

STRIKEGEEK



Taller PHP



Temario

1 - Introducción :

1.0 Instalación de XAMPP

1.1 Windows

1.2 Linux

2.0 Primeros pasos

2.1 Impresión en pantalla

2.2 Comentarios

3.0 Variables

3.1 Tipos de variables

3.2 Valores numéricos

3.3 Cadenas tipo string

3.4 Recibir datos por GET

3.5 Recibir datos por POST

4.0 Estructuras de control condicionales

4.1 If

4.2 Else

4.3 Elself

4.4 Switch

TUTOR

OKOL

Introducción.

Hola, en este taller hablaremos sobre programación en tecnología PHP. Mi nombre es Adrián N.M Hernandez y tengo 16 años y estaré a cargo de llevar este taller para la comunidad de StrikeGeek.

Este taller tratará de cómo podemos involucrarnos con este lenguaje de una manera más sencilla y practica; para eso se harán ejemplos en cada tema y tendrán su respectiva tarea.

Cualquier duda personal que puedan tener pueden enviarme un email a Okoltutos@hotmail.com.

Temario.

1.0 Instalación de XAMPP

1.1 Windows

1.2 Linux

2.0 Primeros pasos

2.1 Impresión en pantalla

2.2 Comentarios

3.0 Variables

3.1 Tipos de variables

3.2 Valores numéricos

3.3 Cadenas tipo string

3.4 Recibir datos por GET

3.5 Recibir datos por POST

4.0 Estructuras de control condicionales

4.1 If

4.2 Else

4.3 Elself

4.4 Switch

Instalación de XAMPP en Windows:

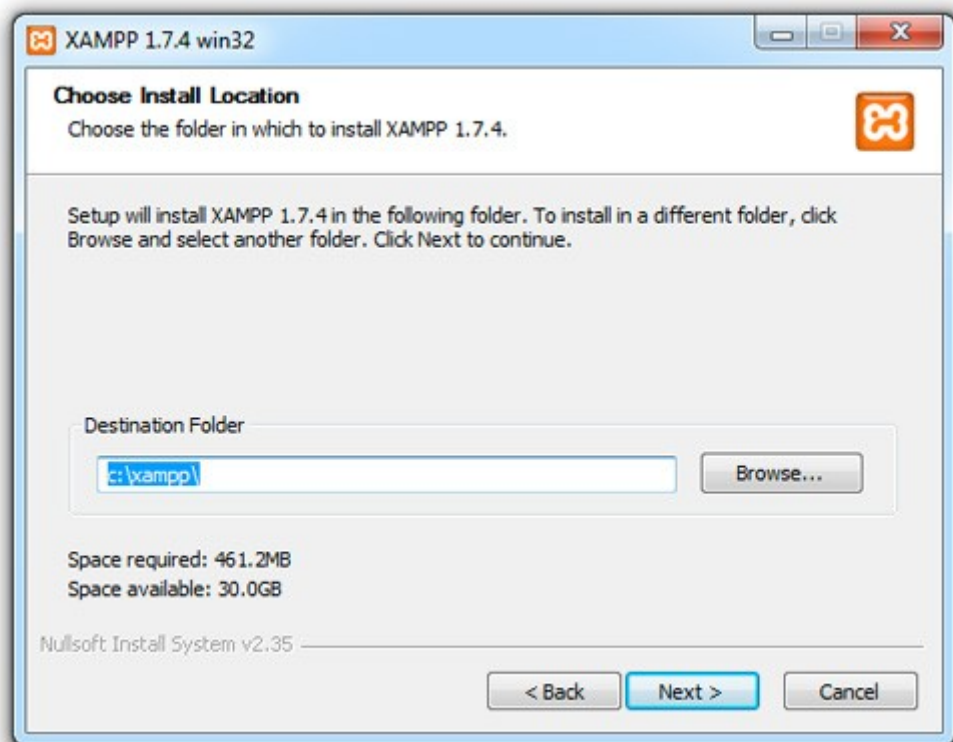
¿Qué es xampp?

XAMPP es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de X (para cualquiera de los diferentes sistemas operativos), Apache, MySQL, PHP, Perl.

