

Guía Básica de PHP

¿Qué es PHP?

- Acrónimo de “Hypertext Preprocessor”
- Lenguaje interpretado de alto nivel
- Embebido en páginas HTML
- Sintaxis similar a C, Java y Perl
- Permite crear página dinámicas

¿Cómo funciona HTML?



Usuario

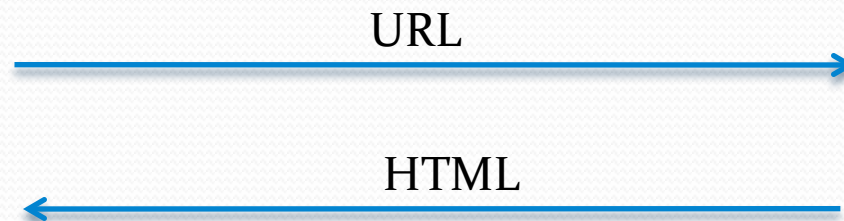
URL

HTML



Servidor

¿Cómo funciona PHP?



Servidor



PHP: Hola Mundo!

- Crear un archivo llamado hola.php

```
<html>
<body>
<h1>Ejemplo 1</h1>

<?php print( "hola mundo!" ); ?>

</body>
</html>
```

- <http://200.9.100.61/~<user>/hola.php>

PHP: Hola Mundo!

- Otro ejemplo usando variables

```
<html>
<body>
  <?php
    $texto = "hola mundo!";
  ?>
  <h1>Otro Ejemplo/h1>
  <?=$texto ?>
</body>
</html>
```

PHP: variables

- Se debe anteponer un signo \$
- No pueden llevar espacio
- No hay que declarar el tipo de dato
- Puedo redeclarar la variable usando otro tipo

```
<?php
```

```
$variable = "hola";  
$variable_2 = 'chao';  
$variable_2 = 5;
```

```
$entero = 5;
```

```
$decimal = 1.1232242;
```

PHP: operadores

- Los mismos que en java...

```
<?php  
  
$a = 20;  
$b = 3;  
  
print( $a + $b ); // suma: 23  
print( $a - $b ); // resta: 17  
print( $a * $b ); // multiplicación: 60  
print( $a / $b ); // división: 6.666666666666676  
print( $a % $b ); // resto: 2
```


PHP: Arreglos

- No se requiere declarar el largo

```
<?php  
  
$frutas = array( "naranja", "manzana", "durazno" );  
$frutas[3] = "pera";  
$frutas[] = "melón";    ← tomará automáticamente la pos. 4  
  
$frutas[23] = "papaya";
```

- Lo que tenemos es:

0	1	2	3	4	23
naranja	manzana	durazno	Pera	melón	papaya

PHP: Arreglos Asociativos

- En vez de usar números, usamos strings

```
$lista = array(  
    'fruta'    => 'manzana',  
    'verdura' => 'lechuga',  
    'numero'  => 4,  
    1         => 'algo',  
);  
$lista['cereal'] = 'maiz';
```

- Y ahora tenemos:

fruta	verdura	numero	1	cereal
manzana	lechuga	4	algo	maiz

PHP: Arreglos Asociativos

- Ejemplo

```
$lista = array(  
    'fruta'    => 'manzana',  
    'verdura' => 'lechuga',  
);  
  
print( 'una fruta es '.$lista['fruta'] );  
print( 'una verdura es '.$lista['verdura'] );
```

PHP: Iteradores

- Parecidos a java. Ej:

```
$lista = array( 'paula', 'javier', 'manuel', 'pedro' );  
  
for( $i=0; $i < count( $lista ); ++$i ){  
    print( 'El valor en la posición '.$i.' es '.$lista[$i].' <br />' );  
}
```

- Equivalente:

```
$lista = array( 'paula', 'javier', 'manuel', 'pedro' );  
  
$i = 0;  
while( $i < count( $lista ) ){  
    print( 'El valor en la posición '.$i.' es '.$lista[$i].' <br />' );  
    ++$i;  
}
```

PHP: Iteradores

- También es equivalente a:

```
$lista = array( 'paula', 'javier', 'manuel', 'pedro' );
```

```
$i = 0;
```

```
do {
```

```
    print( 'El valor en la posición '.$i.' es '.$lista[$i].' <br />' );
```

```
    ++$i;
```

```
} while( $i < count( $lista ) );
```

PHP: iteradores

- ¿Cómo recorro un arreglo asociativo?

```
$lista = array(  
    'javier' => 'hombre',  
    'paula'  => 'mujer',  
);
```

- Así:

```
foreach( $lista as $llave => $valor ){  
    print( $llave.' es un '.$valor.' <br />' );  
}
```

PHP: Condiciones

- `&&` ← y
- `||` ← o
- `==` ← igual a
- `!=` ← distinto a
- `<` ← menor que
- `>` ← mayor que
- `<=` ← menor o igual a
- `>=` ← mayor o igual a
- `===` ← ¿? (prueben `if( o == false )`)

