN TENER UN MAYOR CONTROL SOBRE EL MENSAJE.

```
<html>
<head>
  <title>Ejemplo de PHP</title>
</head>
<body>
<H1>Ejemplo de envio de email</H1>
Introduzca su direccion de email:
<FORM ACTION="email.phtml"
METHOD="GET">
<INPUT TYPE="text"
NAME="direccion"><BR><BR>
Formato: <BR>
<INPUT TYPE="radio" NAME="tipo"
VALUE="plano" CHECKED> Texto plano<BR>
<INPUT TYPE="radio" NAME="tipo"
VALUE="html"> HTML<BR><BR>
<INPUT TYPE="submit" VALUE="Enviar">
</FORM>
</body>
</html>
```