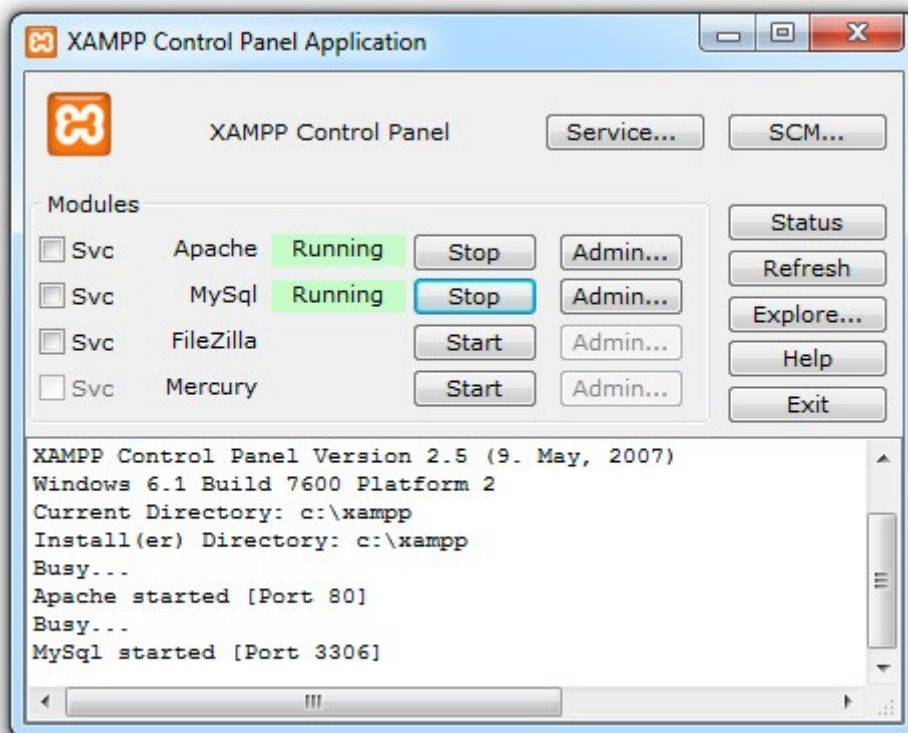
Bueno ahora que ya sabemos lo que es pasemos a instalar en Windows:

Paso 1: Descargamos xampp desde <http://www.apachefriends.org/en/xampp-windows.html>.

Paso 2: Damos una ruta de instalación



Después de que la instalación haya finalizado, encontrarás XAMPP en Inicio | Programas | XAMPP. Puede utilizar el panel de control de XAMPP para iniciar / detener todos los servidores y también instalar / desinstalar servicios.



*Recomendación:

Crear un nuevo directorio llamado Curso1 dentro de C:/Xampp/htdocs. Así para ejecutarlo desde la <http://localhost/Curso1/archivo.php>

Instalación de Xampp en Linux:

Paso 1: Abrimos una terminal y (nos situamos a el directorio Desktop ó Escritorio en el que preferimos que se descargue, con el comando `cd Carpeta`)*Opcional.

Paso2: Descargamos el Xampp (Para Linux es Lampp) desde <http://www.apachefriends.org/en/xampp-linux.html>

Paso3: Extraemos el archivo en la carpeta /opt de esta manera:

```
tar xvfz xampp-linux-1.8.1.tar.gz -C /opt
```

Paso 4: Iniciar lampp de esta manera:

```
/opt/lampp/lampp start
```

Los archivos los subiremos a el directorio htdocs desde /opt/lampp/htdocs (Les recomiendo crear una carpeta llamada Curso1

Advertencia: Xampp al instalar no tiene contraseñas ni seguridad, Para ponerle un poco de seguridad es con este comando:

```
/opt/lampp/lampp security
```