PHP: Condiciones

```
$lista = array( 'javier', 'paula', 'manuel' );  
  
for ( $i =0; $i < count( $lista ); ++$i ){  
  
    if( $lista[$i] == 'javier' ){  
        print( "este es javier!" );  
  
    } else if( $lista[$i] == 'juan' ){  
        print( "y de donde salio Juan?" );  
  
    } else{  
        print( "El resto cae aquí" );  
    }  
  
}
```


Formularios

- ¿Cómo recibimos información del usuario?

```
<html>
<head>
  <title>Ejemplo de Formulario</title>
</head>
<body>
  <h1>Formulario de Ingreso</h1>
  <form method="get" action="validar.php">
    <input type="hidden" name="accion" value="entrar" />
    Nombre: <input type="text" name="nombre" size="40" /><br />
    Password: <input type="password" name="password" size="20" />
    <input type="submit" name="boton" value="Entrar" />
  </form>
</body>
</html>
```

Formularios

- Y nuestro archivo validar.php sería:

```
<?php
```

```
print( 'El nombre ingresado es ' . $_REQUEST['nombre'] );  
print( '<br />El apellido es ' . $_REQUEST['apellido'] );
```

Formularios

- Las variables enviadas las recibe PHP en el arreglo `$_REQUEST`
- Hay otras variables que se llenan dependiendo del atributo `method` del form:
 - `<form method="get" ...>` $\leftarrow$ `$_GET[]`
 - `<form method="post" ...>` $\leftarrow$ `$_POST[]`
- `$_REQUEST = $_COOKIES + $_POST + $_GET;`

Sesiones

- Sirven para pasar variables entre scripts
- Las variables viven al lado del servidor y no del usuario => no pueden ser alteradas por él
- Modo de uso:
 - Archivo 1:

```
<?php  
  
// iniciamos sesion  
session_start();  
  
// guardamos una variable  
$_SESSION['username'] = 'francisco';
```

Sesiones

- Archivo 2:

```
<?php

// iniciamos sesion
session_start();

// imprimimos lo que esta en la variable
print $_SESSION['username'];
```

- Para destruir una sesión se usa `session_destroy()`;

Funciones Útiles

- Manejo de Archivos

```
<?php
```

```
// meter el contenido de un archivo en un arreglo  
$lineas = file( $archivo );
```

```
// leer el contenido de un archivo en un solo string  
$todo = file_get_contents( $archivo );
```

```
// escribir el contenido de una variable en un archivo  
file_put_contents( 'archivo.txt', $variable );
```

```
// revisar si existe un archivo  
print( file_exists( $archivo ) ? "Si existe" : "nones..." );
```

Funciones Útiles

- Manejo de arreglos:

```
<?php
```

```
// separar un string en varios pedazos  
$nombres = "javier#andres#paula#carlos";  
$lista = explode( "#", $nombres );
```

```
// concatenar un arreglo con un string  
$lista = array( 'javier', 'andres', 'paula', 'carlos' );  
$nombres = implode( "#", $lista );
```

```
// sacar el ultimo elemento del arreglo  
array_pop( $lista );
```

```
//sacar el primer elemento del arreglo  
array_shift( $lista );
```

Funciones Útiles

- ¿Cómo Redirigo al usuario a otra página?
 - Se envía un header especial al browser llamado Location.

```
<?php

if( $_REQUEST['password'] != 'hola' ){
    header( 'Location: error.php' );
    exit();    ← es muy importante que vaya esto
} else{
    print( "Bienvenido !!" );
}
```


Referencias

- ¿Donde puedo consultar las funciones de PHP?

- <http://cl.php.net> ← castellano

- <http://www.php.net> ← ingles

Expresiones Regulares

- Sirven para calzar patrones.

```
$texto = "javier#villanueva#18/04/1981|paula#silva#05/09/1981#";  
  
if( ! preg_match( '/(.+)#(.+)#(\d{2})\|(\d{2})\|(\d{4})/', $texto, $calces ) ){  
    exit( 'No encontré nada' );  
}
```

- Lo que vamos a tener en \$calces es:

0	1	2	3	4	5
...	javier	villanueva	18	04	1981

Expresiones Regulares

- `^` ← calza al inicio de la linea
- `$` ← calza al final de la linea
- `.` ← Calza cualquier caracter excepto `\n`
- `()` ← agrupa un calce y lo guarda en `$c[1]`, `$c[2]`...
- `\d` ← cualquier digito
- `\D` ← cualquier no digito
- `\s` ← cualquier espacio(espacios, tabs)
- `\S` ← cualquier no espacio
- `[A-Z]` ← de la A a la Z
- `[^A-Z]` ← que no contenga de la A a la Z



Expresiones Regulares

- Documentación:
 - <http://cl.php.net/manual/es/reference.pcre.pattern.syntax.php>