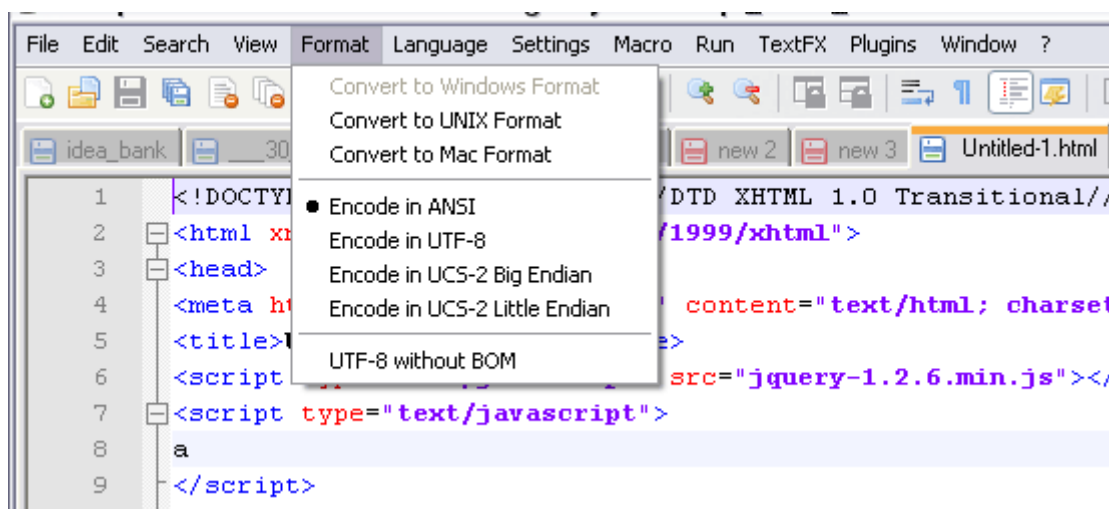
Editores de texto

Para comenzar a programar en cualquier lenguaje debemos tener un editor de texto que se adapte a nuestras necesidades, como por ejemplo la sintaxis remarcada.

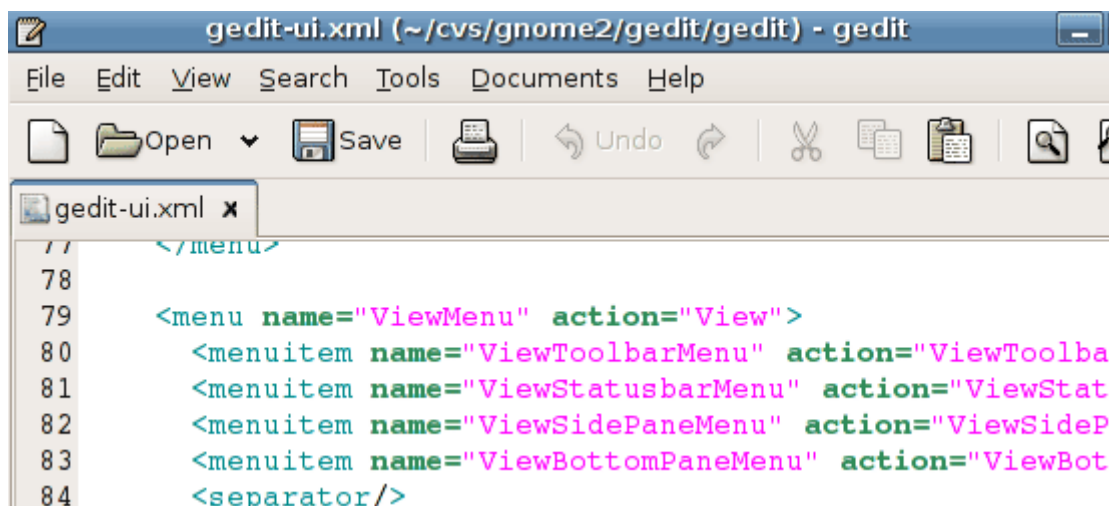
Unos buenos ejemplos de editores de texto pueden ser:

Notepad++ Descarga desde:

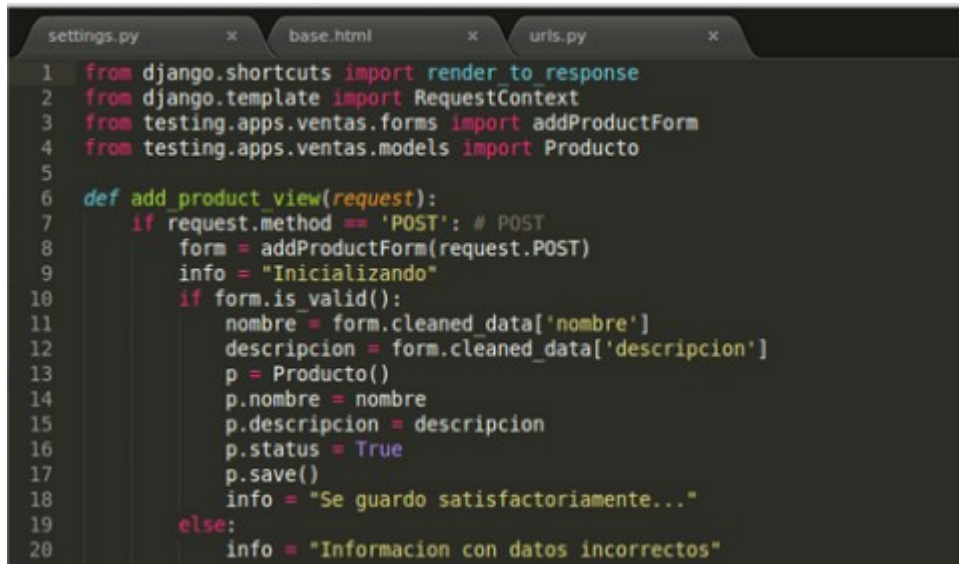
<http://notepad-plus-plus.org/download/v6.2.3.html>



Gedit: Viene instalado en cualquier distribución Linux.

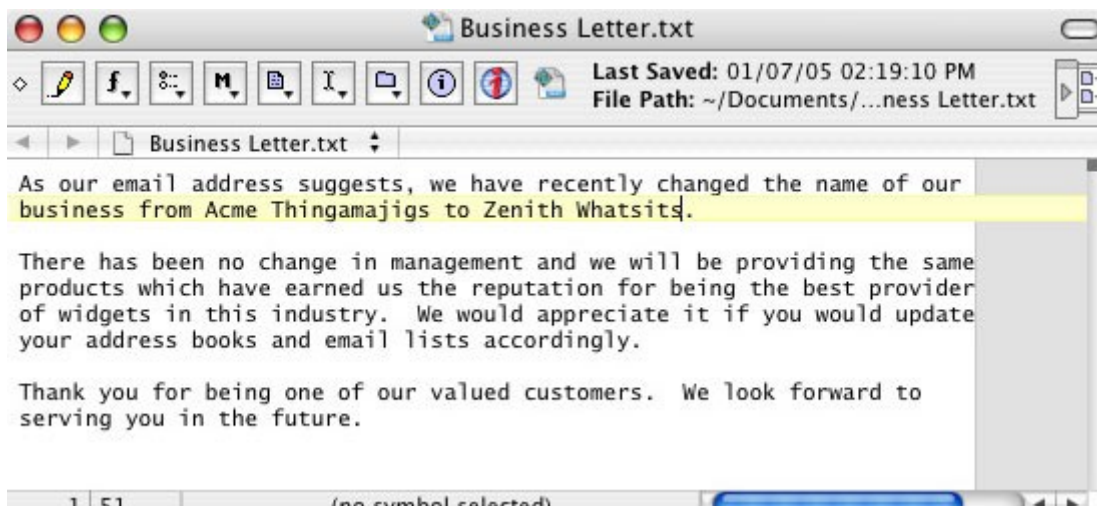


Sublime Text: Descarga desde <http://www.sublimetext.com/2> (Es de paga pero su versión gratuita es super buena).



```
settings.py x base.html x urls.py x
1 from django.shortcuts import render_to_response
2 from django.template import RequestContext
3 from testing.apps.ventas.forms import addProductForm
4 from testing.apps.ventas.models import Producto
5
6 def add_product_view(request):
7     if request.method == 'POST': # POST
8         form = addProductForm(request.POST)
9         info = "Inicializando"
10        if form.is_valid():
11            nombre = form.cleaned_data['nombre']
12            descripcion = form.cleaned_data['descripcion']
13            p = Producto()
14            p.nombre = nombre
15            p.descripcion = descripcion
16            p.status = True
17            p.save()
18            info = "Se guardo satisfactoriamente..."
19        else:
20            info = "Informacion con datos incorrectos"
```

TextWrangler (Para mac):



Después de esto creo que ya has encontrado uno que se adapte a tus necesidades.

Comenzar y terminar un archivo en PHP:

Así como en html se comienza un archivo con <HTML> y se termina con </HTML> en PHP es parecido, Para comenzar un archivo se hace con <?php y para terminarlo se hace con ?>

Ejemplo:

```
<?php
```

Todo el código PHP aquí

```
?>
```

NOTA: Muchos usan <? Sin el php, se puede hacer pero no lo recomiendo ya que en algunos servidores puede marcar error.

Impresión en pantalla:

Para imprimir algún texto en la pantalla es necesario usar la función **echo** seguido de comillas (Simples ó dobles) Después lo que quieras imprimir seguido de el cierre de comillas

NOTA: Para indicarle a PHP que ya terminamos de escribir una función se usa el carácter ; (Punto y coma)

Ejemplo

```
<?php
```

```
echo "Este es mi primer programa en PHP";
```

```
?>
```

Podemos usar print como alternativa a echo:

```
<?php
```

```
print "Este es mi primer programa en PHP";
```

```
?>
```

Dato: Si miras el código de fuente saldrá como si hubiese sido HTML.

Comentarios

Los comentarios sirven para determinar una acción y de esa forma hacer más entendible tu código.

Hay varias formas de comentar un script

1. `<?php`

```
echo "Este es mi primer programa en PHP";  
//Mi primera línea  
?>
```

2. `<?php`

```
echo "Este es mi primer programa en PHP";  
#Mi primera línea  
?>
```

3. `<?php`

```
echo "Este es mi primer programa en PHP";  
/*Mi primera línea*/  
?>
```

NOTA: Cabe mencionar que los comentarios no aparecerán para nada en tu código.

Salto de línea:

Es importante explicar esto ya que si en algún script tu pones:

```
<?php  
echo "Este es mi primer programa en PHP";  
echo "Hola";  
?>
```

Te imprimirá: "Este es mi primer programa en PHPHola"

Hay más de una forma de hacer saltos de línea, la que yo suelo usar es el tag xHtml `
`

Ejemplo:

```
<?php  
echo "Este es mi primer programa en PHP<br />";  
echo "Hola";  
?>
```

Esto me imprimirá:

Este es mi primer programa en PHP
Hola

Tarea: Hacer un script que imprima en pantalla 5 nombres de personas que

quieres, especificando con comentarios quienes son ó porque lo(a) quieres.

Con esto damos por concluido el tema Primeros pasos.

Variables:

Tenemos varios tipos de datos pero por ahora veremos estas 4.

1-Numeric = Valores numéricos enteros

2-Bool= True y false

3-String= cadenas de texto

4-Float = Números decimales

Declaración de variables:

Para declarar una variable se pone un signo de dollar (\$), después se le asigna un nombre a la variable (No puede tener espacios ni caracteres "raros") seguido de el signo igual (=) y el valor de la variable.

NOTA: las cadenas de texto van entre comillas las otras no.

Ejemplo:

```
<?php
$cadena="Okol";
$numerica = 15;
$float = 14.155;
$bool = True;
?>
```

Aquí expusimos los 4 diferentes tipos de datos; para saber de que tipo es una variable podemos usar la función gettype()

Ejemplo:

```
<?php
$cadena="Okol";
$numerica = 15;
$float = 14.155;
$foo = True;
echo gettype($cadena);
?>
```

Para cambiar el tipo de algún dato podemos usar settype().

ejemplo:

```
<?php
$cadena="Okol";
$numerica = 15;
$float = 14.155;
$foo = True;
settype($numerica "string");
?>
```

Como podemos ver le cambiamos de numeric a string.

NOTA: Para imprimir una variable en pantalla no hace falta comillas, para imprimirla frente a un texto se usa la "Concatenación"

Ejemplo:

```
<?php
$var = 'StrikeGeek';
echo 'Pertenesco a la comunidad de '.$var;
echo $var.'Es mi comunidad;
echo 'La comunidad de '.$var.' Es muy buena';
?>
```

Otra forma para no concatenar es poner todo entre comillas dobles.

Ejemplo:

```
<?php
echo "la comunidad de $var es muy buena";
?>
```

Operaciones Matematicas:

En este lenguaje también se pueden realizar operaciones matemáticas, podemos dividir sumar restar multiplicar y más cosas, pero solo explicaré las primeras cuatro.

Los operadores son los siguientes:

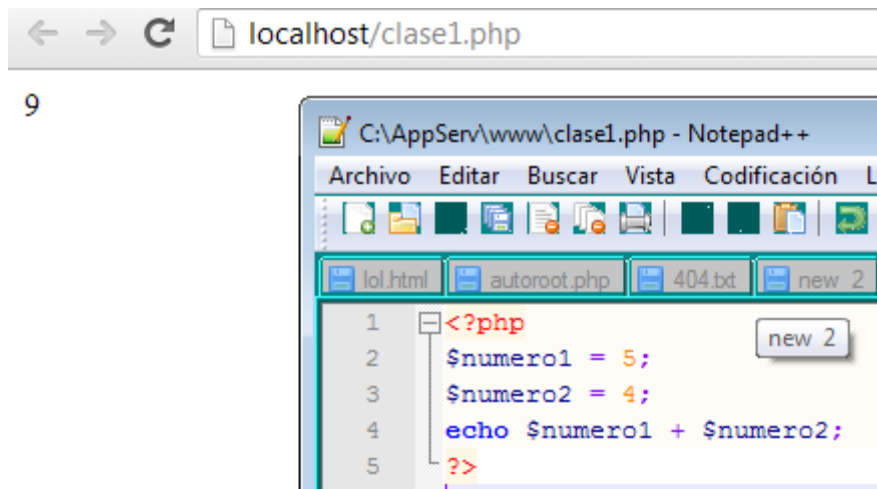
+ = Suma
- = Resta
/=división
*=Multiplicación

Entonces realicemos nuestro primer ejemplo:

```
<?php
$numero1 = 5;
$numero2 = 4;
echo $numero1 + $numero2;
```

?>

Resultado:



Ahora si queremos almacenar algún resultado en una variable seria claramente así:

```
<?php
$numero1 = 5;
$numero2 = 4;
$suma = $numero1 + $numero2;
echo $suma;
?>
```

Cadenas tipo string:

Las cadenas tipo string sirven para almacenar algún texto en una variable.

Ejemplo:

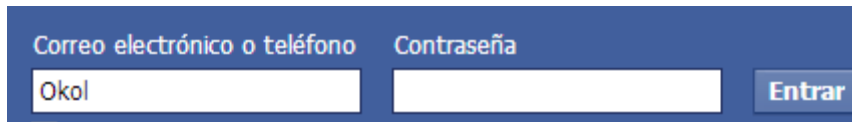
```
<?php
$hola = 'Hola mundo';
$adios = 'Adios mundo';
echo $hola.<br />;
echo $adios;
?>
```

Tarea: Hacer una multiplicación una división una suma y una resta almacenando los valores en variables y escribir 10 oraciones con variables concatenadas.

Formularios HTML, GET y POST

Bueno, hasta ahora vamos bastante bien ¿No? Pues pasaré a explicar algo un poco más de interés...

¿Qué es un formulario?



A screenshot of a login form. It has a blue header bar. Below the header, there are two input fields: the first is labeled 'Correo electrónico o teléfono' and contains the text 'Okol'; the second is labeled 'Contraseña' and is empty. To the right of these fields is a blue button labeled 'Entrar'.

Eso es un formulario. ¿Qué contiene?

Contiene dos input de tipo text y uno de tipo submit.

¿Cómo se hace un formulario?

Para crear un formulario comenzamos poniendo la etiqueta < form>

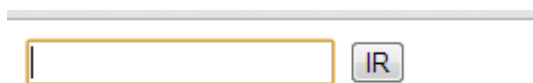
Seguido de sus parámetros (Por que método será enviada la información[GET Ó POST] y hacia que pagina va dirigido)

Veamos el primer ejemplo de formulario:

```
<form method="GET" action="index.php">  
<input type="text" name="hola">  
<input type="submit" value="IR">
```

El formulario será enviado por método GET y va dirigido a el index.php, tenemos algo llamado <input> de texto que su nombre es hola y tenemos un <input> de tipo submit que se encargará de enviar la información.

Así se vería en nuestra pantalla:



A screenshot of the rendered HTML form. It shows a single-line text input field with a yellow border. To the right of the input field is a small, square button with a grey border and the text 'IR' inside.

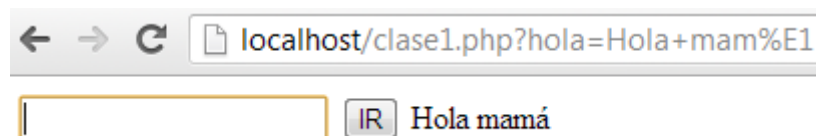
NOTA: la diferencia entre GET y POST es que GET mostrara la información que se esta enviando a través de la url y POST no mostrará eso, por lo tanto para mi es más seguro hacerlo con POST.

Bueno continuando al tema ya tenemos nuestro formulario en HTML, ahora pasaremos a hacer la parte de PHP.

```
<form method="GET" action="clase1.php">
  <input type="text" name="hola">
  <input type="submit" value="IR">

<?php
$variable = $_GET['hola'];
echo $variable;
?>
```

Para declarar \$_GET de preferencia es almacenarlo en una variable y el nombre que se le esta dando a el input tipo texto es el mismo que se le dará a \$_GET entonces de esa forma se podría decir que el valor de \$variable va a ser el mismo que tu ingreses en el input de texto y lo que haremos con echo va a ser imprimir en pantalla el texto que tu ingreses.



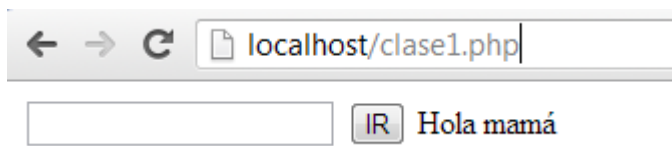
Como puedes ver en la url se muestra la información que enviaste. Si lo hacemos por el método POST solo hay que sustituir en donde dice GET por POST.

Ejemplo:

```
<form method="POST" action="clase1.php">
  <input type="text" name="hola">
  <input type="submit" value="IR">

<?php
$variable = $_POST['hola'];
echo $variable;
?>
```

Como pueden ver no muestra nada en la url:



Tarea: Crear un formulario con 3 campos de texto

1- Que pongas tu nombre

2-Que pongas tu edad

3- Que pongas tu altura

y que el script imprima digamos:

Tu nombre es Adrian Tienes 16 años y mides 1.71

Estructuras de control:

Operadores de comparación:

Ejemplo	Nombre	Resultado
<code>\$a == \$b</code>	Igual	TRUE si <code>\$a</code> es igual a <code>\$b</code> después de la manipulación de tipos.
<code>\$a === \$b</code>	Idéntico	TRUE si <code>\$a</code> es igual a <code>\$b</code> , y son del mismo tipo.
<code>\$a != \$b</code>	Diferente	TRUE si <code>\$a</code> no es igual a <code>\$b</code> después de la manipulación de tipos.
<code>\$a <> \$b</code>	Diferente	TRUE si <code>\$a</code> no es igual a <code>\$b</code> después de la manipulación de tipos.
<code>\$a !== \$b</code>	No idéntico	TRUE si <code>\$a</code> no es igual a <code>\$b</code> , o si no son del mismo tipo.
<code>\$a < \$b</code>	Menor que	TRUE si <code>\$a</code> es estrictamente menor que <code>\$b</code> .
<code>\$a > \$b</code>	Mayor que	TRUE si <code>\$a</code> es estrictamente mayor que <code>\$b</code> .
<code>\$a <= \$b</code>	Menor o igual que	TRUE si <code>\$a</code> es menor o igual que <code>\$b</code> .
<code>\$a >= \$b</code>	Mayor o igual que	TRUE si <code>\$a</code> es mayor o igual que <code>\$b</code> .

IF:

If es una estructura de control condicional; en ejemplos cotidianos es como decir:

Si tienes IFE puedes pasar al bar, sino te quedas afuera

La estructura de el if es así:


```

<?php
$numero = 18;
if($numero >= 18)
{
//Si $numero es igual o mayor a 18 ejecuta esto
}else{
//Si $numero es menor a 18 ejecuta esto
}
?>

```

En el ejemplo de aquí arriba podemos ver que la condición es que la variable \$numero sea igual ó mayor a 18.

Cabe mencionar que envés de los comentarios se puede añadir cualquier código; es decir si la condición se cumple ejecuta tal código y si no se cumple ejecuta este otro.

Elseif

Elseif es también una estructura condicional solo que no ejecuta solo una condición, ejecuta todas las que tu le digas es casi igual que If. Bueno veamos un ejemplo de su funcionalidad

```

<?php
if ($a > $b) {
    echo "a es mayor que b";
} elseif ($a == $b) {
    echo "a es igual que b";
} else {
    echo "a es menor que b";
}
?>

```

Como podemos ver se ejecutan dos condiciones y si ninguna de las dos se cumple se ejecuta un mensaje por “Default” lo cual viene siendo else. Es lo mismo que if solo es saber dónde van los corchetes ({}).

Tarea: Hacer un script que compruebe tres nombres (con formulario) y si no es ninguno de esos tres que muestre un mensaje que diga “No eres ninguno”

Desarrollando un pequeño wargame

Bueno decidí que nos podemos tomar un pequeño descanso con lo que ya sabemos y para que abran un poco mas su mente desarrollaremos un pequeño wargame.

Pueden hacerlo como gusten, yo lo crearé de los más usados (Busca el password).

NOTA: Si no les interesa esto podemos pasar al siguiente capítulo.

NOTA2: debemos un poquito de conocimiento en html.

Yo hare un formulario y con if pondré de condición que la respuesta sea la que yo quiera.

Mi código es así:

```
<form method="POST" action="clase1.php">
  <input type="text" name="respuesta">
  <input type="submit" value="Enviar">
<?php
if(isset($_POST['respuesta']))
{
$res=htmlentities($_POST['respuesta']);
if($res == 'paperbyokolforunderc0de')
{
echo 'Felicidades pasaste';
}else{
echo 'Sigue intentando';
}
}
?>
<br /><A HREF="respuesta.txt">Wargame By Okol</A>
```

Bueno pasaré a explicar un par de funciones que no hemos visto y que me tome la libertad de añadir al código.

isset() = Según php.net determina si una variable esta definida y no es NULL.

es decir que si no pusiéramos if(isset(\$_POST['nombre'])) la variable estaría definida y imprimiría el mensaje de que la respuesta no fue correcta así que por lo tanto si ponemos isset no mostrara nada hasta que la variable tenga un valor.

htmlentities() esa la añadí por seguridad ya que convierte caracteres html por lo tanto evita una inyección XSS.

Mi wargame lo que hace es una condición, si la variable \$res es igual a paperbyokforStrikeGeek le dice que es correcta y si no es asi le dice que siga intentando.

Switch

Switch es otra estructura condicional, en lo personal para mi es más cómodo usarlo cuando tienes varias condiciones.

La estructura es así:

```
1  <?php
2  $var = 100;
3  switch ($var)
4  {
5      case 5:
6          echo 'es igual a 5';
7          break;
8      case 10:
9          echo 'es igual a 10';
10         break;
11     case 15:
12         echo 'es igual a 15';
13         break;
14     default:
15         echo 'No es ninguna de las 3';
16     }
17  ?>
```

Tiene 3 condiciones,

- 1-Que \$var sea igual a 5 y si se cumple imprime "Es igual a 5", si no se cumple pasa a la segunda.
- 2-Que var sea igual a 10 y si se cumple imprime "Es igual a 10", si no se cumple pasa a la tercera.
- 3- Que var sea igual a 15 y si se cumple imprime "Es igual a 15", si no se cumple entonces se imprime un mensaje por default el cual dice "No es ninguna de las 3".

También lo podemos usar con strings de esta forma:

```
1  <?php
2  $var = 'Okol';
3  switch ($var)
4  {
5      case 'Adrian':
6          echo 'Eres adrian';
7          break;
8      case 'Sanko':
9          echo 'Eres Sanko';
10         break;
11     case 'Antrax':
12         echo 'Eres Antrax';
13         break;
14     default:
15         echo 'Puedes ser adrian Sanko ó Antrax pero no eres Okol';
16     }
17  ?>
```

Con esto damos por concluido el taller de este mes, espero hayan disfrutado y aprendido.

No es una despedida, estos talleres estarán saliendo una vez al mes.
Si están siguiendo el taller ya debemos tener un conocimiento de lo que es php y algunas funciones básicas.

Después de leer todo esto recomiendo practicar mucho.

Saludos.